# Augmenting Personal Software Process Analysis For Extreme Programming Teams and Perspective Implication

[1]Shahbaz Ahmed Khan Ghayyur, [2]Abdul Razzaq, [3]Asim Khalil

[1]Department of Computer Science & Software Engineering, Faculty of Basic and Applied Sciences, International Islamic University, Islamabad, PAKISTAN
[2]Department of Information Science Technology & Engineering, Zhejiang University, Zhoushan, China
Department of Computer Science & Software Engineering, Faculty of Basic and Applied Sciences, International Islamic University, Islamabad, PAKISTAN

**Abstract**
The Personal Software Process offers individuals with a particular self-controlled structure for doing a job. To improve individual and team ability is a crucial source of productivity and quality. Measuring an individual's performance is a challenging task in an agile environment as individuals work on many different projects at the same time. No specific criteria exist, which gives personal growth in agile XP. This research study is based on an idea to align the personal software process with agile extreme programming and propose a new model for an individual's professional growth measurement. An evidence-based case study is conducted to accumulate knowledge about the measurement of an individual's performance in the agile extreme programming team. In this paper, systematic mapping is used to collect issues in existence literature. The reason for systematic mapping is needed to recap the enhancement and need to classify the holes also requirements for upcoming studies related to agile with process improvement. This study supports to understand the variance between SPS and XP. This researcher mapping creates awareness about the process improvement with the combination of SPS and XP.

***Keywords***
*Personal software process, PSP, agile, individual improvement in agile, agile and PSP, collaborative software process vs personal software process, XP practicing, PSP and XP*

## 1. Introduction

### 1.1 Personal Software Process

The Personal Software Process (PSPSM) provides engineers with a disciplined personal framework for doing work[1]. The personal software process delivers a way to software engineers to improve the quality, productivity, and probability of their work[2] . PSP aimed to address the improvement of the needs of individual engineers. This helps engineers to understand the quality of their work and to appreciate the effectiveness of the methods they use [2]. PSP was introduced by Watts Humphrey as "A Discipline" for Software Engineering[1]. The PSP process comprises a set of methods, script, and forms that provides a structured approach to individuals to plan and manage their work[2]. The Personal Software Process (PSP) [3] is a controlled framework by which software engineers can plan, track, and manage defects earlier to produce high quality products[3] [1]. The main objective is to produce a quality product with a zero-defect product when individuals use PSP [2]. There are a lot of discrepancies found in the traditional model; a new model "Agile" was introduced to cater to traditional model problems [4]. The Personal Software Process (PSP) [3] is a planned framework by which individual engineers to plan, track, and manage defects earlier to produce high quality products [1]. Data Collection and analysis is evidence of nominal process implementation in PSP[3].

### 1.2 Extreme Programming

A new method called "Agile" and went on to overcome problems with conventional ways [3]. The traditional approach to a different way of adoption. For frequent are the methodologies under the canopy of "Agile" is more than a standard methodology extreme Programming (XP). A set of ideals and XP principles and a deep, quickly develop software that offers quality is perfect competition. Russell Oberlin programming (XP) is lightweight, agile methods to the elastic developers 2 to 10 [15]. They are based on a set of beliefs, XP nearby values, principles, and practices [16]. Values are concepts described to a high level, while these practices are that in the steps taken by the values [16]. Value to the troops bears responsibility values practices. The principles have been the gap between the practical values and a set of rules and practices used to describe the value's victory [17]. Value has five XP, communication, leisure, feedback, and a resolution to the resection [16] [17] the twelve practices [18]. Advantages of XP and a faster processor that includes software development, better quality, better customer satisfaction, and highly respected development teams motivated [19].

Manuscript received December 5, 2025
Manuscript revised December 20, 2025
https://doi.org/10.22937/IJCSNS.2025.25.12.21

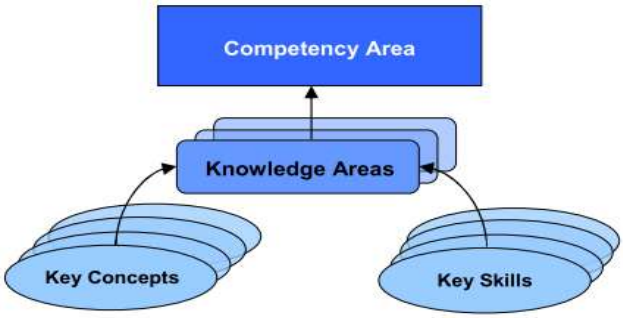



**Figure. 1.** Knowledge area classification [5]

### 1.3 PSP & XP

PSP covers different competency areas, and each competency area further classifies into knowledge areas, as mentioned in Fig. 1. Agile software development can state many specific software development methods, the best known Scrum and XP [5]. XP is a lightweight, agile, rebates, and methodology for developers 2 to 10 [6]. A set of XP has established principles and practices values [7]. The benefits of using the XP software development process include faster time to market form more in-depth, more motivated development teams customer satisfaction and honored [8].

### 1.4 The Need for Systematic Mapping

In this paper, systematic mapping was explored to find problems that are faced during the measurement of an individual's performance [57]. PSP quality principles were explored during systematic mapping, which can be used for an individual's performance measurement in an agile team [58].

High-maturity development processes, making intensive use of metrics and quantitative methods, such as the Personal Software Process (PSP), can generate a significant amount of data to identify performance problems [28]. One collective blame of the software engineering occupation is the weak value of the products that software engineer harvests[37]. The uses of PSP concepts and procedures in their work, engineers in nearly any technical area can increase their planning abilities, the value of their work, and decrease the number of faults in their yields[41] since Personal Software process trusts profoundly on the group and analysis of private data as evidence of effective procedure execution[49]. The necessary concepts' descriptions and PSP's services can help the software specialists in evaluating their own abilities and services and in classifying areas in which they need of enhancement [4].

## 2. Literature & Background Knowledge

The Personal Software Process (PSP) was introduced by Watts Humphrey as "A Discipline" for Software Engineers[1] and a controlled framework by which software engineers can plan, track and manage defects earlier to produce high quality products[1]. Improvements in organizational performance could also result from process improvement by individual engineers. Humphrey recognized that an organization reaches a maturity level beyond which organizational process improvement requires individual process improvement.

Level0 is mainly the present process the individual engineer practice to write programs and software and to provide performance measurements. PSP level0 was improved to Level 0.1 by the addition of size measurement, coding standard, and personal process improvement. Level 0.1 also improves package size capacity, individually count procedures and methods. After Level 0.1, Level 1.0 enhances planning to Level 0, and the early step adds size and reserve estimation. In Level 1.1, planning and status tracking are introduced. In Level 2, a wide-ranging and constant emphasis is sited on refining the individual's capability to quality program production. The main objective is to produce quality work in a consistent way. The intent of introducing Level 2.1 is not to provide a designing process but to tell engineers how to state the criteria for design accomplishment. The Level 3 process presents methods for individuals to developing large scale programs[9].

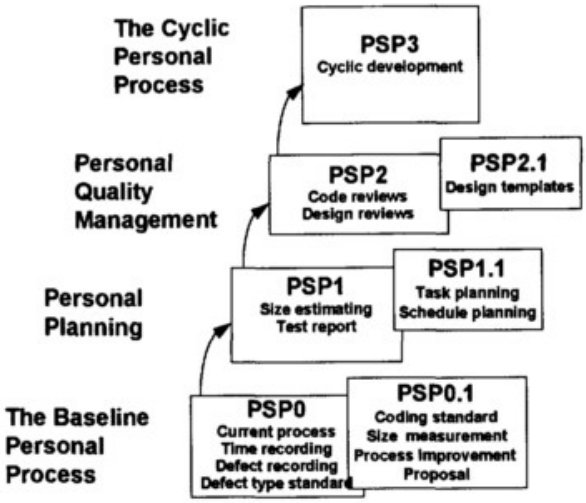



**Figure. 2.** Personal Software Process Phases[9]

## 3. Research Method

### 3.1 Planing of Mapping

This mapping study is highlighting the significant issues that arise due to a lack of process for individual developers to measure their growth in the agile XP team. This mapping will help us to evaluate the benefits [57] that can be achieved if the personal software process introduced in the agile XP team to measure an individual's performance and growth. This study will also help us to find out problems and issues that arise when aligning PSP with the agile XP process.

### 3.2 Research Questions

RQ1. What problems occurred during the integration of the Personal Software Process with an agile extreme programming team? This question has discussed in table 6 and table 8.

RQ2. What challenges have reported in the literature about XP? This question has discussed in table 10.

RQ3. What are the primary motivations for using XP? Motivations of XP have discussed in table 9.

RQ4. What challenges have reported in the literature about PSP? Challenges have discussed in table 12.

RQ5. What are the primary motivations for using PSP? Motivations of PSP have discussed in table 11.

### 3.3 Search Strategy

for searching primary studies, different digital Computing databases are used. In all electronic databases, the Researcher uses search strings, keywords.

### 3.4 Keywords

These keywords are used for searching the studies. {Personal software process*}, {PSP*}, {agile}, {individual improvement in agile}, {agile and PSP}, {collaborative software process vs personal software process}, {personal software process problems*}, {XP practicing}, {PSP and XP}

### 3.5 Search String(s)

***Initial Search String:***

(((Agile AND (XP OR Extreme programming)) AND (personal quality measures OR individual quality measures OR person quality measures)) AND (PSP OR personal software process))

***Final Search String:***

(((Agile OR XP OR Extreme programming) AND personal software process) OR individual OR Solo OR single) AND (PSP)

### 3.6 Primary study

After the text processing is complete, the paper is ready for the template. Duplicate the template file using the Save As command, and use the naming convention that is required by your conference for the name of your paper. In this newly created file, you select all content and import your prepared text file. You are now ready to style your paper; Use the scroll-down window on the left side of the MS Word formatting toolbar.

### 3.7 Search Engine

This paper used the advanced search feature of digital libraries for Search strings. Databases that are selected for retrieving relevant articles are shown in below figure 3 and years wise are shown in figure 4.

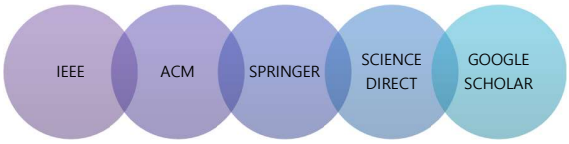



**Figure. 3.** Databases for Paper Selection

### 3.8 Inclusion Criteria

Following inclusion criteria were followed while making a decision.

- Research papers relevant to agile and personal software process (PSP) were included as primary studies
- Research papers relevant to agile XP issues were included
- Research papers relevant to personal and collaborative software process benefits were included
- Research papers which are based on the expert opinion were included
- Research papers that are relevant to individual development in agile XP are included.

### 3.9 Exclusion Criteria

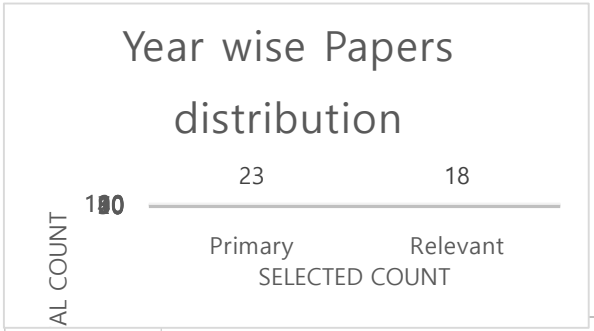
These papers were excluded.

- Books were excluded
- Papers other than primary and irrelevant studies

### 3.10 Conducting mapping

With search string(s) in hand and conferences that were selected for search, 114 papers were selected in total that meet the inclusion criteria for primary study [Table 1]. With the further evaluation of these papers, the Researcher has comprised different types of studies that are most suitable to the problem in hand and filtered papers to 18, as mentioned in Table 1. The first study mentioned in Table 2 and the relevant study mentioned in Table 3. The reference description of primary studies is in Table 4, and the reference description of relevant studies is in Table 5. Aligned parameters extraction detail is in Table 7.

**Table 1.** Publication count




| Databases | Publication Count | | |
|---|---|---|---|
| | *Selected* | *Primary* | *Relevant* |
| IEEE | 52 | 12 | 10 |
| ACM | 35 | 5 | 8 |
| SD[1] | 3 | 1 | 1 |
| Springer | 14 | 1 | 1 |
| Scopus | 0 | 0 | 1 |
| GS[2] | 10 | 3 | 0 |
| **Total** | **114** | **22** | **18** |

TABLE 2. PRIMARY STUDY COUNT

| Database | Primary Study Count |
|---|---|
| IEEE | 12 |
| ACM | 5 |
| Science Direct | 1 |
| Springer | 1 |
| Scopus | 1 |
| GS | 3 |
| **Total** | **23** |

**TABLE 3**. RELEVANT STUDY COUNT

| Database | Relevant Study Count |
|---|---|
| IEEE | 8 |
| ACM | 6 |
| Science Direct | 1 |
| Springer | 1 |
| Scopus | 0 |
| GS | 2 |
| **Total** | **18** |

**TABLE 4**. PRIMARY STUDY REFENRECES

| No | Reference | Primary study |
|---|---|---|
| 1 | [10] | SCRUM-PSP: Embracing Process Agility and Discipline |
| 2 | [11] | Method for personal capability assessment in agile teams using personal points |
| 3 | [12] | Incorporating Lean Development Practices into Agile Software Development |
| 4 | [13] | Faat – Freelance as a Team |
| 5 | [14] | Assessing Individual Performance in Agile Undergraduate Software Engineering Teams |
| 6 | [15] | Extreme Programming for a Single Person Team |
| 7 | [16] | Personal Quality Management with the Personal Software Process |
| 8 | [17] | PSP PAIR: Automated Personal Software Process Performance Analysis and Improvement Recommendation |
| 9 | [18] | Understanding Self-organizing Teams in Agile Software Development |
| 10 | [19] | Trends in Software Process: The PSP and Agile Methods |
| 11 | [20] | Scrum Solo |
| 12 | [21] | Light-Weight Development Method : a Case Study |
| 13 | [22] | There are different types of multiple case studies impact pair programming on the quality of the product. |
| 14 | [23] | Comparing Extreme Programming and Waterfall Project Results |

| | | |
|---|---|---|
| 15 | [24] | Deconstructing Agile Processes: Would Planned Design Be Helpful in XP Projects? |
| 16 | [25] | Factors that Impact Implementing an Agile Software Development Methodology |
| 17 | [35] | A Comparison between Agile and Traditional Software Development Methodologies |
| 18 | [26] | Personal Extreme Programming–An Agile Process for Autonomous Developers |
| 19 | [27] | The Agile Samurai |
| 20 | [28] | Software Defect Detection and Process Improvement using Personal software Process Data M.GOPICHAND |
| 21 | [29] | A Comparative Study of Agile Methods: XP versus SCRUM |

**TABLE 5**. RELEVANT STUDY REFERENCES

| No | Reference | Relevant study |
|---|---|---|
| 1 | [30] | An experiment measuring the effects of Personal Software Process (PSP) training |
| 2 | [31] | An empirical study of the application of PSP methodology with students of a Systems Technology program with different levels of training |
| 3 | [32] | Agile Software Development: The People Factor |
| 4 | [33] | Empirical Study for Improving Personal Software Process Education by Pairing |
| 5 | [34] | Towards Individualized Software Engineering: Empirical Studies Should Collect Psychometrics |
| 6 | [35] | Analysis of Linear Sequential and Extreme Programming Development Methodology for a Gaming Application |
| 7 | [36] | Tool Support for Personal Software Process |
| 8 | [37] | Agile methodologies and plan driver with ccomparison of Student Experiences |
| 9 | [38] | Process ( TSP ): An Overview and Preliminary Results Practices |
| 10 | [39] | Research and improvement of Team software process |
| 11 | [40] | Popular Agile Approaches in Software Development : Review and Analysis |
| 12 | [41] | Agile For Large Scale Projects – A Hybrid Approach |
| 13 | [42] | Literature review of the challenges of developing secure software using the agile approach |
| 14 | [43] | An Empirical Investigation on Effort Estimation in Agile Global Software Development |
| 15 | [44] | Extreme Programming Explained , Second Edition |
| 16 | [45] | Service agile development using XP |
| 17 | [46] | Traditional vs agile development: A comparison using chaos theory |

**Table 6.** Aligning and Conflicting PSP's Concepts with agile XP Team (Aligning Concepts)

| **PSP Key Concepts** | **Aligned with XP principle** |
|---|---|
| PSP Code reviews and PSP standards | XP's Coding Standards practice. |
| PSP's Software Design's Knowledge area | This practice reflects the YAGNI principle for efficient application of templates in your system, in accordance with XP's simple design practice.[47] |
| Right PSP form | This practice is not explicitly described by XP principles and practices, although it is aligned with the XP philosophies of "if you need to do it" and using the most appropriate tool or technique for the job at hand.[48] |
| Tracking software quality | This is a reflection of XP's Testing practice with respect to modeling |
| PSP's process extension knowledge area | This is a modeling-specific practice. XP developers can clearly work on several models—such as CRC cards, acceptance test cases, and sketches |
| Quality measure and methods | Aligned with the refactoring principle of XP. [48] |
| Making and tracking project plans | Aligned with the principle of XP's Planning game |
| User stories, scripts, and guidelines | User stories and requirements [29] |
| Reviews after every phase | Aligned with the Retrospectives practice of XP, in which after each iteration, the team does a short reflection on what went well during [49] |

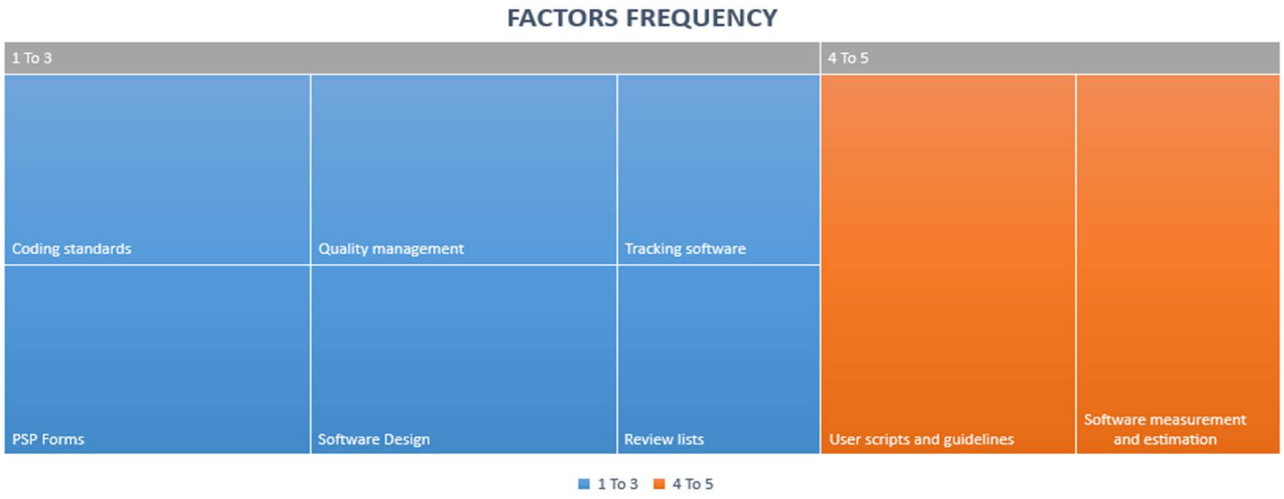



**Figure. 5.** Factors Frequency based on papers

In this segment, the Researcher utilized a Minitab software of statics to find the variation analysis of the elements of XP and PSP. This device makes the distinctive four charts in light of the investigation result [57]. Diagram of scree plot inform us concerning variety amongst each factor and show it by a tapped line that where the variety is happening and the amount it is. Analyst dissects the aftereffect of components by values; these qualities imply that what number of times each factor is utilized as a part of writing that is considered an estimation of the factor for each paper. That is to say; figure 6 will discover the significance of factor and writing center around factor. Analyst connected the consequences of all variables in Appendix 'A' area factor variation result's table. What's more, the Appendix B segment demonstrates the result in various charts.

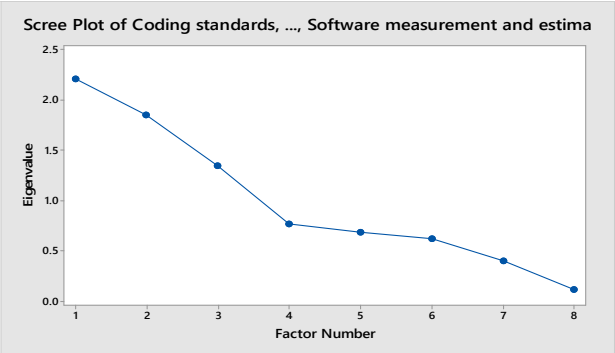



**Figure 6**. Factors list analysis variation graph

**Table 8.** Conflicting concepts between PSP and XP

| PSP Key Concepts | Conflicted with XP Principles |
|---|---|
| Single person programming | Conflicted with the principle of XP Pair Programming |
| Existence of single person ownership | Conflicted with XP's Collective Ownership practice. |
| The complete finished product in a traditional style development | Conflicted with the XP's iterative and increment approach to development.[49] |
| Being traditional in nature | Conflicted with being iterative in nature[53] |
| The complete product is the outcome | Small releases[15] |

**Table 9.** Pros and Cons of XP

| Advantages | Disadvantages |
|---|---|
| • XP encourages pair programming.<br>• Quality planning helps properly trained teams to produce high-quality products | • Having two programmers on the same computer<br>• XP programming is not structured<br>• Extreme Programming is hard to do |

**Table 10.** Challenges in XP

| No. | Challenges | Description | Ref. |
|---|---|---|---|
| 01 | **Requires high investment** | Organizations need to prepare the entire programming improvement group! Also that the rate of XP preparing is considerably more costly than Scrum preparing. | [56] |
| 02 | **Infrastructure** | To run XP, you require speculation for computerized tests and a nonstop conveyance framework. | [56] |
| 03 | **Culture** | You additionally need to change the association culture by uniting various divisions. | [56] |

| | | | |
|---|---|---|---|
| 04 | **XP Coaches** | Employing XP mentors is costly that is on the grounds that they are programming craftsman. A craftsman is costly in light of the fact that they think about quality, and they are optimists. | [56] |
| 05 | **Irrational** | Non-specialized individuals lead most organizations, even IT organizations. Numerous practices in XP is viewed as silly from representatives point of view. | [56] |
| 06 | **Pair programming** | Having two programming engineers on a comparable PC wearing down a comparable segment is extraordinary to pros. For what reason would you waste money for two designers finishing a specific something. | [56] |

**Table 11**. PSP Pros & Cons

| Advantages | Disadvantages |
|---|---|
| • PSP helps engineers how to make better size and resource estimates using statistical techniques and historical data [54]. | • Gathering measurements is a period of the costly undertaking. This is an issue since programming ventures are regularly late, and there is no opportunity to spend in exercises that don't create prompt advantages. [54]<br>• Manual information accumulation is a temperamental movement. An excessive number of blunders or missing information seriously influence the investigation procedure. [54] |

**Table 12.** Challenges in PSP

| No. | Challenges | Ref. |
|---|---|---|
| 01 | **Data Quality Problems in the Personal Software Process** | **[55]** |
| 02 | **Metrics collection and analysis** | **[54]** |
| 03 | **Metrics data exchange** | **[54]** |

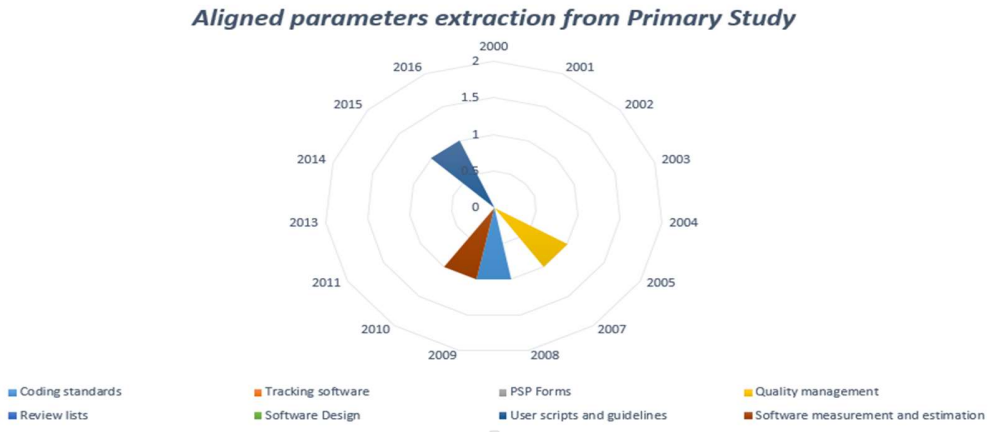



**Figure 7.** Parameters according to years

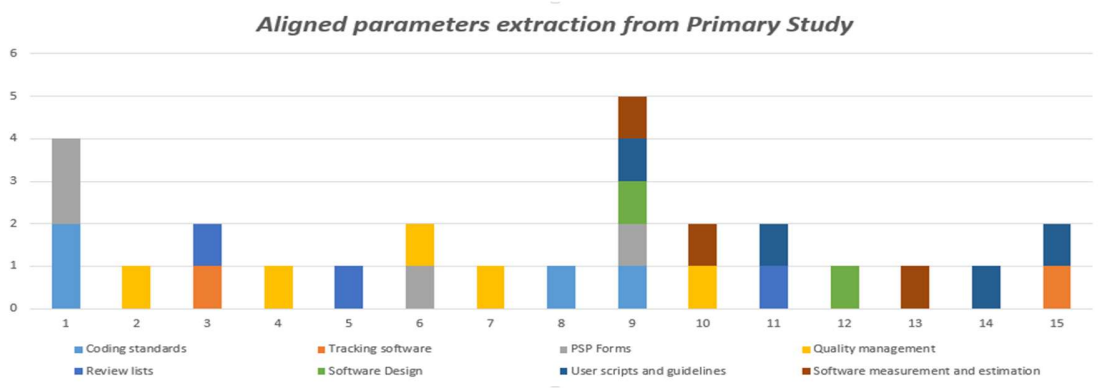



**Figure 8:** Parameters according to papers in years

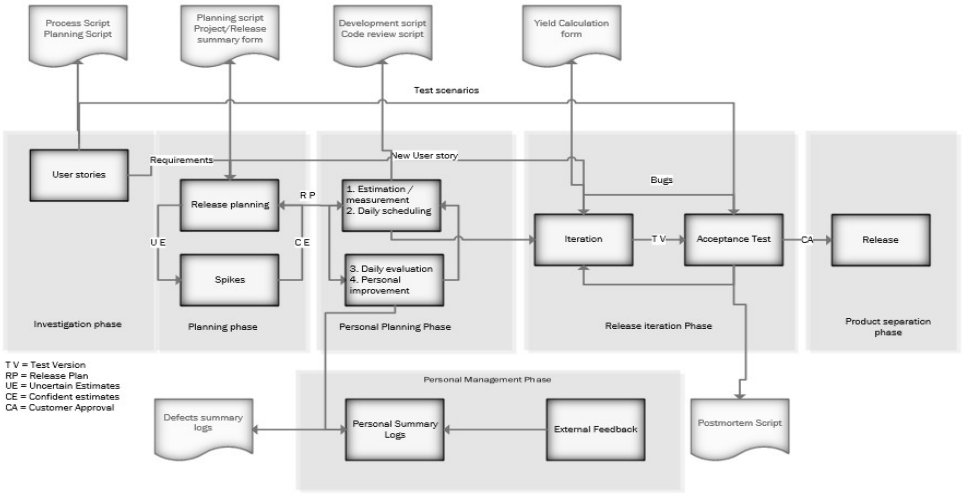



**Figure 9**. Proposed Model

**Benefits of Combining PSP with Agile**. Figure 9, It mainly concludes six points about the benefits of using PSP to complement the Agile. These six benefits are listed as follows:

1. PSP provides the metrics and data collection mechanisms.
2. PSP gives the support to make it a more valuable estimation.
3. PSP provides the benefits to optimize plans.
4. PSP supports to produce the software products up to quality goals.
5. PSP gives additional documentation for agile methods.
6. PSP helps individual engineers to improve their personal performance gradually.

The quality PSP principles that are used for individual growth measurement are;

- Defect density
- Review rate
- Development time ratios
- Defect ratios
- Yield

## 5. Conclusions

This basic assessment and mapping study has inspected painstakingly the given investigations on PSP for XP and the essential engineering challenges revealed in writing. The analyst has talked about in insights about PSP with XP. Compose the arranging of mapping concentrate to deliver the outcomes of how mapping will have appeared in this investigation and the real catchphrases that help to discover the writing identified with PSP and XP. Research Questions is the real piece of this examination. These are affected by after effect of this investigation. RQs have discussed through the table for data. Every engineer has a different nature, and every engineer plans its own work, and make plans for personal data. In this systematic mapping study first, the Researcher collects relevant publications from electronic databases. Then, apply the inclusion and exclusion criteria to extract the main and relevant study. After the final selection, the Researcher applies the methodology of systematic mapping. In this systematic mapping, the Researcher described the concepts of alignment and conflict between the PSP and the agile XP team and discussed all the concepts from 2000 to 2017. Figure 2 shows all phases of PSP. Figure 5 shows the frequency among all factors. Minitab statics software is used to find the variation of the factors of PSP & XP which are mentioned in figure 6 and table 7, and create the result in order to quantitative values and different view that graphs are and the significant graph is Scree plot figure 6 of this analysis, Other graphs and results are shown in Appendix section which are the figure 9, figure 10, and figure 11. Extracted parameters are shown in figure 7. Year wise aligning of parameters are shown in figure 8.

## 6. Future Work

Assessed material depends on the best in class inquire about mined for process improvement and productivity with quality of PSP & XP. The researcher will use the proposed model to process improvement that suggests a new and improved software development model for small and medium scale software industry. The proposed model approach attempt between the light and substantial methodology to improves the software developer productivity and quality. This improved inducts the process in XP, which targets individual performance; however, it will keep the significant practices of XP so that it can be fit for a small development team.

The researcher undertakes the study to improve/evaluate the tailored XP process by integrating the personal software process (PSP) support that was initially part of traditional software development. Since the personal software process is a burdensome activity, which is against the agile core principles so a lightweight version of PSP has proposed, and evaluation will be made on this tailored XP process. There are very few studies available that validate and measure the impact of PSP on the agile XP process. The objective of this study is to evaluate the impact of integrating and aligning PSP with agile XP methodology to measure the individual's performance and growth in an agile XP team

## References


[1] R. L. Cannon, "Putting the Personal Software ProcessSM into practice," Proc. 12th Conf. Softw. Eng. Educ. Train. (Cat. No.PR00131), pp. 34–37, 1999.

[2] W. S. Humphrey, "The Personal Software Process ( PSP ) The Personal Software Process SM ( PSP SM )," Engineering, no. November, 2000.

[3] M. Pomeroy-Huff and T. A. Chick, "USE BY SOFTWARE DEVELOPMENT PROFESSIONALS The," Area, vol. 2009, no. August 2009, p. 92, 2010.

[4] N. Iqbal, M. Hassan, and A. R. Osman, "A framework for partial implementation of PSP in Extreme programming," vol. 3, no. 2, pp. 604–607, 2013.

[5] E. Woods, "Aligning Architecture Work with Agile Teams," IEEE Softw., pp. 24–26, 2015.

[6] R. Juric and T. Interfaces, "WestminsterResearch Extreme programming and its development practices .," 2000.

[7] A. Jansen and J. Bosch, "Software Architecture as a Set of Architectural Design Decisions," 5th Work. IEEE/IFIP Conf. Softw. Archit., pp. 109–120, 2005.

[8] E. Abdullah and E. T. B. Abdelsatir, "Extreme programming applied in a large-scale distributed system," Proc. - 2013 Int. Conf. Comput. Electr. Electron. Eng. 'Research Makes a Differ. ICCEEE 2013, pp. 442–446, 2013.

[9] W. S. Humphrey, "The personal process in software engineering," Proc. Third Int. Conf. Softw. Process. Appl. Softw. Process, no. c, pp. 69–77, 1994.

[10] G. Rong, D. Shao, and H. Zhang, "SCRUM-PSP: Embracing process agility and discipline," Proc. - Asia-Pacific Softw. Eng. Conf. APSEC, pp. 316–325, 2010.

[11] S. Celar, M. Turic, and L. Vickovic, "Method for personal capability assessment in agile teams using personal points," Telecommun. Forum Telfor (TELFOR), 2014 22nd, pp. 1134–1137, 2014.

[12] J. Sutherland and J. Tabaka, "Incorporating lean development practices into agile software development," Proc. Annu. Hawaii Int. Conf. Syst. Sci., p. 7695, 2007.

[13] R. B. Bernabé, I. Á. Navia, and F. J. García-Peñalvo, "Faat," Proc. 3rd Int. Conf. Technol. Ecosyst. Enhancing Multicult. - TEEM '15, pp. 687–694, 2015.

[14] R. F. Gamble and M. L. Hale, "Assessing individual performance in Agile undergraduate software engineering teams," Proc. - Front. Educ. Conf. FIE, pp. 1678–1684, 2013.

[15] R. Agarwal and D. Umphress, "Extreme programming for a single person team," Proc. 46th Annu. Southeast Reg. Conf. XX - ACM-SE 46, no. March, p. 82, 2008.

[16] A. S. Koch, "Personal quality management with the personal software process," Softw. Eng. Educ. Conf. Proc., p. 43, 2001.

[17] C. B. Duarte, J. P. Faria, and M. Raza, "PSP PAIR: Automated personal software process performance analysis and improvement recommendation," Proc. - 2012 8th Int. Conf. Qual. Inf. Commun. Technol. QUATIC 2012, pp. 131–136, 2012.

[18] N. B. Moe, T. Dingsøyr, and T. Dybå, "Understanding self-organizing teams in agile software development," Proc. Aust. Softw. Eng. Conf. ASWEC, no. 3, pp. 76–85, 2008.

[19] J. A. Stark and R. Crocker, "Trends in software process: The PSP and agile methods," IEEE Softw., vol. 20, no. 3, pp. 89–91, 2003.

[20] J. A. Fabri and A. L. Erario, "Scrum Solo."

[21] J. Q. Chen, D. Phan, B. Wang, and D. R. Vogel, "Light-Weight Development Method : a Case Study," 2007.

[22] H. Hulkko, "S8a ~ SUR |."

[23] T. Sedano, “Comparing Extreme Programming and Waterfall Project Results,” pp. 482–486, 2011.

[24] R. Noël, G. Valdes, M. Visconti, and H. Astudillo, “Deconstructing Agile Processes : Would Planned Design Be Helpful in XP Projects ?,” pp. 42–51, 2008.

[25] J. A. Livermore, “Factors that Impact Implementing an Agile Software Development Methodology,” pp. 82–86, 2007.

[26] Y. Dzhurov, I. Krasteva, and S. Ilieva, “Personal Extreme Programming–An Agile Process for Autonomous Developers,” Int. Conf. SOFTWARE, Serv. Semant. Technol., no. May 2016, p. 252, 2009.

[27] F. Breu, S. Guggenbichler, and J. Wollmann, The Agile Samurai. 2008.

[28] M. Gopichand, V. Swetha, and a. Ananda Rao, “Software defect detection and process improvement using personal software process data,” 2010 Int. Conf. Commun. Control Comput. Technol., pp. 794–799, 2010.

[29] Mohammad Almseidin, Khaled Alrfou, Nidal Alnidami, and Ahmed Tarawneh, “A Comparative Study of Agile Methods: XP versus SCRUM,” Int. J. Comput. Sci. Softw. Eng., vol. 4, no. 5, pp. 126–129, 2015.

[30] L. Prechelt and B. Unger, “An experiment measuring the effects of Personal Software Process (PSP) training,” IEEE Trans. Softw. Eng., vol. 27, no. 5, pp. 465–472, 2001.

[31] J. F. Niño Manrique and R. Anaya, “An empirical study of application of PSP methodology with students of a Systems Technology program with different levels of training,” 38th Lat. Am. Conf. Informatics, CLEI 2012 - Conf. Proc., 2012.

[32] A. Cockburn, “Development :,” no. November, pp. 131–133, 2001.

[33] G. Rong, H. Zhang, M. Xie, and D. Shao, “Improving PSP education by pairing: An empirical study,” Proc. - Int. Conf. Softw. Eng., pp. 1245–1254, 2012.

[34] R. Feldt, L. Angelis, and M. Samuelsson, “Towards individualized software engineering: empirical studies should collect psychometrics,” Proc. 2008 Int. Work. Coop. Hum. Asp. Softw. Eng., pp. 49–52, 2008.

[35] P. Sharma and N. Hasteer, “Analysis of Linear Sequential and Extreme Programming Development Methodology for a Gaming Application,” pp. 1916–1920, 2016.

[36] J. Lappalainen, “LNCS 3547 - Tool Support for Personal Software Process,” pp. 545–559, 2005.

[37] C. A. Wellington, T. Briggs, and C. D. Girard, “Comparison of Student Experiences with Plan-Driven and Agile Methodologies,” pp. 18–23, 2005.

[38] T. Report, “Process ( TSP ): An Overview and Preliminary Results Practices,” vol. 2000, no. October, 2001.

[39] P. Ri, R. Q. J. X. Q. X, L. Dqj, H. S. W. Ri, V. Qjlqhhulqj, R. I. Qjlqhhulqj, P. K. B. X. Frp, D. Vfkhphv, F. Dqg, H. Wr, L. Wkh, F. Phwkrg, F. Wkh, L. Ri, D. Q. G. Sudfwlfh, and V. Ri, “5hvhdufk dqg ,psuryhphqw ri 7hdp 6riwzduh 3urfhvv,” 2009.

[40] A. Mahdi, M. Hamed, and H. Abushama, “Popular Agile Approaches in Software Development : Review and Analysis,” pp. 160–166, 2013.

[41] M. Tanveer, “Agile For Large Scale Projects – A Hybrid Approach,” no. Nsec, pp. 14–18, 2015.

[42] H. . Oueslati, M. M. . Rahman, and L. B. . Othmane, “Literature review of the challenges of developing secure software using the agile approach,” Proc. - 10th Int. Conf. Availability, Reliab. Secur. ARES 2015, pp. 540–547, 2015.

[43] R. Britto, E. Mendes, and J. Borstler, “An Empirical Investigation on Effort Estimation in Agile Global Software Development,” Proc. - 2015 IEEE 10th Int. Conf. Glob. Softw. Eng. ICGSE 2015, pp. 38–45, 2015.

[44] K. Beck, Extreme Programming Explained , Second Edition. .

[45] F. Carvalho and L. G. Azevedo, “Service agile development using XP,” Proc. - 2013 IEEE 7th Int. Symp. Serv. Syst. Eng. SOSE 2013, pp. 254–259, 2013.

[46] D. M. Shawky, “Traditional vs agile development: A comparison using chaos theory,” ICSOFT-PT 2014 - Proc. 9th Int. Conf. Softw. Paradig. Trends, pp. 109–114, 2014.

[47] C. Lan, K. Mohan, X. Peng, and B. Ramesh, “How extreme does extreme programming have to be? adapting XP practices to large-scale projects,” 37th Annu. Hawaii Int. Conf. Syst. Sci. 2004. Proc., vol. 0, no. C, pp. 83–92, 2004.

[48] M. A. Awad, “A Comparison between Agile and Traditional Software Development Methodologies,” Univ. West. Aust., vol. 1, pp. 1–300, 2005.

[49] L. Lindstrom and R. Jeffries, “Extreme programming and agile software development methodologies,” Inf. Syst. Manag., vol. 21, pp. 41–52, 2004.

[50] S. Ambler, Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process. 2002.

[51] Y. Chaiyo and S. Ramingwong, “The development of a design tool for personal software process (PSP),” 2013 10th Int. Conf. Electr. Eng. Comput. Telecommun. Inf. Technol. ECTI-CON 2013, 2013.

[52] C. A. D. L. Salge and N. Berente, “Pair programming vs. solo programming: What do we know after 15 years of research?,” Proc. Annu. Hawaii Int. Conf. Syst. Sci., vol. 2016–March, pp. 5398–5406, 2016.

[53] B. Boehm and R. Turner, “Management challenges to implementing agile processes in traditional development organizations,” IEEE Softw., vol. 22, no. 5, pp. 30–39, 2005.