VOL. 3, NO. 4, April 2012 ISSN 2079-8407
Journal of Emerging Trends in Computing and Information Sciences

http://www.cisjournal.org

Effective Pair Programming Practice- An Experimental Study

Venkata Vinod Kumar Padmanabhuni, Hari Praveen Tadiparthi,

Muralidhar Yanamadala , Sagar Madina

Master of Science in Software Engineering, Blekinge Tekniska Hogskola, Sweden.

Email: (hata10, muya10, samc09) @student.bth.se

ABSTRACT

Pair programming is a practice, where two programmers work on a single task. Long research has been carried out in industries, and efficient results were achieved. Research is also being done in implementing the pair programming techniques in academics and good results are seen. In this paper, we suggested a new approach of pairing students from different engineering levels an experiment was conducted on a sample of students. In this research, both qualitative and mixed approaches are used.

Keywords: *pair-programming, PP, experiment.*

1. INTRODUCTION

Pair programming is an eXtreme Programming (XP) practice and an agile software development method [1]. Pair programming is a programming technique where two programmers work on same computer on a single task. One programmer is designated as the driver while the other is a navigator. This is different from collaborative programming, in a pair programming the driver works on the computer while the navigator checks for errors, thinks of alternatives and other resources [2]. The two programmers switch their responsibility periodically [3].

Many Empirical studies and researches show that pair programming is an efficient technique used in the development of software projects to improve the software quality. In software industry pairs produce code with low defect rates; development time is short and is cost efficient [4]. Research suggests that pair programming has increased the students' self confidence and interest in the field of Information Technology [5].

Pair programming helps in knowledge transfer among the student and emphasizes the benefits of the pair designing [6]. Pair programming creates an environment in encouraging the students to learn actively, leading students more confident and developing social interaction [7].

Research has been done in pair programming with students of different skills. The results showed that the best learning environment occurs when one person is slightly more or less skilled. Here the students participated are of same educational background and are undergraduate students. No research study was done by pairing persons with different engineering levels like undergraduate, post graduate and PhD.

2. BACKGROUND AND RELATED WORK

A lot of research on pair programming has been conducted in the field of software engineering, which showed better results when compared to individual programming in Industrial context. Tacit knowledge was shared among the employees, which helps them to increase their learning skills. Previous studies related to pair programming in the field of academics helped both the teaching faculty and students in obtaining good results.

2.1. Literature review

Pair Programming has a great impact in industries and students education. In Software Industry, it has increased the productivity and knowledge transfer between employees. The advantages of pair programming motivated industries and higher educational institution students in the fields of Computer Science (CS) and Software Engineering (SE) to implement this practice.

Pair programmers spent 60 more hours of time in programming to complete the task but after the adjustment period, i.e. pair jelling it was found that compared to solo students paired students completed their task 40% to 50% faster, and many of the students reported that they enjoyed the experience of using pair programming [8].

Data Analysis is done based on the research question, and it was proven that compatibility factors influence the students when they program in pairs. There are three studies who specifically addressed the compatibility of pair programmers [9][10][11]. The results are varied among the compatibility of different groups, students from different courses with different personalities and same self-esteem and same skill are paired [11].

Another three studies [12][13][14] also concentrated on the personality type, Myers-Briggs personality test (MBTT) was the most commonly used instrument to measure the personality type. NEO-PI is also used to measure the personality type [14]. It was discovered that personality of a single programmer does not affect the results of pair programmers but there is a difference when pairs of different personalities work together.

There are also articles that discuss about the compatible factors affect the pair programming [15], [16], [17], [18]. There was a conflict in the idea of pair programming, the two members have to divide the work and at the end

integrate their results but the concept is different where two programmers work on the same task, and their jobs are different.

The advantages of pair programming over individual programming are pairs produce code with fewer defects, produces good quality of code within half the time of an individual programmer, develop good knowledge, builds team work and finally the rotation of partners helps in knowing the overall system [20].

Pairing of students is a great challenge; only the factor of academics is considered by leaving gender and race. The factors should be considered properly. There are also many guidelines to be followed while pairing the students in order to achieve high effectiveness. The students should be supervised by staff or instructors in order to resolve the problem between the pairs. Partners of the students should also be rotated so they can develop communication skills and prevent them from intolerance of same partner [21].

Experience of pair programming shows good results for the Information Systems' students [19]. Industries programmers will come from different educational backgrounds, i.e. people not only come from computer science but also from economics, arts, etc [6]. In these aspect Authors E.Bellini *et al.* conducted an experiment by pairing students of Master of Technologies of Software and Master of Management and Technologies of Software which had shown better results of knowledge transfer between them [6]. Hence, we intend to do research in this field of pair programming by not only implementing on the students of same degree but also extending this concept to pairing students of different degree levels, which may show a great impact in the students' thesis or projects.

3. RESEARCH QUESTIONS

Based on the findings of our literature review and other related work we formulated two research questions for this research paper.

RQ1: How the effectiveness of pair programming increases by pairing students of different education levels?

RQ2: What are the benefits and challenges of pairing students?

4. RESEARCH METHODOLOGY

Research is the systematic way to collect data, analysis and find a solution to the problem [22]. We are following mixed method approach [28] based upon the research questions we have formulated. The RQ1 is answered by using interviews and questionnaire at end of experiment. The RQ2 is answered by analyzing the experiment results.

4.1. Qualitative Research

4.1.1. Context

To understand the benefits and challenges of pair programming we opted for a qualitative procedure interview. We performed this interview process with ten professors who are having interest and experience of research in pair programming. The professors are from Norway, Germany, Sweden and Austria. Among the selected sample professors are related to both academia and industry since they also work as consultants for software companies.

4.1.2. Data Collection

The data collection is done through interviews. We have communicated with professors through communication tools like Skype, Instant messenger and mails. We thought this Interview is useful for collecting the information related to pair programming. Such as benefits and challenges of pair programming. The interview Questionnaire is given in Appendix A.

4.1.3. Data Analysis

The data collected from professors is analyzed by tabulation of a questionnaire survey and narrative analysis. The list of benefits and challenges collected from literature review [5][7][19][25][26][2][30][31][32][33] [34] [35] and interviewers are shown in Table 1:

Table 1: Benefits of Pair Programming practices

Benefits	**No. of professors Supported**	**No. of Professors not supported**
Code Productivity increases	9	1
Knowledge transfer	6	4
Increase in communication skill	7	3
Problem solving skill	8	2
Quality increases	8	2
PP is enjoyable (work satisfaction)	10	0
Builds trust and increase team work	5	5
Increase in student Academic performance	6	4
New coding idioms	7	3
Challenges	**No. of professors Supported**	**No. of Professors not supported**
Deciding on pairs	6	4
Assigning the complexity tasks	7	3
Bonding between team members	5	5

Motivating the students continuously towards the goal	4	6

4.2. Quantitative method

For this quantitative method, an experiment was conducted among the students of different levels, and their results are compared. Claes Wholin defined experiment as "A formal, rigorous and a controlled investigation and the experimentation is a process of systematic, disciplined, quantifiable and controlled way of evaluating the human based activities" [39].

4.2.1. Purpose of the study

The research methods are selected based on the research questions. The quantitative method is used to show the effectiveness of pair programming over individual programming and pair programming among the students of same degree to the students of different educational levels.

4.2.2. Experiment Planning

The experiment was planned to be conducted in our home university Blekinge Teknik Hogskola, Sweden. A total of 45 students were taken from Computer Science and Information Technology branches. Among them 15 members from under graduation (UG), 15 members from post-graduation (PG) and other 15 are from PhD. For this experiment, a Java program is considered as it was a single subject of the course in the UG level that was known to the other two levels (PG and PhD).

The objects needed for conducting the experiment are a class room with good infrastructure, eight computers, and a tutor to guide them all. The interview is conducted after the completion of an experiment so printed material with questions is needed. The experiment is conducted on a Java program. The computers should be provided with Java IDE tools.

4.2.3. Hypothesis

There are two hypothesis null hypothesis and alternative hypothesis.

- Null Hypothesis: This defines that there is no effect in the measurement we have considered.

H0: Effectiveness of pair programming doesn't increase by pairing students from different level.

- Alternative Hypothesis: This converse to the null hypothesis and to prove that the observations considered are to be true.

H1: Effectiveness of pair programming increases by pairing students from different level.

4.2.4. Variable Selection

For conducting an experiment the variables are considered. There are two types of variables independent variables and dependent variables. The variable selection is as follows:

- **Independent Variable:** It includes the same or different personalities of students, no. of students, time, and infrastructure.
- **Dependent Variable:** It includes pair performance, pair participation, code design and communication among the students.

4.2.5. Experiment Design

In experiment design effectiveness was measured. Generally, the related concepts for measuring effectiveness are project duration, Effort and Quality [22]. Since the experiment is designed for a developing a program code the project duration will be the time taken for completing the task. The measure of effort is the person-hours required to complete the task. The quality depends on how good the code was developed and the measure is number of test cases passed. Two experiments were conducted on students'. First Experiment is concerned about the effectiveness of pair programming to individual programming, and second experiment is about the effectiveness of pair programming of students of same level to pairing the students of different level.

Firstly, process metrics like pair speed advantage and defect density are used for measuring the effectiveness of pair programming compared to solo programming [23]. First Experiment was conducted on graduate students. Their time of completions of individual and pair programming were recorded as shown in Table 2.

Table 2: Time duration of first experiment

Programming Type	**Time Duration (in person hours)**
Individual	2.58
Pair	2

It was noticed that pair programmers on average require 29 percent less time than individual programmers.

Pair speed advantage = 100 / (100-29) = 1.4

Hence pair speed advantage was accepted as it lies in the range between 1.3 to 1.8 [24][25][26]. According to W.Humphery good conventional software eliminates upto 70 percent defects and Defect Density was assumed to be 0.03 defects per line of code. Defect Density of the first experiment was shown in Table 3.

Defect Density = Lines of code / defects

Table 3: Defect Density measurements

Programming type	Lines of code	Defects	Defect Density
Individual	120	6	0.05 defect per LOC
Pair	90	3	0.03 defect per LOC

Finally, from the results of process metrics pair speed advantage and defect density in first experiment shows that pair programming is more effective than individual programming.

Secondly program duration, quality and effort are considered in second experiment to examine the pair programming effectiveness of students from same level to the students from different level. The students are formed pairs among them in all three levels and while pairing the students of same level they all are of same skill and it does not show much influence on the program effectiveness. The time duration taken by the students to complete the program, the no defects found which indicates the quality and no. of test cases the program passed are tabulated in Table 4. According to our requirements, the program developed by the students has to satisfy 15 test cases and basing on this programming effectiveness of students was represented using the graphs as shown in Figure 1.

Table 4: Measured values of students paired in same level

Pair level	Person hours	No. of defects	No. test cases passed
UG-UG	13	11	8
PG-PG	10.5	8	9
PhD-PhD	8.25	4	14

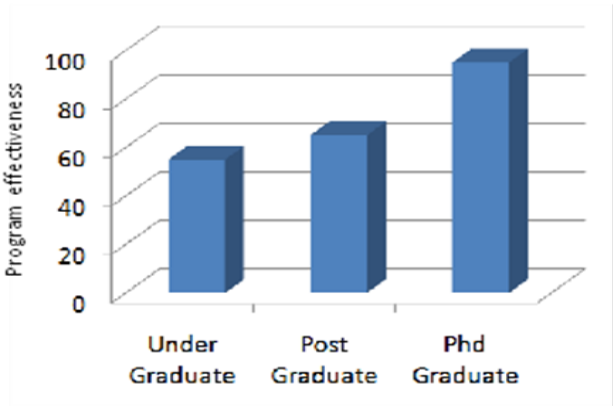

Figure 1:

Programming Effectiveness (based on test cases) = (no. test cases passed / expected test cases to be Satisfied) * 100

Hence programming effectiveness based on test cases for UG-UG is 55%, PG-PG 65% and PhD – PhD is 95%. Now the students of different level are paired like under graduate – post graduate (UG-PG), under graduate – PhD (UG-PhD), PhD – post graduate (PhD-PG). To this mixed personality pair a Java program is given, and their program efficiency is measured and are tabulated in Table 5. Similarly as above the program effectiveness of mixed personality pairs can be seen in Figure 2.

Table 5: Measured values of students paired in different level.

Pair level	Person hours	No. of defects	No. test cases passed
UG-PhD	9.5	7	12
PG-PhD	8	4	14
PG-UG	11	9	10

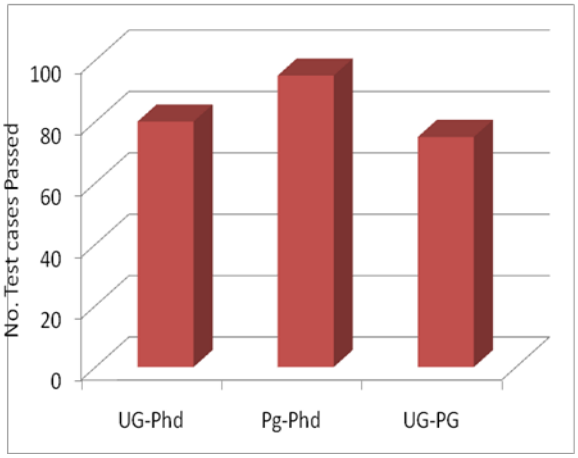

Figure 2: Programming effectiveness of mixed personalities.

The experiment results summarize that the effectiveness of paired PhD students, i.e. higher level was more but their ideas are more complex and go beyond our requirement scope while they are paired with the lower level student it shows the same effectiveness and also more benefits as their scope will be limited to our requirements, lower students are benefited in getting more global exposure, programming level increases and knowledge transfer rate is good. Hence finally from this experiment a sample of students is taken from different engineering levels and pairing is done on the same level and different level and their pair programming efficiencies are calculated. Basing on the results both null and alternative hypothesis are analyzed.

At the end of experiment questionnaire was emailed to students. They were not forced to answer the questions. This helps us to analyze the results of pair programming practice and to know the benefits and challenges of pair

programming. The Questionnaire sends to students is in Appendix.

4.2.6. Data Analysis

4.2.6.1. Hypothesis Testing

To reject a null hypothesis we use two-way ANOVA test to test the hypothesis [37] in section 4.2.3. Firstly, students' feedback is analyzed and the benefits were compared to the benefits drawn in qualitative analysis and are shown in Table 6. For our convenience assume pairs:

A: PG-PG, B: PhD-PhD, C: PG-PhD

D: UG-PhD, E: UG-UG, F: UG-PG

Table 6: Benefits of PP supported by student pairs and professors

Benefits	No. of professors Supported	Pairs supported
Code Productivity increases	9	A,B,C,D,E,F
Knowledge transfer	6	A,B,C,D,F
Increase in communication skill	7	A,B,C,D
Problem solving skill	8	B,C,D
Quality increases	8	
PP is enjoyable (work satisfaction)	10	A,B,C,D,E,F
Builds trust and increase team work	5	B,C,D
Increase in student Academic performance	6	A,B,C,D
New coding idioms	7	C,D

Table 6 signifies that the pair of different education levels has experienced more benefits than other pairs.

Now using the software tool ANOVA test is performed [38]. For this the treatment we have chosen the values in Table 4 and Table 5 which are the measures of pair programming effectiveness obtained by conducting an experiment. The test results are given in Appendix B. Finally, from the data analysis of student feedback benefits and test results the null hypothesis is clearly contradicted.

5. THREATS TO VALIDITY

5.1. Internal Validity

Maturation: The students participated in the experiment may lack concentration or enthusiasm and this might be a threat to experiment result. To prevent this threat experiment is executed with only interested participants.

Instrumentation: The instruments used in experiment like computers can affect the results of an experiment if the systems are not in good working condition, while measuring the time taken by programmers to complete the task. To avoid this threat instructor are advised to check the systems [19].

Selection: Selection of a particular programming language like Java in this experiment could affect the results. As some participants may not have a basic idea of programming in Java, and they may take a lot of time to understand. To minimize this threat prerequisite for the student is to be of academic background of computer science/software engineering and who has completed the basic course of Java in their academics.

5.2. External Validity

Selection-treatment interaction: The experiment is conducted manually and no tools are used and also there is no industrial environment. For result validation this can be a threat.

5.3. Construct Validity

Hypothesis guessing: In an experiment if test subjects are not properly instructed to the students then the outcome of an experiment is affected and the students probably guess the hypothesis and false assumptions may deviate from the main aim. Hence, to avoid this threat the students are given 20 minutes brief instructions about what we want to do them in the experiment [36].

Investigator bias: Information from professors may be indirect and is concluded through the perspective of interviewee's views and also they have answered the questions with some designation but not from natural setting. This threat is to be handled by preparing the questions easily understandable to both interviewer and interviewee and can be easily answered by an interviewer.

5.4. Conclusion Validity

Heterogeneity of participants: The limitation of an interview with professors is that people are not equally articulate and perceptive [28]. The selection of professors from different expertise level and area of research may be a threat. To minimize this threat, we used selection criteria for professors whose area of interest is in pair programming.

Reliability measure: The answers given by interviewers are descriptive. This is a threat while we are analyzing the interview questions. To minimize this threat it is better to provide an option for answers on an ordinal scale which will be easy for measurement purpose.

6. CONCLUSION AND FUTURE WORK

In this research, we finally conclude that pair programming is an efficient technique in programming. While pairing the students from different levels benefits the students in sharing their knowledge, develop communication skill, low level students can gain more knowledge from the higher level students, and also they learn much better than by reading books.

As Researchers suggest that running experiments with students reduce the validity of study and more advantage than an industrial environment (Kitchenman 2002). Hence future work we recommend is to use this pair programming practice of different levels in an educational environment to an industrial environment. We suggest in industries the pairing should be done among the senior programmer and a fresher so that the fresher can easily learn the new technologies with just an initial idea about the technology. Industries can also cut off their costs involved in training the fresher and also the cost required to fix errors and debug them.

7. ACKNOWLEDGEMENT

We would like to thank Dr. Mikael Svahnberg, Jan Kasper Martinsen, Michael Unterkalmsteiner, Criag Lindley and Kranthi Kumar Lakum for spending their valuable time in discussing the ideas and reviewing our research paper.

REFERENCES

[1] Mustafa Ally, Fiona Darroch, and Mark Toleman, "A Framework for understanding the factors influencing pair programming success", Springer-Verlag Berlin Heidelberg, 2005

[2] Williams, L. & R. Kessler, *"Experiments with industry's 'pair programming 'model in the computer science classroom"*, Computer Science Education, Vol.11, No.1,pp.7-20,2001

[3] L. Williams et al., "Eleven Guidelines for Implementing Pair Programming in the Classroom," in *Agile 2008 Conference*, 2008, 445–452.

[4] H. Hulkko and P. Abrahamsson, "A multiple case study on the impact of pair programming on product quality," in *Proceedings of the 27th international conference on Software engineering*, 2005, 495–504.

[5] W.Chigona, And M. Pollock, "Pair programming for information systems student's new to programming: student's experiences and teacher's challenges, IEEE, 2008

[6] E. Bellini et al., "The Impact of Educational Background on Design Knowledge Sharing During Pair Programming: An Empirical Study," *Professional Knowledge Management* (n.d.): 455–465

[7] T. Van Toll, R. Lee, and T. Ahlswede, "Evaluating the Usefulness of Pair Programming in a Classroom Setting," in *6th IEEE/ACIS International Conference on Computer and Information Science, 2007. ICIS 2007*, 2007, 302–308.

[8] L. Thomas, M. Ratcliffe, & A. Robertson, "*Code warriors and code-a-phobes: a study in attitude and pair programming*", ACM. SIGCSE Bulletin, 35(1), 363-367, 2003 .

[9] Katira, L. Williams, E. Wiebe, C. Miller, S. Balik, and E. Gehringer, " On understanding Compatibility of Student Pair Programmers", Proceedings of the 35th SIGCSE technical symposium on Computer Science Education, March 2004, 2004.

[10] N. Katira, L. Williams, and J. Osborne, " *Towards Increasing the Compatibility of Student Pair Programmers*", Proceedings of the 27th International Conference on Software Engineering, May 15 – 21, 2005, 2005.

[11] L. Williams, L. Layman, J. Osborne, and N. Katira**,** *"Examining the compatibility of Student Pair Programmers*", the Agile 2006 International Conference, 2006.

[12] L. Layman, *"Changing students' perceptions: an analysis of the supplementary benefits of collaborative software development"*, Proceedings. 19th Conference on Software Engineering Education & Training, IEEE Computer Society, 2006: 8.

[13] P. Sfetsos, I. Stamelos, & I. Deligiannis, " *Investigating the impact of personality types on communication and collaboration-viability in pair programming - an empirical study. Extreme Programming and Agile Processes in Software .Engineering"*. 7th International Conference, XP 2006. Proceeding (Lecture Notes in Computer Science Vol.4044). Springer-Verlag, pages 43-52, 2006.

[14] S. Heiberg, Puus U., P. Salumaa, A. Seeba, " *Pair programming effect on developers productivity"*, XP 2003 Extreme Programming and Agile Processes in Software Engineering Conference, Proceedings (Lecture Notes in Computer Science), Springer-Verlag,Vol.2675, pages 215-24, 2003.

[15] M. M. Muller, & F. Padberg, *"An empirical study about the feel good factor in pair programming*". Proceedings. 10th International Symposium on Software Metrics . IEEE Computer. Society, pages 151-8, 2004.

[16] E.A. Chaparro, A. Yuksel, P. Romero, and S. Bryant, *"Factors affecting the perceived effectiveness of pair programming in higher education*", Proceedings 17th Workshop of the Psychology of Programming Interest Group, June 2005, pages 5 – 18, 2005.

[17] J. Chao, and G. Atli, *"Critical Personality Traits in Successful Pair Programming"*. AGILE'06, IEEE Computer Society, 2006.

[18] K.M. Lui K.C.C. Chan, "Software process fusion by combining pair and solo programming", *IEEE paper., 2008*

[19] B. A. Kitchen ham, S. Charters, *"Procedures for Performing Systematic Literature Reviews in Software Engineering*", EBSE Technical Report, Software Engineering Group, School of Computer Science and Mathematics, Keele University, UK and Department of Computer Science, University of Durham, UK, 2007.

[20] Barbara Kitchen ham, *"procedures for systematic review*", NICTA Technical report, Australia, 2004.

[21] C. McDowell, L. Werner, H.F. Bullock, J. Fernald, *"Pair Programming improves student retention, confidence, and program quality*", Communications of the ACM, Vol. 49, No. 8, pages 90-95, 2006.

[22] Tore Dyba, Erik Arisholm, Dag I. k. Sjoberg, Jo E. Hannay *"Are Two Heads Better than One? On the Effectiveness of Pair Programming"* ,IEEE Computer Society.

[23] Frank Padberg and Matthias M. Muller. "Analyzing *the Cost and Benefit of Pair Programming"*, Proceedings of the Ninth International Software Metrics Symposium(METRICS03), IEEE Computer Society

[24] A. Cockburn and L. Williams., "The costs and benefits of pair programming. *In extreme Programming and Flexible Processes in Software Engineering XP2000," Cagliari, Italy, June 2000.*

[25] J. Nosek. The case for collaborative programming Communication of the ACM, 41(3):105-108, March.1998.

[26] L. Williams, R.Kessler, W. Cunningham and R.Jeffries. *Strengthening the case for pair programming IEEE Software, pages 19-25, July/Aug. 2000.*

[27] W. Humphrey. *"A Discipline for Software Engineering" Addison*-Wesley, 1997.

[28] John W. Creswell, Research Design:Qualitative, Quantitative and Mixed MethodsApproaches, Sage Publications, Second.

[29] L.Williams and R.Kessler, Pair Programming Illuminated: Addison-Wesley, 2002.

[30] Brian Hanks, Charlie McDowell, David Draper, and Milovan Krnjajic. 2004. Program quality with pair programming in CS1. In Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education, pp. 176-180,2004.

[31] V.Vanhanen & C.Lassenius, "Effects of Pair programming at the development team level: An experiment", IEEE Transactions on Software Engineerin, pp.336-345,2005.

[32] Nachiappan Nagappan, Laurie Williams, Miriam Ferzli, Eric Wiebe, Kai Yang, Carol Miller, and Suzanne Balik. 2003. Improving the CS1 experience with pair programming.Vol 35, No.1 (January 2003), 359-362.

[33] C.McDowell,B.Hanks J.Fernald & L.Werner, "The effects of pair-programming o performance in an introductory programming course", ACM SIGCSE Bulletin, Vol.34, No1,pp 38-42, 2002.

[34] C,McDowell, B.Hanks & L.Werner, "Experiments with pair programming in the classroom", In Proceedings of the 8th Annual Conference on Innovation and Technology in Computer Science Education, pp.60-64,2003.

[35] J.Nawrocki & A.Wojciechowski, "Experimental evaluation of pair programming", In Proceedeings of the 12th European Software Control and Metrics Conferences, pp.269-276,2001.

[36] W.Trochim (2000). The Research Methods Knowledge Base, 2nd Edition. Atomic Dog Publishing, Cincinnati OH.

[37] V. Basili, "The Role of Controlled Experiments in Software

Engineering Research, " Empirical Software Engineering,

LNCS 4336, Springer-Verlag, 2007. pp. 33-37.

[38] Two-Way ANOVA tests, Agilient Technologies, 2005.

Available:http://www.chem.agilent.com/cag/bsp/products/gsgx/downloads/pdf/two-way_anova.pdf.

[39] C.Wohlin, Experimentation in software engineering: an introduction, vol. 6. Springer Netherlands, 2000.

APPENDIX A

Questionnaire for Professors

Q1. What is your designation?

Q2. How many years you are working on the field of pair programming?

Q3. What are the benefits of pair programming?

Q4. What are challenges of pair programming?

Q5. Which programming practice is best solo or pair programming?

Q6. What is your future work on pair programming?

Questionnaire for Students

Q1. How was the co-operation with your team member?

Q2. How was the communication with your team member?

Q3. How is Quality of the program by doing in pairs?

Q4. What is your satisfaction level of knowledge sharing in pair programming?

Q5. How is the work environment in pair programming?

Q6. What benefits you observed while doing pair programming?

Q7. What challenges do you face while working in pairs?

APPENDIX B

Data Entry: ANOVA

Enter in the below set of boxes your data for each group (order makes no difference within a group) and then click on the Calculate Now button. Empty boxes will be ignored.

Calculate Now | Clear All

Data for Group *A*

A_{01}= 13 A_{02}= 11 A_{03}= 8 A_{04}= A_{05}=

Data for Group *B*

B_{01}= 10.5 B_{02}= 8 B_{03}= 9 B_{04}= B_{05}=

Data for Group *C*

C_{01}= 8.25 C_{02}= 4 C_{03}= 14 C_{04}= C_{05}=

Data for Group *D*

D_{01}= 9.5 D_{02}= 7 D_{03}= 12 D_{04}= D_{05}=

Data for Group *E*

E_{01}= 8 E_{02}= 4 E_{03}= 14 E_{04}= E_{05}=

Data for Group *F*

F_{01}= 11 F_{02}= 9 F_{03}= 10 F_{04}= F_{05}=

ANOVA: Results

The results of a ANOVA statistical test performed at 12:21 on 17-MAR-2012

Source of Variation	Sum of Squares	d.f.	Mean Squares	F
between	8.906	5	1.781	0.1627
error	131.4	12	10.95	
total	140.3	17		

The probability of this result, assuming the null hypothesis, is 0.972

Group A: Number of items= 3
8.00 11.0 13.0

Mean = 10.7
95% confidence interval for Mean: 6.504 thru 14.83
Standard Deviation = 2.52
Hi = 13.0 Low = 8.00
Median = 11.0
Average Absolute Deviation from Median = 1.67

Group B: Number of items= 3
8.00 9.00 10.5

Mean = 9.17
95% confidence interval for Mean: 5.004 thru 13.33
Standard Deviation = 1.26
Hi = 10.5 Low = 8.00
Median = 9.00
Average Absolute Deviation from Median = 0.833

Group C: Number of items= 3
4.00 8.25 14.0

Mean = 8.75
95% confidence interval for Mean: 4.588 thru 12.91
Standard Deviation = 5.02
Hi = 14.0 Low = 4.00
Median = 8.25
Average Absolute Deviation from Median = 3.33

Group D: Number of items= 3
7.00 9.50 12.0

Mean = 9.50
95% confidence interval for Mean: 5.338 thru 13.66
Standard Deviation = 2.50
Hi = 12.0 Low = 7.00
Median = 9.50
Average Absolute Deviation from Median = 1.67

Group E: Number of items= 3
4.00 8.00 14.0

Mean = 8.67
95% confidence interval for Mean: 4.504 thru 12.83
Standard Deviation = 5.03
Hi = 14.0 Low = 4.00
Median = 8.00
Average Absolute Deviation from Median = 3.33

Group F: Number of items= 3
9.00 10.0 11.0

Mean = 10.0
95% confidence interval for Mean: 5.838 thru 14.16
Standard Deviation = 1.00
Hi = 11.0 Low = 9.00
Median = 10.0
Average Absolute Deviation from Median = 0.667

AUTHORS BIOGRAPHY

- Venkata Vinod Kumar Padmanabhuni

Masters of Science in Software Engineering
Blekinge Institute of Technology-Sweden

- Hari Praveen Tadiparthi

Masters of Science in Software Engineering
Blekinge Institute of Technology-Sweden

- Muralidhar Yanamadala

Masters of Science in Software Engineering
Blekinge Institute of Technology-Sweden

- Sagar Madina

Masters of Science in Software Engineering
Blekinge Institute of Technology-Sweden