

## Agile Methodologies: Comparative Study and Future Direction

GUL AHMAD  
TARIQ RAHIM SOOMRO\*  
MOHAMMAD NAWAZ BROHI  
Department of IT, SZABIST, Dubai Campus  
United Arab Emirates

### Abstract:

*Several iterative and incremental methods exist to control projects, but currently agile technologies are increasing admirably. Agile Project Management provides an approach that offers not only agility, but retains the concepts of a project, project delivery and project management. This study will explore the agile methodologies focusing on three popular methodologies, such as extreme programming, scrum and rational unified process. Comparisons among these three methodologies and finally discussion on the findings of these comparisons and future direction will be explored.*

**Key words:** Agile Methodology, eXtreme Programming, Scrum and RUP.

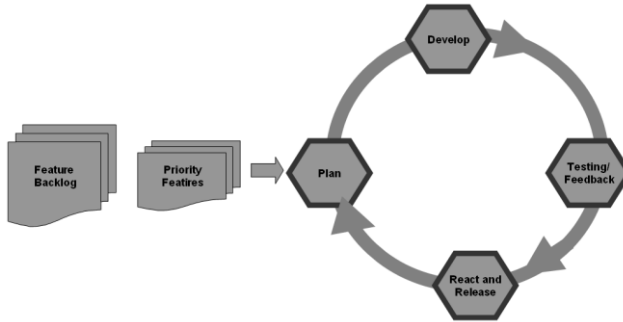
### 1. Introduction

The term agile comes from agility, which means the system capability to reactively and proactively response to the fluctuations and modifications in terms of increasing the business income and productivity. In software project management world Agile Project Management (APM) follows the same carrier path of agility which considers the terms of project development and delivery to the customers. APM has

---

\* Corresponding author: tariqsoomro@ieee.org

the capability to work with other similar disciplines like PRINCE2, CMMI and ISO90001 processes of measuring quality, which qualifies the APM to identify the hidden risks in projects. Agile collaborates between the existing best practices, for example, project management best practices, which are already adopted by the organization [1][2][3]. Agile combines the techniques of iterative development culture and models the listed processes with effective use of human resources expertise, knowledge and experience in order to meet the target within the budget and time. Agile is the systematic and tested process of delivering proper solutions to customer [1]. APM is an approach, which is following the Dynamic Software Development Method (DSDM), “Atern” project management elements and representing as agile project management practices. APM enables the project managers who have experience in class project management to easily adopt agile models for project management within their organizations. Agile is the mature, flexible, changes adoptable model, which empowers the team of the project to boost business value by delivering rapid and reliable solutions according to the customers’ needs. In the business world, customers always have an obsession about rapid feedback, continuous improvement and rapid delivery of services, which are not possible without adopting agile models. APM is an iterative incremental approach for planning and development of projects. Figure-1 shows the agile life cycle, which starts with the stakeholder requirements as input to write big epic, which are user’s stories [4]. These stories or epic need to be broken into small pieces or parts to apply the law of “Divide and Rule”. Then the small stories or pieces are sorted by customers’ priorities and put in a backlog. Agile Software Development (ASD) is an iterative model in which all phases of the project are interconnected; here each phase output is the input for the next phase [2] [4] [5].



**Figure-1: Agile lifecycle**

This paper is organized as follows: section 2 will discuss the agile methodologies in general and extreme programming, scrum and rational unified processes in particular; section 3 will explore these comparisons; and finally section 4 will conclude by discussing along with future direction.

## **2. Agile Methodologies**

The following are the popular Agile Methodologies [2]:

- Extreme Programming (XP)
- Scrum
- Rational Unified Process (RUP)
- Dynamic Systems Development Method (DSDM)
- Crystal Methods (Crystal Clear)
- Lean Development (LD)
- Adaptive Software Development (ASD)
- Kanban
- Agile Modeling
- Rapid Product Development (PRD)
- Feature-Driven Development (FDD)

This research study is limited to only three agile models, their explanation and comparison: Extreme Programming (XP), Scrum and RUP (Rational Unified Process). All these three have

some common characteristics and principles.

### **A. Extreme Programming (XP)**

XP is an agile model which believes in continuous communication between project team members and stakeholders, continuous feedback, simplicity and respect. The most important is the communication principle which encourages the team members to share their knowledge and communicate with client in the best form. Simplicity helps software programmers to look for simple solutions to the problems faced in the project life. In order to think about future problems solution developers also build up a design feature to help them in future. Continuous communication between customers and project team enables the developers to carry on the project on right direction to fulfill the client requirements and support reducing paper work.

Agility nature is based on changes which encourage developers to accept changes collaboratively. In Agile model team members have a positive behavior to encourage and give respect to each other's work. The members of the team respect work from the others and try not to depress the others which leads to good and polite working environment [6] [7]. XP is the bundle of small pieces in which individual piece makes no sense individually but after being assembled and given full paradigm of the project. This is the main difference between traditional/corporate software management methods and the XP management method. XP has a focus on the code written regardless of time factor. In the initial phase of a project, the main focus is on the core product and features are coming in the later phases but coding is an important area where to focus all the time. The software design depends on the evaluation of software changes.

The simplicity principle of XP prefers that the design should be simple, as much as possible, for the present state, not for future needs. Design and analysis practices are applied to

later phases of the project, in the beginning these not being considered as more important. In XP one builds a story card where the user story is written. Each piece or part in a story card is explained before programming a story card. User Story (US) is the assets of customer which is contained on software system requirement (SSR) in small paragraphs in such a language that is known by customer. US are a short way to defined requirements of the stakeholder regardless of managing a big case of documents. Like the corporate project management models, XP also describes the following four variables for a project: cost, time, quality and scope. However, in XP, one of these is always ignored by project managers - this variable is the quality because if the project manager wants that his team to collect some user's requirements with limited resources and to meet the requirements on a specified time frame, then he has to compromise on quality. In XP projects work under stress is always involved and pressure always sacrifices the quality factor. Therefore project managers establish only three parameters in XP [6][8][9].

XP believes on small groups in the development team, ten to twelve people each. The cost variability depends on resources when advanced technology or season experts are required in some areas; under these circumstances the cost of the project will increase. XP aims to maintain higher quality of the project within less possible development time. For this reason the development assigns the task of testing because they have the ideas about the problems and their solutions coming during testing. The development team should be flexible to accept changes in code and design according to the user's requirements in such a manner to maintain quality [6]. XP is based on the iterative and incremental development model which is designed to deliver high quality software according to the customer's needs on deterministic time frame. The main XP objective is to make high quality software within a less possible time. XP believes on the pair programming approach because

the coding is written and review at the same movement and the client should be involved in the development process because he is the one who know the real requirements. The code should be simple and understandable and it will be rewritten to improve the quality as well as performance. The most significant factor which is involved here is the coding which should be common to all within the development team, which every team member can understand, modify or can test. The project team performs the testing before they are moving forward for adding new features or functionalities and each iteration having the phase of code is tested. In other words, they are revising the code at the same movement when you are writing it. The development team share all their work and reuse the same piece of code in other areas or modules of the project. XP has less documentation concept, not the same like user story, but in the end of the project it will rule out. For this reason, the user story can be considered documentation. This is the reason why XP is suitable for small projects with fewer members in the working team, in which less implementation risks are involved. XP is best for small projects but poor performance in medium and large development projects is the limitation of XP [6][7][10][11].

## ***B. SCRUM***

Scrum is an iterative incremental agile practice developed for quick developments and it is used in organizations that are not dependent on deadlines. Scrum provides a methodology that regulates the set of best practices in such a way to work together and gain the beneficial possible outcomes of a project. Like XP, Scrum also believes in simplicity and it is very easy to understand the project implementation processes. This methodology is based on iteration, which is a short round or cycle of repetitive constructs in which each cycle ends with a functional output. Scrum is most suitable for complicated projects environments in which the team is required to gain prompt results, requirements are changing more frequently,

requirements are poorly defined and innovation and productivity are essential. Scrum is a good solution for the situation where the client does not do his/her requirements, when deliveries are delayed, cost are more and there is a poorer quality. It is suitable for competitive situations and when it is essential to identify and resolve the issues systematically [6]. The Scrum development is based on iteration, which is a short cycle of repetitive steps and each cycle is of two to four weeks duration, which is called sprint. Each sprint incorporates a new functionality or feature for the project and their requirements and priorities are reviewed and adjusted to the life cycle of the project in short intervals on regular basis. The regular delivery of scrum sprints incorporates new features and functionalities in the project, playing a vital role in customer satisfaction and meeting customer requirements. This is adapted in real time applications which seek to truly fulfill the customer's needs. The scrum team always focuses on the delivery of qualitative software to meet the client's requirements.

Scrum contains small simple codes of conduct; they are based on principles of regular inspection, innovation, motivation, local management and adaptation. Scrum development base projects are available for everybody in the team, and everyone can review and can get the work of other members. The team's main focus is to achieve the targets within the specified time schedule. Therefore in scrum methodology meetings are important to share the work status with other members of the team. In scrum every sprint starts and ends with a meeting and also the scrum team attempts a meeting on a daily basis to know about each person in the team - what he/she is doing now, what issues he/she is facing and what is his/her next step. On a regular basis, the client is getting response from the project team, because his product is growing with the releasing of sprints, therefore he/she is exited and committed to the project. On the other hand, the project team are learning and getting skills from the environment

which causes motivation in the team. The project developers' team always focuses on coping with achievements which are regulated and aligned to the client's business goals [6][8][12][13].

### ***C. Rational Unified Process "RUP"***

RUP is also like XP and Scrum; it is an incremental and iterative approach. RUP is a centric architectural approach based on principles of software engineering. RUP is widely used along with Unified Modeling Language (UML) for analysis, implementation and documentation of object oriented system. RUP is a generalized process using different types of software systems in various application areas in many types of organizations with many levels of competition and different sizes of projects. RUP is describing a generic structure and is suitable for all kind and size of projects. RUP is also very suitable for user's case to development commercial software products which can continue for many months or years. These big projects are divided into small tasks, each one being an iteration that leads to a new increment in the project. The iteration is referring to the workflow step by step, growth in the product or project. For a successful system the team should know the clients' demands and requirements prospects [6][14][15]. RUP is an approach to make sure that the delivery of qualitative software is in accordance with the end users' prospective and within the budget and scheduled time. RUP is generally represented in two dimensions, which are the horizontal axis and the vertical axis. The horizontal axis represents the process lifecycle based on schedule throughout the development phase of the project. On the other hand, the vertical axis represents the logical grouping of different activities depending on their nature. The two dimensions of the process are representing milestones, phases and iteration in each phase, which are also called dynamic aspects of the process. The second dimension describes the roles in project,



artifacts, activities, disciplines, components and workflow, which are also called static aspects of the process [6][16]. Artifacts are playing an important role in the documentation phase. Different artifacts are done in different phases of the development lifecycle: for example, the elaboration phase uses case diagrams, the construction phase uses class and sequence diagram. In project management, the documents are the complete plan and flow of the project. In project management, the activity is defined as the work in which a person is playing a role but on the other hand, the process is the combination of one or more workflows. RUP is very strict to the plan which is described in documentation [6].

### 3. Comparison (XP, SCRUM, RUP)

The comparison process of XP, SCRUM and RUP depends on their nature and style of each; it also depends on assumptions, description, presentation and structure. This research also explains what the eventual goals are for each process or model and the cost of each use. Some of these feature are shared by all or some processes make a systematic evaluation doable. They require sequences or groups of activities, which are performed by roles to generate artifacts or work products, some or all of which are delivered to a customer. Most processes also acknowledge that instances of the process will have a time dimension, with a beginning and an end, and interesting intermediate milestones that represent the completion of significant activities (or clusters of activities) and the production of associated artifacts [17].

	<b>XP (Development practices)</b>	<b>RUP (Process Framework)</b>	<b>Scrum (Management Strategy)</b>
Objective area	The Customer Drives It's all about the Code	Well Engineered Development Management	Communication and Empowerment Team/Management Interactions
General Perceptions	Cool, trusts the developers, lets me "do my	Manageable, traceable, visible, the way things "should be done"	

	thing” Formalized hacking, waste of resources, no management control	Heavyweight, boring, waste of time	
Basic Principles	<ul style="list-style-type: none"> <li>▪ Rapid Feedback</li> <li>▪ Incremental Change</li> <li>▪ Quality Work</li> <li>▪ Fine Scale Feedback</li> <li>▪ Shared understanding</li> <li>▪ Programmer Welfare</li> <li>▪ Fine Scale Feedback</li> <li>▪ Continuous Process</li> <li>▪ Communication</li> <li>▪ Simplicity</li> <li>▪ Courage/Aggression</li> </ul>	<ul style="list-style-type: none"> <li>▪ Use Case Driven</li> <li>▪ Architecture Centric</li> <li>▪ Iterative, Incremental</li> <li>▪ Uses UML</li> <li>▪ Can be Tailored to a Project’s needs</li> <li>▪ Develop Software</li> <li>▪ Manage Requirements</li> <li>▪ Use component-based</li> <li>▪ Visually Model</li> <li>▪ Verify quality</li> <li>▪ Control Changes</li> </ul>	<ul style="list-style-type: none"> <li>▪ Commitment</li> <li>▪ Focus openness</li> <li>▪ Respect, Courage</li> <li>▪ Built-in instability</li> <li>▪ Self-organizing project teams</li> <li>▪ Overlapping development phases</li> <li>▪ Multi-learning</li> <li>▪ Subtle control</li> <li>▪ Transfer of learning</li> <li>▪ Visibility</li> <li>▪ Communications</li> <li>▪ Impediments</li> <li>▪ No Interference</li> </ul>
Real Motivation	Good clean code is easy to change Customers make all business decisions Developers make all technical decisions Make iterations as short as possible so that Customer can drive with rapid feedback	Making mistakes is expensive, so make sure you know what you want to do before you do it A good solid system has a good solid architecture Plans and Budgets are hard to follow, things change, so be iterative and incremental	Changes may be hard to make, so identify them as soon as possible Developers know how to develop, so just stay out of their way and let them do it Make 30-day
Common Activities	Produce User Stories Release Planning Planning Game (iteration planning) Development Spike Solution Run Acceptance Tests	Consists of cycles Development phases, Project Management, analysis, requirements, Design, Code, Test Activity <ul style="list-style-type: none"> <li>▪ Analysis and Design</li> <li>▪ Business Modeling</li> <li>▪ Implementation</li> <li>▪ Project Management</li> <li>▪ Requirements</li> <li>▪ Test</li> </ul>	Sprint Planning Sprint Daily Scrum Sprint Review
Player / Roles	<ul style="list-style-type: none"> <li>▪ Programmer</li> <li>▪ Customer</li> <li>▪ Tester</li> <li>▪ Coach</li> <li>▪ Tracker</li> <li>▪ Manager</li> <li>▪ Consultant</li> <li>▪ Big Boss</li> </ul>	<ul style="list-style-type: none"> <li>▪ Project Manager</li> <li>▪ Business-Process</li> <li>▪ System Analyst</li> <li>▪ Architect</li> <li>▪ User</li> <li>▪ Reviewer</li> <li>▪ Designer</li> <li>▪ System Integrator</li> <li>▪ Implementer</li> <li>▪ Tester</li> </ul>	<ul style="list-style-type: none"> <li>▪ Scrum Master</li> <li>▪ Scrum Manager</li> <li>▪ Product Owner</li> <li>▪ Scrum Team</li> </ul>
Products	<ul style="list-style-type: none"> <li>▪ Release Plan</li> <li>▪ User Story</li> </ul>	<ul style="list-style-type: none"> <li>▪ Business Case</li> <li>▪ Software</li> </ul>	<ul style="list-style-type: none"> <li>▪ Sprint Backlog</li> <li>▪ Release Backlog</li> </ul>

	<ul style="list-style-type: none"> <li>▪ Code</li> <li>▪ Unit Tests</li> <li>▪ Acceptance Tests</li> </ul>	<ul style="list-style-type: none"> <li>▪ Iteration</li> <li>▪ Analysis Model</li> <li>▪ Design Model</li> <li>▪ Deployment Model</li> <li>▪ Implementation</li> <li>▪ Test Model</li> <li>▪ Architecture</li> </ul>	<ul style="list-style-type: none"> <li>▪ Product Backlog</li> </ul>
Approach	Iterative Incremental	Iterative	Iterative Incremental
Cycle	The length of an iteration in XP is usually 1-3 weeks	Formal Cycle is defined across 4 phases, but a few concurrent workflows are possible too	Each sprint (iteration) is a complete cycle.
Planning	Called Planning Game like release planning, iteration planning	Formal project plan is associated with multiple iterations as well as end-date driven plan is possible along with intermediate milestones.	Project plan, iteration plan is determined at the end of the current iteration. Product Owner determines when the project is done.
Artifacts	<p>Stories</p> <p>Constraints</p> <p>Tasks</p> <p>Acceptance tests</p> <p>Coding</p> <p>Releases</p> <p>Metaphors</p> <p>UML design</p> <p>Documentation</p> <p>Standards</p> <p>Unit tests</p> <p>Planning</p> <p>Release plan</p> <p>Iteration plan</p> <p>Test results</p> <p>Spikes (solutions)</p> <p>Resources</p> <p>Scope</p> <p>Quality</p> <p>Time</p> <p>Tracking results</p>	<p>Vision</p> <p>Glossary</p> <p>Use-Case Model</p> <p>Software Architecture</p> <p>Specifications</p> <p>Supplementary</p> <p>Implementation Model</p> <p>Test Model</p> <p>Product</p> <p>Release Notes</p> <p>Document</p> <p>Design Model</p> <p>Programming</p> <p>Software Development Plan</p> <p>Iteration Plan</p> <p>Business Case</p> <p>Risk List</p> <p>Change Requests</p> <p>Configuration Plan</p> <p>Templates</p>	Operational software is the only formal artifact.
Type of Project / Product	Small projects	With medium to high complexity it is suggested for large, long-term, enterprise-level projects.	Recommended for quick developments for types of organizations that are not deadline dependent.
Scope		Scope is predefined ahead of the project start and documented. Scope may be revised during the project under the strictly controlled procedures.	SCRUM uses a Project Backlog, instead of scope. It is re-evaluated at each sprint.
Practices	<p>The Planning Game</p> <p>Small Releases</p> <p>Simple Design</p> <p>Metaphors</p> <p>Refactoring</p> <p>Testing</p>	<p>Business Modeling,</p> <p>Requirements,</p> <p>Analysis and Design,</p> <p>Implementation,</p> <p>Test,</p> <p>Deployment,</p>	<p>Identify Product Backlog</p> <p>Identify and Remove Impediments</p> <p>Define Sprint Backlog</p> <p>No Interference, no Intruders, no Peddlers</p>

	Pair Programming Collective ownership Continuous integration 40-hour week On-site customer Coding standards	Configuration and Change Management, Project Management, and Environment	Frequent, First-Hand observations
--	--	---	--------------------------------------

**Table 1.1 Comparisons of XP, RUP and Scrum [17] [18]**

#### 4. Discussion and Future Work

In the software world today agile methodologies are the most popular due to delivery of high quality software projects in a shortest possible time according to the customer's needs and business objective. Agile methodologies' main focus is on stakeholder satisfaction and user requirements as well as on organization objectives. XP, Scrum and RUP are the most widely adapted practices of agile development methodologies. Each of them has positive features along with some shortcomings as well. Like Scrum having limitations in engineering practices, XP does not provide much more about management, while the RUP model also has drawbacks, such as for small projects, slow response to frequently changes in requirements and tends to be over budgeted. We are aware that Scrum is a framework, which is not providing proper plan for software development, like a development process to define Software Development Life Cycle. In Scrum practices most of the activities related to project management are done by the developer team, so the team knows better about all activities in the development environment and the problems in the project. The Scrum management being assigned to developer team of the software project, along with combination of XP engineering practices, reduces the rework cost and effort that are considered a burden in the software industries.

As we are also aware of RUP methodology, that is suitable for medium and large scale projects. On the other hand RUP model has some shortcomings as well, such as rapidly requirement changes not being recommended in RUP,

which tends to be slow and over-budgeted; reputation in the sense of small and fast paced projects are substandard. Therefore the combination of the strengths of Scrum and XP into the new proposed model, both are incremental and iterative in nature as well self-managed processes, which will solve the shortcoming of RUP, which is slow in adaptation to frequent changes of requirements. On the other hand, the RUP model strength is to focus on customer satisfactions and business objective by producing and maintaining high quality software projects and provides a comprehensive system plan to the organization. The main shortcoming of XP model is the poor documentation, which gives results in slow performance for medium and large scale projects. This is solved by implementing the RUP model phases in the new proposed XSR model [19][20].

The objective of this study is to propose a novice hybrid model to increase the capability of software industry to produce high quality in the software projects on time and within budget. The hybrid model will consist of the best strengths of XP, Scrum and RUP such as XP providing the best software engineering practices, Scrum offering a best project management features and RUP strengths focusing on business objective and customer satisfactions.

The main strengths of our novice proposed model are to combine the good features of XP, Scrum and RUP and to decrease their shortcomings to provide such a model to produce quality software and give response of requirements changing smartly. Our new proposed model XSR (XP, Scrum and RUP) is intended to combine the management features of software project in Scrum, coding and standard strengths in XP and RUP features of customer satisfaction and focus on organization objective, which will result in producing software products of high quality and thus meeting the business objectives. The main idea, behind creating XSR model, is to have a development methodology that has the capabilities to

produce products of high quality and less shortcomings rate. The new proposed XSR model needs to be tested in the real time project environment. XSR model also further requires proper validation on a small as well as on large scale projects. In the future XSR can be used in outsourcing environments of software development. Further research is needed to extract other good practices of XP, Scrum and RUP to be experienced and it can be a good future research work in this study area.

## **BIBLIOGRAPHY:**

- Augustine, Sanjiv. 2005. *Managing Agile Projects*. Prentice Hall Professional Technical Reference [5]
- Bashir, M. Salman and M. Rizwan Jameel Qureshi. 2012. "Hybrid software development approach for small to medium scale projects: RUP, XP and Scrum." *Sci.Int. (Lahore)* 24(4): 381-384. [15]
- Beck, Kent. 2001. *Extreme Programming explained, embrace change*. Addison Wesley. [9]
- Czarnecki, Krzysztof. 1995. *Software Design and Architecture RUP and XP* [10]
- Fuster, Joan Escamilla and Juan Carlos Ruiz Garca. 2009/2010. *Modern Software Project Management*. University Politecnica De Valencia, Helsingin Yliopisto [6]
- Highsmith, Jim. 2002. *Agile Software Development Ecosystems*, Addison-Wesley [3]
- Jacobon, Ivar, James Rumbaugh, and Gredy Booch. 1999. *The Unified Software Development Process*. Addison Wesley. [16]
- Kniberg, Henrik. 2009. *Scrum and XP from the Trenches*. InfoQ. [8]
- Kruchten, Philippe. 2004. *The Rational Unified Process: An Introduction*. 3<sup>rd</sup> Ed. Addison Wesley. [14]

- Larson and Gray. 2012. *Agile Project Management: The managerial process*. McGraw-Hill [2]
- Lee, Jonathan and Kyle Sakimoto. October 2013. "What is SCRUM?" [http://www.scrumalliance.org/pages/what\\_is\\_scrum](http://www.scrumalliance.org/pages/what_is_scrum) [12]
- Mushtaq, Zaigham and M. Rizwan Jameel Qureshi. 2012. "Novel Hybrid Model: Integrating Scrum and XP." *I.J. Information Technology, and Computer Science* 6: 39-44. Published Online June 2012 in MECS (<http://www.mecspress.org/ijitcs/ijitcs-v4-n6/IJITCS-V4-N6-6.pdf>) [19]
- Nisa, Sadaf Un and M. Rizwan Jameel Qureshi. 2012. "Empirical Estimation of Hybrid Model: A Controlled Case Study." *I.J. Information Technology and Computer Science* 8: 43-50. Published Online July 2012 in MECS (<http://www.mecspress.org/ijitcs/ijitcs-v4-n8/IJITCS-V4-N8-5.pdf>) [20]
- Nisa, Sadaf Un and M. Rizwan Jameel Qureshi. 2012. "Empirical Estimation of Hybrid Model: A Controlled Case Study" *I.J. Information Technology and Computer Science* 8: 43-50. Published Online July 2012 in MECS (<http://www.mecspress.org/>) DOI: 10.5815/ijitcs.2012.08.05 [11]
- Pharro, Richard. 2011. Agile Project Management White Paper, CEO, APMG-International, Sword House High Wycombe, Buckinghamshire HP13 6DG [1]
- Rawsthorne, Dan. 2003. "Comparing/Combining RUP, XP and Scrum – mixing the process Cocktail, Net Objectives." <http://www.netobjectives.com> [18]
- Schwaber, Ken. 1987. SCRUM Development Process, 1 Brooks, F.P. "No silver bullet—essence and accidents of software engineering." *Computer* 20:4:10-19, April 1987. 2 Object Oriented Analysis and Design with Applications, p. 8, Grady Booch, The Benjamin/Cummings Publishing Company, Inc., 1994 [13]
- Sheuly, Sumsunnahar. 2013. "A Systematic literature review

on Agile Project Management.” Lappeenranta University of Technology, Department of Software Engineering and Information Management [4]

Stephens, Matt and Doug Rosenberg. 2003, *Extreme Programming refactored, The case against XP*. Apress. [7]

Smith, John. 2001. *A comparison of RUP and XP*, Rational Software Corporation, 18880 Hoestead Road Cupertino, CA 95014 [17]