ISSN: 2167-1435
v30 n2558

Evaluating Effectiveness of Pair Programming as a Teaching Tool in Programming Courses

Silvana Faja
sfaja@ucmo.edu
School of Accountancy and
Computer Information Systems,
University of Central Missouri,
Warrensburg, Missouri, 64093, USA

Abstract

This study investigates the effectiveness of pair programming on student learning and satisfaction in introductory programming courses. Pair programming, used in the industry as a practice of an agile development method, can be adopted in classroom settings to encourage peer learning, increase students' social skills, and enhance student achievement. This study explored students' perceptions on effectiveness of pair programming and the influence of student's level of experience with this activity and perceived partner involvement on effectiveness outcomes. Findings suggest that the more students are involved in this activity, the more they enjoy it and the more they learn by collaborating with their partners. When comparing different effectiveness measures, their perceived learning, quality of work, and enjoyment during pair programming was found to be at a higher level than increased productivity outcome.

Keywords: Pair Programming, Teamwork, Collaborative Learning, Programming Course.

1. INTRODUCTION

Software development is typically a process that requires the coordinated efforts of the members of one or more teams. As such, it is important that computer programming courses provide students not only with technical knowledge, but also with the skills required to work in real-life projects. It is not sufficient for IS graduates to be technically competent. Social competence, such as teamwork and communication, are also important (Figl, 2010).

Pair programming is one of the collaborative leaning activities that has found support in academic environments as a promising strategy to approach learning programming. This paper presents a study on the implementation of pair programming as a team-based activity in information systems courses. It discusses what was learned about the impact of this type of collaborative activity on students' attitudes and learning. The following section provides a review of existing research on pair programming, collaborative learning and research questions and hypotheses. Next, the description of the study's methodology, results of data analyses and discussion are presented.

2. PAIR PROGRAMMING AND COLLABORATIVE LEARNING

Pair programming is the term used to describe the process in which two programmers work side by side, on the same task at one computer designing and coding the same algorithm. It is suggested that there are typically two roles in pair programming: the"driver", who has control of the mouse and the keyboard and the "navigator", who observes the work of the driver, offers suggestions and corrections to both algorithm and code. Each programmer

takes a turn at being the "driver" and the "navigator". The two programmers collaborate in designing, coding and reviewing. Pair programming is one of the key practices in Extreme Programming (XP) (Beck, 2000). XP is one of the most popular agile software development methodologies with a strong focus on personal interactions among software developers. Benefits of pair programming include ability to notice more details when working in pairs, encouragement of best programming practices and sharing expertise (Wray, 2010).

Collaborative learning involves groups of students working together. Pair programming can be considered a model of collaborative learning. It incorporates the critical attributes of a collaborative learning activity: 1) common task; 2) small group learning; 3) cooperative behavior; 4) interdependence; and 5) individual accountability (Preston, 2005). There has been a significant amount of research on pair programming used as a collaborative learning tool in the traditional classroom. More recently, pair programming has been adopted in distance education, also known as virtual pair programming or distributed pair programming (Hanks, 2008; Edwards, Stewart & Ferati, 2010; Zacharis, 2011).

Studies on pair programming in the classroom have sought to capture both students' attitudes towards pair programming, as well as measure the actual benefits of pair programming such as improved learning and academic performance. Most of the previous research that has explored students' perceptions suggests that students have a positive attitude toward collaboration and communication that takes place during pair programming (Howard, 2007) and that they perceive that pair programming helped them develop teamwork skills (Cliburn, 2003; Edwards, Stewart & Ferati, 2010). It has been reported that students enjoyed working in teams (Williams & Kessler, 2001; Cliburn, 2003; McDowell et al., 2006; Howard, 2007; Chigona & Pollock, 2008; Mentz at al., 2008; Zacharis, 2011). In addition, students who worked in pairs reported higher confidence in the correctness of program solutions compared to individual programmers (Williams & Kessler, 2001; Werner, Hanks & McDowell, 2004; McDowell et al., 2006; Braught, Wahls & Eby, 2011). Several studies indicate that pair programming reduced student frustration (Howard, 2007; Simon & Hanks, 2008; Braught, Wahls, & Eby, 2011). Collaborative learning is considered one of the main benefits of pair programming in both professional and educational setting. Several studies indicated that students perceive they learned more by working with a partner than they would have by working alone (Cliburn, 2003; Carver et al., 2007; Edwards et al., 2010).

Other studies have focused on the impact of implementation of pair programming on academic performance and learning. The findings of these studies are not consistent. Some of them report higher assignment grades for pairs compared to solo programmers, greater likelihood of course completion and higher pass rates (Williams et al., 2002; McDowell et al., 2006; Mendes et al., 2006; Chigona & Pollock, 2008). These findings are not supported by all studies. A study by Zacharis (2011) showed that assignment grades for pairs were not significantly different from solo students. Individual exam grades (a measure of their knowledge of course material) were not different for those who used pair programming vs. those who did not (Williams et al., 2002; McDowell et al., 2006). There were cases when students felt that they understand their programs better when they work by themselves (Simon & Hanks, 2008).

In addition to academic performance, previous research has focused on other potential benefits of pair programming in the classroom, such as program quality and productivity. Many studies showed that pair programming improves the quality of the programs. Studies by Williams & Kessler (2001), McDowell et al., (2006), Chigona & Pollock (2008) and Zacharis (2011) report findings that students working in pairs produced better programs and higher software quality. Muller (2007) also showed that for simple problems, pair programming lead to fewer mistakes.

However, previous research does not provide the same support for the impact on productivity. Zacharis (2011) and Salleh et al., (2011) found that paired students were more productive than individual programmers and they completed tasks in a shorter amount of time. On the other hand, Simon and Hanks (2008) found that pair programming may or may not take less time. Another study by Williams & Kessler (2001) found that time for pairs and individuals was the

ISSN: 2167-1435
v30 n2558

same. This is in line with research findings on pair programming in general that indicate that collaborating pairs do not exceed the performance of its best member working alone (Balijepally et al., 2009).

Outcomes of the adoption of pair programming have not been the only areas examined by previous research. The review of the literature on pair programming in computing education revealed other areas that have been of interest to educators and researchers. Several studies focused on the pair formation approach. In some studies, pairs were assigned randomly (Mendes, Al-Fakhri & Luxton-Reilly, 2006; Muller, 2007; Hahn, Mentz & Meyer, 2009). Some other studies used matching pairs, based on criteria such as academic performance (Williams & Kessler, 2001; Choi et al., 2009; Zacharis, 2011). Van Toll, Lee & Ahlswede (2007) suggested that pair programming works best when programmers in a pair are of slightly different skill level. Bevan et al., (2002) also suggested pairing based on the skill level. Based on their experience of using pair programming in the classroom, disparity between the experience levels of students in a pair was one of the sources of intra-pair stress. Another factor of interest in team design was pair rotation versus same partner throughout the semester. Pair rotation was reported as an approach used by many studies (Carver et al. 2007; Braught et al., 2011). In some other studies students worked with the same partner during the semester (McDowell et al., 2006).

While research described above has provided theoretical and practical contributions to this area, adoption of pair programming as a teaching method can be further investigated using models and theoretical concepts from research on collaborative learning and teamwork effectiveness in educational settings. Little has been done to date to associate these two areas. The purpose of this study is to contribute in this direction by examining students' perceptions of effectiveness of pair programming as a learning activity and some of the factors that might influence these perceptions.

Based on the review of previous research, this study considered these factors of the effectiveness of pair programming as a teamwork activity: confidence in quality of work completed in pair, perceived productivity, enjoyment, and perceived learning. One approach to examine these perceptions would be to analyze which of the outcomes were considered to be more strongly impacted by the use of pair programming. The following research question will be addressed in this study:

Research question 1: What is the relative importance of perceived outcomes of pair programming by students?

Despite significant amount of research conducted on pair programming as a collaborative teaching method, there is still a need to investigate the factors affecting pair programming's effectiveness (Salleh, Mendes, & Grundy, 2011). As indicated in the previous literature review, most of the previous research has focused on factors such as the optimal pair formation approach, or level of complexity of the task. Literature on teamwork in educational settings suggests that previous experience with teamwork and task experience may have a beneficial effect on satisfaction with the teamwork activity. Littlepage, Robison & Reddington (1997) found that both group and task experience leads to better group performance. Wong, Shi & Wilson (2004) found that previous experience with teamwork influenced teamwork satisfaction. Hamlyn-Harris (2006) also looked at this relationship, but their study did not show a significant relationship between these two factors. This factor has yet to be investigated in the context of pair programming. In the reviewed studies, pair programming was implemented in a varying number of course activities ranging from a few labs or assignments to all labs and/or assignments. However, none of them has explored the effect that experience or the number of pair programming activities the students participates might have on their satisfaction and learning.

As such this study will test the following hypothesis:

Hypothesis 1: Students experience with pair programming activities is positively related to their perceptions of pair programming effectiveness.

Like any other team activity, one concern educators have about using pair programming is unequal participation of pair members. Cliburn

(2003) suggested changing partners throughout the semester as an approach to deal with "free-riders". A known phenomenon in teamwork activities is that of social loafing, which occurs when individuals working together in groups put less effort than when they work individually (Balijepally et al., 2009). Hasan and Ali (2007) considered perceived loafing in the context of IS education. Perceived loafing is the perception that one or more other group members are contributing less than they could to the group. In their study, perceived loafing was found to have a significant impact on the success of the project, but not on the student learning from the team project. So the learning was not affected by other members' efforts. Jassawalla, Sashittal & Malshe (2009) explored students' perceptions of consequences of social loafing in group work. They found that the quality of work of the social loafer in the team does not directly affect team performance, because the rest of the team works harder to compensate for the poor work of a loafing team member. Instead, it is the distractive effect of having such a team member that affects the performance.

In the context of this study, it would be of interest to explore how the partner's collaborative effort might influence the outcomes for the other member in the pair programming team. Pair programming approach as a method can help alleviate this issue since students have to switch roles periodically. Based on this and previous research findings, it can be assumed that partners will put a similar effort during pair programming activities and perceptions of partner's effort will not impact the students' perceptions of effectiveness of pair programming activities.

Hypothesis 2: Perceived partner's effort will not influence students perceptions of effectiveness of pair programming.

Next section describes the methodology used in this study to answer the research question and test these two hypotheses.

3. METHODOLOGY

The study was conducted in the context of an introductory programming course for computer information systems students in a midwestern university during the time period of four semesters. The purpose of the course is to introduce principles and practice of software development using the object-oriented programming approach and develop problem solving skills necessary to develop software solutions to problems.

The first semester served as a pilot semester for the implementation of the pair programming approach. The survey was not administered during this first semester to avoid any confounding effect of implementation issues. A total of 82 students were enrolled in the different sections of the course during the three subsequent semesters. The number of students completing the survey was 64 and the number of valid responses was 63.

There were 8-9 hands-on or lab sessions during the course of the semester. In some of the sessions students were asked to work in pairs. Before the first pair lab session, students were introduced to pair programming through a short presentation and a video that demonstrated how pair programming works. In addition to explaining the rules of this class activity, it was important that students understood that this was a component of an actual software development method. Each session covered a different activity. Following the guidelines for implementing pair programming in the classroom by Williams et al., (2008), students were assigned into pairs by the professor instead of letting students choose their partners. Students had different partners during the semester. During the lab sessions where pair programming was used, pairs were closely monitored to ensure that they were using the pair programming method, such as the use of a single computer and role switching, to ensure that both students in the pair had a chance to be in both roles during the session. At the end of the semester students were asked to complete a survey about their experiences with the pair programming activities during the semester. The survey was completed anonymously.

The survey items and the constructs they measured are presented in Table 1 in the Appendix. In addition to questions related to the four effectiveness outcomes, the survey included questions about the partner involvement during pair sessions and the number of pair lab sessions the student participated during the semester. Participants responded to statements using Likert scales

anchored by (1) Strongly Agree to (5) Strongly Disagree.

In this study, confidence in quality represents the strength of the student's perception of the quality of the pair's programming solution. Productivity represents student's perception of the time used to complete the exercise, and perceived learning represents student's perceptions of the learning that took place during the pair activities compared to individual ones. Most of the items were adopted by Chigona and Pollock (2008) and the last item was adopted by Howard (2007).

4. RESULTS AND DISCUSSION

A summary of students' responses for effectiveness items is presented in Appendix, Table 1. Data showed that except for the productivity, the majority of students either agree or strongly agree with statements that compare benefits of pair programming over working individually. Reliability analyses were initially performed on the items used to measure students' perception of pair programming activities. The alpha scores were within the acceptance range. Cronbach's alpha for confidence in quality was 0.74, enjoyment was 0.93, and perceived learning 0.84.

The first research question aims at exploring the relative importance of perceived outcomes and benefits of pair programming by students. Table 1 shows descriptive statistics for the variables that represent the various outcomes of pair programming activity.

Table 1. Descriptive statistics for effectiveness outcomes

Variables	Mean	Standard deviation
Perceived learning	2.22	0.82
Confidence in quality	2.37	0.88
Enjoyment	2.38	1.10
Productivity	2.70	1.12

Results show that the mean for perceived learning is the lowest among the four outcomes measured in this study, which means that the students have a higher level of agreement with statements that represent enhanced learning during pair programing activities. Perception of productivity, as measured by the time to complete the activity, has the highest mean. This indicates that students do not perceive that their productivity was increased during pair programing at the same level as the other effectiveness measures. In order to draw any conclusion about these differences in perceptions about the outcome measures and their ranking, paired samples t-tests were performed to determine whether these mean values are significantly different. Results of these analyses are shown in Table 2.

Effectiveness measures comparisons	Mean diff.	t	df	Sig
Perceived learning vs. Confidence in quality	-.15	-1.81	62	.074
Perceived learning vs. Enjoyment	-.16	-1.64	62	.106
Perceived learning vs. Productivity	-.48	-3.92	62	.000*
Enjoyment vs. Confidence in quality	.02	.20	62	.843
Confidence in quality vs. Productivity	-.33	-2.99	62	.004*
Enjoyment vs. Productivity	-.32	-2.56	62	.013*

Table 2. Results of t-tests for the differences among outcomes

These tests showed that students' perceptions of learning, their enjoyment and confidence in the quality of the program developed during the pair programming activity were significantly higher than their perceptions of productivity. Results of the tests also indicate that there are no significant differences among perceived learning, enjoyment and confidence in quality. Based on these analyses we can conclude that in this study students' perceptions of increased learning, improved quality of their work and enjoyment during the collaborative work in the pair programming activities are at similar levels. However, their perceptions of improved productivity due to working in pair are significantly lower than the three other outcomes. This findings does not necessarily diminishes the benefits of this activity. Compared to industrial settings where productivity and quality are the main concerns

associated with the adoption of pair programming, it can be argued that in educational setting productivity is of a lesser importance relative to learning and student engagement outcomes.

Another purpose of this study was to examine the role of students' level of experience or exposure to pair programming on the effectiveness outcomes. Hypothesis 1 states that this experience would be positively related to their perceptions of pair programming activities. Experience with pair programming was measured by the number of pair programming sessions students completed during the semester. In the sample used in this study, 7 students or 11% attended only one session, 28 students or 44% attended 2 sessions and 28 students attended 3 sessions. To test this hypothesis, regression analyses were conducted. The independent variable in these analyses was number of sessions attended. The dependent variables for each of the analyses were the effectiveness outcomes. Summary of results for these analyses are presented in Table 3.

Test	Outcome variables	β	t	p
1	Perceived learning	-0.27	-2.20	.032*
2	Confidence in quality	-0.17	-1.35	.183
3	Enjoyment	-0.27	-2.19	.032*
4	Productivity	-0.08	-0.62	.539

Table 3. Summary of regression analyses with number of sessions attended as independent variable

Results showed that the number of sessions completed has a significant effect on students' perceptions of learning and their enjoyment during these sessions. In other words, the more session they attended, the more they perceived to have learned more during pair programming and the higher the level of enjoyment. The results also indicate that the number of sessions does not have a significant effect on the confidence in quality and productivity. One explanation for these findings might be the fact that this study used pair rotation. Attending more sessions provided more opportunities to work with different classmates with different levels of skills. Also, as they became more familiar with the procedures of the pair programming activities they were able to enjoy the activity more.

In addition to the role of experience, this study considered the role of perceived partner effort. The second hypothesis stated that the perceived partner effort would not impact effectiveness outcomes. To test this hypothesis, regression analyses were performed. Summary of these analyses are presented in Table 4.

Test	Outcome variables	β	t	p
1	Perceived learning	0.37	3.08	.003*
2	Confidence in quality	0.42	3.65	.001*
3	Enjoyment	0.46	3.99	.000*
4	Productivity	0.40	3.35	.001*

Table 4. Summary of regression analyses with partner effort as independent variable

Results indicate that perceived partner effort had a significant effect on the effectiveness outcomes. This leads to the rejection of hypothesis 2. In this study, students' perceived partner effort has a positive significant effect on all effectiveness outcomes. The greater the student perceived their partners were equally contributing to the group activity, the greater was their perceived learning, confidence in work quality, enjoyment and productivity. These findings contradict those of some of the previous research where social loafing did not affect learning from the group activity or the quality of work. In this study, lack of partner contribution affects both learning and confidence in the quality of work. As for the other relationships, it is understandable that lack of participation from the other pair member would reduce the level of enjoyment and increase the time the complete the work.

5. CONCLUSIONS

Educators are always trying to find way to incorporate activities that would increase student engagement, learning, and foster their collaborative skills. Despite good intentions, these activities may not always have the desired outcomes of encouraging peer learning, increasing students' social skills, and enhancing student achievement. The objective of this study was to examine effectiveness of pair

programming as a collaborative learning activity in IS education, as perceived by students.

Several factors that represent the effectiveness of this collaborative activity were considered and two main research questions were addressed. First, students' perception for each effectiveness factor was considered in order to determine which outcome had the higher level of achievement. Second, the study focused on the effect of two factors identified by the teamwork literature, experience with the activity and partner involvement on the effectiveness outcomes.

Findings provide some useful insights on the adoption of pair programming as a classroom activity to encourage collaborative learning and foster teamwork skills. Participants in this study perceived participation in this activity as beneficial in several areas such as learning, confidence in the quality of their work and being able to accomplish work faster than they would be working individually. They also enjoyed this type of teamwork in the classroom.

This study also showed that the more students are involved in pair programming exercises, the more they perceive they learn through it more than they would if they worked individually, and the more they enjoyed it. This has important implications for educators interested in adopting this activity in the classroom. To increase the benefits to students, they should plan for several programming activities, to allow for student to become familiar with the procedures and some of the unique aspects of this type of collaboration work. Also, educators should implement measures to ensure that pair members contribute equally to the team activity. This study indicated that this aspect has a significant impact on the outcomes of this activity. It is important to enforce role switching and incorporate peer assessments as a motivation for student to be involved. Pair rotation, having different pair assignments for each session, could contribute to mitigation of social loafing. On the other hand, this could prevent bonding between team members, also known as "pair jellying" in pair programming.

Future research can examine which approach might be more beneficial for student learning. This study focuses on effectiveness of this collaborative activity from the students' perspective. Future studies can examine the impact of the factors analyzed here on more objective measures of effectiveness such as academic performance and work quality as assessed by the instructor.

6. REFERENCES

Balijepally, V., Mahapatra, R., Nerur, S., & Price, K.H. (2009). Are Two Heads Better than One for Software Development? The Productivity Paradox of Pair Programming. *MIS Quarterly, 33(1),* 99-118.

Beck, K. (2000) *Extreme Programming Explained,* Reading, MA: Addison-Wesley.

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,* 100-107.

Braught, G., Wahls, T., & Eby, L. M. (2011). The Case for Pair Programming in the Computer Science Classroom. *ACM Transactions on Computing Education, 11*(1), Article 2.

Carver, J.C., Henderson, L., He, L., Hodges, J., & Reese, D. (2007). Increased Retention of early Computer Science and Software Engineering Students usign Pair Programming, *20th Conference on Software Engineering Education and Training,* 115-122.

Chigona, W., & Pollock, M. (2008). Pair Programming for Information Systems Students New To Programming: Students' Experiences and Teachers' Challenges. *PICMET 2008 Proceedings,*1587-1594.

Choi, K.S.., Deek., F.P. & Im., I. (2009). Pair Dynamics in Team Collaboration. *Computers in Human Behavior, 25(4),* 844-852.

Cliburn, D. C. (2003). Experiences with Pair Programming at a Small College. *Journal of Computing Sciences in Colleges, 19*(1), 20-29.

Edwards, R. L., Stewart, J. K., & Ferati, M. (2010). Assessing the Effectiveness of Distributed Pair Programming for an Online Informatics Curriculum. *ACM Inroads, 1*(1), 48-54.

Figl, K. (2010). A Systematic Review of Developing Team Competencies in Information Systems Education. *Journal of Information Systems Education*, *21*(3),323-337.

Jassawalla, A., Sashittal, H. & Malshe, A. (2009). Students' Perception of Social Loafing: Its Antecedents and Consequences in Undergraduate Business Classroom Teams. *Academy of Management Learning and Education, 8*(1), 41-54.

Hahn, J., Mentz, E., & Meyer, L. (2009). Assessment Strategies for Pair Programming. *Journal of Information Technology Education, 8*, 273-284.

Hanks, B. (2008). Empirical Evaluations of Distributed Pair Programming. *International Journal of Human-Computer Studies, 66,* 530-544.

Hasan, B, Ali, J. (2007). An Empirical Examination of Factors Affecting Group Effectiveness in Information Systems Projects. *Decision Sciences Journal of Innovative Education, 5*(2), 229-244.

Howard, E. V. (2007). Attitudes on Using Pair-Programming. *Journal of Educational Technology Systems, 35*(1), 89-103.

Katira, N., Williams, L., Wiebe, E., Miller, C., Balik, S., & Gehringer, E. (2004). On Understading Compatibility of Student Pair Programmers. *ACM Technical Symposium on Computer Science Education,* pp.7-11.

Littlepage, G., Robison, W. & Reddington, K. (1997). Effects of Task Experience and Group Experience on Group Performance, Member Ability, and Recognition of Expertise, *Organizational Behavior and Human Decision Processes, 69*(2), 133–147.

McDowell, C., Werner, L., Bullock, H.E., Fernald, J. (2006). Pair Programming Improves Student Retention, Confidence, and Program Quality. *Communication of the ACM,49(8),* 90-95.

Mendes, E., Al-Fakhri, L.B., & Luxton-Reilly, A. (2006). A Replicated Experiment of Pair-Programming in a 2nd year Software Development and Design Computer Science Course. *Proceedings of the 11th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education,* 108-112.

Mentz, E., van der Walt, J., & Goosen, L. (2008). The effect of incorporating learning principles in pair programming for student teachers. *Computer Science Education, 18*(4), 247-260.

Muller, M. (2007). Do Programmers Pairs Make Different Mistakes than Solo Programmers? *The Journal of Systems and Software, 80,* 1460-1471.

Preston, D. (2005). Pair Programming as a Model of Collaborative Learning: A Review of the Research. *Journal of Computing in Small Colleges, 20*(4), 39-45.

Salleh, N., Mendes, E., & Grundy, J. (2011) Empirical Studies of Pair Programming for CS/SE Teaching in Higher Education: A Systematic Literature Review. *IEEE Transactions on Software Engineering*, 37 (4), 509-525.

Simon, B., & Hanks, B. (2008). First-Year Students' Impressions of Pair Programming. *ACM Journal on Educational Resources in Computing, 7*(4), Article 5.

Sfetsos, P., Stamelos, I., Angelis, L., & Deligiannis, I. (2009). An Experimental Investigation of Personality Types Impact on pair effectiveness in Pair Programming. *Empirical Software Engineering, 14,* 187-226.

Van Toll, T., Lee, R., & Ahlswede, T. (2007). Evaluating Usefulness of Pair Programming in a Classroom Setting. *Proceedings of the 6th IEEE/ACIS International Conference on Computer and Information Systems,* 302-308.

Werner, L., Hanks, B., & McDowell, C. (2004). Pair-Programming Helps Female Computer Science Students. *ACM Journal of Educational Resources in Computing, 4*(1), 1-8.

Williams, L. & Kessler R. (2001). Experiments with Industry's "Pair-Programming" Model in the Computer Sciene Classroom. *Computer Science Education, 11(1),*7-20.

ISSN: 2167-1435
v30 n2558

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

Williams, L., McCrickard, D. S., Layman, L., & Hussein, K. (2008). Eleven guidelines for Implementing Pair Programming in the Classroom. *Agile 2008 Conference,* 445-452.

Wray, S. (2010). How Pair Programming Really Works. *IEEE Software, January/February*, 50-55.

Wong, Y. K, Shi, Y., & Wilson, D. (2004). Experience, Gender Composition, Social Presence, Decision Process Satisfaction and group performance, *ACM Computer Science Press*, Dublin, Ireland, pp. 452-461.

Zacharis, N. Z. (2011). Measuring the Effects of Virtual Pair Programming in an Introductory Programming Java Course. *IEEE Transactions on Education, 54*(1), 168-170.

ISSN: 2167-1435
v30 n2558

APPENDIX

Outcome	Items	Strongly agree	Agree	Neither agree or disagree	Disagree	Strongly disagree
Confidence in Quality	I find that pair programming develops better projects than programming by myself.	21%	33%	27%	14%	5%
	More errors were found and fixed when we pair programmed.	16%	56%	14%	14%	0%
	I was more confident in the work when we pair programmed.	22%	46%	11%	18%	3%
Perceived Productivity	The work was finished quicker because of the pair programming.	13%	37%	25%	19%	6%
Enjoyment	I enjoyed programming with a partner more than programming alone.	21%	35%	17%	22%	5%
	If I had the choice I would work in a pair programming team again.	30%	30%	23%	11%	6%
	I liked using pair programming during the in-class labs.	30%	33%	19%	16%	2%
Perceived learning	I learnt more from doing the work because of the pair programming	21%	36%	27%	16%	0%
	It was helpful to discuss programming problems and solutions with a partner.	33%	52%	10%	5%	0
	I think that using pair programming during the in-class labs helped me better understand the concepts.	21%	36%	24%	19%	0

Table 1. Items used to measure effectiveness outcome and frequencies of student responses.