Hindawi
Scientific Programming
Volume 2018, Article ID 6523538, 8 pages
https://doi.org/10.1155/2018/6523538

Research Article

Examining Student Performance and Attitudes on Distributed Pair Programming

Maya Satratzemi,[1] **Stelios Xinogalos**,[1] **Despina Tsompanoudi**,[1] **and Leonidas Karamitopoulos**[2]

[1]*Department of Applied Informatics, School of Information Sciences, University of Macedonia, Thessaloniki 54636, Greece*
[2]*Department of Information Technology, Alexander TEI of Thessaloniki, 57400 Sindos, Greece*

Correspondence should be addressed to Maya Satratzemi; maya@uom.edu.gr

Received 1 June 2018; Revised 4 September 2018; Accepted 2 October 2018; Published 24 October 2018

Academic Editor: Emiliano Tramontana

Pair programming (PP) has become popular in the research and software industry as well as being studied for a number of years in computer science courses with positive findings on student performance and attitudes. Advantages of PP reported in the literature are satisfaction, design quality, code productivity, team building, and communication. More recently, distributed pair programming (DPP), which enables two programmers to work remotely, has also attracted the interest of researchers and instructors. The difference between DPP and PP is that the former allows geographically distributed teams to collaborate and share program code. Such collaboration is, thus, only feasible if an underlying infrastructure supports all necessary interactions. The integrated development environments (IDEs) for DPP should cover the basic requirements for remote software development as well as address common PP problems, such as unequal contributions from each member of a pair, feedback during DPP sessions, and communication problems. This paper presents the findings of a study on student performance and attitudes towards DPP in an object-oriented programming (OOP) course. The factors examined were student performance, in terms of assignment grade, exam grade and implementation time in relation to students' programming experience, and confidence, as well as student attitudes towards DPP, i.e., the feelgood factor, working alone or with a partner, and the perception of their partner's technical competence. The results suggest that a students' performance is associated with their programming experience and confidence in programming but not with how comfortable they feel during DPP sessions. Students evaluate the DPP sessions positively regardless of their confidence on programming or their perception of their partners' technical competence. Students who consider themselves to have about the same programming competence as their partners tend to be more satisfied with DPP sessions. Overall, students prefer working with a partner regardless of their confidence on programming.

1. Introduction

Pair programming (PP) has become popular in the research and software industry as well as being studied for a number of years in computer science courses with positive findings on student performance and attitudes [1–6]. The literature reveals that the collaborative nature of pair programming helps students to increase confidence and improve their grades on programming assignments. Other studies indicate that PP leads to higher program quality, continuous knowledge transfer, and greater student enjoyment [7]. More recently, distributed pair programming (DPP) has made it possible for programmers to develop software with a partner from anywhere and at anytime. As an alternative to pair programming, DPP is more demanding because each member of the pair is not colocated while their common work is dependent on the features of the integrated development environment (IDE), as well as an infrastructure that has to be set up and configured by the students themselves. Although DPP can be realized with a general screen sharing application, when it comes to education, a DPP system is usually used to support students. The IDEs for DPP should cover the basic requirements for remote software development, such as a shared editor, supporting the roles of the driver and navigator, and a communication tool, as

well as should address common PP problems, such as unequal contributions from each member of a pair, feedback during DPP sessions, and communication problems. Most of the DPP IDEs were built as Eclipse plugins.

Performance is one of the most investigated factors regarding the effectiveness of PP, indicating that it has a positive effect on students' grades [1, 2]. Implementation time is also a common measure used in PP studies to evaluate its effectiveness [8]. Most of the studies reported that students working on PP require less time to complete assignments compared with students on solo programming. Another well-studied factor is pair compatibility [5]. Research suggests that pairing students with similar programming skill levels or programming experience has positive results on motivation and participation [5, 7]. Similarly, it has been shown that a pair's performance is correlated with how comfortable students feel during a PP session, the so-called "feelgood" factor as coined by Muller and Padberg in their study [8]. Finally, a factor used to evaluate pair programming satisfaction is the students' self-rated confidence, which, however, has presented mixed results [3, 6, 9, 10].

In order to draw safer conclusions on distributed pair programming, we developed an educational DPP system, called SCEPPSys ("Scripted Collaboration in an Educational Pair Programming System"). SCEPPSys [11] is an Eclipse plugin that has some unique features in comparison to other plugins such as Sangam, RIPPLE, XPairtise, and Saros. Specifically, SCEPPSys saves and analyzes students' interactions, helps educators in organizing and monitoring DPP classes, and supports the creation of programming assignments that are comprised of small and manageable tasks associated with specific didactical goals (OOP concepts). The building capabilities of SCEPPSys for assessment and evaluation provide data to form the basis for further research, since extensive studies of DPP are still lacking. For example, the statistics reported by SCEPPSys for each student (e.g., total time spent on solving a problem, implementation time for each student, and each student's contribution to the project) can be used to research the following [12]: students' progress in programming; any undesirable behaviour (e.g., plagiarism); and problems in collaboration between the members of a pair.

We have been conducting research in real-world situations for the last 5 years, studying the different aspects of DPP effects on the learning of programming. In a study [13] carried out during the academic year 2013-14, we investigated how scripted roles affected students' engagement and knowledge building in the context of DPP. The effects of two different approaches on performance and distribution of activities between student pairs were examined. In the first approach, pair members switched roles after each task, while in the second, the role assignment of the driver or navigator depended on task type and students' personal skills. The findings showed both approaches to be equally effective. In another study [14] during the academic year 2015-16, we investigated whether prior programming skills assessed at the level of the student, the partner, and the pair as a whole, as well as pair compatibility was related to student performance in an OOP course supported by DPP assignments. In this study, students chose their partner, whereas the task distribution policy was that of rotating roles, meaning that each member switched roles after each task. The findings showed that the actual skill of student, partner, and pair affected each student's performance, whereas there was no association between pair compatibility and the student's own performance.

In the present study, we examined student performance and attitudes towards DPP in an object-oriented programming (OOP) course based on Java over one semester in the academic year 2016-17. Performance was studied based on the pair's mean grade in the assignments, each student's grade in the final exam, and implementation time. Student's attitudes towards DPP were studied based on the reported feelgood factor, the preference of working alone or in pairs, and the student's perception of their partner's programming competence. Correlations between the student's performance, confidence in programming, programming experience, the "feelgood" factor, and preference in working alone or in pairs were examined. Although all these factors have been studied regarding the effectiveness of PP, they have not been examined in the context of DPP. Most of the studies on DPP have focused on either the comparison between solo programming and DPP students' groups or on studying student interactions emphasizing users' contributions, coordination, and communications [15–17].

The remainder of the paper is organized as follows: Section 2 gives the related studies on PP and DPP; Section 3 describes the study methodology; and Section 4 presents the results and discussion. Finally in Section 5, conclusions are drawn.

2. Related Work

Salleh [18] performed a systematic literature review (SLR) of empirical studies that investigated factors and studies that measured the effectiveness of PP for CS/SE students. The results showed that the most significant factor in PP effectiveness was student skill level, while the most common measure used to gauge PP effectiveness was time spent on programming. In addition, there was overall higher student satisfaction when using PP than when working solo. Their meta-analyses showed that PP was effective in improving students' grades on assignments. Finally, in the studies that used quality as a measure of effectiveness, academic performance and expert opinion were the quality measures mostly applied.

The motivation for the SLR conducted by da Silva Estácio and Prikladnicki [19] was the lack of studies on the use of DPP in industry. They reported that the majority of the studies concerned PP, and only 22 papers were related to DPP, most of which focused on tool proposals, while only a few described case studies on the adoption of DPP in industry. They gathered data from a field study concerning variables (code quality, team productivity and communication, and difference of knowledge between the pairs), DPP aspects (company guidelines for using DPP, infrastructure and methods for DPP, development tool for DPP, facilitator

to support DPP, and experience between pairs of DPP), benefits (execution time and motivation), challenges of using DPP (collaboration and communication challenges), and opinion (suggestions). Based on the literature review and the field study, they concluded by suggesting twelve practices that could help professionals in the use of DPP.

Umapathy and Ritzhaupt [20] conducted a meta-analysis on the effects of PP on educational outcomes. The results showed that pair programming can have a positive impact on students' programming assignment grades, exam scores, and also persistence in computer programming courses. However, in order to achieve these positive results, PP has to be implemented properly by both students and educators. Specifically, the following conditions must be met: regarding students, they should be supported in understanding the PP practice; they must alternate the roles of the driver and the navigator while working together on programming activities at the same time; and equal learning experiences must be ensured by having the driver and navigator alternate roles after a given duration [21]. Regarding educators, they should not just simply pair off students to carry out assignments, but they must ensure that PP is appropriately implemented.

Muller and Padberg [8] conducted two controlled experiments with 38 subjects on PP. They first studied the correlation between a pair's feelgood factor and the pair's implementation time and programming experience. In the second phase, rather than looking at the pairs, they focused on the individual's programming experience and feelgood factor. The findings showed that a pair's implementation time was uncorrelated to the pair's programming experience, but there was a significant correlation with how comfortable the developers felt with PP during the session (the "feelgood" factor). They did not find any significant correlation between the individual's programming experience and the pair's implementation time, nor the pair's implementation time and the individual's feelgood factor. The only statistical significant relationship they found was between implementation time and the member of each pair who felt less comfortable with pair programming. In their study, programming experience was measured in a subjective way using the number of years and number of lines of code written, which was data provided by the students in a pretest questionnaire. In their study, Muller and Padberg [8] used implementation time, which reflected the elapsed time that it took a pair to finish the assignment at the prescribed quality level, as a measure for the performance of a pair.

Thomas et al. [10] examined how self-confidence is reflected in students' reactions to the PP technique for developing software. Students who had programming experience before university were given a survey that placed them on a scale they called CodeWarrior to Code-a-phobe. They then placed students in 'opposite' and 'similar' pairs for a PP exercise and surveyed their reactions. The evidence indicated that students who had considerable self-confidence did not enjoy the experience of PP as much as the other students and the former produced their best work when placed in pairs with students of similar self-confidence levels.

Williams et al. [5] conducted a two-phased study on PP from 2002 to 2005 to determine if teachers can proactively form compatible pairs based on any of the following factors: personality type, learning style, skill level, programming self-esteem, work ethic, or time-management preference. They found (a) students prefer to pair with someone they perceive to be of similar technical competence, (b) pairing sensors and intuitors together yields very compatible pairs, and (c) pairing students with strongly dissimilar work ethics will more likely yield incompatible pairs.

Canfora et al. [22] conducted two experiments to study the impact of DPP regarding two productivity metrics: time and code quality. The results indicated that each member of the pair tended to work alone. The four factors reported to explain the results were failure to establish a working protocol, conflicting ideas between pair members, problems with the chat software, and different levels of experience between the pair members. In their second experiment, the two factors reported for successful DPP were appropriate communication between pair members and collaboration support.

In his paper, Hanks [17] discussed the development and empirical evaluation of a DPP tool, the important feature of which was the presence of a second cursor that supported gesturing. Students who used the tool in their introductory programming course performed as well as collocated students on their programming assignments and final exam. These students also spent less time working by themselves. They also felt that the gesturing feature was useful and used it regularly.

Zacharis [23] conducted a study investigating the effectiveness of virtual pair programming (VPP) on student performance and satisfaction in an introductory Java course. The two groups consisted of VPP students and solo students, and the two factors examined were code productivity and software quality. In addition, a comparison was made of the midterm and final examination scores between VPP and solo students. Finally, a survey of students' perceptions of VPP was administered. The results suggested that VPP is an effective pedagogical tool for flexible collaboration and an acceptable alternative to the individual/solo programming experience, regarding productivity, code quality, academic performance, and student satisfaction.

3. Methodology of the Study

3.1. Course Outline. The study was conducted in the 3rd semester of an undergraduate course on OOP in the academic year 2016-17. The course uses the Java programming language and runs over thirteen weeks with a 3-hour lab per week. Eighty-eight (88) students chose their partner, forming 44 pairs. Students were assigned five Java projects as homework to be solved in their pairs using the DPP system, SCEPPSys. The grade of each DPP assignment was the same for both students in the pair. Students had to take a final exam in the lab, where they were required to complete a program individually.

We developed an educational DPP system, calling it SCEPPSys [11], which supports the basic requirements of

remote collaboration. Pair programmers can edit the source code in real time using a shared editor which supports the driver's and the navigator's roles. The driver edits the source code, while the navigator monitors the changes made to the source code and comments on them. Programmers are able to discuss and coordinate their actions using an embedded text-based communication channel. Remote code highlighting (a basic gesturing feature) enables the navigator to point out code parts in order to indicate potential problems. The remaining features, the so-called "awareness indicators", aim to provide pair programmers with information about the user's status and performed actions within the workspace (such as editing and saving). Distributed pair programming is guided by collaboration scripts, which consist of a number of components and mechanisms [24]. In the administration environment of SCEPPSys, these scripts have been adapted to meet DPP requirements. Specifically the script authoring procedure includes participants' settings; pair formation; programming tasks; and turn-taking policies. Programming tasks are the subtasks of a major programming assignment. A hint provides support to students in completing the subtask with which it is associated. The turn-taking policies specify the distribution of the driver/navigator roles between the programmers. A significant feature of SCEPPSys is its ability to distribute the driver's and navigator's roles to the pair members. In previous works [12, 14], alternative turn-taking policies supported by SCEPPSys were investigated. In the present study, the chosen task distribution policy was free collaboration. Free collaboration policy allows students to distribute tasks based on their own decisions. Although this type of task distribution policy is more acceptable to students, it could lead to unequal participation. Thus, in order to avoid this, throughout the session, SCEPPSys displays metrics, such as the pair's total time and individual participation rates to help students balance their participation.

3.2. Research Objectives. From research on pair programming in the literature review, the factors most investigated were students' performance in terms of achieved grades, implementation time for assignments, quality of code, and factors concerning pair compatibility (personality type and programming competence). In contrast, research on distributed pair programming focused mainly on the features of the relevant tools, the performance of students, and the quality of their code.

In this study, we investigated the correlations between students' performance and factors related to their prior programming skills and confidence in programming, as well as factors related to students' attitudes towards DPP, such as the feelgood factor, working alone or with a partner, and the student's perception of their partner's technical competence. It should be mentioned that, to the best of our knowledge, the feelgood factor and students' perception of their partners' technical competence have been studied only in the context of pair programming.

The following are the meanings and measures of the factors in our study.

Student performance is measured by the following:

(a) The grades achieved in the final exam of the OOP course (exam grade)

(b) The mean value of the 5 assignment grades (mean assignment grade)

(c) The implementation time, which is the total time spent by each member of the pair to complete all the submitted assignments

Regarding student grades, it must be noted that our decision to examine the two measures separately rather than combining them, was made on the basis that they represent two distinct phases in the student's progress in the course: the collaborative solution of DPP assignments covering a specific part of the syllabus and the final outcome for each student based on their individual examination covering the whole syllabus. We also considered implementation time as a measure of a student's performance during DPP sessions. The literature shows that this has been used as a measure for student performance in a number of studies in a variety of ways seen indicatively [8, 22].

Programming experience is measured as the mean value of the student's grades in two introductory courses "Procedural Programming (C programming language)" and "Algorithms in C" of the previous academic year.

Confidence in programming is measured by the students themselves prior to the beginning of the OOP course. We asked students to place themselves on a scale of 1 to 9 based on their self-confidence of their ability to program, and inspired by the study of Thomas et al. [10], we classified them into three groups: warriors: 7–9; middle: 4–6; and phobes: 1–3.

The feelgood factor shows how comfortable each member of a pair felt during the DPP sessions. It was introduced and studied by Muller and Padberg [8] in the context of PP. We asked students to evaluate their experience of DPP on a five-point scale (1: very bad, 2: bad, 3: neutral, 4: good, and 5: very good).

Perception of partner's competence in programming is the perception each member of the pair has about their partner's technical competence in regards to their own competence on a three-point scale (better, about the same, and weaker).

The following research questions were investigated:

Research questions examining student performance

Performance has been examined in relation to other factors in most of the studies on PP and DPP as it is a crucial factor, in a way, related to students' learning outcomes. Therefore, in RQ1.1 to RQ1.5, we look to see if there are correlations between a student's performance and their prior programming skill (*programming experience)* and attitudes on programming (*confidence in programming*) prior to the OOP course based on the DPP assignments.

RQ1.1: Does a student's mean assignment grade or exam grade correlate with their confidence in programming?

RQ1.2: Does a student's implementation time correlate with their confidence in programming?

RQ1.3: Does a student's mean assignment grade or exam grade correlate with their programming experience?

RQ1.4: Does a student's implementation time correlate with their programming experience?

RQ1.5: Does a student's mean assignment grade or exam grade correlate with their implementation time?

Research questions examining student attitudes towards DPP

RQ2.1 to RQ2.4 aim to examine if the feelgood factor on DPP is related to other factors such as programming experience and attitudes towards programming, as well as factors more linked/relevant to DPP such as student performance in the current course on OOP (assignment grade, exam grade, and implementation time).

RQ2.1: Does a student's feelgood factor correlate to their confidence in programming?

RQ2.2: Does a student's feelgood factor correlate to their programming experience?

RQ2.3: Does a student's feelgood factor correlate to their mean assignment grade or exam grade?

RQ2.4: Does a student's feelgood factor correlate to their implementation time?

RQ3.1, RQ3.2 and RQ4.1, RQ4.2 examine any patterns that may exist between groups of students with specific attitudes. The *feelgood factor groups*, based on students' estimation of the DPP sessions, are bad/very bad, neutral, and good/very good. The *confidence in programming groups*, based on students' rating of themselves, is Warriors, Middle, and Phobes. In addition, there are students who prefer to work alone or those who prefer to work with a partner, and finally, the partner's competence in programming groups, which are better, about the same, or weaker. The investigation of these last four RQs gives us the chance to record the distribution with respect to groups of students with different attitudes concerning the DPP:

RQ3.1: Is there a relationship between a student's feelgood factor group and their confidence in programming group?

RQ3.2: Is there a relationship between a student's preference in working alone or with a partner and their confidence in programming group?

RQ4.1: Is there a relationship between a student's confidence in programming group and their perception of their partner's technical competence?

RQ4.2: Is there a relationship between a student's feelgood factor group and their perception of their partner's technical competence?

3.3. Instruments and Data Analysis. The data analysed in this paper were gathered from the following:

(a) The grade achieved in the final exam and the mean grade from the assignments in the OOP course.

(b) The implementation time of each assignment and for each member of the pair recorded by the DPP system SCEPPSys.

(c) A presemester questionnaire distributed to students as a Google form that recorded the grades obtained in the introductory courses "Procedural Programming (C programming language)" and "Algorithms in C" of the previous year, as well as their confidence in their ability to program. Since both these courses are introductory, their syllabi and assignments are quite typical of universities around the world. Students were asked to answer the following question prior to the OOP course:

Q1. Place yourself on a 1 to 9 scale with the following endpoints:

1 = I do not like programming, and I think I am not good at it. I can write simple programs, but have trouble writing new programs for solving new problems.
9 = I have no problems at all completing programming tasks to date, in fact they were not challenging enough. I love to program and anticipate no difficulty with this course.

In their studies on Pair Programming, Thomas et al. [10] (pp. 364) and Williams et al. [5] (pp. 417) posed Q1 in order to measure students' self-confidence on their ability to program. Consequently, although one might argue that Q1 asks how challenging the assignments were, and how much students like programming, it should be noted that given the specific context, the question deals mainly with students' confidence in their ability to program.

(d) A questionnaire was distributed to students as a Google form on completion of the DPP assignments at the end of the semester. To investigate students' attitudes on DPP, the following questions were included in the questionnaire:

Q2. How would you evaluate the distributed, collaborative solving of assignments as an overall experience (1 = Very bad, 2 = Bad, 3 = Neutral, 4 = Good, 5 = Very good)?
Q3. Based on your experience in DPP would you prefer to work alone or with a partner in programming assignments?
Q4. Assess the technical competency of your partner relative to yourself [Better, About the same, Weaker].

The questionnaire was adapted from similar researches conducted in the context of PP. Muller and Padberg [8] used Q2 to measure how comfortable students felt during the pair programming session. As mentioned in previous sections, they called this metric the individual "feelgood" factor of a developer. Williams et al. [5] asked students to evaluate their partners' technical competence through question Q4.

Out of the 88 students, the statistical analysis was compiled on 78, as these students answered both questionnaires.

Statistical analysis was performed by using IBM SPSS Statistics (version 19.0.0). Pearson's product-moment correlation coefficient (r) was computed to assess the relationship between quantitative variables. Moreover, the chi-square test of independence was utilized for RQ3 and RQ4.

4. Results and Discussion

In this section, the results of the study are analysed and discussed.

The results of RQ1.1 to RQ1.5 are given in Table 1.

For the variations of RQ2, the results (Table 2) showed that the feelgood factor did not correlate with any of the following factors: confidence in programming, programming experience, mean assignment grade, exam grade, or implementation time.

In the attempt to spot any patterns that may have formed between the groups of students with specific attitudes (RQ3), we combined students' responses in Q2 with Q1, and Q3 with Q1.

The chi-square test of independence was conducted to investigate whether there was a relationship between the groups for feelgood factor and confidence in programming. The findings showed that there was no relationship between the two groups ($X^2 = 3.300$, df = 4, $p = 0.039$).

From the data in Table 3, it can be seen that a total of 76% of students in the study found distributed pair programming as good/very good.

The chi-square test of independence was conducted to investigate whether there was a relationship between the student's preference in working alone or with a partner and their confidence in programming group. A statistically significant relationship was observed between the two ($X^2 = 6.500$, df = 2, $p = 0.039$).

Table 4 clearly shows that only a small percentage of the warriors preferred working alone and all others preferred working with a partner.

In order to further investigate students' attitudes towards DPP, we considered students' responses to Q4 (partner's assessment) in relation to Q1 (confidence). The chi-square test of independence showed that there was no relationship between the two ($X^2 = 8.769$, df = 4, $p = 0.067$).

Even though the statistical result is not significant, the data presented in Table 5 give us useful information about the synthesis of the pairs. From Table 5, it can be seen that the highest total percentage of students in the study (71.8%) stated that they perceived their partner as having about the same technical competence as themselves. This applies to all three categories of phobes, middle, and warriors with 62.5%, 71.0%, and 74.4%, respectively. None of the phobes assessed their partners were weaker than themselves, whereas only 5.1% of the warriors stated that they perceived their partner as better than themselves.

In order to further investigate students' attitudes towards DPP, we considered their responses to Q4 (partner's assessment) in relation to Q2 (feelgood factor groups). The chi-square test of independence showed that there is no relationship between a student's feelgood factor group ($X^2 = 6.261$, df = 4, $p = 0.180$) and their perception of their partner's technical competence.

The data presented in Table 6 show that 81% of students who believed that they have about the same programming competence as their partners found the DPP sessions as being good/very good, whereas only 5% said that they were bad/very bad.

Table 1: Results summary of RQ1.1 to RQ1.5.

RQ: result	r	p
1.1: Mean assignment grade correlates positively with the student's confidence in programming	0.233	0.040
1.1: Exam grade correlates positively with the student's confidence in programming	0.591	$p < 0.001$
1.2: Implementation time correlates negatively with the student's confidence in programming	−0.342	0.002
1.3: Mean assignment grade correlates positively with the student's programming experience	0.334	0.008
1.3: Exam grade correlates positively with the student's programming experience	0.619	$p < 0.001$
1.4: Implementation time correlates negatively with the student's programming experience	−0.245	0.05
1.5: Mean assignment grade does not correlate with the student's implementation time	0.136	0.208
1.5: Exam grade does not correlate with the student's implementation time	−0.065	0.582

Table 2: Result summary of the variations of RQ2.

RQ: result	r	p
2.1: Feelgood factor does not correlate with the student's confidence in programming	−0.027	0.818
2.2: Feelgood factor does not correlate with the student's programming experience	−0.062	0.648
2.3: Feelgood factor does not correlate with the student's mean assignment grade	0.028	0.809
2.3: Feelgood factor does not correlate with the student's exam grade	−0.016	0.897
2.4: Feelgood factor does not correlate with the student's implementation time	−0.032	0.778

5. Conclusions

The aim of this study was to investigate the effectiveness of DPP in an object-oriented programming (OOP) course in the academic year 2016-17. The factors examined were student performance, in terms of assignment grades, exam grade and implementation time in relation to students' programming experience, and confidence, as well as student attitudes towards DPP, i.e., the feelgood factor, working alone or with a partner, and the perception of their partner's technical competence.

The results suggest that a student's performance is associated with their experience and confidence in programming rather than on how comfortable they felt during the DPP session. Even though some of the above results may appear to be rather obvious; nonetheless, there is still a serious lack of empirical data in the context of DPP. Muller and Padberg [8] in their study on PP found that individual performance does not correlate with the programming experience and that the feelgood factor of that pair member who felt less comfortable with pair programming correlates with the pair performance. The findings of Muller and Padberg could not be compared with ours even though they examined the same factors: performance, programming experience, and feelgood factor, as we did. The reason is that

Table 3: Students' distribution with respect to feelgood factor groups and confidence in programming groups.

Confidence in programming groups	Feelgood factor groups			Total
	Bad/Very Bad	Neutral	Good/Very Good	
Phobes	0 (0%)	3 (38%)	5 (63%)	8
Middle	3 (10%)	5 (16%)	23 (74%)	31
Warriors	2 (5%)	6 (15%)	31 (80%)	39
Total	5 (6%)	14 (18%)	59 (76%)	78

*Data are presented as frequencies and row percentages.

Table 4: Students' distribution with respect to confidence in programming groups and working alone or with a partner.

Confidence in programming groups	Working alone or with a partner		Total
	Alone	With a partner	
Phobes	0 (0%)	8 (100%)	8
Middle	0 (0%)	31 (100%)	31
Warriors	6 (15%)	33 (85%)	39
Total	6 (8%)	72 (92%)	78

*Data are presented as frequencies and row percentages.

Table 5: Students' distribution with respect to their confidence in programming group and their perception of their partners' technical competence.

Confidence in programming groups	Perception of the partner's technical competence			Total
	Weaker	About the same	Better	
Phobes	0 (0.0%)	5 (63%)	3 (38%)	8
Middle	3 (10%)	22 (71%)	6 (19%)	31
Warriors	8 (25%)	29 (74%)	2 (5%)	39
Total	11 (14%)	56 (72%)	11 (14%)	78

*Data are presented as frequencies and row percentages.

Table 6: Students' distribution with respect to their feelgood factor and perception of their partner's technical competence.

Perception of their partner's technical competence	Feelgood factor			Total
	Bad/Very Bad	Neutral	Good/Very Good	
Weaker	0 (0%)	4 (36%)	7 (64%)	11
About the same	3 (5%)	8 (14%)	45 (81%)	56
Better	2 (18%)	2 (18%)	7 (64%)	11
Total	5 (6%)	14 (18%)	59 (76%)	78

*Data are presented as frequencies and row percentages.

they used different metrics for performance and programming experience as described in the related work section, and they examined slightly different RQs.

A strong majority of students had a positive attitude, regardless of the feelgood factor or their perception of their partner's technical competence. The study findings clearly indicate that the vast majority of students preferred to work with a partner rather than alone. Thomas et al. [10] reported similar findings concerning warriors and their preference for pair programming. However, Hanks [9] reported partly contradictory results in that although the highly confident students liked pairing the most and those with low confidence liked it the least. It seems plausible that our finding where most students of all confidence levels (Phobes, Middle, and Warriors) preferred pairing could be a result of the structured pair programming scripting in the SCEPPSys tool that was not present in the work of Hanks. One of the main hypotheses behind the decision to incorporate the ability of structured DPP scripting in SCEPPSys was that it would provide guidance during problem solving and would further support weaker students [25].

The majority of students in our study in the confidence in programming groups perceived their partners' technical competence to be about the same as theirs. The findings showed that, as regards students' feelgood factor, those students who believed their partners had about the same programming competence as them tended to be more satisfied with the DPP sessions, keeping in mind that the pairs of students chose their partner themselves. This finding is supported by Jacobson and Schaefer [26] who in their study on PP noted that allowing students to freely form pairs themselves leads to a high degree of satisfaction.

It appears that our findings concerning student attitudes towards distributed pair programming are in accordance with most of the findings on similar studies on pair programming. This is an encouraging result as DPP is more

demanding than PP. In this study, SCEPPSys the educational system developed for a typical undergraduate OOP course promotes student collaboration and balanced participation in DPP. Clearly, the results of this study that used SCEPPSys and a specific set of assignments cannot be generalized for all DPP educational settings. Despite the limitations, this research adds to the body of studies on distributed pair programming since factors, such as feelgood factor, confidence in programming, and perception of the partner's programming competence, have not been examined in the context of DPP.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] C. McDowell, L. Werner, H. Bullock, and J. Fernald, "The effects of pair-programming on performance in an introductory programming course," *ACM SIGCSE Bulletin*, vol. 34, no. 1, pp. 38–42, 2002.

[2] C. McDowell, L. Werner, H. E. Bullock, and J. Fernald, "The impact of pair programming on student performance, perception and persistence," in *Proceedings of the International Conference on Software Engineering*, pp. 602–607, Portland, OR, USA, May 2003.

[3] C. McDowell, B. Hanks, and L. Werner, "Experimenting with pair programming in the classroom," *ACM SIGCSE Bulletin*, vol. 35, no. 3, pp. 60–64, 2003.

[4] C. McDowell, L. Werner, H. Bullock, and J. Fernald, "Pair programming improves student retention, confidence, and program quality," *Communications of the ACM*, vol. 49, no. 8, pp. 90–95, 2006.

[5] L. Williams, L. Layman, J. Osborne, and N. Katira, "Examining the compatibility of student pair programmers,"*in Proceedings of the conference on AGILE 2006 (AGILE '06)*, IEEE Computer Society, pp. 411–420, Minneapolis, MN,USA, July 2006.

[6] L. Williams, C. McDowell, N. Nagappan, J. Fernald, and L. Werner, "Building pair programming knowledge through a family of experiments," in *Proceedings of 2003 International Symposium on Empirical Software Engineering*, pp. 143–152, IEEE Computer Society, Washington, D.C., USA, September-October 2003.

[7] S. Faja, "Pair programming as a team based learning activity: a review of research," *Issues in Information Systems*, vol. 12, no. 2, pp. 207–216, 2011.

[8] M. M. Muller and F. Padberg, "An empirical study about the feelgood factor in pair programming," in *Proceedings of International Software Metrics Symposium*, pp. 151–158, Chicago, IL, USA, September 2004.

[9] B. Hanks, "Student attitudes toward pair programming," *ACM SIGCSE Bulletin*, vol. 38, no. 3, pp. 113–117, 2006.

[10] L. Thomas, M. Ratcliffe, and A. Robertson, "Code warriors and code-a-phobes: a study in attitude and pair programming," *ACM SIGCSE Bulletin*, vol. 35, no. 1, pp. 363–367, 2003.

[11] D. Tsompanoudi, M. Satratzemi, and S. Xinogalos, "Distributed pair programming using collaboration scripts: an educational system and initial results," *Informatics in Education*, vol. 14, no. 2, pp. 291–314, 2015.

[12] S. Xinogalos, M. Satratzemi, D. Tsompanoudi, and A. Chatzigeorgiou, "Monitoring an OOP course through assignments in a distributed pair programming system," in Z. Budimac, Z.Horvath, and T. Kozsik, eds., in Proceedings of *the SQAMIA 2016: 5th Workshop of Software Quality, Analysis, Monitoring, Improvement, and Applications*, vol. 1677, pp. 97–104, Budapest, Hungary, August 2016, http://ceur-ws.org/Vol-1677/.

[13] D. Tsompanoudi, M. Satratzemi, and S. Xinogalos, "Evaluating the effects of scripted distributed pair programming on students' performance and participation," *IEEE Transactions on Education*, vol. 59, no. 1, pp. 24–31, 2016.

[14] S. Xinogalos, M. Satratzemi, A. Chatzigeorgiou, and D. Tsompanoudi, "Factors affecting students' performance in distributed pair programming," *Journal of Educational Computing Research*, 2017.

[15] J. Schenk, L. Prechelt, and S. Salinger, "Distributed-Pair Programming can work well and is not just Distributed Pair-Programming,"in Companion Proceedings of 36th InternationalConference on Software Engineering. ACM, pp. 74–83, Hyderabad, India, May 2014.

[16] T. Schümmer and S. Lukosch, "Understanding tools and practices for distributed pair programming," *Journal of Universal Computer Science*, vol. 15, no. 16, pp. 3101–3125, 2010.

[17] B. Hanks, "Empirical evaluation of distributed pair programming," *International Journal of Human-Computer Studies*, vol. 66, no. 7, pp. 530–544, 2008.

[18] N. Salleh, E. Mendes, and J. Grundy, "Empirical studies of pair programming for CS/SE teaching in higher education: a systematic literature review," *IEEE Transactions on Software Engineering*, vol. 37, no. 4, pp. 509–525, 2011.

[19] B. J. da Silva Estácio and R. Prikladnicki, "Distributed pair programming: a systematic literature review," *Information and Software Technology*, vol. 63, pp. 1–10, 2015.

[20] K. Umapathy and A. D. Ritzhaupt, "A meta-analysis of pair-programming in computer programming courses: implications for educational practice," *ACM Transactions on Computing Education*, vol. 17, no. 4, pp. 1–13, 2017.

[21] D. W. Govender and T. P. Govender, "Using a collaborative learning technique as a pedagogic intervention for the effective teaching and learning of a programming course," *Mediterranean Journal of Social Sciences*, vol. 5, no. 20, pp. 1077–1086, 2014.

[22] G. Canfora, A. Cimitile, G. A. Di Lucca, and C. A. Visaggio, "How distribution affects the success of pair programming," *International Journal of Software Engineering and Knowledge Engineering*, vol. 16, no. 2, pp. 293–313, 2006.

[23] N. Z. Zacharis, "Measuring the effects of virtual pair programming in an introductory programming java course," *IEEE Transactions on Education*, vol. 54, no. 1, pp. 168–170, 2011.

[24] L. Kobbe, A. Weinberger, P. Dillenbourg et al., "Specifying computer-supported collaboration scripts," *International Journal of Computer-Supported Collaborative Learning*, vol. 2, no. 2-3, pp. 211–224, 2007.

[25] P. Dillenbourg and F. Fischer, "Computer-supported collaborative learning: the basics," *Zeitschrift für Berufs-und Wirtschaftspädagogik*, vol. 21, pp. 111–130, 2007.

[26] N. Jacobson and S. Schaefer, "Pair programming in CS1," *ACM SIGCSE Bulletin*, vol. 40, no. 2, pp. 93–96, 2008.