*International Journal of Computer Applications (0975 – 8887)*
*Volume 171 – No. 7, August 2017*

# Proposal of Tailored Extreme Programming Model for Small Projects

Faiza Anwer
Department of Computer Science
Virtual University of Pakistan

Shabib Aftab
Department of Computer Science
Virtual University of Pakistan

Iftikhar Ali
Department of Computer Science
Virtual University of Pakistan

## ABSTRACT

Extreme Programming is a well-known agile process model that can fulfill the needs of today's software industry. It is suitable for small to medium scale projects. Its strength lies in its practices that are applied in extreme manner to get the best results. However in some scenarios these practices overburdened the software development process. In small scale projects where requirements are almost stable and no detailed design and planning activity is required, the overall structure and some of XP practices require extra effort and cause unnecessary delay in project completion. Practices like on-site customer, continuous testing and continuous integration can be a hurdle in timely completion of small project. To overcome these problems a modified form of Extreme Programming model called Tailored Extreme Programming (TXP) is presented in this research that can be applied to small scale projects to make the development process effective and efficient.

## Keywords

Extreme Programming,Modified XP, Tailored XP, Customized XP, XP for small projects, TXP

## 1. INTRODUCTION

Researchers and Software practitioners combined the best practices and principles of software engineering in agile methodologies to overcome the drawbacks of previously known software development models. Agile models provide a lightweight, iterative and incremental way of software development [1] [2]. These development methods are more focused towards customer's satisfaction, team collaboration, fully functional software and accommodating change in requirements even in later phases of development [1] [4] [29] [30]. Agility of these models helps to cope with the today's software development needs. Due to their distinguishing features, agile development models got vast acceptance in software industry. Some of the well-known agile models used in software industry are Extreme Programming (XP), Scrum, Feature Driven Development (FDD), and Dynamic System Development Method (DSDM). Extreme programming is one of most popular agile models developed by Kent Beck in 2000. It is a combination of best software engineering practices, principles and values. Extreme programming uses software engineering practices in a disciplined way to develop a high quality software in less time and cost [2] [24]. It is used for small to medium scale and low risk projects with changing requirements. In XP, the development process consists of six phases i.e. Exploration phase, Planning phase, Iteration to release phase, Productionizing phase, Maintenance phase and Death phase[2] [3]. In exploration phase requirements are collected from customer through story cards. Different architecture possibilities are also considered in this phase [3]. Planning phase deals with project planning issues. In this phase order of development is defined for each iteration. An iteration plan is also developed that includes detail like number of stories to be implemented, effort and time estimation [5]. Actual development is take place in iteration to release phase. This phase may consists of several iteration that ends with a workable software product [2] [3]. In productionizing phase, developed software is tested before handing over to the customer [6]. In XP new functionality can be added, keeping the old one in running condition in maintenance phase [3] [7]. Finally when implementation of all customer's requirements are completed, development process enters in death phase. Necessary documentation is completed and the product is released in this phase [3] [7].During these phases XP uses twelve best practices of software engineering. These practices include planning game, small releases, metaphor, simple design, continuous testing, refactoring, pair programming, collective ownership, continuous integration, 40-hour week, on-site customer and coding standards [2] [3].XP uses these practices rigorously to get high quality software [2] [3]. But these practices cannot be beneficial in each and every situation. Especially in small scale projects, XP practices like on-site customer, continuous testing, and continuous integration are difficult to implement. Moreover the structure of XP makes it rigid for small scale software development. Six phases and activities performed in each phase put extra burden on schedule when we deal with small project where requirements are well defined with less tendency to change. In such situation we need a process model which deals with the small scale project development where no detail design and planning activity is required. Many modified XP models are available but nota single one tried to cover these problems for small projects in an effective way. In this paper a tailored version of XP called TXP is presented which provides a better, easy and flexible development approach for small scale projects.

Remaining paper is organized in section 2which is about related work. Section 3 defines the problem areas and section 4 presents TXP model as a solution of the problem. Finally section 5 concludes the paper.

## 2. RELATED WORK

In [8], authors have presented a simplified version of classical extreme programming model called SXP for small to medium scale projects. The purpose of this research was to make the XP model more effective which can deliver the qualitative product with in time. However SXP does not use the pair programming and onsite customer practices and introduced explicit analysis and design phases which make it complex to apply on simple small scale projects where requirements are clear and unlikely to change. This paper lacks the empirical proof. In [9], an enhanced extreme programming model is proposed by authors. The proposed model focuses on solving the problems of documentation, design and quality without losing agility. This model introduced parallel quality iteration to basic XP iteration. The refinement cycle executes with basic XP iteration to produce quality product. However, development of software with higher interdependencies among subsystems cannot be well-supported by this model. Parallel execution of refinement and basic XP iteration demand more resources that increase overall cost of software development. This paper also lacks empirical proof of the model. In [10], authors proposed software maintenance process model based on XP practices. This model uses XP practices in software maintenance process to improve the productivity. The proposed model makes the maintenance process speedy and qualitative which produces more maintainable code with less effort. This approach enhances learning as well as productivity of the team. However authors evaluated proposed model in academic projects only. In real business projects, situations can be very different from academic projects and can be even more complex. To judge the true strength of proposed model, it should be evaluated using real life projects.In [11], authors proposed a hybrid model that integrates Scrum, XP and Dynamic System Development Model (DSDM). This model combined the strengths of these different process models in one hybrid model. Proposed model tried to cover the different aspects of software development by taking project management practices from Scrum, engineering practices from XP and project initiation practices from DSDM. This model introduced a new role called technical writer for proper documentation of the system. These models have complementary practices and the proposed model can be heavily loaded with these practices which can affect the agility of proposed integrated model.In [12], authors suggested a hybrid model called XSR. This model providesa generalized framework which integrate XP, Scrum and RUP. XSR integrates management related aspects from scrum, engineering and coding related aspects from XP and business related aspects from RUP. Authors claimed that this model can deliver a high quality end product with low bug rate but there was no empirical proof in the paper to support the claim.[13] Proposed an enhanced Scrum framework by combining XP practices in Scrum activities. This model incorporated simple design practice of XP during product backlog activity, user stories during sprint planning and pair programming, coding standards and testing during sprints. However this paper lacks the implementation detail and empirical proof of the proposed framework to ensure the effectiveness.In [14], authors proposed a framework named IXSCRUM that integrates the management aspects of Scrum and engineering practices of XP for the sake of quality software development in timely manner. Proposed model is validated using the results of a case study however there is a need to evaluate the effectiveness of proposed model via comparative analysis with other agile models. In [15], authors proposed a hybrid model called eXRUP. Proposed model integrates XP and RUP process models. The purpose of this research was to target the small to medium scale project with the focus on high quality and timely delivery. Authors have taken the software engineering practices from XP and maintenance & documentation activities from RUP. This model tried to cover almost all aspects of software development however in this model time to manage change request was comparatively high. This integration can work better for medium scale projects but for small scale project its phases and activities will put extra burden on development team. Furthermore this paper did not provide any guidancefor coordination and communication aspect of teams.In [16],authors discussedthe principles, values and practices of XP followed by the analysis for the support of XP for SOA (Service Oriented Architecture). They proposed an approach which is composed of a set of guidelines for the effective use of practices of extreme programming with the context of SOA.However this paper did not pay any attention towards SOA complexities that can reduce the agility of XP. This paper also lacks empirical proof to prove the effectiveness of proposed solution.In [17], authors used AHP (Analytical Hierarchy Process) for CRC cards in prioritization process in the design activity of XP. It is also highlighted by the authors that AHP is an important tool for the team of XP to analyze that which CRC cards can impact the system most. With AHP, cooperative decision making environment can be built which can make the XP development process more effective and efficient. The proposed solution is tested by a case study in which participants are the graduate students.In [18], author presented an extended XP model to overcome one of the main limitations of classical XP which is its non-suitability for medium and large scale projects. Proposed model introduced some new phases in XP to handle the medium to large scale projects. Due to its extension in phases, it loses the concept of agility.In [19] authors proposed a model that helped development team as well as the customer in the release planning activity. This model developed a release plan by keeping in view the size of stories, priorities and precedence relations among stories. Proposed model can work effectively for projects having 50 or less stories. Exponential complexity of model makes it difficult to use it in large scale projects where number of stories is more than 50.In [20].the authors proposed XP process model framework for e-commerce application development. The authors focused on inbuilt security features and parameters for XP.The proposed approach ensures the involvement of development team as well as the client in the start which allows the identification of important security threats at initial level and also the dealingof these threats in early stages. The risk management activity is performed in iteration to find out any more threats which might have left undetected in early stages. But incorporating these security checks in all phases of XP can affect the agility of the model. To prove the effectiveness of model, it is required to validate it using real life project.In [21], authors worked to improve customer awareness by reducing the misunderstanding of software development process and barriers & potential threats. They introduced two-dimensional analysis model which improves the XP's traditional demand method. The proposed solution improves XP demand module by using Kano model's quality features. However this paper does not provide any real life project evaluation of proposed method.In [22], authors proposed an adaptive XP model. This adaptive XP model introduced analysis, design and deployment phases to provide better adaptability to different software projectsi.e. simple, average and complex moreover it eliminates the limitation of development of reusable components, large development teams, documentation and quality. However there was no empirical proof in the paper to support the claim. In [23], authors proposed an improved XP model to eliminate the weaknesses of classical XP. Proposed XP model supports component based development with risk management in distributed environment where large teams are involved. However this paper lacks implementation detail of analysis and risk management activities. Furthermore there was no empirical proof given to support the claim.

## 3. RESEARCH PROBLEM

Extreme programming is a renowned agile software development model that provides agility in software development process and can handle changing requirements with good level of customer satisfaction [2] [24] [25]. It uses best software engineering practices in extreme manner to get a quality software [2].

XP process model is used for low risk and small to medium scale projects. However conventional XP can delay the end product when it is used for small scale projects where the requirements are predefined and having very low tendency to change in later stages. The structure of conventional XP, its six phases and activities in each phase are not suitable for small projects. Some of XP's practices can prolong the development process such as On-site customer, continuous testing and continuous integration.

It will be unnecessary to involve customer throughout the development process when the project is small and requirements are clear/predefined and have less tendency to change. Moreover this practice is very sensitive to implement as far as quality product is concerned, because Inexperienced staff member cannot contribute effectively instead of that he may spoil the project by providing wrong information [26], [27].

Continuous integration is also not implementable with small projects as small project can easily be completed in one iteration however even if more than one iterations are required such as in case of customer's demand then the product can be developed in more than one modules but integration and integration testing will need more time specially when there is only one developer in the team [28]. Many researchers have presented different modifications in classical XP process model to reduce the limitations however above identified problems are still unaddressed. Keeping in view the above identified problems we have to find the answer of following question.

*How to modify classical XP process model to make it suitable for small scale projects with effective and efficient development process?*

## 4. PROPOSED SOLUTION

As a solution of the above identified problem, a tailored version of XP model for small scale projects is presented called TXP. This section presents a detail description of proposed model. TXP removes unnecessary practices of conventional XP and also cover its drawbacks. Proposed model provides a flexible and simple approach for small scale software development which can result in efficient and effective development process. TXP consists of three phases: *Initialization phase*, *Development & Testing phase* and *Release Phase*. Activities of each phase will be monitored by the Project Manager who is the key role of this process model.

### 4.1 Initialization Phase

Initialization phase deals with project initiation activities. Requirement gathering, project planning and software designing are the main activities of this phase. Budgetary decisions are also taken with the consent of both sides (customer & Project Manager). Successful completion of this phase provides the sound base for the development of high quality software.

Below is the detail of activities which are performed in this phase:

#### *4.1.1 Story Writing & Prioritization*

Customer writes story cards to describe the features to be included in the system. Each story card explains one feature or functionality along with its priority. Development team extracts functional and non-functional requirements from these story cards. No technical detail is added in these cards; only purpose of these cards is to collect desired functionality of the system to be developed.

#### *4.1.2 Project Planning*

This activity is completed by the development team with the consent of customer. A project plan is prepared after negotiating with customer regarding project scope, cost and tools/technology to be used. Iteration plan contains detail about number of iterations, duration of iterations, number of stories implemented in each iteration and hardware/software resource estimation. Structure of system is finalized in the form of architectural diagram during this phase.

#### *4.1.3 Designing UML Diagrams*

This activity is also completed by development team. Two types of UML diagrams are designed Use Case and Sequence diagrams. These diagrams provide the pictorial view of system's requirements for the development team.

### 4.2 Development and Testing Phase

This is the second and main phase of TXP. The input of this phase is the UML diagrams which were developed in the first phase. These diagrams will be used in each activity of this phase. There are three activities of this phase Coding, Functional Testing and Integration & Integration Testing. We called this phase as the main phase because it includes the iteration part. Moreover the development and testing requires the maximum resources as compared to other activities of the process model.Below is the detail of activities which are performed in this phase:

#### *4.2.1 Coding*

The second phase starts with the coding activity in which a programmer writes code for the selected stories according to design, prepared in first phase. Pair programming practice is followed in which two programmers write code by using one computer. However if the development team consists of only one member then this practice can't be followed.

#### *4.2.2 Functional Testing*

Test cases are written by keeping in view the UML Diagrams and then developed module/software is tested to get rapid feedback. In case of any problem during functional testing, the code is reviewed to find the bug and properly noted also the coding activity can be repeated. These tests are performed by programmer(s).

#### *4.2.3 Integration & Integration Testing*

After the successful functional testing of each module it has to be integrated with the previously developed module(s) (if any). This activity is optional in this process model because this model is designed for small scale projects, but even then if the Project Manager or customer wants then this model can provide the facility to develop a small project in chunks/components.

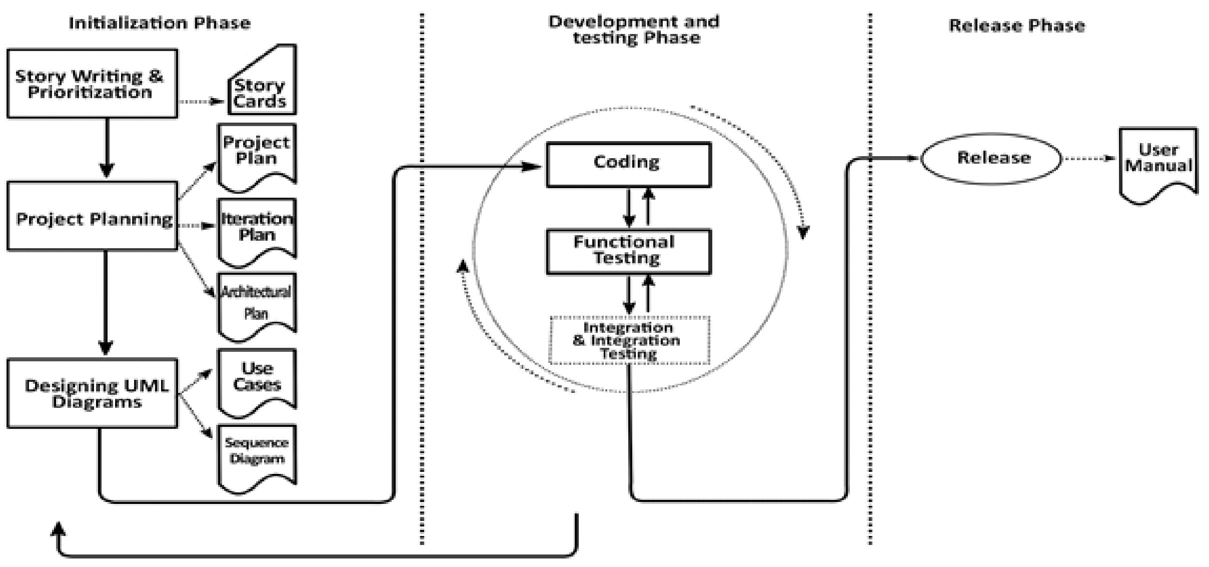

**Fig 1: TXP Process Model**

If more than one iteration are required then code developed during each iteration is integrated and tested to validate its correctness. If any bug occurs during the integration testing then the codeis reviewed to find the problem. This activity is properly noted for future reference moreoverthe first two activities of the phase can be revised if the problem exists.

## 4.3 Release Phase

This is final phase of this process model where a complete workable product along with the user manual is delivered after customer's approval.

## 5. CONCLUSION

Extreme programming is a well-known agile model that helps in developing high quality software products.It is used for low risk, small and medium scale projects. XP greatly stress on implementation of its practices to extreme level in order to get high quality end results.XP practices are greatly helpful in developing software according to customer choice however these are not as fruitful in every situation. It is claimed that XP is suitable for small scale projects however some of its practices slow down the development process. Practices like on-site customer, continuous integration and continuous testing can be helpful in medium scale projects but in small projects these can be an extra burden over development process.In this paper a tailored version of XP called TXP is proposed that is specially designed for small scale projects which have predefined requirements. This model removes extra practices and activities from classical XP and provides best solution for the development of small scale projects without losing agility.

## 6. REFERENCES

[1] Williams, L. (2010). Agile software development methodologies and practices. Advances in Computers, 80, 1-44.

[2] Anwer, F., Aftab, S., Shah, S. S. M., &Waheed, U. (2017). Comparative Analysis of Two Popular Agile Process Models: Extreme Programming and Scrum. *International Journal of Computer Science and Telecommunications*, 8(2), 1-7.

[3] Beck, K. (2000). Extreme programming explained: embrace change. addison-wesley professional.

[4] Anwer, F., Aftab, S., Waheed, U., & Muhammad, S. S. (2017). Agile Software Development Models TDD, FDD, DSDM, and Crystal Methods: A Survey. *International Journal of Multidisciplinary Sciences and Engineering*, 8(2), 1-10.

[5] Dudziak, T. (1999). Extreme programming an overview. Methoden und Werkzeuge der Softwareproduktion WS, 2000, 1-28.

[6] Juric, R. (2000, June). Extreme programming and its development practices. In Information Technology Interfaces, 2000. ITI 2000. Proceedings of the 22nd International Conference on (pp. 97-104). IEEE.

[7] Abrahamsson, P., Salo, O., Ronkainen, J., &Warsta, J. (2002). Agile software development methods: Review and analysis. 3-107.

[8] Anwer, F., & Aftab, S. (2017). SXP: Simplified Extreme Programing Process Model. International Journal of Modern Education and Computer Science (IJMECS), 9(6), 25.

[9] Qureshi, M. R. J., &Ikram, J. S. (2015). Proposal of Enhanced Extreme Programming Model. International Journal of Information Engineering and Electronic Business, 7(1), 37.

[10] Choudhari, J., &Suman, U. (2014). Extended iterative maintenance life cycle using eXtreme programming. ACM SIGSOFT Software Engineering Notes, 39(1), 1-12.

[11] Sultana, S., Motla, Y. H., Asghar, S., Jamal, M., & Azad, R. (2014, February). A hybrid model by integrating agile practices for pakistani software industry. In Electronics, Communications and Computers (CONIELECOMP), 2014 International Conference on (pp. 256-262). IEEE.

[12] Ahmad, G., Soomro, T. R., &Brohi, M. N. (2014). XSR: Novel Hybrid Software Development Model (Integrating XP, Scrum & RUP). International Journal of Soft Computing and Engineering (IJSCE), 2(3), 126-130.

[13] Darwish, N. R. (2014). Enhancements in Scum Framework Using Extreme Programming Practices. International Journal of Intelligent Computing and

Information Sciences (IJICIS), Ain Shams University, 14(2), 53-67.

[14] Malhotra, C., & Chug, A. (2013). IXSCRUM-A Framework Combining Scrum and XP. International Journal of Scientific & Engineering Research, 4(7).

[15] Rasool, G., Aftab, S., Hussain, S., &Streitferdt, D. (2013). eXRUP: A Hybrid Software Development Model for Small to Medium Scale Projects. Journal of Software Engineering and Applications, 6(09), 446.

[16] Carvalho, F., &Azevedo, L. G. (2013, March). Service agile development using XP. In Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on (pp. 254-259). IEEE.

[17] Alshehri, S., &Benedicenti, L. (2013, May). Prioritizing CRC cards as a simple design tool in extreme programming. In Electrical and Computer Engineering (CCECE), 2013 26th Annual IEEE Canadian Conference on (pp. 1-4). IEEE.

[18] Qureshi, M. R. J. (2012). Agile software development methodology for medium and large projects. IET software, 6(4), 358-363.

[19] vanValkenhoef, G., Tervonen, T., de Brock, B., &Postmus, D. (2011). Quantitative release planning in extreme programming. Information and software technology, 53(11), 1227-1235.

[20] Musa, S. B., Norwawi, N. M., Selamat, M. H., & Sharif, K. Y. (2011, March). Improved extreme programming methodology with inbuilt security. In Computers & Informatics (ISCI), 2011 IEEE Symposium on (pp. 674-679). IEEE.

[21] Z. Li-li, H. Lian-feng and S. Qin-ying, "Research on Requirement for High-quality Model of Extreme Programming," in Information Management, Innovation Management and Industrial Engineering (ICIII), Shenzhen, 2011.

[22] Qureshi, M. R. J., &Hussain, S. A. (2008). An adaptive software development process model. Advances in Engineering Software, 39(8), 654-658.

[23] Qureshi, M., &Hussain, S. A. (2012). An Improved XP Software Development Model. arXiv preprint arXiv:1202.2501.

[24] Mahajan, E. R., &Kaur, E. P. (2010). Extreme Programming: Newly Acclaimed Agile System Development Process. International Journal of Information Technology, 3(2), 699-705.

[25] Newkirk, J. (2002, May). Introduction to agile processes and extreme programming. In Proceedings of the 24th international conference on Software engineering (pp. 695-696). ACM.

[26] Khalaf, S. A. J., & Maria, K. A. (2009, December). An Empirical study of XP: the case of Jordan. In Information and Multimedia Technology, 2009. ICIMT'09. International Conference on (pp. 380-383). IEEE.

[27] Dalalah, A. (2014). Extreme Programming: Strengths and Weaknesses. Computer Technology and Application, 5(1).

[28] Rao, K. N., Naidu, G. K., &Chakka, P. (2011). A study of the Agile software development methods, applicability and implications in industry. International Journal of Software Engineering and its applications, 5(2), 35-45.

[29] Fowler, M., & Highsmith, J. (2001). The agile manifesto. Software Development, 9(8), 28-35.

[30] Ashraf, S., & Aftab, S. (2017). Latest Transformations in Scrum: A State of the Art Review. *International Journal of Modern Education and Computer Science* (IJMECS), 9(7), 12-22.