A REVIEW OF AGILE METHODOLOGY IN SOFTWARE DEVELOPMENT

Karambir¹, Anjali Sharma²
¹Assistant Professor, Department of CSE, UIET, KUK, India
²Student of Department of CSE, UIET, KUK, India

Abstract – The Agile Methodologies were introduced to meets the new requirements of the software development. This paper presents a review of thirty papers and papers are based on agile methodologies including Extreme Programming (XP), Scrum, Crystal, Lean Software Development (LSD), Kanban Software Development (KSD), Feature Driven Development (FDD), Adaptive Software Development (ASD) and Dynamic Systems Development Method (DSDM), describes the differences between them and recommends when to use them. Agile Methodologies are the best software development approaches and these methodologies are very important for researchers.

Key words: Agile, Extreme Programming, Scrum, Crystal, lean Software Development.

1. INTRODUCTION

Agile means able to move quickly and easily and this is what an agile software development methodology refers to. An agile software development methodology fully accepted these days. It is an iterative approach to keep action with dynamic development environments. During the past years, a new software development approaches were introduced to fits new cultures of the software development companies. Most software companies today aim to produce high quality software in short time period with minimal costs, and within unstable, changing environments. Agile Methodologies were thus introduced to meet the new requirements of the software development companies [1].

Agile methodologies are used to develop and implement software quickly according to customer requirements. Agile SDMs (Software Development Methodologies) share several features including prototyping, iterative development, and minimal documentation. Agile software development methodologies are used to produce the high quality software in the shorter period of time. It is an alternative to the traditional project management used in software development. Agile software development is a methodology for creative process that expects the need for flexibility and applies a level of practicality into the delivery of the complete product. Agile software development focuses on keeping code simple, testing and delivering functional bits of the application. The goal of the ASD is to build upon small client approved parts as the project progresses as opposite to delivering one large application at the end of the project.

Agile software development encourages promotes adaptive planning, evolutionary development, continuous improvement, early delivery and promotes quick and flexible response to change. Agile software development (ASD) is a methodology for the creative process that expects the need for flexibility and applies a level of practicality into the delivery of the complete product. It focuses on keeping code simple, testing frequently, delivering functional bits of the application as soon as they are ready. The goal of ASD is to build upon small client approved parts as the project progresses, as opposed to delivering one large application at the end of the project. It is a lightweight software engineering framework that promotes iterative development during the life-cycle of the project.

2. RELATED WORK

Andreas Schmiertendorf [2] presented an analysis of the effort estimation possibilities within agile software development methodologies. Agile methods of the software development were increasingly used for industrial projects. The application of effort estimation methods in such kind of projects was very difficult, but an important task. Classical estimation methods were needed well defined requirements. It provided an investigation about estimation possibilities, especially for the extreme programming paradigm. The XP projects built the basis of the effort estimation. Xiaofeng Wang [3] demonstrated agile
methods such as Extreme Programming (XP) and Scrum to lean software development in the last several years, which was suggested as “from agile to lean”. The aim of this research was to investigate how agile and lean approaches had been merged in software development. The patterns of merging agile and lean in reports were described and categorised in a more universal way. Pankaj Kamthan [4] explained a number of differentiate changes were shown in industrial software engineering, including the movement towards agility. The agile manifesto identify the term “agile” and lists a set of principles that form a basis for agile methodologies, such as Extreme Programming (XP), Scrum, and OpenUP. The creation, consumption, and communication defined of knowledge during agile software development. For the success of an agile project, it was crucial that such knowledge be managed effectively. Fahad Almudarra and Basit Qureshi [5] demonstrated a mobile cloud based environment for content media management. It showed that Mobile Cloud application development integrated with agile development methodologies reduced the cost, time and improving software quality. Result showed the positive indications to applying Extreme Programming (XP) for both mobile and web application integrated with hybrid cloud. R. Steven Wingo and Murat M. Tanik [6] developed software development for complex problem domains was a difficult task with challenges to successful outcomes. The XP Methodology was a good choice for a skilled and disciplined software development team to use in creating software solutions for complex problem domains. For critical problem domains such as core business systems, XP used to deliver high-quality software that keeps up with a changing environment.

Maria Paasivaara and Casper Lassenius [7] described communities of practice groups of experts who shared a common interest and collectively want to deepen their knowledge. It described on how a large organization within Ericsson with 400 persons in 40 Scrum teams at three sites adopted the use of Communities of Practice (COP) as part of transformation from a traditional plan-driven organization to lean and agile. In the case organization, COPs were originally used to support the agile transformation, and as part of the distributed Scrum implementation. Sergio Galvan et al. [8] presented software process standards (e.g. ISO/IEC 12207, ISO/IEC 15504) and models (e.g. CMMI) provided a set of best practices and guidelines for improving the quality of the software. In this research, the particular issue of compliance of Agile Software Development Methodologies (SCRUM, XP, and UPEDU) and the new ISO/IEC 29110 standard was studied. The main findings indicate that the UPEDU and SCRUM methodologies were presented and the high compliance level with the ISO/IEC 29110 Project Management process. David P. Harvie et al. [9] presented software engineering and mission command was two separate but similar fields, as both were instances of complex problem solving in environments with ever changing requirements. The research hypothesis was that modifications to agile software development based on inspirations from mission command. Targeted Scrum was a modification of Traditional Scrum based on three inspirations from Mission Command: End State, Line of Effort, and Targeting. Martin Tomanek and Tomas Klima [10] introduced the penetration testing was embedded into the scrum framework that represents the most used agile software development framework. The scrum was helped to automate the penetration tests during the software development projects, incorporate the specific penetration tests into the regular software releases and improved the overall resistance of developing software. The research was focused on development of the penetration testing methodology PETA. Bruno Antunes [11] proposed an analyzed waterfall vs. agile methodology that used an agile methodology, scrum. It described a solution approach of a virtual team, as well as shows some examples of using the Microsoft Visual Team Foundation Server tools to address the challenges. Software development environment dictated speed, flexibility and a people centered focus.

Jeffrey A. Livermore [12] demonstrated the agile software development methodologies (SDMs) were developed by software developers for utilizing iterative development, prototyping and templates. The agile SDM was using an online survey. The data of survey was used to identify factors related to agile SDM implementation. Factors included training, management involvement access to external resources, and corporate size were found to impact implementation of an agile software development methodology. Torgeir Dingsoyr et al. [13] described the status and main challenges for research on agile software development and proposed a preliminary roadmap for research on agile software development. The preliminary roadmap served as a starting point for creating a common research agenda. It gave the brief overview of the current state of research in related core fields: Software engineering, information systems, and empirical software engineering. Tore Dyba and Torgeir Dingsoyr [14] described agile software development represented a major departure from traditional and plan-based approaches to software engineering. A
systematic review of empirical studies of agile software development was conducted. The search strategy was grouped into four themes: introduction and adoption, human and social factors, perceptions on agile methods, and comparative studies. Ying Wang et al. [15] demonstrated agile software development methods were widely used globally, especially in building information systems for enterprises. Software developing vendors were suppliers in an informationalization supply chain (ISC) of the customer. To analyze the impact of agile software development on the management of ISC, a brief introduction of agile software development methods were presented. Anfan Zuo et al. [16] demonstrated the formal method Refinement of Component and Object Systems (RCOS) was applied into the agile software development. The development process was divided into four stages. In it a formal method was applied into agile software development and investigates the method to adopt the RCOS in agile software development. Markus Kohlbacher et al. [17] proposed agile software development approaches successfully applied to support systems engineering projects in dynamic environment. The system engineering projects were developed products consisting of hardware and software and associated to the interaction effects between change in requirements and agile methods on customer satisfaction. Customer satisfaction was chosen as performance factor. Tobin J. Lehman and Akhilesh Sharma [18] explained the term of implementation to software development framework was “Software Development as a Service”. The subject matter experts (SMEs) specified the overall mission statement, offered problem statements and provide feature suggestions. In past years it had been a change from classical plan-based methods i.e. Waterfall, to a new agile methods. Siva Dorairaj et al. [19] described software development teams needed highly valuable knowledge to carry out knowledge-intensive development activities. Knowledge sharing was most difficult for distributed agile teams due to spatial, temporal, and cultural barriers, which negatively impact face-to-face interaction, communication and collaboration. Agile team was a cross-functional team that the team as a whole had the collection of skills required to execute software development activities and deliver business values to customers. Lotfi Ben Othmane et al. [20] proposed the agile software development approach did not stop ensuring the security of software increments produced at the end of iteration. It proposed a method for security reassurance of software increments and integrates security engineering activities into the agile software development process. The method enabled ensuring the delivery of secure software at the end of each iteration.

Johannes Holvitie et al. [21] mentioned a major reason for the popularity of agile and lean software methods. It contributed to the technical debt discussion by showing differences in assumed and indicated technical debt knowledge. Also, components closest to implementation and its maintenance were supposed to have the most positive effects on technical debt management. Hassan Soltan and Sherif Mostafa et al. [22] explained lean concept focused on eliminating non value added activities while agile discovering and responding to uncertain changes of the market. Both concepts were able to achieve strategic objectives (competitiveness, productivity, profitability, and survival) through improving the overall performance. Many research suggested that combining lean and agile via decoupling point increases the organization benefits. Tomohiro Hayata et al. [23] described the agile software development and lean architecture both had been examined in the software engineering field. Merging two approaches had also been investigated. The existing investigation and practices only present the principles but fail to develop a framework of the Software development. A framework developed about how primitive agile practices were fulfilled by introducing “lean” practices under the Data Context Interaction (DCI) paradigm. Lean practice was an end user-focused and value-centric system design.

Jitender Choudhari and Ugrasen Suman [24] demonstrated agile methodology applied analogy and expert opinion based estimation techniques such as planning poker but in software maintenance, historical data and experts did not present. A heuristic method was expected for the calculation of maintenance efforts. The SMEEM technique applied story points to calculate the volume of maintenance and value adjustment factors that was affecting story points for effort estimation. Rashmi Popli and Naresh Chauhan [25] described the estimation in Agile Software Development methods depended on an expert opinion and historical data of project for estimation of cost, size, effort and duration research on estimation had been conducted for decades with huge quantities of models and tools produced. With the proposed method in research work the correct velocity of the project calculated. Kamran Ghane et al. [26] demonstrated the agile methodologies break software development into shorter cycles that produce more detailed empirical data about process activities and attributes such as time, resource, cost and scope. The System introduced historical data to
model the error rate of activity estimates. It fitted them into probability distribution functions where such distributions were applied to future activity estimates to predict the actual future values using the Monte Carlo simulation. Sakshi Garg and Daya Gupta [27] proposed a new cost estimation model for agile software development projects. The methodology was found to think appropriate for agile projects as it uses constraint programming to openly check for satisfaction of agile manifestos. The methodology also used in case of unavailability of historical data or expert opinion. The proposed cost estimation approach increases the correctness and accuracy of estimates and it was applied for the Agile Software Development Projects. Aditi Panda et al. [28] described agile software development process had become famous in industries and substituting the traditional methods of software development. One popular approach of calculating effort of agile projects mathematically was the Story Point Approach (SPA). An effort had been made to enhance the prediction accuracy of agile software effort estimation process using SPA. For it, different types of neural networks General Regression Neural Network (GRNN), Probabilistic Neural Network (PNN), Group Method of Data Handling (GMDH) Polynomial Neural Network and Cascade-Correlation Neural Network was used.

3. CONCLUSION

Agile methodologies are widely used in a variety of software industry projects, their flexibility provides the means to deal with many common problems faced in the development of software systems. Agile methodologies provide some practices that facilitate communication between the developer and the customer. The main aim of agile methodologies is to deliver what is needed when it is needed. This paper presented agile methodologies that provide results according to customer satisfaction by early and continuous delivery of software. This paper also described what are difficulties faced to develop software in object oriented software development. Agile Methodologies have to be applied in order to overcome these problems.

As a future work, estimate the effort required during the agile software development using machine learning techniques. Agile Methodologies are the best software development approaches and these methodologies are very important for researchers and the researchers will continue research on this topic.

4. REFERENCES


[9] David P. Harvie, Graduate Student Member, IEEE, and Arvin Agah, Senior Member, IEEE, “Targeted Scrum: Applying Mission Command to Agile Software


