INFLUENCE OF PERSONALITY SHORTFALLON THEEFFECTIVENESS OFDEFECT DENSITY IN PAIR PROGRAMMING

[*]Mrs. k.s.sunitha ,[#]Dr. K. Nirmala

[*]Research Scholar, PG& Research Department of Computer Science,
Quaid-E-Millath Government College for Women(A), University of Madras, Chennai,
sunithasura2001@gmail.com

[#]Associate Professor, PG& Research Department of Computer Science,
Quaid-E-Millath Government College for Women (Autonomous), Anna Salai, Chennai,
nimimca@yahoo.com

ABSTRACT - Various software developmental issues in pair programming environment such as,reducing defects, better bug fixing, optimumknowledge combinationofexpertise and thereby reducing developmental time etc.,have been reported in literature. They are however found mostlyoncomparative studies with single programmingofs/w projects.These characteristics arealso found to be dealt withmostly in isolation bythe available literature.Whether certain personality shortfalls among pair programmers in some of the above specified programmer characteristics would influence the efficiency in reducing s/w defects?Thisissue mightdemand forempirical social factors of programmer characteristics for further research purposes.We have reported in our earlier work that defect density (defective codes to overall code ratio) has reduced in small sized s/w projects, when domain experts have been paired with conventional programmers.This paper will be an extension of that, which attemptsfor an investigation on delimited personality shortfalls, namely talent in combination with domain job matching. The investigation aims at studying the influence of cross trainingmade on inexperienced programmers (of the pair) in cross domains with an objective to determine whether these shortfalls would be rectified and therebythe defect density of small sizeds/w development projects would decrease?Experimental setups, social surveys and the results reported in this paper form a part of a whole research program of the authors, and hence some of the relevant input data have beendrawn from our earlier published work [SunithaK. S &Nirmala, K, 2015]. When s/w defects relate to lines of coding (LOC) or in other words the developmental efforts, due to personality shortfalls might directly influencethe concerned paired humans. It is hypothesized that the domain experts (navigators) could controland influence the programmers (drivers) one anotheror in a combined fashion.The paper presents relational study results between defect densities and the talent personality shortfall in combination with job matching of 5 selected small sized s/w development casestudies. For the purpose of experiments,apart from the experts who would act as navigators, the driver programmers are selected from a control group of graduate students. For the experiments presented in this paper, the personality shortfalls are represented in qualitative metric forms and the results would provide an empirical cue to theinfluence of the defect densities.The results will demonstrate the major role played bythe cross training component on the non-domain programmers (the drivers). The results reported in this paper will be of importance to s/w engineering researchers and also provide utility values to s/w project managers.

KEYWORDS: Pair programming; Personality shortfalls; S/w defect density; Pair programmingpsychology; Domain specific experts; Cross training.

1.0 INTRODUCTION AND BACKGROUND

One of the practices of eXtreme Programming (XP) is paired programming which puts two programmers together for developing one application codes. Literature indicates that personality traits such as communication, comfortableness, confidence and compromising abilities are beneficial to such s/w developmental practices. Therefore personality traits of pair programmers are needed to be checked in s/w developmentalprojects of XP [Andrew J. Dick & Bryan Zarnett, 2002].Programmer pairs generally proceed to develop codes when a task is assigned and each one of the pair taking in turn the role of driver (the one who writes the code) while the other whoacts as a navigator (or directs the logics). To maximize the production, exchanging the driver with navigator is recommended after each task [Sallyam Bryant, 2004]. But the psychological issue that would be raised is: whether often interchangingthe driverswith navigators is advantageous? This could be worth studying when both the participants of the pair are competent enough programmers. But results [Sunitha, K. S&Nirmala, K, 2015] are encouraging when the navigator of the pair is of a domain expert (whether proficient in programming or not) while the driver might remain a professional programmer (whether sufficiently experienced or not).

Psychological aspects of programmers (of pair programming) with their underlying behavior need to be researched upon for the implications of debugging purposes [Sallyann Bryant, 2004]. Literature on study of personality traits for selecting best methodology or increasing the team spirit or to determine the causes of success or failure of development etc., have been found to be plenty. But study on the influence in reducing the s/w defects by pair combinations particularly with an expert and also incorporating cross training is rare to be seen in literature. Under this specific background, this paper attempts to make an empirical study on reduction of defect densities of certain code defects due to fitting the apt developer personalities of the pair. The paper also attempts to concentrate on the navigator's role from domain experts who would pair with relatively inexperienced programmers who on the other hand would be taking the driver's role. The main aim of the proposed experiment is to study whether cross training of the programmers by the domain experts themselves would influence in reducing the defect density?With strong literature support for the purpose of the proposed experiments, the programmers who will form the driver's role and also who would pair with the experts are selected from a pool(control group) of students of Computer Science and Application of the authors' institution.The experts are invited from small/medium sized consulting firmswhoare also engaged as part time research scholars/higher education learners but they are however experienced in their own domain expertise. The role of student programmers for experiments is supported heavily by literature. Majority of the research studies on software engineering and pair programming have been taken place in academic environment [Williams, L et. al. 2000]. The findings presented in this paper will be of immense use to s/w engineering researchers and the utility value will be useful to small and medium sized s/w project developers, who intend to adopt pair programming or XP.

2.0 LITERATURE SUPPORT AND PROBLEM FORMULATION

Psychological problems among programmers have been reported ontheir understandingsthrough system metaphors [Sallyann Bryant, 2004].To tackle such issues, it is demonstrated that experts could lead the design processes through proven developmental models that might transform it into working metaphors. To maximize the spread of knowledge, particularly in pair programming environments,the suitability of programmers need to be studied. It is therefore suggested that pair programmers need to be interviewed and assessed with their communication abilities, comfort levels, confidence and ability to compromise [Sallyann Bryant, 2004]. Program coding under logical conditions (such as defect densities) might be affected due to programmercharacteristicsor personality traits thatcould dominateamong pairsthat would seriously influence the quality of the development[Kwak and Stoddard 2004]. Application dependent errors could lead to multitude of other common errorsin addition to popular errors [Benjamin L et. al. 2005]. Software developers in general attend more to commonly known errors caused by operating systems, but it is reported that there are lesser known application dependent errors in software. Besides, these application specific errors are responsible for multitude of other errors. Only application specific domain experts can be able to discover such errors and can fix them quickly. It is therefore concluded that application domain experts could discover execution errorsbetter than programmers who would concentrate more on mathematical errors.

There is a relationship between personality and cognitive capability of s/w developers [Abdu Mekonnenand WorkshetLamenew, 2013]. Personality of developers indicated team maturity level. The relationships of personality traits have shown influencingconsequences on the appropriate s/w development methodology, such as agile / plan-driven. It is reported that the preferences for development methodology has been found to be linked with developer personalities. Thereby personality traits could be used for the study of suitability of developmental methodologies. Hypotheses like significant relationships among personality factors and s/w development methodologies and personality factors, and also on agile method have been tested and validated. It was also reported that s/w engineering schools could not provide proper training in agile techniques.

Interview techniques could be administered to gauge personality traits like communication, comfortableness within a team [Andrew J. Dick and Bryan Zarnett, 2002]. In single programming environment, introverted developers could flourish since interpersonal communication is not a mandatory. It is also found that extremely uncompromising developer could have an absolute control over the development. Hence interviewscheduleswith programmers would be the best towards determining aptitude for pair programming. Success ratesmight be high when pairs are built with appropriate personality traits which have been determined in addition to technical skills. In XPs, where pair programming is more promoted, questions like why a project has failed or succeeded will throw more light on personality traits of developers [Petre, M, 2003]. For experimentations to determine personality traits, applications of external representations were tried out[Kim Nilsson, 2003]. It was found that the productivity gain by two programmers producing a single set of code, compared with twice the cost of a single programmer. In such cases it might be worthwhile to replace the one programmer (the navigator) with an expert; but then a serious question arises: whether the expert may not be proficient in programming, while the programmer on the other hand might not understand the domain problem logically. It is therefore due to the abovegaps found in the literature, it is proposed to conduct experiments with and without cross training on the programmers having personality shortfalls by domain experts themselves and empirically confirm the influence of personality traits on the defect density of pair programming.

3.0 METHODOLGY AND EXPERIMENTAL SETUP

Empirical studies in academic environment on pair programming for over-arching understanding about pair programmers' personality traits have been reported [Sallyann Bryant, 2004]. In Delphi method for expert judgments, the participants in the survey are involved in the decision-making processes [Rowe and Wright, 2001]. Domain experts answer to a set of structured questions raised through interviews for judging personal traits, in a couple of rounds as per this technique. In our experiments, the programmers are selected from a control group of young inexperienced students who although have programming talents do not have application domain knowledge (or job mismatching). But on the other hand, the intended navigators are experienced experts from commercial s/w concerns (Table 1.0). Comfort levelis thus assured as age difference is even though much between drivers and navigators, but the latter onesare experienced domain expertswhere asthe driversare programmers who do not have application domain expertiseand therefore there will not be any fear for ridiculing in domain areas and no comparisonswould be made on the job positions as expertsare different in their profilesfrom programmers as contradicted and cited otherwise by [Andrew Dick et. al2002].

The sampling technique adopted for the intended social survey is 'Purposive' [Sharma, B.A.V, 1988], as it is known to be representative of the total required input for the specific purpose of the survey. In view of the above, the survey methodology was administered withactual demographic data presented in Table 1.0. Five s/w projects are identified for the experiments and the initial first round (before training) LOC in Java are: P1- Hotel management s/w LOC: 2633; P2: Travel and Tours management s/w LOC: 3450; P3: Banking s/w LOC: 4128; P4: Financial & Banking lending/borrowing concern s/w LOC: 7402; and P5: Combined Tourism & Hotel Management Concern s/w LOC: 10239.The sample size for the survey: 32 (Students/programmers = 26 and Experts/navigators & trainers = 6 – see Table 1.0). The talent is infused with job matching for the purpose of experiments. The hypotheses considered for the 10 questions (presented in the next section) are: 1. Experts generally will adhere to high standards than students (programmers); 2. Experts prefer to work alone compared to students (programmers); 3. Experts generally do not work well with students (programmers); 4. Experts do not in general wish to involve in confrontation unlike students (programmers); 5. Experts will not be warm and lively with students (programmers); 6. Experts will generally be dominating than students (programmers); 7. Experts think more abstractly thanstudents (programmers); 8. Experts are more perfect than students (programmers) in their behavior; 9. Experts are more perfect than students (programmers) in their approaches; and 10. Experts felt more tension than students (programmers) while handling pressure. Hypotheses have been tested by one way ANOVA (using SPSS 17.0).

Personality trait factor for expert is considered to be equal withprogramming talent and working experience in the domain (projects). Even 'novice' to 'medium' knowledge of programming is considered to be half of the programming talent factor. Shortfall is 1.0 – the total factor. But for student programmers the total final factor is obtained from interview schedule and opinion received from the job matching talented experts. As the survey is based on Delphi technique, both the participants (respondents) of the experiments were subjected to social survey. The demography along with talent shortfall factors are presented in Table 1.0.

Table 1.0 Demography and Talent Factors of Experts and Student Programmers

Id of Project	**Project Domain**	**No. & nature of Respondents**	**Years of experience**		**Talent Shortfall Factor in relation to Job matching**
			Specific to Domain	**Specific to Programming**	
P1	Hotel Management	1 Expert	5	Novice	0.25
P3 & P4	Banking and Financial & Banking Lending/Borrowing Concern.	3 Experts	12, 8, 5	Medium & Novice	0.25
P2 & P5	Combined Tourism & Hotel Management Concern	2 Experts	5, 4	Nil	0.5
P1, P2, P3, P4 & P5	All the above	26 Student Programmers	Virtually Nil	Sufficient proficiency of few years	0.0 in Job matching and 1.0 in programming

4.0. RESULTS AND DISCUSSIONS

Many researchers could not assess pair programmers' cognitive perspectives [Williams et. al. 2000]. It is reported that as long as the quality of software is concerned, pair programming is favored. Hence an attempt is made to make the pair with least talent (highest personality shortfall) student programmer with expert for the first three projects and high talent (lowest personality shortfall) with expert as pair for the later two projects. The experiments were repeated after appropriate training in that domain to the selected programmers by the experts. The cognitive perspectives are determined using a combination of talent and job matching in the following survey for determining highest and lowest talented student programmers. Unless otherwise specified, each question is raised on talent in relation with job matching. The scale for questions 1 to 4 is presented that precedes the questions 1 to 4 presented in Table 2.0.

Scale for Q.Nos. 1 to 4:

1.0	**2.0**	**3.0**	**4.0**	**5.0**
Strongly Agree	Somewhat Agree	No Opinion	Somewhat Disagree	Strongly Disagree

Table 2.0 Questions 1 to 4 as Variables and the Intended Measure

Q. No.	**Question**	**Measure of**	**Preferable Range**
1	I adhere to high standards.	Talent & Job Matching	Lower
2	I prefer working alone.	Job Matching	Higher
3	I work well with others.	Job Matching	Lower
4	I don't like confrontation.	Talent	Lower

The appropriate scale values are presented for questions 5 to 10 for each question in succession presented below.

Q. No. 5: Relating to pair partner**.**

Scale for Q. No. 5:

1.0	**2.0**	**3.0**	**4.0**	**5.0**
Warm	Live	Bold	Private	Self-reliant

Q. No. 6: Influencing or Collaborating.

Scale for Q. No. 6:

1.0	**2.0**	**3.0**	**4.0**
Dominating	Bold	Vigilant	Open

Q. No. 7: Style of Thinking.

Scale for Q. No. 7:

1.0	**2.0**	**3.0**	**4.0**
Warm	Sensitive	Abstract	Open

Q. No. 8: Flexibility.

Scale for Q. No. 8:

1.0	**2.0**	**3.0**	**4.0**
Lively	Rule oriented	Abstract	Perfect

Q. No. 9: Structured Approach.

Scale for Q. No. 9:

1.0	**2.0**	**3.0**	**4.0**
Casual	Syntactic	Abstract	Perfect

Q. No. 10: Pressure Handling.

Scale for Q. No. 10:

1.0	**2.0**	**3.0**	**4.0**
Emotional	Vigilant	Cannot Say	Tension

The measuring talent metrics along with questions 5 to 10 and also the preferred range of result are presented in Table 3.0.

Table 3.0 Measuring Metrics for Questions and Preferred Range

Q. No.	Question	Measure of	Preferable Range
5	Relating to pair partner	Job Matching	Lower
6s	Influencing or Collaborating	Job Matching& Talent	Higher
7	Style of Thinking	Talent	Higher
8	Flexibility	Talent	Higher
9	Structured Approach	Job Matching	Higher
10	Pressure Handling	Talent	Lower

For want of conserving space only a couple of survey results are presented, while the hypothesis test is presented with final results (Table 4.0).

4.1 Sample Survey Results

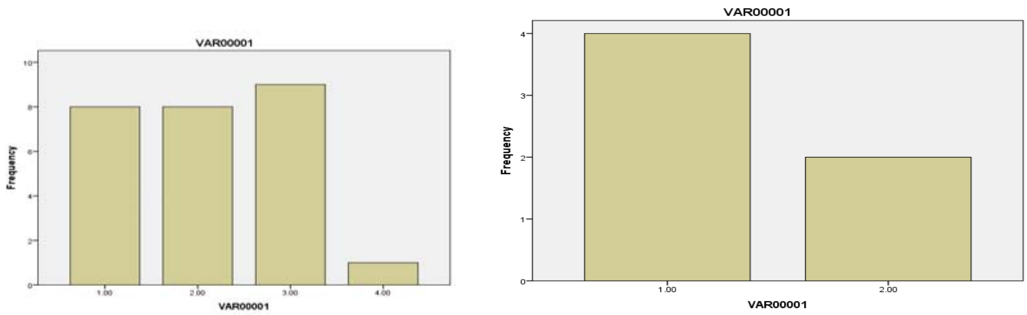
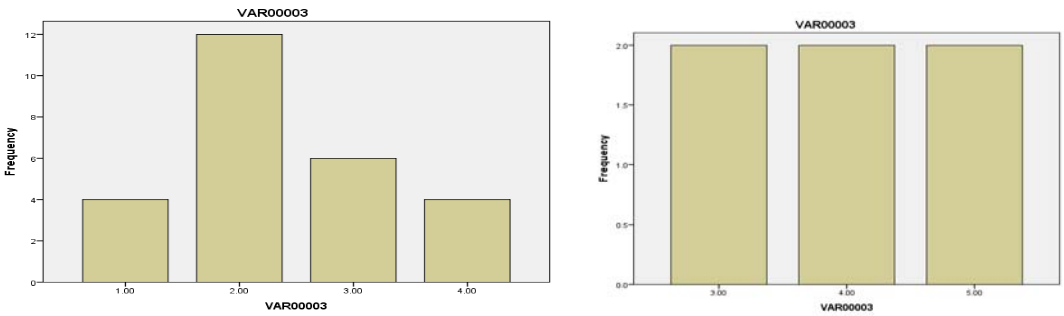


Figure 1.0 Sample Survey Results of Students and Experts on adaptation of High Standards



Figures 1.0 and 2.0 show the responses and comparisons made between student programmers and experts on question 1 and 3 respectively. Figure 1.0 show a clear positive responses provided by both the groups. This is perfectively reflected in the ANOVA result shown in Table 4.0 (see the null hypothesis on variable 'VAR00001' of first row. Figure 2.0 show a significant difference between responses provided by both the groups on question 3 viz. willingness to work in pairs. This is once again reflected by the ANOVA results of Table 4.0.

Table 4.0 Test on Hypotheses through One way ANOVA

Q.No.	Variable	Standard Deviation		'F'	F_{crit} @ 0.05 Significance	Result on Hypothesis
		Programmers	Experts			
1	VAR00001	.90893	.51640	0.00	1.00	Null
2	VAR00002	.89443	1.0488	0.278	0.840	Insignificant
3	VAR00003	.94136	.89443	1.800	0.306	Significant
4	VAR00004	1.07917	81650	0.281	0.773	Insignificant
5	VAR00005	.94787	.75277	1.219	0.410	Significant
6	VAR00006	1.06699	.75277	13.875	0.030	Very Significant
7	VAR00007	96157	.54772	0.125	0.742	Insignificant
8	VAR00008	1.10732	.98319	0.214	0.818	Insignificant
9	VAR00009	1.32723	.54772	0.063	0.815	Insignificant

As the number of respondents is high with students and also the responses are wide, the standard deviations (Table 4.0) of some responses (variables) are slightly more than 1.0.

From the above survey results, the personality shortfalls are empirically arrived at for each student. When the factor value is less than 0.50 the student is termed as less talent whortfall while if it is more than 0.5 then the personality shortfall is considered high.

Legend for Table 5.0:

T: Talented; **TS_f:** Talent Shortfall; **JM:** Job Well Matched; **JMS_f:** Job Match Shortfall

The values that were arrived at through normalizing the feedback scale values are presented in Table 5.0.

Table 5.0 Normalized Talent and Shortfall Factors from the Survey

Variable	**Student Programmers**				**Experts**			
	T	**TS_f**	**JM**	**JMS_f**	**T**	**TS_f**	**JM**	**JMS_f**
1	15	11	15	11	6	0	6	0
2	-	-	6	20	-	-	3	3
3	-	-	16	10	-	-	0	6
4	3	23	-	-	5	1	-	-
5	-	-	12	14	-	-	0	6
6	-	-	13	13	-	-	5	1
7	16	10	-	-	6	0	-	-
8	11	15	-	-	3	3	-	-
9	-	-	11	15	-	-	6	0
10	14	12	-	-	No reliable feedbacks received			

From the Table 5.0, three least talented student programmers who have got the highest shortfall factors have been selected to form pair in the first three projects namely P1, P2 and P3. Two high talented student programmers who have obtained small shortfall factors have been selected to pair with experts in the two projects namely P4 and P5.

4.2 Effect of Defect Densities Before and After Cross Training in Domain Areas

For the purpose of testing experimentally on the influence of personality traits on defect densities, cross training on the project domains were provided by the experts themselves to the selected programmers after the first stage of developments were completed. The experiments were repeated with pair programming. It is reported that the speed of development after training was fast and the cost analysis and speed of development of expert-programmer pairis beyond the scope of this paper. Tables 6.0 and 7.0 show the three selected defects namely syntax, execution and application dependent have been analyzed and presented before training and after training for projects P1, P2 and P3 (Table 6.0) and for P4 and P5 in Table 7.0 respectively.

Table 6.0 Defect Densities before and after Cross Training of Personality ShortfallStudent Programmers

Project Id	Defect Density Before Cross Training			Defect Density After Cross Training		
	Syntax Error	Execution Error	Application Error	Syntax Error	Execution Error	Application Error
P_1	0.101	0.072	0.002	0.054	0.068	0.002
P_2	0.098	0.063	0.003	0.049	0.030	0.002
P_3	0.103	0.067	0.003	0.034	0.030	0.002

Table 7.0 Defect Densities before and after Cross Training of Talented & Job Matched Student Programmers

Project Id	Defect Density Before Cross Training			Defect Density After Cross Training		
	Syntax Error	Execution Error	Application Error	Syntax Error	Execution Error	Application Error
P_4	0.094	0.024	0.001	0.018	0.012	0.001
P_5	0.097	0.078	0.002	0.003	0.026	0.001

The results are presented figuratively in Figure 3.0 for syntax defects; Figure 4.0 for execution defects and Figure 5.0 for application defects.

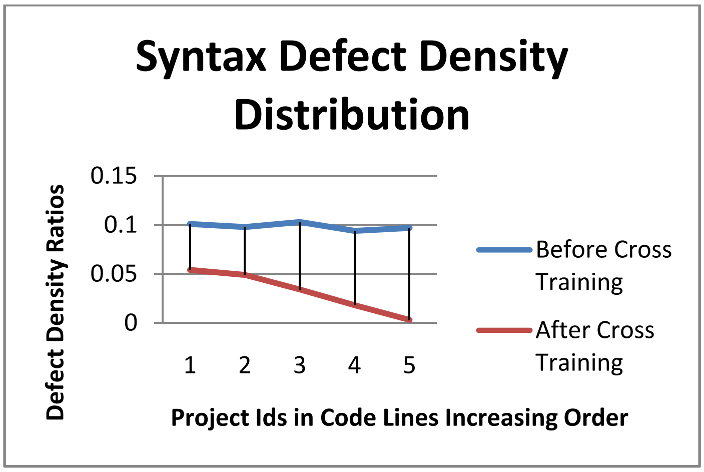


Figure 3.0 Distribution ofSyntax Defect Densities Before and After Training

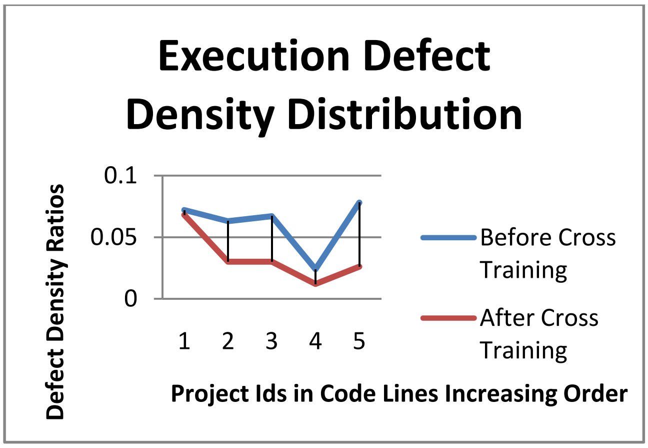


Figure 4.0 Distribution ofExecution Defect Densities Before and After Training

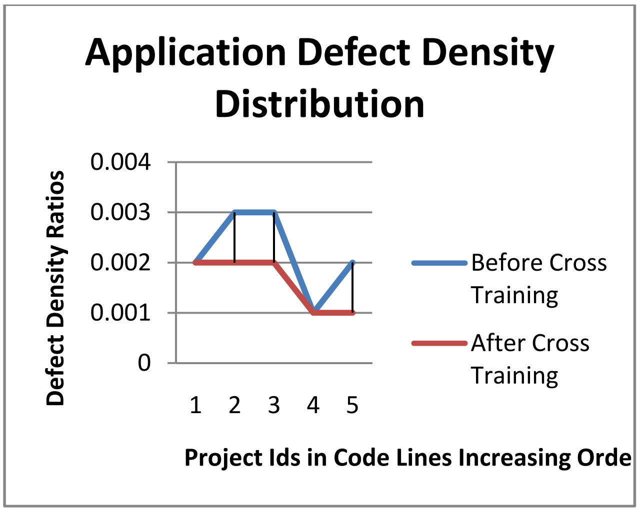


Figure 5.0 Distribution ofApplication Defect Densities Before and After Training

It is observed from the figures and results presented in the above tables that there is certainly a reduction in defect densities after cross training is provided by the experts to the programmers. Various conclusions are drawn from the above experiments.

5.0 CONCLUSIONS

1. No serious confrontation caused due to personality traitshas beennoted in pair programming, when programmers who acted as drivers but experts acted as navigator is paired with; irrespective of the lack of domain knowledge in programmers and lack of programming knowledge in experts.

2. It is clearlydemonstrated that cross training in domain areas is effective in reducing defect density in all cases.

3. Experts were not to be found dominating the programmers; on the other hand were found to be more open with them.

4. Improvements due to cross training seemto be more uniform in the case of personality shortfall student programmers, rather than that is with talented student programmers.

6.0 REFERENCES

[1] Abdu Mekonnen and WorkshetLamenew, (2013), “The Role of Personality of Software Developers in Selecting Software Development Methodology”, HiLCoE Journal of Computer Science and Technology, Vol. 1, No.2, June 2013.

[2] Andrew J. Dick and Bryan Zarnett (2002), “Paired Programming and Personality Traits”, Workshops on Database Theory – XP, 2002, pp: 82-85.

[3] Benjamin L and Zimmermann T, 2005, “DynaMine: Finding Common Error Patters by Mining Software Revision Histories”, Proceedings of the 10th European Software Engineering Conference, 13th ACM SIGSOFT International Symposium on Foundations of Software Engineering, ACM, NY, USA, pp: 296-305.

[4] Kim Nilsson, (2003), “Increasing Quality with Pair Programming”, http://uppsatser.se/om/Kim+Nilsson/

[5] Kwak Y.H. and Stoddard J. (2004),”Project Risk Management: Lessons Learned From Software Development Environment’, Technovation Vol.24,Sciencedirect, pp.915-920.

[6] Petre, M, (2003), “Team Coordination through Externalized Mental Imagery”, Proceedings of the Co-located 15th Annual Psychology of Programming Internet Group Workshop and Empirical Assessment of Software Engineering Conference. 2003

[7] Rowe and Wright, (2001), “Expert Opinions in Forcasting. Role of the Delphi Technique. In “Principles of Forecasting: “A Handbook of Researchers and Practitioners”, Ed. Armstrong, Kluwer Academic Publishers, Boston, USA, 2001.

[8] Sallyann Bryant (2004), “XP: Taking the Psychology of Programming to the eXtreme”, 16th Workshop of the Psychology of Programming Interest Group, Ed. Dunican, E & Green T.R.G, Proceedings of PPIG, Carlow, Ireland, April, 2004, pp: 121-132.

[9] Sharma, B.V.A, (1988), “Research Methods in Social Sciences”, Sultan Chand Publishers, New Delhi, 1988, India.

[10] Sunitha, K. S and Nirmala, K, (2015), “Correlation Study on Defect Density with DomainExpert Pair Speed for Effective Pair Programming”, Indian Journal of Science and Technology, Vol 8, No. 34, Dec. 2015.

[11] Williams, L, Kessler, R.R, Cunningham, W, Jeffries, R (2000), “Strengthening the Case for Pair Programming”, IEEE Transaction on Software, Vol. 17, No. 4, 2000, pp: 19-25.