VOL. 10, NO. 8, MAY 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

ENHANCING LEARNING EXPERIENCE OF E-LEARNERS IN LABORATORY COURSES USING PAIR PROGRAMMING

N. Mohanraj[1], A. Sankar[2], and V. Senthil Kumaran[1]
[1]Department of Applied Mathematics and Computational Sciences, Coimbatore, India
[2]Department of Computer Applications, PSG College of Technology, Coimbatore
E-Mail: vsk.mca@gapps.psgtech.ac.in

ABSTRACT

Laboratory courses constitute one of the core competencies that graduates from information systems discipline are expected to possess. Laboratory courses in e-learning are just a curricular formality without bothering about the part played by such learning experiences. Lot of practice is required for e-learners for acquiring a good learning experience, for which motivation is an essential factor. Research has suggested that the lack of a formalised structure for laboratory courses may be one of the factors responsible for learners' negative impressions of e-learning and also for the high failure rate in e-learning. Ability to work in teams has been considered one of the most important learning outcomes of the laboratory courses. This study highlights the importance of laboratory courses in e-learning and investigates whether the use of pair programming in laboratory courses would enhance the learning experience of e-learners. The final objective is to provide new learning experience to motivate e-learners and present laboratory courses as an easy and attractive challenge using pair programming. Experiments were conducted in data structures, problem solving and C programming courses. Results indicate that the learning experience of both the learners and teachers were improved in laboratory course and also showed an improvement in failure rate.

Keywords: e-learning, learning experience, laboratory course, pair programming.

INTRODUCTION

Presently, teaching learning process in majority of e-learning systems is theoretical. Practical work is not given desired serious in most of the e-learning systems. Learners of information systems courses cannot be trained with focus on theory only which is going to be forgotten with passage of time (Van Der Vyver and Lane, 2003). Good programming skills are one of the core competencies that information system learners are expected to develop. However, learners and teachers agree that learning laboratory courses is a hard task in e-learning course. Learners need to be adequately motivated in order to learn programming in a successful and effective manner. Learners will be motivated when they interact with other learners and/or teacher (Furberg *et al*., 2013).

The main issue which may exacerbate e-learners' difficulties with laboratory courses is the lack of a formalised environment for collaborative peer learning (Preston, 2005). Some of the challenges that e-learners face in laboratory courses may be overcome by allowing learners to collaborate with their peers. The pedagogical advantages of learner interaction in collaborative construction of knowledge are grounded in the social constructivist perspective of learning. Based on the constructivist pedagogical approach, actual learning takes place when students actively construct their knowledge through social interactions with their peers (Van Der Vyver, 2003). Knowledge is discovered and constructed through communication and collective sense making. Collaborative learning benefits educators in computing domain. Engagement in collaborative activities causes individuals to master something that they could not do before the collaboration. We are interested in investigating how collaborative learning can be used to enhance learning experience of e-learners in laboratory courses. In e-learning, learners are located in geographically different locations; the current study employs a distributed collaborative programming technique referred to as distributed pair programming.

Distributed Pair programming is a novel and successful collaborative paradigm used in software industry (Salleh *et al*, 2011). The idea is that two programmers work collaboratively on the same program from the different locations. One programmer is designated as the 'driver' and has control of the input devices. The other programmer is designated as the 'navigator' and has the responsibility of reviewing the code that has been typed to check for deficiencies, such as erroneous syntax and logic, misspellings and design issues (Braught, Wahls and Eby, 2011). The navigator continuously examines the work of the driver, thinking of alternatives and asking questions. The driver and the navigator change roles frequently and different pairs are formed to facilitate the spread of information through an organisation. It is the opinion of the industry experts that programmers working in pairs produce shorter programs with better design and fewer bugs than those working alone (Vanhanen and Lassenius, 2007). This collaborative technique has also been successfully applied to the teaching of computer programming for beginners in classroom and a wide range of benefits have been reported, such as improved quality of code, decreased time to complete, improved understanding of the programming process, enhanced communication skills and enhanced learning (Salleh *et al*, 2011; Williams and Upchurch, 2001). In pair programming, one leaner will follow another learner and will try to imitate. One learner will be the camaraderie of another learner. It is found that mathematical logic skills were enhanced when pair programming practice was followed. It is also proved that collaborative learning enhanced student experience in producing Wiki websites (Tsai *et al*., 2011). Pair programming had positive effects on student engagement and performance within computer science lectures (Maguire and Maguire 2013).

The intricacies of learning programming laboratory courses in e-learning environment

Computer programming laboratory courses may be viewed as a method for some problem solving. Knowledge transfer is expected to be easier if the prior knowledge and/or experience of the learners are similar to the knowledge transfer being done. Programming laboratory courses are greatly enhanced through learner-learner interaction. Where there is problem in this knowledge transfer such as an error which the learner cannot explain, overcoming that problem is faster through minimised distance between teaching and teaching (Denner *et al*., 2014). Laboratory courses are not similar to other courses. There is some uniqueness to laboratory courses in e-learning that must be considered when one contemplates an ideal environment for learning programming courses. Teaching and Learning programming courses have their intricacies as well as problems not fully overcome (Carver *et al*., 2007). Online learning has major benefits but when programming is taught online, another set of concerns must be considered. Additionally the benefits derived from an online environment for different courses differ. International Data Corporation reports that enrolments in e-learning courses are growing at thirty three percent a year and will continue to climb. Most of the e-learning systems provide Virtual Learning Environment (VLE) and Integrated Development Environment (IDE) to offer laboratory courses. There are new ways in learning program as such using virtual learning environments, evolving programming environments and software programs and applications. The information technology exists for this application providing for computer-based instruction or asynchronous and synchronous learning networks. Virtual learning environment (VLE) is a set of teaching and learning tools intended to develop a student's learning capability via computers and the Internet in the learning process (Rosenberg, 2001). Learners with low motivation or bad study habits may fall behind.

Research overview and hypothesis

Much of the research on distributed pair programming as a pedagogical technique has focused on the teaching of introductory programming to beginners, with fewer studies investigating its applicability for expert programmers. Also, while there is a growing body of research in the area, more studies have focused on pair formation and its effectiveness. In the current study we describe the outcomes of adopting a collaborative pair programming paradigm for enhancing learning experience of e-learners in laboratory courses. We compare the learning efficacy of e-learners in laboratory courses with pair programming and without pair programming. The continuity of the learner cohort permits analysis of various outcomes of the pedagogical intervention, such as learning experience and efficacy. Thus, the following hypothesis is formulated about the e-learners in laboratory courses using pair programming in the improvement of learning experience and efficacy:

H1: The e-learners who use pair programming will have better learning experience than those who do not use it.

H2: The e-learners who use pair programming in laboratory courses will obtain higher grades than those learners who do not use it.

H3: To establish whether learners benefit from a peer programming intervention in terms of their academic performance in both continuous assessment and examination results,

H4: Dropout in e-learning will be decreased because of the satisfaction level of e-learners.

METHOD

Participants

Experiments were conducted with the e-learners of our e-learning system “psglms” taking data structures laboratory, and Problem Solving and C programming laboratory courses. These two laboratory courses are offered at both UG and PG level in science and engineering stream. Table-1 shows the total participants and the same is shown as chart in Figure-1.

Table-1. Total participants

	Name of the laboratory course	Sex						Total
		Male	Female	Science stream		Engg. stream		
				CS	Non-CS	CS	Non-CS	
UG Level	Problem Solving and C programming Lab	180	114	74	47	116	57	294
	Data Structure Lab	152	97	87	30	91	41	249
PG Level	Problem Solving and C Programming Lab	45	92	58	19	43	17	137
	Data Structure Lab	41	67	53	21	18	16	108

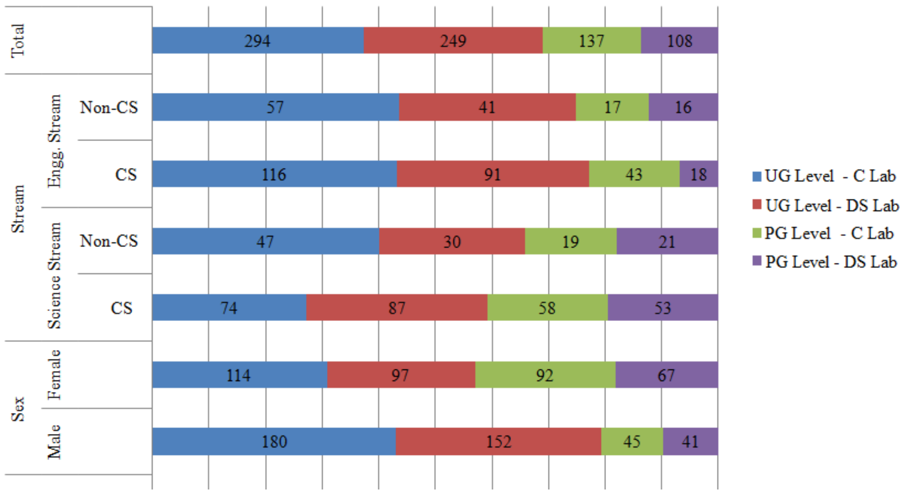

Figure-1. Chart showing the total number of participants.

Design

A quasi-experimental design was employed with discipline of study, level of study and gender as the key independent variables. Students' performance in lab examination was the key dependent variable, but measures of programming confidence, perceptions of the pair programming intervention, dropouts were also examined.

Procedure

The experiment took place during December 2013 and April 2014. Students' motivation, learning experience and satisfaction were analysed by means of observation and satisfaction questionnaires. In addition it was necessary to analyse the effects of the system on students' academic outcomes and dropout of students. For this, the experimental research method was applied (Oncu and Cakir, 2011) and experimental and control groups were established in order to identify a relationship between variables. In order to evaluate the experience all students took the two courses during the academic year 2013-2014 were considered as belonging to the experimental group. Their academic results and drop outs would then be compared with those obtained by the students taking the same courses during the next academic year. The following premises were established:

- The only difference between the two academic years would be the use of this system in the first and not the second.
- A total of 788 students with different gender and enrolled in the two lab courses in UG / PG level participating in the experiment.
- Necessary training was provided.

Instruments and data collection

Two instruments were used in this study: a) the students' final exam grades in the courses for both academic years, and b) a survey, which measures students' satisfaction with their learning experience using the system. Specifically, the survey was composed of three different parts:

- Personal data for statistics: age, gender, computer skills.
- Five-score Likert-type scale items, which ranged from "Strongly Disagree" to "Strongly Agree", for analysing the level of satisfaction.

- Yes/no items for assessing both the quality of the problems posed and the functionality of the on-line Judge.

Data from the survey was collected on-line when the courses finished. The survey was completed by all the students. The data collected on students' outcomes (final exam grades) were analysed for group comparison using the Student T-Test. This statistic indicates whether the means of two groups are statistically different from each other in order to be able to compare them. In addition, in order to check whether students' satisfaction differed according to gender, subject of study and UG/PG level and to investigate whether there was any interaction among these variables, a two-way analysis of variance (ANOVA) was also conducted.

RESULTS AND DISCUSSIONS

Comparisons in Time spent on learning, and academic performance

Table-2 outlines the overall mean and standard deviation (SD) of the scores for total time spent on learning, and final assessment across the two academic years. A series of dependent t-tests revealed a significant difference between the two groups on any of these measures (p > .05).

Table-2. Mean, standard deviation (SD) and T test of the scores for total time spent on learning and final assessment mark across the two academic years.

	Academic year 2013-2014 (without pair programming)		**Academic year 2014-2015** (with pair programming)		**T test**	
	Mean	**SD**	**Mean**	**SD**	**T**	**p**
Total time spent on the system	162.45	2.0256	276.87	0.819	2.017	0.030
Final exam result	52.59%	1.383	87.37%	0.753	2.142	0.025

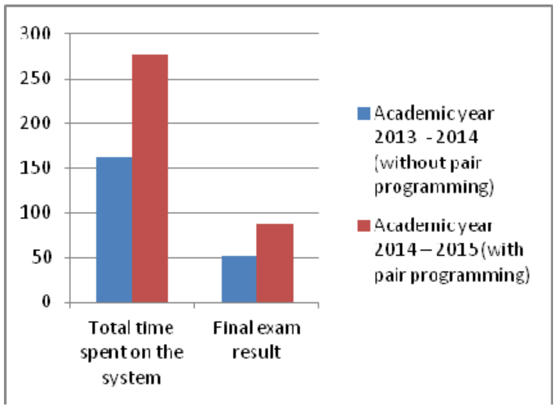

Figure-2. Chart showing the comparisons in Time spent on learning and academic performance

An improvement in the final score can be observed when pair programming is applied to learning programming in e-learning. According to the results in Figure-2, students who used pair programming achieved significantly better academic outcomes than those who did not use it across all courses. This result also shows that students more interestingly spending more time in learning. This result indicates that the hypothesis H1 is supported. The trend, as seen in Table-3, clearly indicates that as the semester progressed, the students showed more interest in learning programming and the drop out rate is reduced. This also shows that for students having no programming background, the maximum learning experience came from the lab work. By the end of the semester, all students performed well in the final examinations.

Comparisons in dropout rate and failure rate

Table-3 gives the overall failure rate and dropout rate across the two academic years.

Table-3. Overall failure rate and dropout rate across the two academic years.

	Academic year 2013 - 2014 (without pair programming)	**Academic year 2014- 2015** (with pair programming)
Failure rate	0.317	0.048
Dropout rate	0.108	0.019

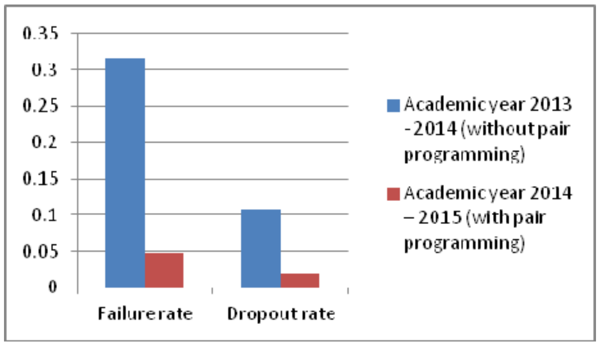

Figure-3. Chart showing the overall failure rate and dropout rate across the two academic years.

The major problem in e-learning is the lack of confident and motivation in learning. It can be observed that when pair programming is applied to learning programming laboratory courses in e-learning the overall failure rate and dropout rate is significantly reduced. According to the results in Figure-3, learners who used pair programming are confident in completing the course successfully. This gives a positive sign for the universities and organizations that uses e-learning system.

Analysis of the satisfaction

In this section we analyse the students' degree of satisfaction with the use of pair programming in learning programming in e-learning system based on the survey data. The purpose of this analysis is to validate the usefulness of the system, since several studies (Donohue and Wong, 1997; Levy, 2007) suggest that students' satisfaction and motivation are important factors in measuring the success or effectiveness of the e-learning process. The analysis of results is done in general terms and also answering the research question and testing the hypotheses formulated.

Once the results of the surveys were available, we have studied their reliability. Cronbach's alpha was tested and the calculated alpha value for the learning experience of e-learners in programming laboratory courses was 0.95, indicating very high reliability (Straub, 1989). In general terms, the survey data shows the learning experience was evaluated positively by students. Figure-4 summarises the survey results for each course in UG/PG level, where bars represent the average score assigned to each item (5 being the maximum score). It is also clear that the students think that it helped to achieve academic excellence in learning goal. Most students reported a high learning experience with the pair programming.

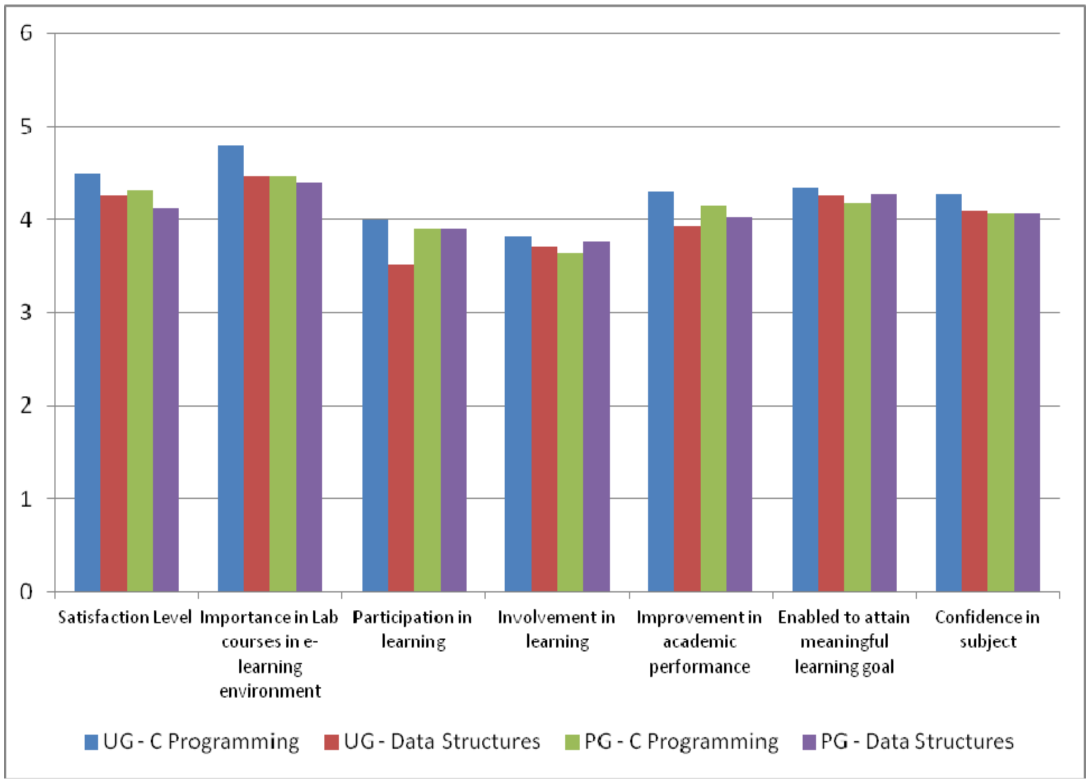

Figure-4. Summary of the survey results.

Table-4. Mean and Standard Deviation (SD) of Learner's opinion regarding learning experience.

Survey items regarding learners' learning experience	**Mean**	**SD**
Satisfaction Level	4.301	0.725
Importance in Lab courses in e-learning environment	4.524	0.779
Participation in learning	3.828	1.018
Involvement in learning	3.75	0.952
Improvement in academic performance	4.103	0.871
Enabled to attain meaningful learning goal	4.271	0.729
Confidence in subject	4.132	0.785

Table-4 shows a more detailed statistical study (with the mean and the standard deviation) of the different items in the opinion survey. In summary, the results indicate that the learners' level of satisfaction and learning experience in learning laboratory courses in e-learning environment that uses pair programming was 4.301.

Table-5. Two-way ANOVA for the learner's satisfaction.

	Sum of squares	**Df**	**Mean square**	**F**	**P**
Gender	4.919	1	4.919	4.88	0.037
Level of study	3.634	1	1.817	1.8	0.184
Gender x level of study	0.378	1	0.189	0.19	0.828
Error	28.229	785	1.008		
Total	37.159	788			

Finally, the different hypotheses proposed about the relationship between the level of satisfaction and gender and level of study had to be validated. Results of Table-5 indicate that students' learning experience was not different in relation to gender (F ¼ 4.88, $p > 0.05$), level of study (F ¼ 1.8, $p > 0.05$) or the interaction of both (F ¼ 0.19, $p > 0.05$). Therefore, no differences are found in learning experience between male and female students and among students with different levels of study.

Based on the results shown in Table-4 and the responses and statements of the learners, some of the evident advantages of pair programming that we could bring out effectively in our e-learning laboratory course were:

Results of Using Pair Programming in Industry are positive - Many teams report improved product quality (Dybå *et al*., 2007; Vanhanen *et al*., 2007) when using pair programming. Specifically, one large telecommunications company in Finland whose software engineers almost exclusively worked in pairs had only five field failures in one and a half years of production.

Economic feasibility - The tradeoff between increased speed and quality vs. increased resources has been examined in two economic models (Padberg and Müller, 2003). The model developed by Erdogmus and Williams (2003) is a net present value model to examine the economic feasibility of pair programming. Central to his model is the consideration that value lies in the ability of a pairs to deliver and get paid for a working product, even a partial working product, to customers more rapidly.

Collaboration and confident building - Studies have shown that pair programming creates an environment conducive to more advanced, active learning and social interaction, leading to students being less frustrated, more confident, and more interested in IT (McDowell *et al*., 2006). Students who work in pairs tend to produce programs of higher quality and have higher course passing rates (Nagappan *et al*., 2003) even when students pair program in a distributed manner. It has improved the team work quality among learners. Learners feel the fact that paired programmers were more comfortable in clearing their doubts with their partners. When they worked in pair, learners showed the confidence in learning the subject. They were able to state when something was right and the ability to admit when something was wrong. Another advantage that was found in the students' responses was that paired learners developed the tendency to work together even outside the class.

Learning efficacy - According to Bevan *et al*. (2002), pairs spend less time working on assignments than individuals. In our experiment also inexperienced-pair programmers could produce code of the same quality in the same time as experienced-solo programmers. Although paired programmers had to write more code, (individual and combined tasks), they seldom took more than an hour to complete the task. This happened because when students attacked the combined task, the students working in pairs could work out the logic much easily and in less time as they had already grasped the concept while working on the individual tasks.

Skill development - Collaborative programmers talked, discussed, and argued more than the individual programmers. They had the additional and increased opportunity to learn by watching how their partners approach a task, how they use programming language features, and how they use the development tools (Williams, Kessler, Cunningham, and Jeffries 2000). They had the opportunity to better understand someone else's view by understanding how an issue looks from their partner's perspective. At such times, drawing from each person's unique talents and experience, a process known as 'pair brainstorming' occurs resulting in highly effective problem solving. The simple act of explaining an issue often leads to the solution faster.

Quality in learning - Knowledge is constantly shared between pairs (Jason, 2004). Though no specific measure were made about program defects, the instructors felt that compared to earlier batches when such a pairing was not tried out, the quality of the programs produced by learners improved significantly.

4. CONCLUSIONS

While there is much research to suggest benefits of pair programming (McDowell *et al*, 2003; 2006; Preston, 2005; Shalleh *et al*, 2011; Williams and Upchurch, 2001) the current study is focused on using pair programming for laboratory courses in e-learning. This paper reports on a study using the pair programming for the programming laboratory courses in e-learning teaching-learning process in four laboratory courses offered at UG and PG level. This approach has resulted in benefits such as enhancement of problem solving skills, efficiency, quality, trust, and teamwork skills. We have also observed that paired laboratory experience is especially advantageous to e-learners. A hidden advantage that was evident from the learners' responses was that learners were motivated to work collaboratively even for other tasks. Firstly, learners like this approach since they regard it as useful, facilitating the learning process, enabling to attain the learning goal and good learning experience. Moreover, the results of this study indicate that the use of pair programming in e-learning has important effects on the learners' academic outcomes and also the dropout rate is also reduced. The learners were motivated and involved in laboratory courses that created a confident in them. The learners obtained better final grades. Therefore, the results hereby presented suggest that this system can support effective learning strategies for laboratory courses in e-learning. The research showed several benefits of using pair programming in laboratory courses in e-learning such as enhanced learning, greater confidence in work quality, higher problem solving skills, enhanced interaction skills, and improved team building skills. The result also shows that e-learners had a good learning experience. The study also indicated several areas for future research. Future studies can examine the effects of pair forming and automatic formation of pairs. Future studies can examine the use of pair programming in Non-Computer Science curriculum.

REFERENCES

Bevan J., Werner L and McDowell C. 2002. Guidelines for the use of pair programming in a freshman programming class. Conference on Software Engineering Education and Training (pp. 100-107). KY: IEEE Computer Society.

Braught G., Wahls T. and Eby L. M. 2011. The case for pair programming in the computer science classroom. ACM Transactions on Computing Education (TOCE). 11(1): 2.

Carver J.C., Henderson L., He L., Hodges J.E. and Reese D.S. 2007. Increased retention of early computer science and software engineering students using pair programming. Conference on Software Engineering Education and Training. 115-122.

Denner J., Werner L., Campe S. and Ortiz E. 2014. Pair Programming: Under What Conditions Is It Advantageous for Middle School Students? Journal of Research on Technology in Education. 46(3): 277-296.

Donohue T. L. and Wong E. H. 1997. Achievement motivation and college satisfaction in traditional and nontraditional students. Education. 118(2): 237-244.

F. Padberg and M. Müller. 2003. Analyzing the Cost and Benefit of Pair Programming. in International Software Metrics Symposium (METRICS) 2003, Sydney, Australia. pp. 166-177.

Furberg A., Kluge A. and Ludvigsen S. 2013. Student sense making with science diagrams in a computer-based setting. International Journal of Computer-Supported Collaborative Learning. 8(1): 41-64.

H. Erdogmus and L. Williams. 2003. The Economics of Software Development by Pair Programmers. The Engineering Economist. 48(4): 283-319.

J. Vanhanen and C. Lassenius. 2007. Perceived Effects of Pair Programming in an Industrial Context. in 33rd EUROMICRO Conference on Software Engineering and Advanced Applications. Lubeck 2007. pp. 211-218.

Jason A. 2004. Technical and human perspectives on pair programming. ACM SIGSOFT Software Engineering Notes. 25(5): 1-14.

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

Levy Y. 2007. Comparing dropouts and persistence in e-learning courses. Computers and Education. 48(2): 185-204.

Maguire P. and Maguire R. 2013. Can Clickers Enhance Team Based Learning? Findings from a Computer Science Module. AISHE-J: The All Ireland Journal of Teaching and Learning in Higher Education. 5(3).

McDowell C., Hanks B., Werner L. 2003. Experimenting with pair programming in the classroom. SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE '03). pp.60-64.

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

N. Nagappan, L. Williams, M. Ferzli, K. Yang, E. Wiebe, C. Miller and S. Balik. 2003. Improving the CS1 Experience with Pair Programming. in ACM Special Interest Group Computer Science Education (SIGCSE) 2003, Reno. pp. 359-362.

Oncu S. and Cakir H. 2011. Research in online learning environments: priorities and methodologies. Computers and Education. 57(1): 1098-1108.

Preston D. 2005. Pair programming as a model of collaborative learning: A review of the research. Consortium for Computing Sciences in Colleges. 39-45.

Rosenberg M. J. 2001. E-learning: strategies for delivering knowledge in the digital age. McGraw Hill, New York.

Salleh N., Mendes E. and Grundy J. 2011. Empirical studies of pair programming for CS/SE teaching in higher education: A systematic literature review. Software Engineering, IEEE Transactions on. 37(4): 509-525.

Straub D. W. 1989. Validating instruments in MIS research. MIS Quarterly. 13(2): 147-169.

T. Dybå, E. Arisholm, D. Sjøberg, J. Hannay, and F. Shull. 2007. Are Two Heads Better Than One? On the Effectiveness of Pair Programming. IEEE Software. 24(6): 12-15, November/December.

Tsai W. T., Li W., Elston J. and Chen Y. 2011. Collaborative learning using wiki web sites for computer science undergraduate education: A case study. Education, IEEE Transactions on. 54(1): 114-124.

Van Der Vyver G. and Lane M. 2003. Using a Team-based Approach in an IS Course: An Empirical Study. Journal of Information Technology Education. 2, 393-406.

Williams L. and Upchurch R. 2001. In support of student pair programming, SIGCSE Conference on Computer Science Education. pp. 327-331.