ABSTRACT

In a software development organization, where the traditional way of developing a software program is adopted, a programmer will use a computer terminal for writing software programs, which are necessary for a project. In this approach, the programmer has to consult a systems analyst for implementing the system requirements in his/her software program. This process is time consuming, may lead to miscommunication and thus results in low user satisfaction and low productivity. Also such traditional approaches involve large amounts of documentation like requirements specifications, architecture document, design document and test plans etc., instead of giving out useful functionality to the end user. Due to such user unfriendly approaches sometimes projects are abandoned even before it is deployed.

The solution for such type of problems like project abandonment before deployment, low user satisfaction and low productivity is provided by using Agile software development methods like Pair Programming, Distributed Pair Programming and Extreme Programming. When a single programmer uses the system, many tools are not required for synchronizing the activities, but it is required in the case of Pair Programming or Extreme Programming. The tools such as to replicate a user’s desktop onto multiple computers in particular two in the case of pair programming are required. All input and output methods should be shared between multiple computers and the application to be developed should also be deployed on both or multiple computers. It should be noticed that direct communication is better than a detailed documentation. However, documentation is also important in the software development. In pair programming, the limitations are scalability as well as co-located pairs in the same physical location. But, due to the advances of internet and social networking, one can foresee an approach

which uses the advantages of such technologies. Hence, there is a need to address distributed pair programming where there is a possibility of scalability as well as the users need not be co-located in the same physical location.

Distributed pair programming is a practice where two pairs are geographically separated and are working for the same problem. High quality products are produced more rapidly. Remote pair programming, also known as virtual pair programming or distributed pair programming, is a pair programming in which the two programmers are in different locations, working via a collaborative real-time editor, shared desktop or a remote pair programming IDE plugin. But remote pairing has difficulties like extra delays for coordination, loss of verbal communication resulting in confusion and ego conflicts. Software tools may be necessary for screen sharing and for audio chatting through the use of headsets which will be of very useful in distributed pair programming.

In the first part, the effectiveness of the distributed pair programming was analysed using an experimental technique in an academia environment for laboratory courses. This experiment demonstrates that distributed pair programming could be very effective in promoting a student's ability to learn programming concepts in laboratory courses in a faster and more efficient way when compared to solo programming. This experiment was conducted in the computer laboratories in an engineering college. The solutions given by pairs in distributed environment are analysed and shown using various charts. The results provide support for distributed pair programming in the software engineering curriculum of an academia. The results proved that distributed pair programming is better than compared with solo programming in helping a student to learn the programming concepts in the laboratory, easily and efficiently.

Traditionally, pairs are formed based on individual preferences or administrative authority’s decision to support organization requirements as there are no standard procedures for forming pairs. A performance will be highly productive, if they are more compatible with each other. Incompatible pairs have less understanding and have communication gap between them. This leads to reduced performance, demotivation and disengagement from work. Then, there is a need for finding compatible pairs. In the second part, a novel method is proposed to form student pairs for programming laboratory courses based on weighted graph matching technique incorporating necessary psychological factors for compatibility between pairs. The concept of graph matching is used in many industrial applications. For instance, the assignment of individual workers to tasks, jobs to processors, etc can be modelled using graph matching. The experimental results demonstrate that the proposed method yields better performance of the pairs in the task assigned to them.

Laboratory courses constitute one of the core competencies that graduates from computer science discipline are expected to possess. Research has suggested that the lack of a formalised structure for laboratory courses may be one of the factors responsible for learners’ negative impressions of E-learning. In order to motivate E-learners and present laboratory courses as an easy and attractive challenge, pair programming was used as an effective tool. In the third part, experiments were conducted to analyse whether pair programming can enhance the E-learning system and thereby encourage the E-learners by motivating their E-learning experience. The results show that for the students having no programming background, they gain maximum learning experience from the laboratory work. Most students reported a high learning experience and satisfaction level when E-learning was employed with pair programming.

In the fourth part, a pair recommender system is proposed based on association rule. Association rules are used in data mining to discover interesting relations. In pair programming experiment, association rules are used to discover pair compatibility. Pair compatibility based on previous successful projects, skill levels, designation, personal interests are also identified. Then association rules can be built and also they can be analysed whether they are strongly based on threshold values. To generate strong association rules Apriori algorithm was used. Experimental results show that the proposed approach is better than solo programming and pair programming based on weighted graph approach.

The experiments conducted in the academia environment to evaluate the performance of distributed pair programming proved that distributed pair programming is a better approach than compared with solo programming in improving the ability of student community to learn computer programming without any difficulty and students were also motivated during the distributed pair programming which improved the quality of software products delivered by them. The experiments conducted to find the pair compatibility found to be successful in forming better pairs to produce a better quality software.