Review On Extreme Programming–XP

Conference Paper · April 2019

CITATIONS
0

READS
1,254

1 author:

Maleeha Arif Yasvi
Indraprastha Institute of Information Technology
3 PUBLICATIONS **0** CITATIONS

SEE PROFILE

Review On Extreme Programming-XP

Khusbhu Sahendrasingh Yadav
Department Of CSE
IIITD
Delhi,Indiai
yadav18087@iiitd.ac.in

Maleeha Arif Yasvi
Department Of CSE
IIITD
Delhi,India
maleeha18112@iiitd.ac.in

Shubhika
Department Of CSE
IIITD
Delhi,India
shubhika18085@iiitd.ac.in

***Abstract*—Extreme programming is an iterative software development methodology which aims to produce higher quality software and helps in providing an optimal solution. Extreme Programming differs from other software development methodologies as it focuses more on adaptability and responsiveness to the changing customer requirements. By using extreme programming as a software development methodology, better results have been obtained in software development.**

Keywords—Extreme Programming, Testing, Pair Programming, Refactoring, Small releases

I. Introduction

Extreme Programming is a software development model which primarily focuses on the software quality improvements and responsiveness to the changing customer requirements. It is a type of agile software development model. It advocates frequent releases in short development cycles. These releases are intended to improve the productivity and enhance the quality of the software by introducing certain measures that focus on initiating certain checkpoints at which the customer requirements can be adopted and met.

Extreme Programming focuses on light-weight processes. The main phases involved in the cycle of XP are- Planning, Design, Coding, Testing [2]. As it is an iterative model, the system is developed by dividing the overall project into small functions. The cycle of development from the design to the test phase is performed for one function. After executing for one function and debugging it properly, the developers then shift to the next [2].

XP is based on rapid release cycles and continuous communication between the developers and the stakeholders; that is, the customers [4]. It strongly relies on oral communication, frequent testing, code review and designing [5]. Communication is regarded as an important criteria in XP. The communication should frequently happen among the concerned parties like the developers, customers, and managers [6].

II. Background

XP was created by Kent Beck during his work on Chrysler Comprehensive Compensation system (C3) payroll project. Kent Beck became C3 project leader in March 1996 and refined the development methodology used in the project and wrote a book on the Extreme Programming methodology in 1999.

XP is regarded to be better than the traditional waterfall model. It is increasingly adopted by companies worldwide for software development. Since in the waterfall model, the development process is completed on a single project and the requirements have to be known beforehand, so there is no scope of changing the requirements once the project development starts. So the waterfall model is not flexible to the changing requirement needs. On the other hand, XP focuses on iterations and allows for changing requirements even after the initial planning has been completed [2]. Also, the waterfall model does not require the participation of customers while as XP focuses on customer participation and satisfaction and regards it as a primary goal during software development process. So, XP is a preferred model in the software development these days.

XP is a framework of the agile software development model, and it primarily aims at producing higher quality software. Agile software development advocates adaptive planning, evolutionary development, continuous improvement and encourages rapid and flexible response to changes. It focuses on customer satisfaction, simplicity and continuous attention from the developers as well as the customers. XP is the framework of the agile software development and hence all these practices are implemented in XP.

III. Traditional vs XP

The Waterfall model, often termed as a linear sequential development model has been used widely in the last few decades whereas extreme programming (XP) has shown its popularity in the recent past. There was a need to analyze both these methodologies to choose one that serves as the best fit for any software development process. The factors upon which decision lies are time constraint and risk mitigation and adaptability to the requirement changes during any phase of software development.

The Waterfall methodology comprises of five phases including requirement gathering and specifications, design phase, implementation phase testing and deployment phase. As stated in [19] and [20], waterfall methodology spent most of its time in documentation i.e. during the first phase, and implementation phase also shows a significant amount of time consumption whereas extreme programming methodology devotes maximum time in the testing phase.

Extreme Programming (XP) being iterative in nature, proposes short releases periodically. It follows an automated testing strategy that prevents risk propagation in the later stages. To avoid any kind of risk, XP performs continuous checks in each phase and resolves it in that particular stage if found any. In the case of waterfall methodology, if the requirements are misunderstood in the initial phase then it may lead to the development of a faulty system as it lacks communication process throughout the development. As waterfall tries to find errors in the later

phase, risk propagation in the upcoming phases increases which eventually end up costing high amount to fix the risks.

Waterfall model is sequential. Hence there is no way one could backtrack to the previous stage. Hence it causes a problem in making corrections. It is not possible to adapt to the changing business requirements in the later stages. In Extreme Programming, several checkpoints are being introduced which could be used for adapting the new customer requirements. However, adapting frequent requirement changes late in the development process may cause a delay in the completion of the project. It may also result in increasing the complexity of the project.

XP model helps to deliver fast with minimum risk exposure and is well suited for small sized software project whereas waterfall serves as the foundation for a large scale software project.

IV. Keys to Success In Extreme Programming

- Management Backing: Management encourages the team to try different scenario related to the size of teams, duration of iterations.
- Coach Role: Technical lead and Project Lead on the project, guide and push the teams to follow the process to which team has committed.
- Simple Coding: XP works best when the simplest solution is implemented for the current iteration and refactoring is done after each iteration when stories require it.
- Adaptation: If the process is the same as one year before, then the process is not agile as it was a year ago. So, adaptation is an important step in the success of XP.
- Team Coordination: The coordination between the members of a team plays a vital role in the success of an XP project.
- Tester Roles: Testers are a critical part of the success of an XP project, while their role may look different on an XP team, make them part of the team.
- Test First: Testing saves developers from coding the same model again and again. Testing after almost every iteration shows the flaws in the system and hence provide a means to achieve good quality software.

A healthy XP project can run f long time, but you need to change and adapt along the way. Change is good and necessary, including changing the process along way.[14]

V. Practices

XP has 12 basic practices which are always implemented. These practices ensure better code readability and understandability and also help in enhancing communication between the developer and the customers so that a better product is made.

The 12 practices of XP are listed as:-

1. Planning Game
2. Small Release
3. Metaphor
4. Testing
5. Refactoring
6. Pair Programming
7. Collective Ownership
8. 40-hour week
9. On-site customer
10. Coding Standards

Planning Game: This involves the developmental strategy of releasing plans for the project and the meetings between the developers and the customers. The strategies to improve communication among the stakeholders are also focused upon in this practice of XP. The system releases plans and dates for the meetings and the project reviews [1]. General discussions to know about the progress of work are held through XP. Discussions among the stakeholders ensure that the queries are resolved and both the developer as well as the customer has a proper understanding of the system and the project flow [3].

Small Release:- The entire project is distributed into functions, and after completion of each function, the development team releases its version to the customers. The release cycles in XP are shortened to speed up the feedback from the customer. It is done to handle the risks and errors according to the release of each version [3]. It ensures better accountability and efficiency. After the release of each function, it is integrated with the previously released functions. So, continuous integration is done [5].

Metaphor:- It describes how the program looks. It is a document that describes the working of the system and expresses the evolving project vision that would define the system scope and purpose. The metaphors are directly derived from the principle and standards of the project architecture and requirements [4].

Testing:- XP focuses on frequent testing. Testing is done to ensure that the code is free from errors. Unit testing is performed in which every piece of code that is written of a particular function is tested before moving on to the next function. Apart from the testing involving rectification of the coding errors, acceptance testing is also done. Acceptance test verifies the requirements as understood by the developer and whether it meets the requirements of the user or not. Integration testing is also done which helps in testing the system when different functions of the system are integrated into one unit [1]. The testers and developers work together to find out faults. Therefore, in XP the tests are initially run on small codes and then the whole system [3]. In some of the software applications, J-Unit is used for testing. It allows testing localization classes, entire package or even entire project [8].

Refactoring:- It is a way by which the code is kept in an easy to understand form. It ensures the removal of duplicate code and makes the code easy to understand. The use of complex coding schemes is not appreciated in XP and stress is given on forming an easy code [3]. Long methods, use of

unnecessary classes are avoided [5]. In this phase, restructuring is implemented. It is the change made to the internal structure of the software to make it easier to understand. Refactoring helps to modify the structure without changing its observable behavior [1].

Pair Programming:- It is the concept of having two programmers work together for a particular function code. One programmer writes the code while as the other is the observer who reviews the code written by the programmer [1]. It ensures better efficiency and helps in determining more alternatives [5].

As Extreme Programming focuses on pair programming for implementing the user stories, it gives rise to observe the personality traits which will impact the development process. It is being observed that the efficiency of pair programming depends on whether both the persons are working remotely or at the same location. It has been checked by a well-known personality test model namely Myer-Briggs Indicator which takes into account following traits: extraversion/ introversion (how people get influenced by surrounding), sensing/intuition, thinking/feeling (how an individual takes a decision) and judging/perceiving (organizing plan of developers). The final results stated in [23] showed that overall efficiency reduces if two different personalities are working together on a specific module at a foreign location. On the other hand programmers with similar traits tend to understand and communicate easily regardless of the location thereby leading to enhance the efficiency.

Collective Ownership:- It is a practice in XP which ensures that all the team members should be familiar with the project code. This practice enables any programmer to change the code in the system at any time. It works like open-source programming [4].

40-Hour Work:- It is the practice in XP which ensures that the team members in a project should work the hours that they can sustain quality [1]. The important thing is to recognize the time agreeable by all the team members for the number of hours that have to put in for a week. After every week, the work done will be reviewed [3].

On-Site Customer:- To ensure that the developers have a proper understanding of the desired project and all the requirements are being met, a single customer from the customer team is always available to answer all the questions of the developers, resolve disputes and set small scale priorities all the time [2].

Coding Standards:- For ensuring collective ownership so that any programmer can change the code in the system at any time, the practice of collective ownership is implemented [6]. XP mandates the use of coding standards. The programmers must follow a common coding standard so that all the code appears to have been written by one person [4].

VI. Life-cycle Of XP

The iterative life-cycle followed in XP goes through the analysis, planning, designing, implementation and the delivery phase [1]. All these phases are executed following the practices of XP. Initially, stories are created that give the estimate, release plan, iteration duration, etc. In the analysis phase, the stories are analyzed by the developer to see if the implementation can be carried through them or not. The output of the analysis phase is the feasibility report. For forming the stories, brainstorming sessions are encouraged among the stakeholders [2]. Value graphs which help to answer questions such as 'why', 'how' are also followed. Value graph is a tool used for finding value and requirement functions [2].

Followed by the analysis phase is the planning phase in which the strategy is planned. It is done according to the planning game practice of XP. Failed stories will be analyzed in this phase and new stories will be created to overcome the failed stories.

After the analysis phase, designing and implementation phase is executed which ensures extensive testing. Pair programming is followed while coding.

The last stage of the life-cycle is the delivery stage. This phase includes the activities like the installation of the software, training of the customer to accustom with the enhancements or changes [1].

VII. Applications Of XP

XP practices are widely used in the software development process.

XP practices are used in the development of web-based applications [7]. This is done to ensure software organization responsiveness while decreasing the developmental overhead [7].

By ensuring frequent meetings between the customer and the developer, XP has also been helpful in global software development [6]. When customers and developers are not co-related, then the feedback between them is not timely enough which may become an obstacle. So, XP practices when adopted result in a shortcut communication channel [6].

XP implements reverse engineering practices while developing the software model [5]. The developers comprehend the source code of the old system and deliver its features to the customer. The customer validates the specification set and the developers again revise the features and start building the system according to the XP practices [6]. In the designing of some software models, there are separate developers for reverse engineering phase and for implementation that is the forward engineering phase, separate developers are present. Reverse engineering group shares documents and specifications with the forward engineering group and consultation between the two groups happen [6].

XP practices have also been implemented in university projects among the students [8]. In some of the projects, JCVS (Java Concurrent Version System) tool is used for providing a standard for coding and providing collective ownership. The JCVS tool leads to a more consistent and relaxed method of coding. By containing all code in one server and accessing the server for the most recent version, all programmers can get to know the recent release.

For maintaining the documents of the project system and extracting stories, automatic generation tools are used which record the insights during the analysis phase and then the grouping of similar ideas is done to derive a proper description of the system [2].

The customers have ample knowledge regarding product requirements and background. The bidirectional communication between customers and the development team is essential as it allows developers to gain adequate insight regarding the requirements of the project so that they can develop a high-quality product. Hence XP communication is divided as internal communication among the project members, communication between the customers and communication between developers and customers been discussed in [22].

The XP bidirectional communication module involves formal (includes conferences) as well as informal communication (which includes team discussion, customer discussion). But sometimes customers may specify requirements in abrupt manner i.e, there could be a case when they specify the requirements in a fuzzy manner leading to requirement deficient condition or sometimes may over expect from the developers resulting in providing excessive requirements which in real sense would not be possible to satisfy in the given time frame of the project. This problem could be solved by introducing the two dimensional characteristic model in XP that analyzes the satisfactory situation of customers leading to the developments of a high-quality product with the best features. It can be achieved by comparing the customer's requirements and developer needs. The customer requirements could further be classified as Expectation, No Opinion, Bear and No Way, whereas the developer needs could be classified as Expectation, Possible, Reluctantly and Refuse. Both the sides then reach a consensus to fix the requirements that need develop and thereby eliminating the undesirable needs.

The demand modules of XP serves as a base to improve the product quality and increasing the customer awareness of the project along with reducing the risk factor involved in the project development. The demand module comprises of concept stage, prospect stage, requirement stage high-quality requirement stage and tool stage. The concept stage performs preliminary analysis to identify the main objective of the project eliminating the problem of deficient requirement. In the prospect stage simple planning is done using all the key records to get the characteristics of the product thereby designing the work process and managing the flood requirements. The requirement stage focuses on combining the use cases modeled by developers and stories being prepared by customers and analyze the data to avoid frequent changes. In the high requirement stage where both customers requirement and developer need to meet a consensus followed by tool stage, where defects are being tracked and version control is looked upon. This leads to enforce a better insight for the developer to understand the requirements and functionality clearly.

The efficiency of the various software methodologies could be enhanced by incorporating XP principles within their life cycle. One of them includes the Rational Unified Process (RUP) which is a process oriented development methodology that focuses more over on the process control and spending a significant amount of time to meet the process requirements. In contrast to XP, it does not pay much attention to the developing program and improving its agility. Hence RUP could be improvised by adopting agility feature of XP as mentioned in [24] and XP could incorporate certain RUP principles to strengthen the theoretical foundation.

The requirement changes often lead to a considerable amount of rework in the software development process. Extreme programming helps to reduce this rework by forecasting the changes based on business and technical perspective using enhancing story card and adjusting the planning game i.e. whenever the customer presents a story, the developers analyze the stories and break them up into various features and organize them into components. These components are then assigned risk, cost, schedule and likeliness of getting changed discussed in [25]. Thereby components with minimum chances of getting changed were implemented first, except few times where there exists dependency among the components and therefore those components are developed sequentially.

Security is a matter of concern while developing a web-based application as these applications are highly prone to vulnerabilities. The most popular threats include Cross-Site Scripting and Structured Query Language Injection as discussed in [21]. It becomes important to incorporate certain security measures while adopting any software development methodology. Extreme programming could be improved by tightening security controls all over the development stages.

The improved model thereby involves both the development team and the business representatives in the early phase that helps to identify security threats and to work over it at the early stages in the development lifecycle. This way they define security requirement and acceptance testing requirement within the company policies. Misuse cases along with use cases are introduced in the model that is being carried out from the requirement and design stage as it gives a proper insight regarding how will the system be exposed to threats. It also emphasizes on following secure coding standards that enforce secure naming convention, remarks, and security infrastructure. Pair programming also helps to review code by double checking the areas that are prone to vulnerabilities. There is an iterative risk assessment during the entire lifecycle that keeps track of the security issues as few of the threats may go undetected in the early phases.

Various other applications of XP are as follows:

A. Extreme Programming Development through Dialog

The two primary roles in XP are customer and programmer. The customer is responsible for identifying the features (known as stories) that the programmer must implement, providing detailed acceptance tests for those stories and assigning priority to them. On the other hand, programmers are responsible for estimating how long it will take to implement those stories [10].

With this division of responsibility in place, a project plan represents a dialog between the customer and the programmers. The protocol of that dialog is also clear. The customer tells the programmers what stories he or she wants

in the next iteration. The programmers add up the estimates for those stories and tell the customer whether they are possible [10]. The customer can remove or swap stories but cannot get more in the iteration than the programmers estimate is possible. The programmers can tell the customer how long something will take, but cannot select stories to be implemented.

Thus, there are several layers of dialog in an XP project [10]. The customer has an ongoing dialog with the programmers, the programmers have an ongoing dialog with each other, and the test cases have an ongoing dialog with the program. In this way, the programmer and customer learn to trust each other.

B. *Extreme Usability (XU) in Software Engineering Education*

In software engineering, usability is a degree to which software can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use. Extreme Usability (XU) is such that all best practices of Usability Engineering (UE) are kept in XP during planning games, with a restriction of the usability aspects in next iteration and the equal treatment of Usability and Functionality [11].

A practice of XP, which is often difficult to achieve in a realistic setting, is the customer-on-site because of the heavy time-restraints this poses on the customer [10]. In XU, this difficulty could be transformed into an advantage by allowing different customers to take part in different iterations, if not releases, thus solving two problems at once:

- From the point of view of standard XP, the requirement that the same customer has to be present all the times can be relaxed, thus possibly achieving a better overall coverage of customer time in the team.
- From the point of view of Usability Engineering, the usability of the real system can be tested all times on several different real end users, one at a time, but at any stage in greater depth with the possibility of redesigning the user interaction with the state of the system, for a cost that can be accurately specified.

The combination of Extreme Programming (XP) and Usability Engineering (UE) which leads to a new method: Extreme Usability (XU), is very promising, especially for Software Engineering education [11].

C. *XP Labs Design in Universities*

Planning Game, which is one of the 12 practices in XP help students to learn how to divide requirements into User Stories and how to prioritize and estimate the costs of those stories [6]. A preparation class conveys the principles of agile methods and XP practices to the students. Afterward, a multi-day block course is conducted, which is the main part of the lab. The students have to develop an application for a customer using the XP practices. The on-site customer is played by a tutor. Another tutor performs the coach role [12].

The best results were achieved when the project was for 9 block days and each iteration was of 2 days. Also, the team size played a vital role, so a team of 8 students was best.[12] It is difficult to efficiently enforce test-first development when supervising a larger group of students. So, a small group shows a better result than the large one.

By tweaking the properties of the XP lab, we could improve the overall learning outcome over the years. The XP lab has been a very successful class in the computer science curriculum so far.

D. *Industrial Extreme Programming implementation (IXP) in Rational Unified Process (RUP) on Agile Development theme*

Rational Unified Process (RUP) is a *case-driven* and *architecture-centric* based IBM Theory. The main purpose of RUP is to ensure that the resulting software has high-quality as the user's need. It also ensures that software is produced within time and cost specified. RUP life cycle is divided into 4 different phases: *Inception*, *Collaboration*, *Construction*, and *Transition*. In each phase there are 9 processes, they are :
Business Modeling, Requirements, Analysis and Design, Implementation, Test, Deployment, Configuration and Change Management, Project Management, Environment.

Industrial Extreme Programming (IXP) is a development of the failed Extreme Programming (XP) which is intended for large scale software development [23].

Implementation of Industrial Extreme Programming (IXP) in Rational Unified Process (RUP) is done for every process of RUP. Additions or modification are done on the basis of the workflow of RUP practices offered by IXP. IXP supported by RUP is a better scope for software development which can be used for large scale software development providing structured and applicable methods [12].

VIII. Quality Control in XP

The biggest challenge in XP is maintaining quality control over various iterations. Reworking on the code in different iterations can lead to an unmaintainable mess. But XP has different means to have control over quality. First, XP demands simplicity from the programmers- they must leave the code in the simplest possible state that passes all the acceptance test. Thus, when the code is reworked from iteration to iteration, it is also continually reduced to the simplest state programmer can find. Second, programmers are not allowed to work on code alone, the programmers team up in pairs. Finally, before adding code to the system, the programmers must write a failing unit test that the new code must make successful. This ensures that as the program grows, a copious suite of tests grows with it. These tests keep the quality of the software high and give the programmers the courage they need to continually rework the code into its simplest form—an operation known as *refactoring.*[10].

IX. Advantages of XP

- The greatest advantage of XP is that it allows software development companies to save their cost and time required for project realization. XP

eliminates unproductive activities to reduce the cost and frustration of everyone involved. XP allows developers to focus on coding.

- The Simplicity of code. With different iterations over time, refactoring of code is done i.e, developers are restricted to write code in the simplest form and after every iteration, the code becomes more simple.
- XP reduces the risks related to programming and project failure. The customer gets what he or she wants at the end.
- Constant feedback after every iteration from customers helps programmers to proceed in the right direction.
- XP helps in increasing employee satisfaction and retention. The breakdown of the project into subcomponents and constant feedback helps employees in completing the project within deadline without overtime.
- This approach creates working software faster. Regular testing at the development stage ensures detection of all bugs, and the use of customer-approved validation tests to determine the successful completion of a coding block ensures implementation of only what the customer wants and nothing more.

X. XP AND DATA WAREHOUSE

Some of the practices of XP can be used in the data warehouse to enhance the efficiency of the data warehouse systems.

The data warehouse focuses on the subject of the decision to be taken. By subject, we mean that the information should be completely present for a particular problem. The XP practice of planning game and continuous interaction and feedback between the customer and developer can be a helpful practice to implement in the data warehouse. By interacting among the stakeholders and specifying the problem statement clearly, the data warehouse can be more helpful in providing clear information and hence, support in managing the decisions.

XI. CONCLUSION

Extreme Programming is hence categorized by short iterative cycles, incremental planning, evolutionary design and its ability to response to the changing business needs. It strongly practices oral communication among stakeholders. It encourages unit testing of each functional unit before the integration has to be done. It encourages the programmers to follow a coding standard and work in pairs while coding. This is mainly done to achieve better results. XP is a software development model which emphasizes on better building and understanding of the system.

Extreme programming does not follow the traditional approach of maintaining voluminous documents for requirement specification rather it follows to maintain source code well documented. But in reality, the documentation plays a vital role as the complete product knowledge resides in the software requirement specification rather than merely on a development team's intelligence.

To cope up with this problem, XP sets up an oral communication between the development and maintenance team immediately after the completion of the project as stated in [26]. It also requires skilled programmers to incorporate frequent changes in the project and to ensure that the simple design is maintained. XP is not well suited for mission-critical or safety-critical applications as large scale projects would not find it possible to arrange stand up meetings on a regular basis and having an oral handoff seems to be difficult.

Although extreme programming is a good software development practice, yet there are some of the problems that can be faced by the developers. The timing issues can be faced among the programmers who are involved in pair programming [8]. When standard data structures are used, then there is no such requirement of having two people involved for the same portion of code. Due to the practice of pair programming, the management might have to face the cost issues and expenses of two instead of one for a particular work. Even with some of the drawbacks that can be present in XP, XP still outperforms the traditional models of software development. It provides better efficiency and better system and software understanding.

REFERENCES

[1] J.Choudhari and Dr. U.Suman, Iterative Maintenance Life cycle using Extreme Programming, in International Conference on Advances in Recent Technologies in Communication and Computing in 2010.

[2] T. Goto, K.Tsuchida and T.Nishino, EPISODE:An iterative programming method for innovative software based on system designs in 3rd International Conference on Advanced Applied Informatics in 2014.

[3] B.Xu, Towards high quality software development with extreme programming methodolgy:Practices from real software projects.

[4] A.English, Extreme Programming: It's worth a look.

[5] A.V Deurson, Program Comprehension risks and oppurtunities in extreme programming.

[6] Yang Xiohu, X.Bin,H.Zhijun, Extreme Programming in global software development.

[7] F.Maurer and S.Martel, Extreme Programming : A rapid development for web-based applications.

[8] J.Kivi, D.Haydon, J.Hayes, R.Schmeider, G.Succi, Extreme Programming: A university team design experience.

[9] S.Alsheri and L.Benedicenti, Prioritising CRC Cards as a simple tool in extreme programming in IEEE Canadian Conference of Electrical and Computer Engineering (CCECE) in 2013.

[10] Robert C Martin, Extreme Programming development through Dialog, soapbox

[11] [1]Andreas Holzinger, [1]Maximilian Errath, [1]Gig Searle, Bettina Thurnher[2], Wolfgang Slany,1Institute for Medical Informatics, Statistics & Documentation, Medical University Graz, 2 Institute of Software Technology and Interactive Systems, Vienna University of Technology, 3 Institute for Software Technology, Graz University of Technology, From Extreme Programming and Usability Engineering to Extreme Usability in Software Engineering Education (XP+UEoXU)

[12] Putu Edy Suardiyana Putra, Arlisa Yuliawati, Petrus Mursanto, Industrial Extreme Programming Practice's Implementation in Rational Unified Process on Agile Development Theme, ICACSIS, 2012

[13] [13] Yang Yong, Bosheng Zhou, Evaluating Extreme Programming Effect through System Dynamics Modelling, Beihang University, Beijing

[14] [14] Brian Spears, The Bold New Extreme Programming-Now in its Ninth Year, Agile Conference, 2009

[15] [15] Angela Martin, James Noble, Robert Biddle, Programmers are from Mars, Customers are from Venus: A practical guide for customers on XP projects.

[16] Sallyan Bryant, Double Trouble: Mixing qualitative and quantitative methods in the study of eXtreme Programming

[17] Kai Stapel, Daniel Lubke, Eric Knauss, Best Practices in eXtreme programming Course Design, Leibniz Universitat Hannover

[18] James Newkirk, Introduction to Agile Processes and Extreme Programming, Chicago, USA

[19] Pooja Sharma, Nitasha Hasteer, "Analysis of Linear Sequential and Extreme Programming Development Methodology for a Gaming Application", in international Conference on Communication and Signal Processing, 2016.

[20] Feng Ji, Todd Sedano, "Comparing Extreme Programming and Waterfall Project Results",

[21] Bala Musa S., Norita Md Norwawi,Mohd Hasan Selamat, Khaironi Yetim Sharif, "Improved Extreme Programming Methodology with Inbuilt Security", in IEEE Symposium on Computers & Informatics, 2011

[22] Zhai Li-li., Hong Lian-feng, Sun Qin-ying,"Research on Requirement for High-quality Model of Extreme Programming", in International Conference on Information Management, Innovation Management and Industrial Engineering, 2011

[23] Ramlall Poonam, Chuttur M. Yasser,"An Experimental Study to Investigate Personality Traits on Pair Programming Efficiency in Extreme Programming", in 5th International Conference on Industrial Engineering and Applications, 2018

[24] *Xiaobo Wu*, Chang Ge, "The Research on Necessity and Plan for Using Extreme Programming in Rational Unified Process"

[25] Xu Bin, Yang Xiaohu, He Zhijun, Srinivasa R. Maddineni, "Extreme Programming in reducing the rework of requirement change"

[26] Colin J. Neill, "The Extreme Programming Bandwagon: Revolution or Just Revolting? " , in IEEE Computer Society, 2003

[27] Sara Shahzad, "Learning From Experience: The Analysis of an Extreme Programming Process", in Sixth International Conference on Information Technology: New Generations, 2009

[28] Zahid Hussain, Martin Lechner, Harald Milchrahm, Sara Shahzad, Wolfgang Slany, Martin Umgeher, Thomas Vlk, "Optimizing Extreme Programming", in Proceedings of the International Conference on Computer and Communication Engineering, 2008

[29] H. Kiwan, Y. L. Morgan, Luigi Benedicenti,"Two mathematical modeling approaches for extreme programming",in 26th IEEE Canadian Conference Of Electrical And Computer Engineering (CCECE),2013

[30] Elmuntasir Abdullah, El-Tigani B. Abdelsatir,"Extreme Programming Applied in a Large-scale Distributed System", in International Conference on Computing, Electrical and Electronic Engineering (ICCEEE), 2013