# Empirical Comparison of XP & SXP

**Faiza Anwer†, Shabib Aftab†, Muhammad Salman Bashir†, Zahid Nawaz††, Madiha Anwar††, Munir Ahmad†**

†Department of Computer Science, Virtual University of Pakistan
††Department of Computer Science, University of Gujrat, Lahore Campus, Pakistan

**Summary**
Extreme Programming (XP) is a renowned agile model, commonly used for small scale projects. It uses an iterative approach for software development, assisted with agile practices used in extreme manner. Although XP provides the opportunity to handle shortcomings of traditional software development models however it is not exempt from limitations. Lack of proper design, no documentation and poor architectural structure are some of its drawbacks. Furthermore, some of its practices like on-site customer and pair programming are not beneficial in every situation and may cause an extra burden on development process. Simplified Extreme Programming (SXP) process model was proposed to cover these problems without affecting the agility of development process. This paper compares classical XP and proposed SXP with the help of empirical case studies.

***Key words:***
Agile Models, Extreme Programming, XP, SXP,Modified XP, Comparative Analysis, Empirical Comparison.

## 1. Introduction

Agile software development methodologies provide light weight, iterative and incremental way of software development with evolutionary principles and values [1],[2],[3],[34]. These methodologies emerged in 2001 while the software industry was looking for better software development processes, which could mitigate the project's failure risks and also meet the needs of new business environment [2],[4],[5]. Agile methodologies changed the development paradigm and explored the hidden aspects of software development to get better results. These methodologies deeply valued good team collaboration, frequent customer interaction and change in requirements with constant pace of development [5],[6],[34]. Agile methods are the collection of best software engineering practices and values used to cope with challenges of delayed, canceled or failed projects [1] [31],[33],[35]. Although most of these practices were not new for the software industry, however in agile umbrella these are used in a novel manner and the encouraging results of these practices convinced the software developers to use agile models to handle software failure risks [6],[7]. Today, many agile software development models are used by software industry such as Extreme programming (XP), Scrum, Feature driven development (FDD), Dynamic system development method (DSDM), Kanban, Lean software development (LSD) and Adaptive software development (ASD). Extreme programming (XP) is one of the widely used agile models [3]. It was developed by Kent Beck to overcome the limitations of traditional software development methodologies. It consists of principles, values and practices, which work together rigorously to develop high quality software [6],[8],[9]. Like other agile methods, XP provides a flexible and adaptive approach which can handle the changing business needs in a better way. Its twelve practices provide guidelines to govern the whole development process. With all the advantages XP provides, it lacks in some areas as well. Poor architecture, weak system design and lack of documentation are the major issues with XP [10],[11]. Moreover some of its practices like 'pair programming' and 'on-site customer' are little bit controversial and are not applicable in every situation [12]. Pair programming needs mutual understanding, common skills, personality traits and good coordination among developers [11],[13]. It is also possible that particular project does not have enough resources to use pair programming practice. Similarly, the practice of on-site customer can cause serious issues if not implemented properly [11],[12],[13]. Customer presence can cause problems if he does not understand the system requirements properly. To tackle the mentioned issues of XP, SXP [14] was proposed for small to medium scale projects. This paper performs an empirical comparison of conventional XP and SXP with the help of case studies. Both models are used to develop client oriented projects and extracted empirical data of development is used for comparison.

Further organization of this paper is as follows. Section 2 describes the various attempts of XP customizations. Section 3 provides brief overview of XP & SXP. Section 4 compares both the models with empirical analysis. Section 5 finally concludes the paper.

## 2. Related Work

Many researchers have discussed and customized the XP model; some of the selected studies are discussed here. In [32], researchers presented a customized form of XP named Tailored Extreme Programming (TXP). The proposed model was designed specifically for small scale projects where requirements have fewer tendencies to change. In [15], authors proposed a customized version of XP model which introduced the feature of reusability. Proposed model used a framework which added the ability of component based architecture refinement reusability in traditional XP. This framework provided a way to develop simple and loosely coupled design which has made the future modifications easy. Researchers in [16] customized the XP model and introduced parallel refinement iteration along with development activities to enhance quality without affecting the agility. Proposed

Manuscript received March 5, 2018.
Manuscript revised March 20, 2018.

model is not suitable for software projects having a lot of inter dependencies among modules. In [17], researchers proposed Formal Extreme Programming (FXP), a modified form of XP model. FXP introduced formal methods in XP to deal with safety critical projects. In proposed model, authors combined the agility of XP with precision of formal methods to overcome the drawbacks of both models.FXP used formal methods like Software Cost Reduction (SCR), Algebraic Specification and Design by Contract (DbC) in different phases of XP to make it suitable for safety critical projects. In [18], authors presented an extended software maintenance model. The proposed model used many XP practices such as: on-site customer, planning game, small releases, pair programming, metaphor, test driven development and refactoring etc. In [19], authors presented an integrated model by incorporating the practices of Personal Software Process (PSP) in XP. The proposed model introduced "Personal planning phase" in which developers can plan their activities using PSP practices. 6 crucial practices from each model (PSP and XP) are integrated in proposed model for effective development. In [20], a modified XP model was presented to develop medium scale projects with a large team. Moreover the proposed model also targeted the drawbacks of XP such as weak design and lack of documentation.A phase named "analysis and risk management" was introduced to handle failure risks. In [21], researchers used Analytical Hierarchy Process (AHP) with CRC cards during designing phase of XP. AHP was used to design a systematic approach of CRC card's prioritization. AHP is a five step hierarchal model which reflects the human thinking process. Use of AHP enables the developers to select, design and implement the most important classes first. In [22], researcher proposed a new XP model by extending classical XP for medium and large scale projects. According to author, XP has some drawbacks which make it suitable only for small scale projects. The drawbacks include weak design, poor architecture, lack of risk management and lack of documentation. To handle these issues, new phases were introduced. In [23], authors presented an improved XP methodology, designed for the security critical projects. The proposed model included security checks in all the phases of XP and involved developers and business representatives from the beginning of project to identify security threats.

## 3. Material and Methods

XP is one of the widely accepted agile models by software industry. It is commonly used for small scale and low risk projects. Along with all the benefits XP provides, it reflects some limitations as well. Lack of documentation and weak system architecture make it a bad choice for medium and large scale projects. Absence of proper system design makes the development task difficult and time consuming [11],[26]. Moreover practices like pair programming and on-site customer can create extra burden over the development process if not implemented properly [11],[12],[13]. To get the maximum benefits from XP, its limitations have to be eliminated. For this reason, many researchers have presented the customized versions of XP. Moreover, XP has been tailored and integrated with other models to accommodate different business needs and product requirements [25]. Simplified Extreme Programming (SXP) was proposed in [14] to fix the maximum issues of XP without affecting the simplicity and agility.

### 3.1 ExtremeProgramming (XP)

XP is a lightweight and flexible approach of software development which focuses on customer's satisfaction, frequent communication, quick feedback and acceptance of changing requirements [3],[24]. Its iterative and incremental approach helps in managing the vague and constantly changing requirements with maximum level of customer's satisfaction [29]. Development process starts with the basic functionality of the system (module) which incrementally developed in to a complete product [6]. XP is a collection of values, principles and best practices that may not be new for software industry but arranged here in a novel way to achieve effective and efficient software development as well as to get the trust of software industry [6]. These practices and principles are used in extreme manner to mitigate the chances of project failure. XP uses twelve best practices including planning game, small releases, pair programming, metaphor, refactoring, collective code ownership, on-site customer, 40-hour week, simple design, continuous integration, continuous testing and coding standards [3],[24]. The development process of XP consists of six phases: Exploration, Planning, Iteration to release, productionizing, maintenance and death (Fig. 1). Exploration phase deals with the requirement gathering activity. In this phase, customer provides the system requirements through writing the story cards [24],[27],[28]. Each story card describes a small functionality to be developed without any technical detail. Development team considers different options regarding tools and technology for system development [3]. Next is the planning phase, in which a complete iteration is planned by prioritizing the collected requirements. Moreover, effort and time estimation is also completed in this phase to make a realistic plan. In iteration to release phase, developers perform analysis, designing, coding, testing and integration activities of selected requirements iteratively. Pair programming practice is used for development in this phase [30]. Unit testing and integration testing are performed to get quick feedback regarding the implemented code. In productionizing phase, further testing is performed and with the approval of customer, workable module is released. Maintenance phase is used to handle maintenance process of software in which existing functionality can be updated or a new functionality can be added [6]. Death phase is last phase of XP where development process ends after releasing the final product with complete required functionalities.

### 3.2 Simplified Extreme Programming (SXP)

SXP [14] is the customized form of XP designed for small to medium scale projects.

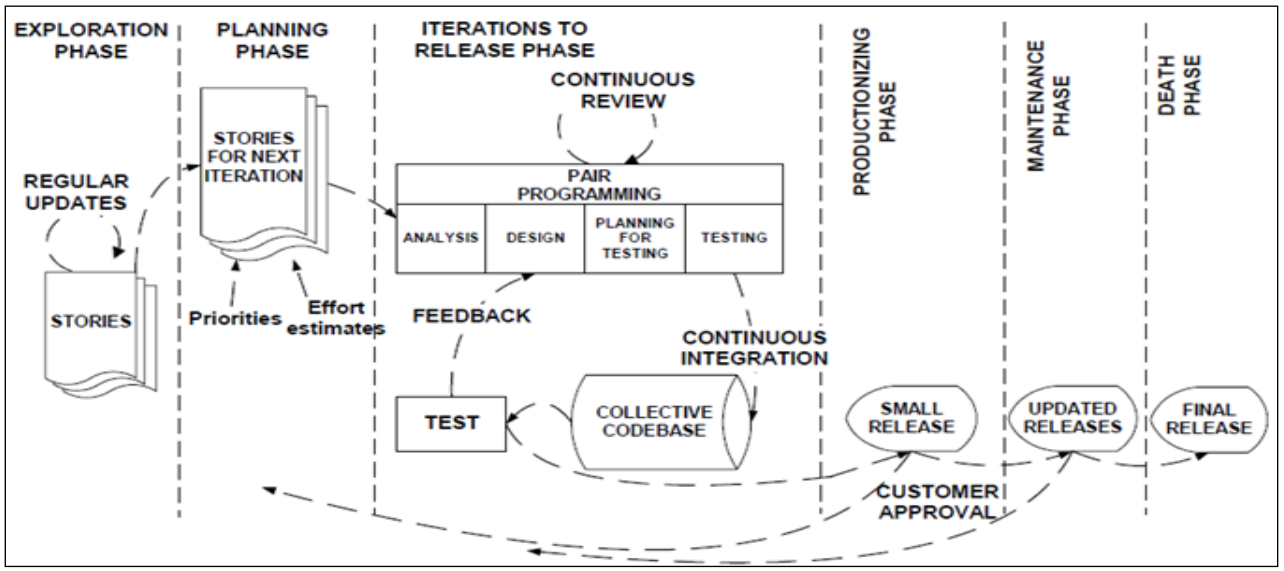


Fig. 1Extreme Programming Life Cycle [3],[6]

SXP explicitly pays attention towards system architecture, design and documentation activities without affecting the agility. This model simplified the structure of XP by removing the practices of pair programming and on-site customer. SXP consists of five phases: Initialization, Analysis, Design, Development & Testing, and Release (Fig. 2). Initialization phase is the first phase of SXP which consists of two basic activities; requirement gathering & prioritization and project planning. In this phase, representatives from customer side and from developer side sit together to extract the requirements of system. These collected requirements are then prioritized according to customer demand and finally documented for later reference. During the activity of project planning, project scope and project cost are finalized.

Moreover development tools are also selected in this phase. During analysis phase, an iteration plan is developed, which includes the detail regarding number of iterations needed to develop a complete system and the number of stories which would be implemented in each iteration. Moreover, time span of each iteration and final budget estimation of the project is also the part of this phase. All these activities are performed by development team only. Design phase of SXP provides the opportunity to explicitly focus on system design. In this phase, use case and sequence diagrams are developed. This phase also includes the test planning activity in which test cases for the functionalities are written before the development. Writing the test cases before coding activity provide better design opportunities.

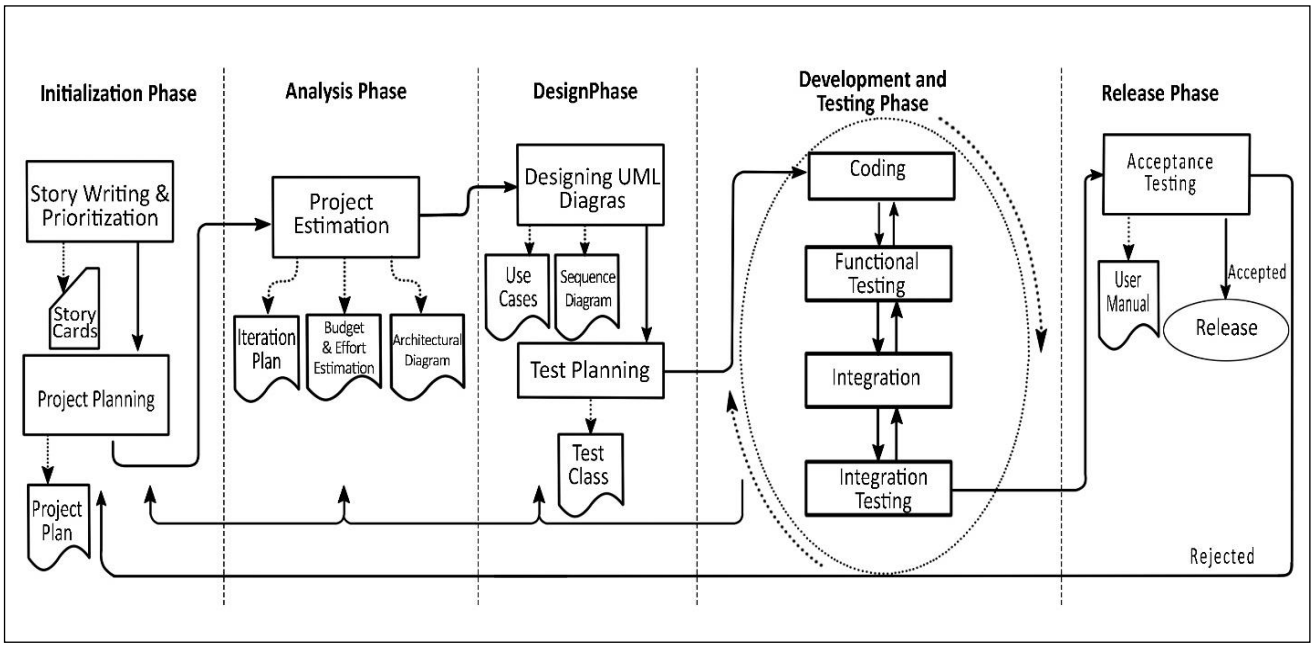


Fig. 2 Simplified Extreme Programming (SXP) Process Model

in which coding and testing activities are performed in an iterative manner.This is the phase where actual implementation takes place. During coding activity only one developer can work on selected tasks. In this phase activities of coding, testing, code integration, and integration testing are performed in an iterative manner until a workable product is ready.The final product is handed over to customer in release phase after performing acceptance testing. As shown in Fig. 2, these phases can be revisited in case of any deficiency or problem.

## 4. Results and Discussions

Two case studies are selected for the empirical comparison of XP and SXP. The selected case studies are the part of a research project in which multiple agile models are used to develop client oriented projects in a software house, situated in Islamabad, capital of Pakistan. In selected case studies, two small scale projects of same nature were developed by two teams with XP and SXP respectively.

Table 1: Case Study Selected for XP

| **Characteristics** | **Description** |
|---|---|
| Size | Small |
| Iterations | 3 |
| Programming Approach | Object Oriented |
| Language | C#, ASP.NET |
| Documentation | MS Office |
| Testing | Browser Stack |
| Web Server | IIS |
| Product Type | Human Resource |
| Project Type | Average |
| Project Duration | 4 Weeks |
| Team size | 5 Member |
| Feed back | Weekly |
| Development Environment | Visual Studio 2012 |
| Other Tools | MS Visio |
| Reports | Crystal Report |

Table 2. Case Study Selected for SXP

| **Characteristics** | **Description** |
|---|---|
| Size | Small |
| Iterations | 4 |
| Programming Approach | Object Oriented |
| Language | C#, ASP.NET |
| Documentation | MS Office |
| Testing | Browser Stack |
| Web Server | IIS |
| Product Type | Human Resource |
| Project Type | Average |
| Project Duration | 4 Weeks |
| Team size | 5 Member |
| Feed back | Weekly |
| Development Environment | Visual Studio 2012 |
| Other Tools | MS Visio |
| Reports | Crystal Report |

Both teams have same work environment and each team consisted of 5 members. Description of case studies of both the models is given in Table 1 and Table 2.

For comparison, following quality parameters are selected [36],[37]:

- Completion Time (weeks)
- Budgeted Work Effort (hours)
- Actual Work Effort (hours)
- Post Release Defects
- Team Productivity
- Time to Manage Change

Table 3 presents the comparison of empirical results according to above mentioned quality parameters. It can be seen that SXP has improved overall development process as its analysis and design phase helped to overcome the limitation of weak design and produced necessary documentation for future reference. Removing the practices of pair programming and on-site-customer made the development process simple and productive.

Table 3: Comparison of XP and SXP model

| **Parameter** | **XP** | **SXP** |
|---|---|---|
| Completion time (in weeks) | 4 | 3.8 |
| Total line of Code (LOC) | 2812 | 3060 |
| Budgeted work effort (in hours) | 800 | 760 |
| Actual work effort (in hours) | 710 | 695 |
| Post release defects | 25 | 15 |
| Team productivity | 3.96 | 4.4 |
| Pre-release Change Requests | 10 | 10 |
| Time to manage change (in hours) | 14 | 11 |

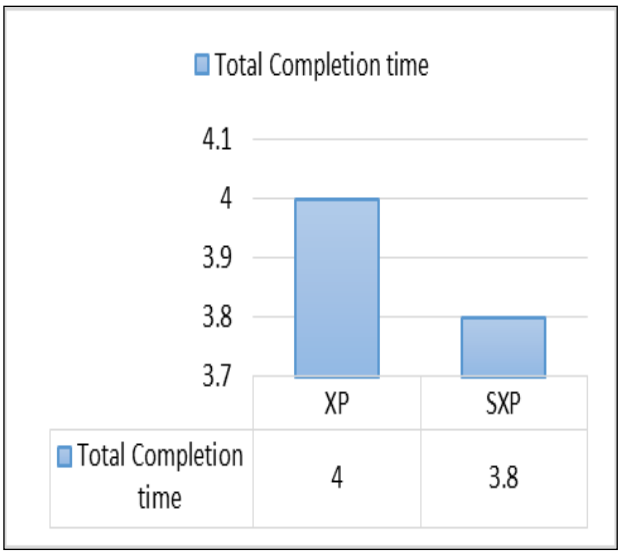


Fig. 3Total Completion Time (Weeks)

Total project completion time is less in SXP (Fig. 3) because of the effective customization.
The activities of analysis phase helped in proper estimation and provided an upfront design document which reduced the total completion time. Moreover the removal of the practices of pair-programming and onsite-customer also reduced the completion time as these practices were creating an extra burden on overall schedule.
Total budgeted work effort for a project is an important quality metric and calculated using the following formula;
Total budgeted work effort (h) =No of hours in a day (8) * No of days in a week (5) *No of weeks* Total team size (5).
As the completion time reported for SXP is already less then XP so obviously budgeted work effort would also be less (Fig. 4) as in both projects remaining parameters are same such as no of hours, no of days and team size.

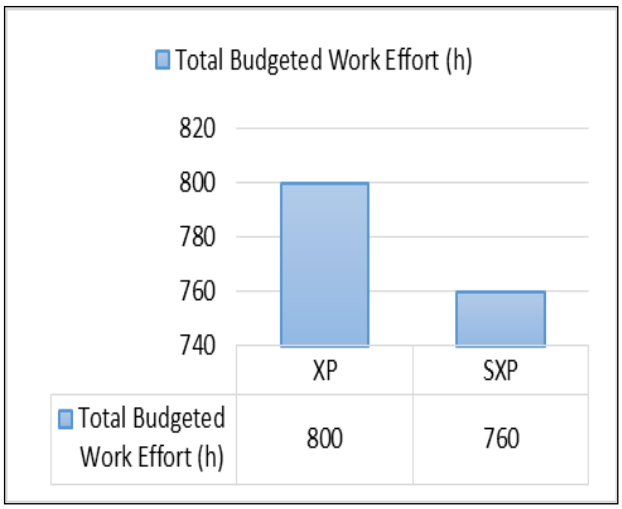

Fig. 4 Budgeted Work Effort (Hours)

Actual work effort depends upon the actual time in hours spent in a day for project development, the remaining formula is same as budgeted work effort.

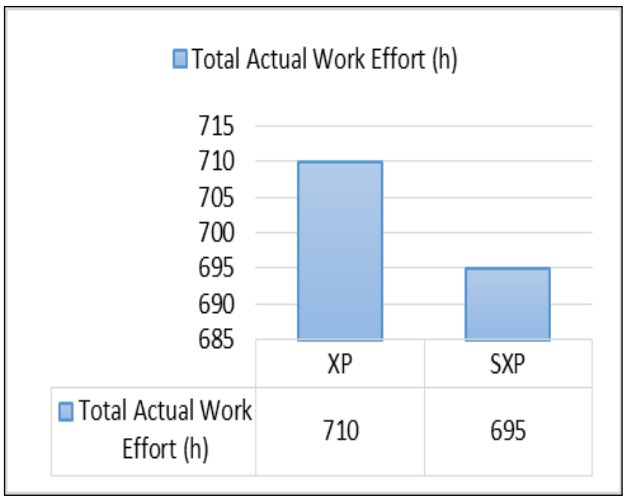

Fig. 5Actual Work Effort (Hours)

Actual spending hours are always less from budgeted time as some time is consumed on other related activities. In SXP, actual work effort is 695 hours however in XP it is reported 710 hours. Number of post release defects is very important quality parameter which also reflects the customer's satisfaction. In case of SXP, 3060 lines of code were written and 15 defects were reported. Whereas for XP project, 2812 lines of code were written and 25 post release defects were reported (Fig. 6).

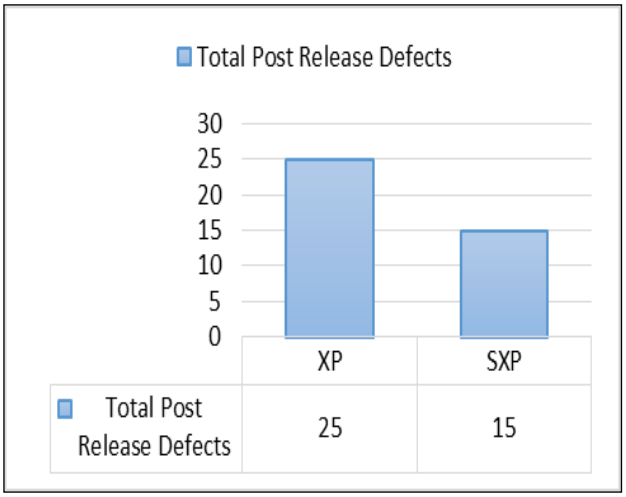

Fig. 6 Post Release Defects

Team productivity is calculated using the following formula;
Productivity= line of code/ actual time spent in hours.
As in SXP, more lines of code were written in less time that's why team productivity is higher than XP (Fig. 7).

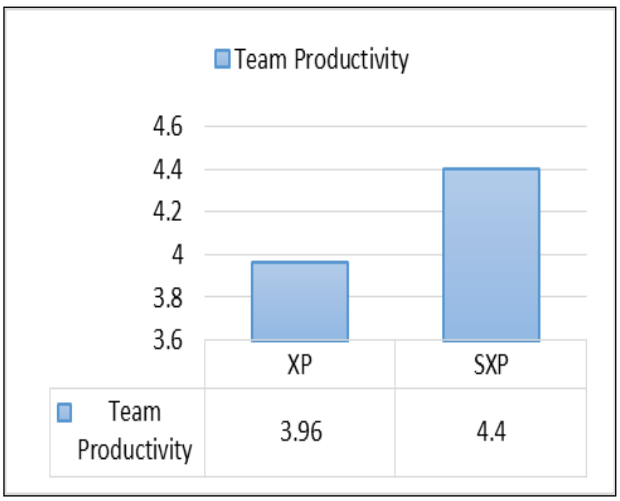

Fig. 7 Team Productivity

Post release change requests are 10 in both the cases, SXP implemented the changes in 11 hours whereas XP have taken 14 hours (Fig. 8). SXP provides the necessary documentation which helps to manage the change in an efficient manneras compared to XP where no design diagrams and documents are used (Fig. 8).

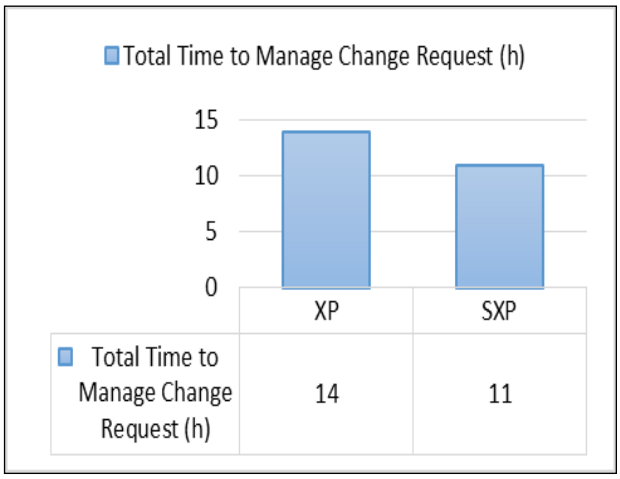


Fig.8Total Time to Manage Change Request (Hours)

## 5. Conclusion

Agile models have explored the new directions of software development. Extreme programming (XP) is one of the widely used agile model which have the ability to accommodate changing requirements with good level of customer satisfaction. Although XP uses best software engineering practices however it has some drawbacks as well. Due to these limitations it is not considered suitable for medium and large scale projects. To overcome its limitations many customized version of XP were presented by different researchers. Simplified Extreme Programming (SXP) model was also a contribution in this regard which targeted to eliminate maximum limitations of classical XP without effecting its simplicity and agility. This study empirically compared proposed SXP and classical XP. For empirical comparison, two small scale projects of same nature were developed using SXP and XP respectively. Results of selected quality parameters were compared and it was concluded that the SXP performed far better than classical XP. For future work, it is suggested that the proposed SXP model should be tested further in medium scale complex projects.

## References

[1] L. Williams, "Agile software development methodologies and practices," in *Advances in Computers*, vol. 80, Elsevier Inc., pp.1-44, 2010.

[2] M. Fowler and J. Highsmith, "The agile manifesto." *Software Development*, vol. 9, no. 8, pp. 28-35, 2001.

[3] F. Anwer, S. Aftab, U. Waheed, and S. S. M. Shah, "Comparative Analysis of Two Popular Agile Process Models: Extreme Programming and Scrum,*"International Journal of Computer Science and Telecommunications* vol. 8, no. 2, March 2017.

[4] D. Cohen, M. Lindvall, and P. Costa, "An introduction to agile methods." *ADVANCES IN COMPUTERS*, vol. 62, pp.1- 66, 2004.

[5] F. Anwer, S. Aftab, U. Waheed, and S. S. Muhammad, "Agile Software Development Models TDD, FDD, DSDM, and Crystal Methods: A Survey," *International Journal of Multidisciplinary Sciences and Engineering*, vol. 8, no. 2, 2017.

[6] P. Abrahamsson, O. Salo, J. Ronkainen and J. Warsta, "Agile software development methods: Review and analysis," *VTTpubl.*,pp. 3-107 2002.

[7] T. Dyba, and T. Dingsoyr, "Empirical studies of agile software development: A systematic review," *Information and Software Technology*, vol. 50, no. 9-10, pp. 833-859, 2008.

[8] E. Mnkandla, and B. Dwolatzky, "A survey of agile methodologies," *The transactions of the SA institute of electrical engineers*, vol. 3, pp.236-247, Dec. 2004.

[9] E. R. Mahajan and E. P. Kaur, "Extreme Programming: Newly Acclaimed Agile System Development Process," *International Journal of Information Technology*, vol. 3, no. 2, pp.699-705, 2010.

[10] R. Crocker, "The 5 reasons XP can't scale and what to do about them," *Proceedings of XP*, 2001.

[11] A. Dalalah, "Extreme Programming: Strengths and Weaknesses," *Computer Technology and Application*, vol. 5, no. 1, 2014.

[12] S. Beecham, H. Sharp, N. Baddoo, T. Hall and H. Robinson, "Does the XP environment meet the motivational needs of the software developer? An empirical study," in *Agile Conference (AGILE), 2007* pp. 37-49, IEEE.

[13] K. S. Choi and F. P. Deek, "Extreme Programming Too Extreme," *New Jersey Institute of Technology*, 2002.

[14] F. Anwer and S. Aftab, "SXP: Simplified Extreme Programing Process Model," *International Journal of Modern Education and Computer Science (IJMECS)*, vol. 9, no. 6, pp. 25-31, 2017.

[15] S. Nagalambika, R. Majunath and K.S. Praveen, "Component Based Software Architecture Refinement and Refactoring Method in Extreme Programming,"*International Journal of Advanced Research in Computer and Communication Engineering*, vol. 5, no. 12, 2016.

[16] M. R. J. Qureshi and J. S. Ikram, "Proposal of Enhanced Extreme Programming Model," *International Journal of Information Engineering and Electronic Business*, vol. 7, no. 1, p.37- 42, 2015.

[17] T. Saeed, S.S. Muhammad, M.A. Fahiem, S. Ahamd, M.T. Pervezand and A.B. Dogar, "Mapping Formal Methods to Extreme Programming (XP)–A Futuristic Approach," *International Journal of Natural and Engineering Sciences*, vol. 8, no. 3, pp.35-42, 2014.

[18] J. Choudhari and U. Suman, "Extended iterative maintenance life cycle using eXtreme programming," *ACM SIGSOFT Software Engineering Notes*, vol. 39, no. 1, pp.1-12, 2014

[19] N. Iqbal, M.U. Hassan, A.R. Osman and M. Ahmad, "A framework for partial implementation of PSP in Extreme programming,"*International Journal of Engineering Research and Applications*, vol. 3, no. 2, pp.604-60, 2013.

[20] M. Qureshi, "Estimation of the new agile XP process model for medium-scale projects using industrial case studies,"*arXiv preprint arXiv: 1408.6228*, 2014.

[21] S. Alshehri and L. Benedicenti, "Prioritizing CRC cards as a simple design tool in extreme programming," in *Electrical and Computer Engineering* (CCECE), Regina SK, 2013.

[22] M.R.J. Qureshi, "Agile software development methodology for medium and large projects,"*IET software*, vol. 6, no. 4, pp.358-363, 2012.

[23] S. Musa, N. Norwawi, M. Selamat and K. Sharif, "Improved Extreme Programming Methodology with Inbuilt Security," in *Computers & Informatics* (ISCI), Kuala Lumpur , 2011.

[24] K. Beck, “Extreme programming explained: embrace change,” addison-wesley professional, 2000.
[25] S. Shahzad, “Learning from experience: The analysis of an extreme programming process,”Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference, pp. 1405-1410, IEEE, 2009.
[26] R. Fojtik, “Extreme Programming in development of specific software,” Procedia Computer Science, vol. 3, pp.1464-1468, 2011.
[27] T. Dudziak, “eXtreme programming an overview,” Methoden und Werkzeuge der Softwareproduktion WS, 2000/1999, pp.1-28.
[28] R. Juric, “Extreme programming and its development practices,” in. Proc. 22nd Int. Conf. Information TechnologyInterfaces, IEEE, pp. 97-104, Jun. 2000.
[29] J. Newkirk, “Introduction to agile processes and extreme programming,” in Proc. 24th Int. conf. Software engineering, pp. 695-696, 2002.
[30] O. Kobayashi, M. Kawabata, M. Sakai and E. Parkinson, “Analysis of the interaction between practices for introducing XP effectively,” in Proc. 28th Int. conf. Softw. Eng, pp. 544-550, May 2006.
[31] Z. Nawaz, S. Aftab and F. Anwer, “Simplified FDD Process Model,” International Journal of Modern Education and Computer Science (IJMECS), vol. 9, no.9, pp. 53-59, 2017.
[32] F. Anwer, S. Aftab and I. Ali, “Proposal of Tailored Extreme Programming Model for Small Projects,” International Journal of Computer Applications (IJCA), vol. 171, no. 7, pp. 23-27, 2017.
[33] G. Rasool, S. Aftab, S. Hussain and D. Streitferdt, “eXRUP: A Hybrid Software Development Model for Small to Medium Scale Projects,” Journal of Software Engineering and Applications, vol.6, no. 09, p.446, 2013.
[34] S. Ashraf and S. Aftab, “Latest Transformations in Scrum: A State of the Art Review,” International Journal of Modern Education and Computer Science (IJMECS), vol. 9, no.7, pp.12-22, 2017.
[35] F. Anwer and S. Aftab, "Latest Customizations of XP: A Systematic Literature Review,” International Journal of Modern Education and Computer Science (IJMECS), vol. 9, no. 12, pp. 26-37, 2017.
[36] S. Aftab, Z. Nawaz, M. Anwar, F. Anwer, M. S. Bashir, and M. Ahmad, "Comparative Analysis of FDD and SFDD," International Journal of Computer Science and Network Security (IJCSNS), vol. 18, no. 1, pp. 63-70, 2018.
[37] S. Ashraf and S. Aftab, "Pragmatic Evaluation of IScrum," International Journal of Modern Education and Computer Science (IJMECS), vol. 10, no. 1, pp. 24-35, 2018.