iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 5 **Issue:** XI **Month of publication:** November 2017

DOI: http://doi.org/10.22214/ijraset.2017.11051

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887
Volume 5 Issue XI November 2017- Available at www.ijraset.com

Effectiveness of Programming in Pair Comparative Analysis through Project Experiments

Tushar Sambare [1], Gulabchand K. Gupta [2]
[1]*Research Scholar, Dept. of Comp. Science and Engg., Shri JJT University, Jhunjhunu, Rajasthan, India*
[2]*Adjunct Professor, Dept. of Comp. Science and Engg., Shri JJT University, Jhunjhunu, Rajasthan, India*

Abstract*: The empirical study current SW methodologies proven that the Extreme Programming (XP) is an emerging standard for the SW development nowadays. One of the popular practices called as "Programming in pair" is part pf XP. It is one of the essential and main practices of the lightweight development methodology. The Programming in pair refers to the process of two developers working together in a single computer terminal to develop and test programs. The parts of programming in pair are driver and guide where the driver composes the code and the guide helps the driver and searches for problems in code. The research paper explains the research work done through experimental projects to measure the efficiency of Programming in pair for learning process of SW skills. The paper demonstrates the overall productivity effects of Programming in pair for process and quality metrics of the project.

Keywords*: Programming in pair, Waterfall Model, Quality Metrics, Productivity, Measurement, Project Experiment, Efforts Measurement

I. INTRODUCTION

Programming in pair is a social expertise that sets aside opportunity to learn. This is like taking a stab at a helpful approach to work that incorporates give and take from the two accomplices paying little heed to corporate status. The best match developers know when to state "we should attempt your thought first." Don't anticipate that individuals will be great at it from the beginning. It helps on the off chance that there is a need have somebody on group with understanding to demonstrate everybody what it should feel like. Programming in pair utilizes the four eyes principal, guaranteeing that two arrangements of eyes audit the code that is being delivered notwithstanding when there is a bifurcation of work. While one individual composes test cases utilizing test-based development procedures, for instance, the other individual may compose code to get the tests to pass. Commonly, the two engineers will turns coding or looking into, checking each other's work as they go. Figuring out how to accomplice adequately in a group that nearby and sharing a work PC takes aptitudes that not all software engineers have. It requires the two developers to have the fundamental delicate abilities required for coordinated effort and also the vital hard aptitudes to compose and test code.

However there some myths about Programming in pair as such, it halves the productivity of developers. This can be only true if the toughest part of programming was typing. It was believed that t a pair is actually more productive that two separate developers. This is because of the constant discussion and review that blending presents. One can come up with better designs, make less mistakes, and make more people familiar with the code. All of these things offset having less people typing. Of course, since we Cannot Measure Productivity we can't know for sure. There is need to try it and the team should reflect on whether they feel they are more effective with pairing that without. As with any new practice make sure you allow enough time so you have a good chance of crossing the Improvement Ravine. There is also a misconception that the Programming in pair is only worth on complex code, rote code yields no advantage. There is a point to this - pairing is about improving design and minimizing mistakes. Repetition of code that is easy to compose yields little open doors for matching to have any kind of effect. Except this writing boring repetitive code is a smell. On the off chance that one needs to compose exhausting monotonous code it's generally a sign that possibility to miss an essential abstraction, one that will definitely limit the measure of repetitive code to compose. Programming in pair can help to find out that abstraction.

In this research project experiments are performed to find out the better side of the Programming in pair. The findings of the experiments are observed with different parameters.

II. PROBLEM IN HAND

SW development is a learning process, developers cannot program faster than they can learn [5]. The efficiency of the learning process and the productivity can be measured using the following parameter:

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887
Volume 5 Issue XI November 2017- Available at www.ijraset.com

A. Schedule

It is defined as the total development time elapsed between the start and comple of programming task including debugging.

B. QUALITY OF DESIGN

The amount to which the design satisfies the completeness and correctneof the requirements.

D. ACQUIRED KNOWLEDGE

It indicates individual's gained knowledge and programming skills.

E. SIZE

It is the variance between the estimated size of the project and the actual size of the project at the end of project experiment.

F. REQUIREMENT STABILITY

It provides visibility to the magnitude and impact of requirements changes.

G. PRODUCTIVITY

It is a measure of output from a related process for a unit of input with test case preparation and execution as well as defect identification and fixation.

There are some other parameters related to quality of the project can be considered for the measurement to get the insights of quality processes followed with Programming in pair strategy and with traditional programming approach.

1)Quality Cost in project processes: It is a measure of the performance of quality initiatives in an organization. It's expressed in regards with efforts invested in project for quality processes. It is also measured for Poor Quality cost to measure the rework efforts.

2)Efficiency in Review and Testing: It is characterized as the effectiveness in identifying review and testing defects in the verification phase.

3)Defect Density: It is the quantity of defects recognized in the product development partitioned by the span of the product (commonly in KLOC).

4) Defect removal efficiency: It measures the productivity with which defects were identified and kept from reaching the client.

It is argued in literature that Programming in pair, an important practice of XP, enhances the learning efficiency. Even though many reasons were stated for increase in learning efficiency by the use of Programming in pair, the research is to be conducted to evaluate the effectiveness of Programming in pair quantitatively. In this study, the researchers analysed the performance of the research participants through formal experiments when they adopt Programming in pair for completing the programming tasks as well as with solo programming practices. However these studies on Programming in pair have the following limitations:

5) Elapsed Duration: Programming in pair routine with regards to XP proposes the rotation of pair accomplices a few times in a day. The planning game proposes that a client story is separated into assignment. Viewing from these points of view, a couple works for around 3 hours to finish a task of programming.

6) Design Quality: SW design assumes a vital part in development. The design quality between Programming in pair and traditional method gatherings can be measured through the length of the code to actualize a similar usefulness. This measure is for the most part not considered as a compelling design quality measure. The viability of Programming in pair on design quality has not been investigated successfully up until now and it is to be assessed. The accompanying are the vital elements to be considered for assessing the learning productivity:

a) Accuracy (the degree to which the design represents the requirements correctly).
b) Inclusiveness (the degree to which the design represents the requirements completely).

7)Knowledge Acquisition: In a large portion of the past research works, the proficiency of Programming in pair was assessed by measuring the attributes of the product (PC programs) that were created by the research members. Nonetheless, to demonstrate that Programming in pair is a powerful learning technique, the improvement in person's subject information and programming abilities must be measured. Just couple of endeavours have been made to gauge the individual execution in the academic and in addition in

professional environment. Hence, it is necessary to conduct an experimental study to evaluate the effects of Programming in pair on learning efficiency in the context of project experiments.

8)Size: It is not practically possible to perform project experiment with industrial setting for various sizes scale of projects as there can be many options.

9)Requirement Stability: The requirements can vary client to client as per business need and the understanding of the software, so it is difficult to state stabilization impact with simple experiment findings.

10) Defect finding and Removal efficiency: This measurement is totally depends upon the knowledge and expertise of the developers, testers most important the customer itself client. There are chance of missing the defects by these participants. So it's not perfectly measured in the project experiments.

III. RESEARCH APPROACH

In the present study, formal experiments is chosen as investigation approach due to the advantages allied with the approach. It was decided to conduct the formal experiment using the last year undergraduate students of Bachelor of information Technology. The teams mentioned here have been undertaken task to complete the two identical projects using two disciplines such as Waterfall Model and Extreme Programming. The focus of present study was to compare the effectiveness of adopting Programming in pair in terms of learning efficiency and for computing the overall productivity improvement. The formal experiments are to be conducted to validate the following objectives:

The defect finding and removal efficiency is better programming in pair.
The programming pair invests more efforts in quality perspective processes.
The programming in pair consume less time for completion of the project.
The quality of design improves with programming in pair.

The above objectives are proposed based on the theoretical explanation on the benefits of Programming in pair and from the results of the earlier research on Programming in pair. In all of the above objective validation purpose, a project experiment is considered to be done under the supervision of a researcher.

A. Experiment Variables

1) *Independent Variable*: these variables represent the cause for the effect in any experiment. The effect in the present study is efficiency and the conjectured cause is the learning methodology. Thus, the independent variable for the study is learning methodology. The independent variable can be manipulated in three different ways of manipulating the independent variables namely presence or absence of technique, amount of technique and type of technique [6]. For this experiment, the type technique is chosen as it the present our problem. The two types of independent variables namely xp and waterfall method (solo programming) was considered for experiment
2) *Dependent Variables:* These variables represent the effects that are specified in the objective.

B. Quantification of Learning Efficiency

The improvement in learning efficiency (if any) of the Programming in pair group compared to that of the traditional method group is quantified by adopting the following steps:

1) The improvement in design evaluation due to the adoption of Programming in pair is expressed as a percentage variance between the mean values of design evaluation of Programming in pair and traditional method.
2) Weights are assigned to the three values (computed in step 1) as follows: Weights of 40% and 60% are assigned to the improvement of design evaluation and elapsed time. The higher weight is assigned to elapsed time as it involves the coding and testing skills.
3) The improvement in learning efficiency is computed by calculated on the basis of elapsed time and improvement design evaluation in percentage.

IV. THREATS

Threats to validity are factors beyond our control that can affect the dependent variables. Such threats can be considered unknown independent variables causing uncontrolled rival objectives to exist in addition to the research objectives. One crucial step in an experimental design is to minimize the impact of these threats [1]. Two different classes of threats to validity are internal validity threats and external validity threats. Threats to internal validity constitute potential problems in the interpretation of the data from

the experiment. If the experiment does not have a minimum internal validity, valid inference cannot be made about the cause effect relationship between independent and dependent variables. On the other hand, the level of external validity is an indicator of the generalizability of results. Depending on the external validity of the experiment, the data can be assumed valid in other populations and settings. The threats that are anticipated and tried to control are explained in following sections.

A. Internal Validity Threats

The group design of the present experiment eliminates many of the internal validity threats [4]. However, it was found that there were still some other internal validity threats in the experiment and they were handled as explained below.

1)Instrumentation Threats: The experiment involved 18 students and 9 developers. It is necessary to ensure that all the students and developers involved are doing the project of same duration and same level of difficulty. The uniformity in duration and the difficulty level of the exercises were verified by conducting the same set of project experiment using traditional method, with students who were not involved in the present study. It was also ensured that the team chosen for the experiment in a year had been trained by the same trainer.

2)Selection Threats: Each student and developer in a team was assigned randomly to either the Programming in pair group or the traditional method group.

B. External Validity Threats

Generally, the students are very cordial to each other and they are ready to help each other. If this kind of environment does not exist, present research experiment results will not be valid.

V. RESULTS

There are two experiment project have been performed during this research with two project development techniques. For both the project team a project “online jewellery shop management” project is allocated. Also both the teams had used same technologies for development as “ASP Dot Net C# and MSSQL server”. These settings are kept in experiment to have fair analysis between two project experiment with only concentrating the variance between project implementation process rather than technology used as per the research paper objective.

Both the team have given one day extensive training about their respective development procedure with seminar and some short hands on practices. There are 6 developers at each team allocated for one month specified duration of project. There are significant difference in the processes followed by Waterfall (WFM) approach and in XP procedure. The WFM is step down approach while the XP is an incremental approach. All the processes of WFM are not matching with processes of XP, so comparing them becomes difficult task depending on the processes followed.

In following section two projects results are presented using only matching factors of comparison of two project strategies. While the further section presents result details of only WFM and XP strategies as all the processes are matching for fair comparison. The next subsequent sections presents the results of the experiment in regards with the objective of the research.

A. Data Collection

Both quantitative and qualitative data were collected. The quantitative data was grounded on three basic data points, i.e. time, size and defect. While a number of other interesting data points could have been captured, these three metrics were seen to be the most beneficial for setting some references for other scholars and specialists. The following sections provide results on the basis of nine project metrics and seven process metrics.

B. Project Experiment Metrics Observations

In the observations presented here Project 1 is referred in regards with solo programming project results with Waterfall approach and Project 2 is referred in regards with programming in pair project results with XP approach.

It is observed that the schedule variance in Project 1 was 6.41, in Project 2 was 0. The results presents the fact that Project 1 process took more time to complete(2 days extra) the task than the scheduled, while Project 2 project finished as per schedule. This results are satisfying the research objective of time efficiency. Though the students are working on the project and same schedule allocated with same project domain and same technologies are used, the results of programming in pair presented the effective time management.

TABLE I

Sr.	Work Procedure Metrics	Project 1	Project 2
1	Variance in Schedule	6.41	0
2	Variance in Effort	22.8	20.3
3	Variance in Size	3.7	3.9
4	Stability Index Variance	2.6	1.65
5	Productivity of Project	0.041	0.046
6	Test case preparation productivity	2.2	2.3
7	Test case execution productivity	2.1	1.85
8	Defect detection productivity	0.67	0.72
9	Defect fixation productivity	0.41	0.44
Sr.	Metrics Related to Quality	Project 1	Project 2
1	Quality Cost	52.5	55
2	Poor Quality Cost	7.01	5.34
3	Density Defect	6.56	5.34
4	Efficiency of Review	34.6	38.8
5	Efficiency in Testing	56.6	60.13
6	Efficiency Defect removal	62.5	57.14
7	Density Residual Defect	27.4	33.14

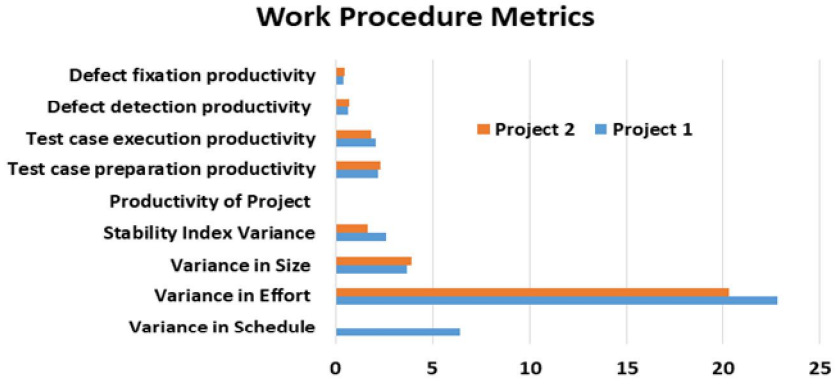
The experiment performed in this research are very much clarifying the fact that the programming in pair is not only saves the time but also proven to be better than the traditional approach. The variance of effort observed in the Project 1 project was 22.8, in Project 2 it was 20.3. Here the Project 1 team had spent more hours than planned while Project 2 teams have spent less hours than planned to complete the tasks. The effort measurement obtained here because of timely completion of tasks in Project 2, it result in less number of hours of work to be done , while in Project 2, as the project taken 2 days extra time for completion, it turns to increase in the efforts of team members also. The size of the project is measured in KLOC. The results shows that the both project types produced more lines of the code than the planned. This can be explained as all the students are working on the project not able to map the requirement properly into lines of code. However the difference in not too much hardy few hundreds of lines of code observed to be produced more both project types. The results also demonstrates the fact that the Project 2 produced more number of lines than project one, the reason behind this as there is live customer support for the project type 2 which turns to be more additional demands in functionality, adding the changes and finally turn to write more number of lines.

The requirements change in Project 1 is observed as 2.6 and in Project 2, it was 2.9. This results obtained as the all the requirements are specified in early stage of the project. While in case of Project 2, as the customer is available most of the time, the requirements are observed increased. The customer on site always result in increase of change as more they see, more they expect from the project.

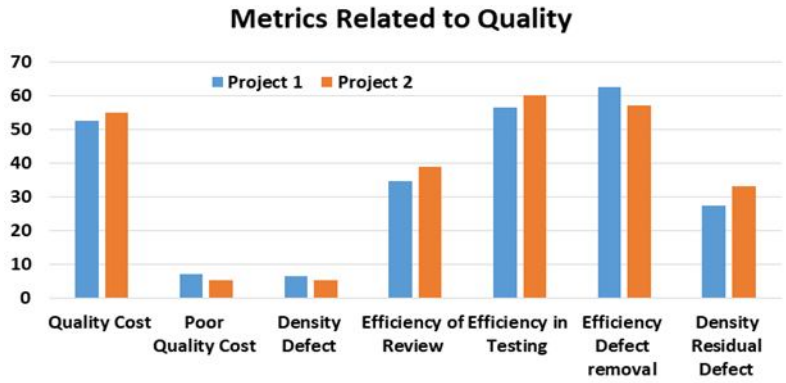
The productivity of the project observed in Project 1 is 0.041, which slightly less than Project 2 as 0.046. Here the productivity of the project two increased because of implementing more number of lines of code in less time, once again proving the programming in pair effect. The productivity of test case preparation is found somewhat similar in Project 1 and Project 2, still there very small amount increase is observed in Project 2 because of increase in change caused increase in number of test cases, as discussed earlier. The productivity of executing the test cases as 2.1 in Project 1 and 1.85 in Project 2. This results present the fact that Project 2 taken less time for executing the test cases than Project 1 even though the number of test cases are more, this is due to use of the automated tools support, which is allowed in XP procedure to build the test case.

The defect detection productivity was observed for Project 1 as 0.67 and for Project 2 as 0.72. The measurement results were pointing the fact that, programming in pair also effective in finding the defects and resolving the defects as explained in next category of results. SMXP projects while it is little bit more in Project 2 project. The reason behind it was, the defects found in

Project 1 were more and the effort spent is also more. While in Project 2 defects are less but effort spent are also less. The productivity of defect fixation is observed as 0.41 in Project 1and 0.44 in Project 2. These results was obtained due to number of defects and time taken to fix it were less in the Project 1 and more in the Project 2 comparatively.



In metrics related to quality aspects of project process, the quality cost measured in regards with the efforts. The results clearly presents that the Project 2 team took more efforts than Project 1 team. Here the Project 1 is observed to produce comparatively better quality. The cost of measurements of quality considers the effort in hour's invested in quality related aspects only, here the pair programming team again at the top in case of results of current experiment. The cost of poor quality, which is related to effort needs to be taken for rework activity was observed more in Project 1(7.01) than the Project 2(5.34). Here it was observed significantly less in Project 2 teams work than other Project 1. The reason behind this was the Project 2 team's maximum utilization of project tracking and monitoring tools prevents the maximum rework in the project. Also there was always presence of frequent reviewer in terms of second person in pair, which helps to the get the task right in first go, eliminating the rework.



The density of defect was observed little bit less in Project 2 (5.34) compare to Project 1 (6.56). As the number of lines of code produced more and defects found was less in in Project 2 these results were obtained. The research findings here also agreeing on the fact after project experiment that the pairs itself cautious about not to leave defects and in any case if it was found then it must be eliminated rapidly.

The efficiency of review was observed as 34.6 in Project 1 and 38.8 in Project 2. The results presents the fact the Project 2 reviews was conducted efficiently due to its enhanced review techniques with pairs. The efficiency of testing was observed as 56.6 in Project 1 and 60.13 in Project 2 as the number of test cases formed are more and executed in less time in Project 2. In both the project types the defects found in acceptance testing are comparatively low than the overall defects.

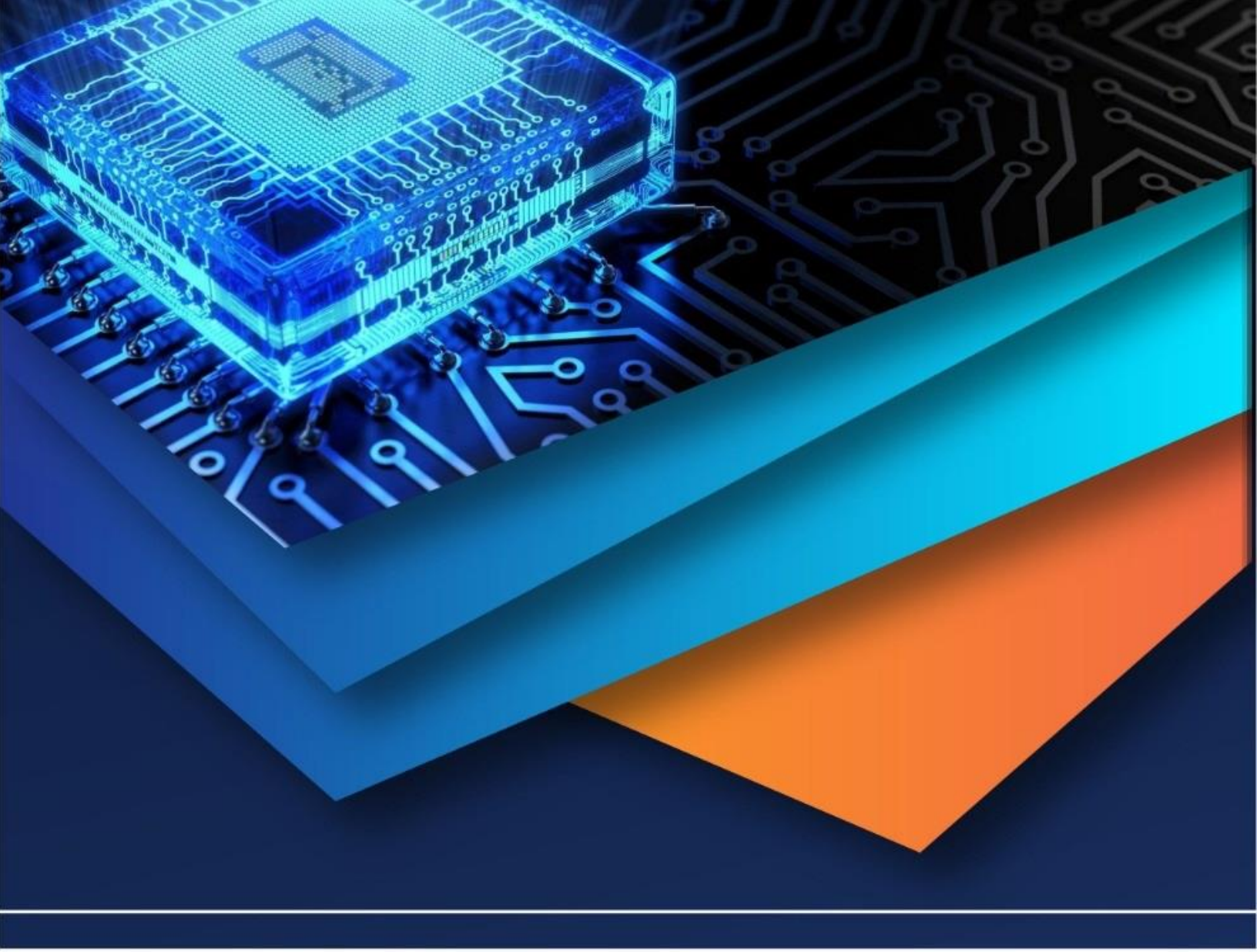
The efficiency in removing the defect quantifies the efficiency with which defects were detected and prevented from reaching the customer. In Project 1 it was observed as 62.5 and it was 57.14 in Project 2. On the first sight the results are seems to be better here in case of Project 1, but fact is that as the number of defects found were less in Project 2, so efficiency calculated presented the lower percentages. The density of residual defect observed as 27.4 in Project 1 project and 33.14 in Project 2. Here the Project 2 produced better results as defects found by the customer are less than the overall defects. This states that the procedure followed in Project 2 itself capable finding and fixing the defects well in time before the customer event though customer us always present.

VI. CONCLUSION

The findings of research experiment are clearly presents the facts that the programming in pair practice with extreme programming methodology is effective for building the Software's. The results presented in the research are based on valid measurement of several parameters which are essential for the software products. The several positive points presented by this research are contrary to the most of the previous results of the researchers. One can point out the boundaries of this research as there are small type of projects are implemented for short duration and that too with the students' participants. But the fact is that for small scale project it has produced very good results to support the efficiency of programming in pair. The training session conducted for one day, before this project experiments was very helpful to gain the better results and perform the experiment in disciplined manner.
However there is still further need to have an experimental research in regards to measure and prove effectiveness of programming in pair in industrial setup and with various size of projects.

REFERENCES

[1] Basili, V.R., Green, S., Laitenberger, O., Lanubile, F., Shull, F., Sonungard, F., and Zelkowitz, M.V, "The empirical investigation of perspective reading", Journal of Empirical Sofhvare Engineering, 1 (2), 133-164, 1996
[2] Beck. K., "Extreme programming explained: embrace change", Addison Wesley, 2000
[3] Laurie Williams et al., "Strengthening the Case for Pair-Programming", IEEE SOFTWARE, 2000, 0740-7459 © 2000 IEEE
[4] Trochirn, W.M.K., "The research methods knowledge base. http:lIw.trochim.human.wrnell.edu/kb/ index.html, 2000
[5] Dudziak. T., "Extreme Programming: an overview", Methoden und Werkzeuge der Sofiwareprodubion WS, 2000
[6] Johnson, B., and Christensen, L.B., "Educational research: quantitative, qualitative, and mixed approaches", (2nd Ed.). Allyn, & Bacon, 2003
[7] J E. Terry et al., "Student Perceptions of Problem Solving through a Pair Programming Technique", EDU-COM International Conference, 2006
[8] E. Mnkandla, "Online tools for Virtual Pair Programming", 11th Annual Conference on World Wide Web Applications, Port Elizabeth, Sepetmber 2009.
[9] Sanjay Goel, Vanshi Kathuria, "A Novel Approach for Collaborative Pair Programming", Journal of Information Technology Education, Volume 9, 2010, ISSN: 2165-316X
[10] Ilenia Fronza, Alberto Sillitti, Giancarlo Succi, and Jelena Vlasenko, "Analysing the Usage of Tools in Pair Programming Sessions", 12th International Conference, XP 2011, May 10-13, 2011, ISSN 1865-1348
[11] Laura Plonka, Judith Segal, Helen Sharp, and Janet van der Linden, "Collaboration in Pair Programming:Driving and Switching", 12th International Conference, XP 2011, May 10-13, 2011, e-ISSN 1865-1356
[12] Nattakarn Phaphoom, Alberto Sillitti, and Giancarlo Succi, "Pair Programming and Software Defects – An Industrial Case Study ", Springer, XP, 2011, e-ISSN 1865-1356
[13] Venkata Vinod Kumar Padmanabhuni, Hari Praveen Tadiparthi, Muralidhar Yanamadala, Sagar Madina, " Effective Pair Programming Practice- An Experimental Study", Journal of Emerging Trends in Computing and Information Sciences, VOL. 3, NO. 4, April 2012, ISSN 2079-8407
[14] Manisha Giri , "Predicting Performance in Pair Programming using Programmer Ranker Algorithm", International Journal of Advance Research in Computer Science and Management Studies, Volume 1, Issue 5, October 2013, ISSN: 2321-7782 (Online)
[15] http://www.extremeprogramming.org/rules/pair.html
[16] http://searchsoftwarequality.techtarget.com/definition/Pair-programming
[17] https://martinfowler.com/bliki/PairProgrammingMisconceptions.html.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY