ISSN 2277-3061
Volume 15 Number 04
INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY

Multi-factor analysis of pair programming based on PSP methodology

Nabiollah Bayatmoghadam[1] (M.S), Ashkan Sami (Ph.D.)[2]
[1]Department of Computer Science, Engineering, Shiraz University
Email: Pazhooam6@gmail.com
[2]Department of Computer Science, Engineering, Shiraz University

ABSTRACT

In regard with designing software, users play key role. In order to design software, it is necessary to observe standard principles of designation, using templates and using modern methods. Over the decades, using development methods, XP and one of the XP methodologies of paired programming used to design software. These methods have been designed for purpose of enhancing quality of product and rapidresponse to need of market and customer and overcoming weaknesses of traditional methods based on long-term programming and waterfall method. Therefore, every programmer and developer can pass a series of processes for constructing computer software. The processes can be changed daily and efficient processes maynot be effective and useful; although they can be considered as process. The main objective of the present study is multi-factor analysis of pair programming based on PSP methodology. Practical and analytical methodand two PSP methods have been applied for investigations.

Key words: software engineering, XP, pair programming, PSP methodology, multi-factor analysis

INTRODUCTION

In designing software, users play key role. In order to design software, it is necessary to observe standard principles of designation and using templates and using modern technologies. In this regard, many engineers and scholars have applied different models and methodologies such as waterfall model, spiral model, enhancing and iterative model for purpose of reducing costs and developing quality of a software product. However, with presence of positive effects of these models, because of lack of flexibility, low speed and high costs of development and lack of response to customer needs; mistakes and repetitive errors and finally, low quality of final product and some other problems were remained dissolved. In addition, some other scholars of software established a group named Agile Alliance, who claimed that there is better solution for production of software that can facilitate task of programmers. One of the approaches of this type of development was Xtreme Programming (XP) and the other one of the methodologies of XP was pair programming. The method caused paying attention to agile methods and especially XP and pair programming method of XP over the years in the US, Europe, Japan and other developed countries (Arisholm et al, 2007). This is because; these methodshave been designed for purpose of enhancing quality of products and rapid answering to needs of the market and customer and overcoming weaknesses of traditional methods based on plan-deriven and waterfall method over the decades. On the other hand, in order to reduce costs and overcome problems of human resource, outsourcing is also an emerging phenomenon that is being developed constantly, so that American companies and other companies in developed countries assign their activities and relevant jobs of information technology to the relatively developed and developing countries such as India, China, Russia, Philippine and so on.

In regard with using the modern methods and especially pair programming in the developed countries and especially in the USA since 2000 and according to presented literature, some studies have been conducted in the mainly academic environments. As the studies have considered only a single part of relevant factors of pair programming, it seems that simultaneous use of PSP method and scientific survey can make it possible to investigate several factors with each other at the same time. On the other hand, except for a case, similar studies have not been conducted in developing countries like Iran and it seems that specific consideration of such studies in these countries and providing suitable conditions for using modern technologies and development of software and encouraging use of modern technologies of development of software and culture making for team works can enhance level of software products and can also decrease production costs and provide also opportunities for attending global market and having competent proportion in this field.

WHAT IS SOFTWARE ENGINEERING AND FOR WHAT PURPOSE HAS BEEN DEVELOPED?

In general,engineering refers to art and skill of learning and applying sciences (mathematics, economics, etc.) for purpose of designing, producing or developing a secure product to promote life level of people and software engineering means designing and developing software based on certain principles and methods. In regard with development of software product, some issues should be noted as follows:

- Duration of software development
- Cost (money)
- Quality ofproduct

Along withdevelopment of technology and narrowing of competitions and loss of companies, which were developing software traditionally an empirically, need to create suitable methods in field of software development were established and finally, they caused emergence of software engineering.

Ma reasons for rapid growth of software engineering are presented as follows:

- Rapid development of hardware technologies
- Rapid development of software technologies
- Emergence of new needs and requirements
- Emergence of large and complicated systems

For example, one can refer to considerable emergence and growth of relevant technologies of web and internet, which brought their needs. In addition, some steps have been passed for development of software engineering as follows:

1. Feasibility study
2. System analysis
3. System deign
4. System implementation
5. System test and integration
6. System validation
7. System maintenance

In general,one may face in fact information systems in field of software engineering. In fact, system is a collection of components that have specific target. In order to analysis information systems, 5 components of information systems should be considered as follows:

1. A series of inputs
2. A series of outputs
3. A series of information files, in which information of system can be stored.
4. A series of processes that can present operations of the system.
5. Target

WHAT IS SOFTWARE DEVELOPMENT METHODOLOGY?

Software development methodology means method and framework of constructing, programming and controlling development steps of an information system in software engineer. Methodologies of software development have been emerged and developedsince 1960. In addition, individuals like EdsgerDijkstra, Bohm, Jacopiny, Nikolas Worth, David Parnas, Halan Di Millis and Larry Constantin have presented some methods and insights in field of preventing occurrence of error instead of solving it after occurrence, enhancing capability and power of programming, especially in large projects and preventing bad design. Some of these insights and theories are as follows:

- Functional analysis of problems like structured programming
- Data-based attitudes and data structure
- Official attitudes (mathematics) like Vienna Development Method (VDM)

Structured attitudes were more common than others.

In early 80s decade, many methodologies have been presented, which following them separately needs long time. In order to get information about them, one can refer to conducted studies in this field. By that time, some concepts were common as follows:

- Prototyping
- Real time system problems
- Software engineering using computers

Following in 80s decade, new concepts such as object-oriented methodologies and subjects such as inheritance, data abstraction and reuse and retrieval of factors were considered significantly. However, considering this issue is time consuming and is also out of framework of this thesis. Following, some traditional models of software development consider advanced and modern method of software development named Agile and then, consider XP methodology and pair programming.

WATERFALL MODEL

Figure 1 has illustrated waterfall model of development process including identification of needs (need analysis, feasibility study), design, implementation, test and system maintenance. Main weakness points of the model can be lack of flexibility, low speed and high costs of development and lack of rapid response to customer needs.

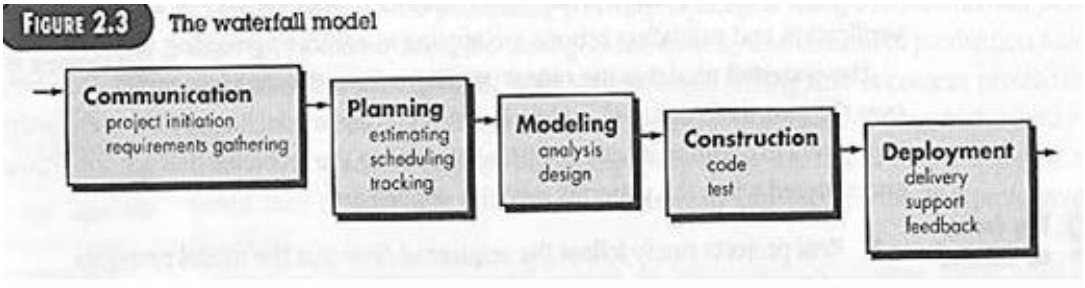

Figure 1: waterfall model (Pressman, 2001)

SPIRAL MODEL

As it is illustrated in figure 2, key feature of spiral model is risk management in all stages of software production cycle. Spiral model of the method that indicates following activities using 4 diagrams can depict iterative steps of performing the processes:

1. Codifying and formulating planning for identification of system targets, selected sections for program implementation, certain limitations of the project
2. Risk analysis and system hedges: analytical evaluation of selected programs for specifying how to detect and remove risks
3. Project implementation: implementing software production and system efficiency confirmation

Risk-oriented Spiral model emphasizes authority of choosing options and limitations in orders for supporting reuse of software and that software quality can help integration of special goals in producing software.

The first step is associated with codifying and formulating a program for purpose of achieving goals with these limitations and aft that, attempting to find and remove all potential risks through careful analysis and prototyping if necessary. If some risks are not solvable, customers should make decision that whether they want to end the project or neglect desired risks and continue in any way. Finally, the results would be evaluated and design of the next step would be stated. Main weakness points of this model can be rise of costs and reduction of interest for the developers for purpose of risk management.

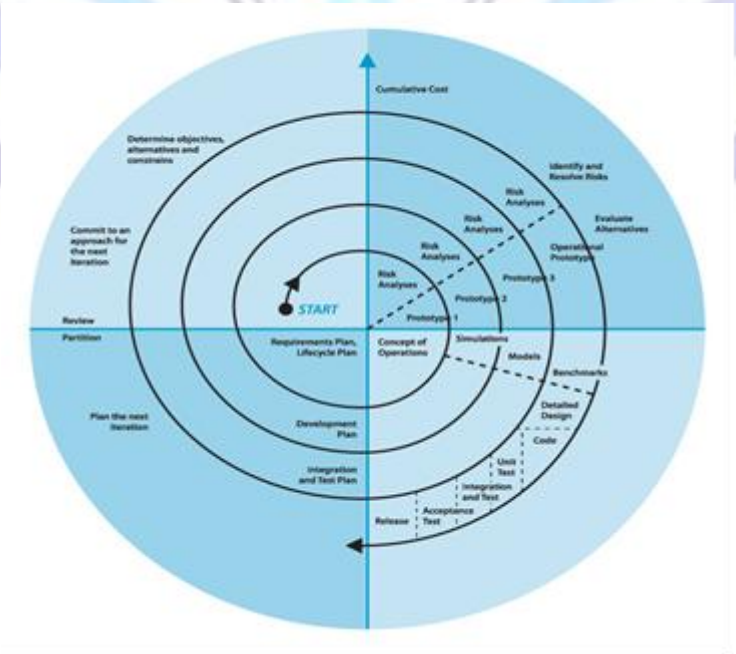

Figure 2: spiral model

ITERATIVE AND INCREMENTAL MODEL

As it is obvious in figure 3, the project would be started from small section and would be developed gradually. In this model, while development and before completion of development process, important difficulties would be removed and destruction of the system as a result if improper assumptions would be prevented. Iterative model can be applied by suppliers of commercial software, since the models allows to apply needs of users over the time, which are not clear exactly by the time of designation. The model has been considered as the basis for development method of Agile.

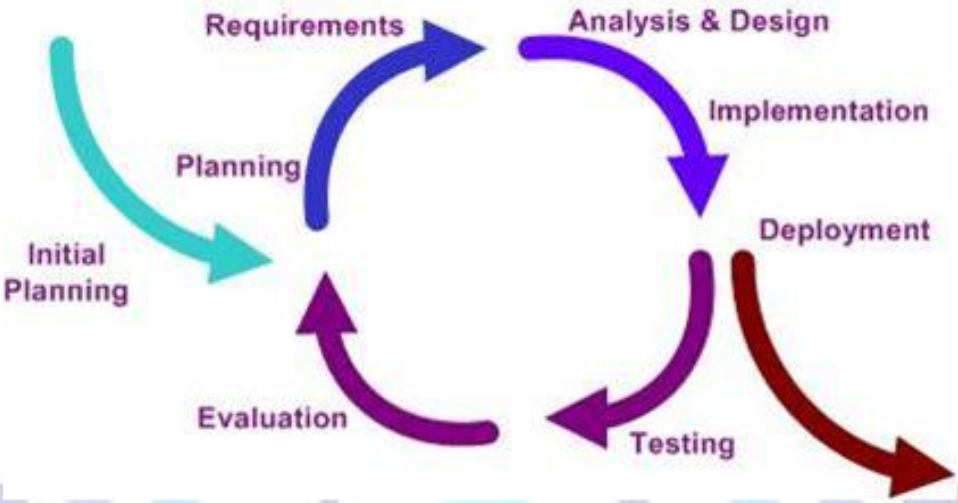

Figure 3: iterative and incremental model

METHODOLOGY OF AGILE DEVELOPMENT

Many people face software projects, which can be produced with no principle and it has been observed that it is hard to work on such projects. In these projects, main problems include inability of suppliers for detecting needs of users, presence of iterative events and errors, delay in supplying the product and so on. On the other hand, customers of such software have complaint for inattention of presenting exact schedule by designers of system, low quality of produced software and increase in costs.

In such projects, programmers consume a lot of times on producing software, which is full of problems and their efforts is not sufficiently effective. When one faces these problems, may think that a proper method should be applied, in which relevant activities of the projects can be specified; users' needs can be cleared and outputs of the software and project's products can be also produced successfully.

In early 2001, some scholars of software established a group named Agile Union, which could find a solution for software teams, so that they can produce software rapidly and with high quality and can also control changes in the software and apply required modifications.

It could be mentioned that in this method of producing software, iterative and incremental method has been applied as a basis; although in Agile method, the main mechanism of controlling the project is feedback instead of planning. Feedbacks would be produced by a series of tests and publication of developing software.

They have considered several important principles as manifest or declaration. Among these important principles, one can name priority of customer satisfaction through fast delivery of small, high-quality and usable versions; importance of role of team members in software project; production and use of suitable documents for software; importance of role of system users and using them in steps of producing software; and ability of applying changes in software in all steps of production.

In fact, the main objective of each programmer and all software teams is presenting the best service and product with high quality to the customers. Agile development method is a solution for achieving this goal. Approaches or methodologies of Agile Method include Scrum, adaptive software development and Extreme Programming (XP), which can be useful.

XP METHOD (VERY FAST METHOD)

One of the powerful and intelligent methodologies in Agile Software development is XP. It could be mentioned that XP is nothing other than observance of a series of relevant practices and principles. XP method has been constructed based on 5 key values as follows:

- Communication: it means that, creating constant, active, effective, informal and away from formalities is existed among technical team, representatives of customer and other beneficiaries.
- Simplicity: XP encourages development team for designing for the current needs and avoiding complexity instead of focusing on far future needs.

- Feedback: the aim of feedback is the feedback of user, other team members and the software.
- Courage: having three above mentioned factors can practically provide required courage for accepting changes and fighting problems in all steps of development.
- Respect: through observing all of the mentioned values, practically a respectful space with high ability to accept responsibilities would be created among all team members and especially users.

XP methodology has been designed for purpose of fast and high-quality development of small products in order to fight unexpected changes and meet needs and new applications that can be performed in the next steps. Hence, the method can constantly adjust the project due to the conditions using its 5 key values. However, in the groups using other methods than XP, these new applications and inputs would be neglected because of this issue that project framework has been determined previously and these changes can be applied easily or they would be handled as a new series of information but in the same framework of total pattern of the project. In XP, team members can observe completion process of software production and follow the process. XP is a suitable method for small projects, in which team members have been formed of 2-12 programmers, who work in pair form. In this method of development, software product would be produced during 4 main steps as follows.

- Planning step
- Initial design
- Program coding
- Testing codes

In XP, programmers use pair programming method and are constantly writing and revising codes. In this method, two programmers can work on a single project, function or form and criticize or challenge work of each other. In XP, customer or user is a member of development team and is completely informed of process of problems. User and other team members can record gradually and with cooperation of each other needs and necessities of the product in form of a series of prioritized user stories on some papers. These papers can case remembering initial needs and facility of time and cost estimation by the team members. One of the tasks of the customer or user is testing different versions of the system and reporting results and outputs constantly to the development team.

Figure 4 has illustrated process of software development using XP method. Details and explanations of the XP method have been presented in all references such as Pressman (2001). However, it could be mentioned on figure 4 that final product can't be prepared in one step, but also the team would deliver some versions for use and reporting feedback in short iterations of 3 weeks to the customers. Interestingly, before delivery of each version or iteration to the customers, acceptance tests would be performed based on user stories. The tests would be implemented once more when a new part has been added to the program and the program is being compiled. Hence, if there is a problem, the error message would be sent. As a result, whenthe system was completed, it could be ensured that the system is without error. Another important issue is that in XP, all codes would be produced in form of test-based development; meaning that code test would be written before writing program code and programmers produce program codes based on following results and outputs of the test, so that they can remove possibility of any kind of error or defect. After implementing and confirming all iterations, final version would be created. At the end, the final version should be tested and confirmed by the user. There is no doubt that considering flexibility and dynamicity of the methodology, probable errors and difficulties would be removed constantly in each step.

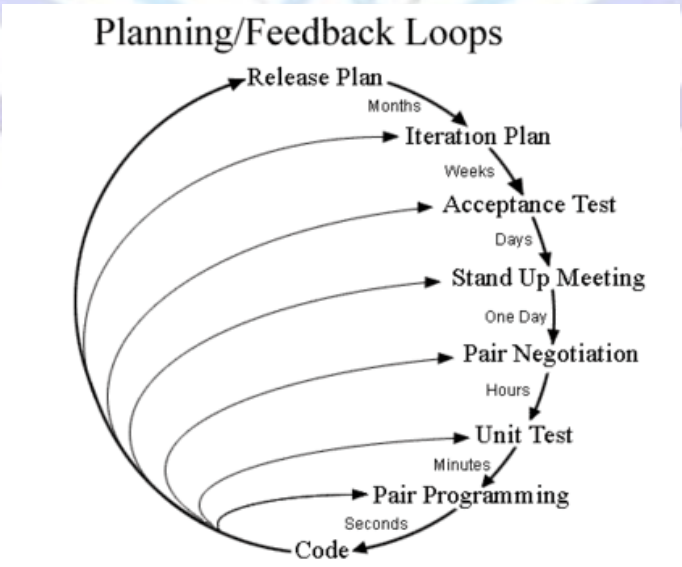

Figure 4: software development process in XP model

PAIR PROGRAMMING

Programmers in XP field would be divided to 2-person groups, which every group is responsible for completing a specific part of the program. Both programmers discuss on needs of users and prepare classes of the program step by step. Hence, firstly a class would be created in elementary mode and with no initial design and the class would be tested. If the class has no problem, original code of the program would be selected based on the class test. When one of the programmers is writing a section of the program, the other programmer is responsible for controlling accuracy of these codes and if there is a problem, the composer of code would be informed. In fact, in pair programming, typically two persons would work on a single computer and compose a program. One of the programmers is responsible for implementing code and has the keyboard practically. The other member is focused on algorithm or solution that should be implemented in form of program and maintains it in the mind or on the paper. Actually, one person is involved in high level and logic level and program algorithm and the other one is involved in low level and in level of programming and coding language. Two persons have cooperation and conversation for composing a program and every person has special tasks.

A REVIEW ON PERSONAL SOFTWARE PROCESS (PSP)

Every programmer or developer should pass a series of processes in order to construct computer software. The processes can be changed day by day and the processes may not be even efficient and useful; although they can be considered as process. In order to avoid such waste processes and for purpose of developing high-quality and errorless software based on scheduling, Watts Humphrey has presented PSP methodology, in which as it is illustrated in figure 5 a team member or programmer should act based on predetermined scripts and can record different logs such as errors, type of errors, time of error occurrence and time of removing errors in specific forms and present summary of forms in a project plan summary in phases of planning, design, design review, coding, review coding, compile, test and final assessment as it is obvious in figure 5 (Pressman, 2001).

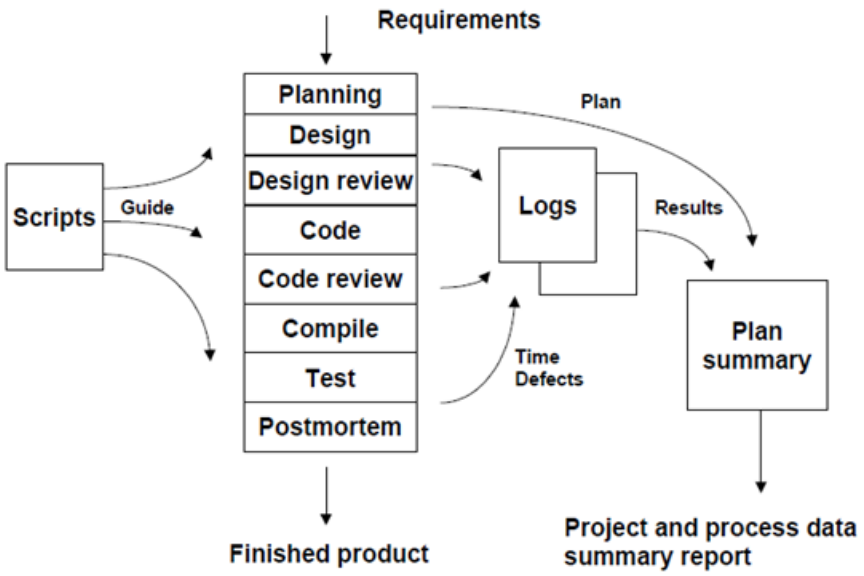

Figure 1: PSP Process Flow

Figure 5: schematic of processes of PSP (Humphrey, 2000)

In this regard, according to the explanations, scholars have conducted different studies in scope of this study, which would be investigated here. By 1998, Nosek compared difference of team programming in a manner that two programmers work together on a single code and design. For this purpose, an X-Windows system and C programming language have been applied. In this test, participants were asked to write output errors in a file and if there was no error, the program should compose in the file that no error found. No one of the participants had experience of composing such program. Members of groups were selected randomly and 45min time was considered for the participants to fulfill the program. For statistical calculations of the program, T-test was applied. Obtained results indicated that in pair programming, created code is more efficient and programmers would enjoy their design.

By 2001, Jerzy Nawrocki and Adam Wojciechowski have conducted a study under the title of experimental evaluation of pair programming. In the paper they compared pair programming with two variants of individual programming: one of them is based on Personal Software Process that has been proposed by W. Humphrey, and the other is a variant of extreme Programming tailored to individuals. Four experiments are described that has been performed at the Poznan University of

Technology. During those experiments 21 students wrote 4 C/C++ programs ranging from 150 to 400 LOC. Considered factors in this test have been development time and number of errors. This test was conducted in the university and on students in 4th year of Computer Major. All students had similar conditions in programming and they found based on obtained data that according to the development time and software size, pair programming can be more predictable and amount of rework is low in pair programming. However, it should be noted that this test was conducted on programs with small size.

Mendes et al (2005) have also investigated pair-programming in a 2nd-year software development and design computer science course. This paper presents the results of a pair programming experiment conducted at the University of Auckland (NZ) during the first semester of 2004. It involved 300 second year Computer Science students attending a software design and construction course. In order to investigate obtained data, Chi-Square test has been applied. According to the obtained results, using pair programming is significantly effective in raising learning level and students enjoyed pair working and believed that pair programming is more interesting than single programming.

Williams et al (2006) have also investigated impacts of pair programming on learning in the classroom and presented 11 methods for improving use of this method. According to obtained results, pair programming can provide a suitable environment for learning and can enhance self-confidence of the students.

Sison et al (2008) have investigated Pair Programming in a Software Engineering Course in an Asian Setting. Sison has investigated impact of pair programming using two factors of defect density and loc/hour factor and using 24 students in one of the universities of Philippine. Obtained results indicated that in pair programming, defect density can be declined significantly compared to single programming; although loc/hour of code would not be changed considerably. In this paper, it has been found that such test has not been conducted yet in the developing countries.

Salleh and Mendes (2009) have conducted a study based on impact of personality in pir programming. In a test, Salleh and some others in The University of Auckland found that dir to the different personalities and difference in personality of members of a group, again pair programming can be significantly effective in learning and can enhance its quality.

Salleh and Mendes (2009) have conducted another study in this filed. This paper describes two experiments involving the development of small systems and very small programs by student programmers in an Asian university. The results involving the small systems, which were actually complete systems designed to support the Personal Software Process (PSP), showed that defect densities of systems written by the pair programming teams were significantly lower than those written by the teams that used the traditional approach of individual coding and testing of units, followed by integration testing. On the other hand, results involving the very small (and therefore much less complex) programs did not show any significant differences between the defect densities of the programs written by the pair programmers and of those written by the solo programmers, though they did show significantly greater productivity of the solo programmers when writing simpler code. The combined results of the two experiments suggest that pair programming might increase software quality without decreasing productivity when projects are sufficiently large (or complex) for the programmers working on them.

METHODOLOGY

At the present study, the main objective is using two practical and analytical methods and using two methods of PSP and survey method to test wide range of qualitative and quantitative factors of software development using pair programming in Iran as a developing country. In the empirical test of the study, factors of quality and software development time have been investigated separately in phases of software development including analysis and design, coding, code review and test phase. In the analytical method, creating responsibility against the project, more focus on the project, amount of knowledge transfer among pairs, enjoying software development in pair programming was compared to single programming. In addition, factors such as cultural difference between pairs, skill adaptation between pairs, different gender of pairs and previous experience of pairs in programming have been also investigated. In addition, data collection has been conducted using PSP method. PSP dashboard software has been also applied for purpose of arrangement and maintenance of collected data. Practical test has been conducted in engineering university of Shiraz on 36 students of third year in Software Engineering Major and the survey has been also conducted in Iran.

Statistical population

Statistical population of this study includes students of third year of software engineering major in Shiraz University.

Statistical sample

Statistical sample of the study includes 36 students of third year of Software Engineering major in Shiraz University. Data collection method has been conducted using questionnaire. The questionnaire included two sections. At the first section, a questionnaire has been presented in regard with effects of pair programming on presented factors and at the second section, a questionnaire has been provided in regard with investigating effective factors in pair programming.

RESULTS

Obtained results from investigating common factors of practical and analytical test, except for designing phase, have been same. If the main index has been considered practical test, results indicate positive effect of pair programming on design phase. In general, results of the study are as follows:

- Pair programming can enhance quality of produced software and this can confirm obtained results from studies of Nosek (1998) and Sison (2008) and Salleh and Mendes (2009).
- Pair programming has no significant impact in productivity of development team and this can confirm obtained results by Sison (2008) and Salleh and Mendes (2009).
- Pair programming can provide useful results for knowledge transfer level between couples and enjoying software development compared to single programming. This can confirm obtained results from studies of Nosek (1998); Mendes (2005), Kelly (1992) and Salleh and Mendes (2009).
- Pair programming can enhance responsibility against the project and can cause more focus on the project. The test has not been conducted by the previous studies.
- Some factors such as cultural difference of couples, adaptation of skill of couples, different genders of couples and experience of couples can have positive effect on pair programming. The test has not been conducted by previous studies.
- Pair programming can't provide considerable positive effects in designing and testing phases. The test has not been conducted by previous studies.
- Pair programming can create no significantly positive effect in designing and testing phases. The test has not been conducted by previous studies.
- According to variety of qualitative and quantitative factors involved in productivity, it is impossible to express certain idea in this field.
- Among obtained results of the study, it could be mentioned that when pressure of the market for rapid production of high-quality is considered, pair programming can be significantly effective. However, it can raise costs to some extent based on the results.
- For purpose of cost reduction, it would be better not to use pair programming in phases of designing and testing the software and mainly coding and code review phases can be applied for this purpose.
- Although based on results of the study pair programming acts good in preventing creation of syntax defect compared to checking and function errors, it seems that this issue can't be significantly effective.

CONCLUSION

According to the mentioned and through studying previous studies, it could be found that a part of relevant factors of pair programming and simultaneous use of PSP and survey method can provide conditions for assessment of several factors at the same time. In fact, when pressure of the market for rapid and high- quality product is considered, pair programming can be significantly effective. On the other hand, XP method is a suitable method for fast production of high-quality products for purpose of fighting unexpected changes and meeting needs and new applications. This method is a suitable method for small projects, in which team members have been formed of 2-12 programmers working in pair groups. As a result, for purpose of avoiding waste processes and for purpose of developing high-quality and errorless software, PSP has been recognized as the most adequate way for preventing these factors.

REFERENCES

1. E. Arisholm, H. Gallis, T. Dyba, and D. Sjoberg. 2007. "Evaluating Pair Programming with Respect to System Complexity and Programmer Expertise," IEEE Trans. on Software Engineering, vol. 33, no. 2, pages: 65-86
2. E. Mendes, L. Al-Fakhri, and A. Luxton-Reilly.2005. "Investigating Pair-Programming in a 2nd-year Software Development and Design Computer Science Course," Proc. ITiCSE, Monte de Caparica, Portuga, pages: 296-300
3. J. Nawrocki and A. Wojciechowski.2001. "Experimental Evaluation of Pair Programming," Proc. European SoftwareControl and Metrics Conf. London, England, (ESCOM)
4. J. Nosek, 1998."The Case for Collaborative Programming," Comm.ACM, vol. 41, no. 3, pages: 105-108,.
5. J.C ,Kelly, J.S. Sherif, and Hops, J. "An Analysis of Defect Densities Found During Software Inspections," Journal, Systems and Software, Vol. 17, No. 2, Feb. 1992, pp.111-117.
6. L. Williams, D. McCrickard, L. Layman, and K. Hussein, "Eleven Guidelines for Implementing Pair Programming in the Classroom," Agile 2006 Conference, Toronto Canada, pages: 445-459, august 2006.
7. N. Sallehand E. Mendes.2009/ "An Empirical Study of the Effects of Personality in Pair Programming using the Five-Factor Model", Third International Symposium on Empirical Software Engineering and Measurement, Florida USA, pages: 214-225
8. N. Salleh and E. Mendes.2009. "Investigating the Effect of Pair Programming and Software Size on Software Quality and Programmer Productivity" APSEC, Penang Malaysia, pages: 187- 193
9. R.S.Pressman.2001. Software Engineering: A practitioner's Approach. McGraw-Hill Science

10. Sison.2008. “Investigating Pair Programming in a Software Engineering Course in an Asian Setting,” Proc. APSEC, Beijing china, pages: 325-331, December
11. W. Humphrey.2000.”The Personal Software Process”, CMU/SEI-2000-TR-022, ESC-TR-2000-022.