JOURNAL OF SOFTWARE ENGINEERING & INTELLIGENT SYSTEMS
ISSN 2518-8739
30 April 2020, Volume 5, Issue 1, JSEIS, CAOMEI Copyright © 2016-2020
www.jseis.org

AGILE SOFTWARE DEVELOPMENT METHODS AND CURRENT TRENDS

MUBARAK HIMMAT [1], AHMED SEEDAHMED ALI OSMAN [2]

Future University Khartoum, Sudan[1]
Al-Neelain University Khartoum, Sudan[2]
Email: barakamub@yahoo.com [1], aalageed@neelain.edu.sd [2]

OPEN ACCESS

Academic Editor: Abid Mehmood, Saudi Arabia.
Published: 30 April, 2020
Funding: Authors have no funds.
Article Type: Review Article

ABSTRACT

The last three decades era witnessed rapid development in the field of software, and a large number of software are produced and applied in different real-life aspects. The dependence on software have experienced a massive growth. There is a lot of work that has been done in the field of software development and many different methods and models are proposed to develop the software and software process. One of these concerned models is the agile model that is discussed in detail. This paper discusses the software development models in general and mainly it covers the Agile Software Development (ASD), moreover, it also discusses the ASD teams. There is a focus on agile methodologies and emerging themes in ASD, and there is a demonstration of the success rates of ASD methodologies, challenges and future trends.

Keywords: software development methods; agile software development.; SCRUM; lean; XP; Kanban;

1. INTRODUCTION

There is no doubt that the software development and its practices have gained the attention of the large number of software companies, software developers and computer scientists all over the world, and software development has evolved steadily over the years since computer has been invented till now decades. There are numerous models [1-3], methods [4, 5], tools [6, 7] and techniques that have been proposed to enhance its efficiency and effectiveness. The testing centric view of software development practices will be covered deeply and specifically this article reviews software development methodologies, identifies the current situation and the latest trends in the industry and discusses their implications. The practical applications of agile methods and techniques that have been proposed are used and discussed in many works to see their impact on the productivity and efficiency of software development dominate [8-10].

In the late 1990s, agile method for software development is proposed and originated [11], Agile approaches were created as a response to traditional software development models' perceived weaknesses. The Agile Software Development (ASD) refers to a group of software development methodologies based on iterative development, these methodologies are considered as the basic requirements and solutions evolve through collaboration between self-organizing cross-functional teams which will be discussed in the next sections. These methodologies consist of several principles of Agile Method that are shown in the Figure1 and have been proposed after getting an inspiration from Agile Manifesto [12].

The process of software development is considered as a critical task and it requires detailed and well-structured methods and tools that may help in defining the guidelines for the software development process. The good software development process model will lead to high quality software, and the software process plays a major role in developing high quality software. The software development models, such as the traditional software development models, like waterfall model and its extension such as Rational Unified Process (RUP), Verification and Validation model (V-Model) and Spiral Model are still dominant in software industry. Currently ASD is one of the common and mostly used software development model, the agile model was proposed in 2001 by a group of software developers that met in Utah to explore new and improved ways of software development [12], where they valued some criteria over other such as :

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Corresponding Author: Mubarak Himmat
Email Address: barakamub@yahoo.com

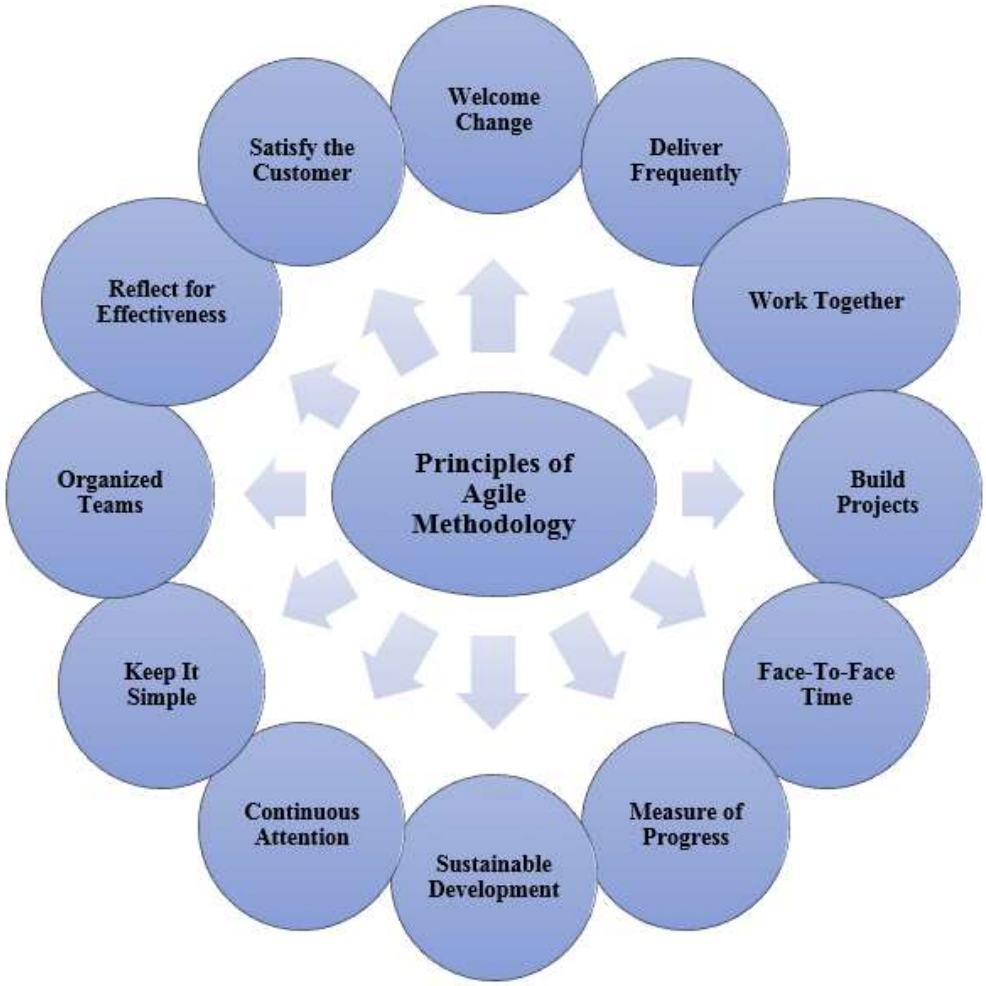

Figure. 1 Agile principles of agile methodology

The principles of Agile Methodology shown in Figure 1 try to facilitate the software development by speeding up and reducing the timescale of the software delivery, and provide better environment to build projects around motivated teamwork and individuals. Agile projects focus and concentrate on self-organizing teams who have a capability and are able to guide and manage the teamwork and the work at the same time. The agile method of face-to-face time conversation is considered an efficient and effective way of conveying information between development team [13]. Moreover, the Measure of Progress is a method that focuses mainly on the objective of the software project and measure the progress of the development. The Sustainable Development, that is promoted by Agile, enabled the developers, sponsors, and users to maintain a constant pace indefinitely. The Continuous Attention method in agile continuously focuses on technical excellence and good design, and it improves agility [13-15].

An innovative model is important, a solution that lasts the test of time is even more valuable. Perhaps a solution that has the ability to be updated to keep it up-to-date is even more important, Continuous attention results in technical excellence and good design in order to improve agility. Another important method is Keep It Simple. It is related with the art of optimizing the unfinished amount of work-essential. The most reliable apps, of course, are those that are not yet built ... they can't fail. But apart from that, nearly 30 percent of the features we create are rarely or never used. Agile is ruthless about cutting features and functions that don't give value. The important aspect of the agile model is the development of the teams and to Organized Teams through self-organizing teams emerge the best architectures, requirements, and designs [16, 17].

Cross-functional groups are self-organizing as well. It's better to spot challenges before they become actual obstacles; the people nearest to the solution, of course; also reflect effectiveness of methods whereas the team group focuses on how to become more successful at regular intervals, then monitors and changes adapt its behavior accordingly [18, 19]. Many methods have been developed to implement these principles, generally the agile methods tried to implement these principles by addressing the technical challenges or/and the management challenges.

2. AGILE METHODOLOGIES

Currently there are many agile methods in use, the most popular methods are described briefly in this section.

2.1. SCRUM

SCRUM is a framework to manage complex problems by applying the agile principles, it is lightweight and simple to understand [20]. SCRUM is built on three pillars namely transparency, inspection and adaptation [20]. The rules of SCRUM bind together the events, roles and artifacts. SCRUM develops the product incrementally through the sprint. The events of SCRUM are related to the sprint and start with the SCRUM planning meeting which set the plan for the coming sprint by selecting the features that can be implemented or realized during the sprint. During the sprint there should be the daily meeting called daily SCRUM, it is a follow up meeting to let the team to monitor and know the status. This meeting discusses what has been done from the last meeting, the plan for today and what are the impediments. Once the sprint is finished a sprint review meeting is conducted, the aim of this meeting is to evaluate what has been done during the sprint by the key stakeholders and the users of the product. SCRUM specified three main rules that are the SCRUM team, SCRUM master and the product owner. The SCRUM team is cross-functional and self-organized team there are no titles and all the team members are responsible for the result. The SCRUM master is a facilitator who helps the SCRUM team, product owner and the organization to apply the SCRUM and he is not a project manager. The product owner is responsible for the requirements and he is the customer representative. SCRUM defined two artifacts that are the product backlog and the sprint backlog. The product backlog is an ordered list that contains all the defined requirements, while the sprint backlog is the list that contains the requirements that will be implemented during the current sprint [20].

2.2. Lean software developments

Lean invented in the manufacturing environment to optimize the production line by minimizing the waste and maximizing the customer value, despite the clear difference between the manufacturing and the software development; but software development faces the same challenges. Lean has seven principles that can be applied in the area of software development [21]:

1- Eliminate waste: in software waste includes the extra features, functions or part of code.
2- Build quality in: developing quality software can only be achieved by emerging the quality in the processes followed to develop the software.
3- Create knowledge: sharing the knowledge and the lessons learned among the team and the organization is so important for the success of the project and the organization as well, this can be achieved through documentation and pair programming.
4- Defer commitment: by planning as per the needs of execution, the team is responsible for the decisions.
5- Deliver fast: this is achieved through the incremental delivery, that provides value for the customer.
6- Respect people: agile focuses more on people rather than process, this concept should be applied throughout the project and appears for example in decision-making process.
7- Optimize the whole: by adapting the practices based on what has been learned.

2.3. Extreme programming

Extreme Programming (XP) is one of the most popular agile methods, used since 1996. XP focuses more on the technical side of the software development applying the agile principles. XP is based on values which are: simplicity, communication, feedback, respect, and courage [22]. XP set rules for the different stages of the development that includes:

- Using the user stories to collect and manage the requirement.
- The releases should be small and frequent.
- The planning and design should be done incrementally.
- The team works in pair, and all the team works in an open workspace.
- Sustainable peace policy where the overtime work is not allowed.
- Maintaining the simplicity for both the processes that followed to develop the software and the software itself.
- Refactoring the software to enhance the quality.
- Continuous integration, the integration can be done many times in a day.
- Encouraging the collective ownership of the code.

Test driven development where the testing is done automatically and the testing code is developed before the development of the function itself.

2.4. Kanban

Focusing on visualizing the work in the knowledge-based projects, it is used as a mean to design, manage and improve the workflow. The values of Kanban are [23]:

- Transparency in communication and information / knowledge sharing.
- Balance the different aspects and viewpoints to reach better results.
- Collaboration between the team.
- Focus on the customer by delivering values to the customer.
- Flow of the work is continuous and should add value.
- Respecting the different views of the people and the roles.

Kanban practices include visualizing the work to facilitate the communication and get the focus of the team on what should be accomplished. Kanban aims to minimize the work in progress (WIP) at any time. To improve the processes of the development Kanban implements feedback loop.

3. AGILE METHODOLOGY

Agile approaches adopt an iterative and gradual design model in which collective self-organizing teams respond quickly to evolving customer needs. The Agile Manifesto's principles include quick, regular, reliable and continuous delivery of working software; adapting to changing requirements; encouraging effective communication; empowered and well-supported self-organizing teams. In response to the weaknesses of traditional software development models, agile methods were developed and proposed. The common agile team members are:

1 Team lead
2 Team member
3 Product owner
4 Stakeholder

Team lead who is called also as “Scrum Master” in Scrum or team coach or project lead in other methods. Team Lead is responsible for organizing, managing securing and defending the group from problems.

Acts as the coach responsible for organizing and directing the team, securing resources when necessary, and eliminating barriers that hinder the team from doing their job. The position of Scrum Master also includes project management's soft skills more than preparation and technical skills that are often left to the entire team. Manager. Rather, this role should reflect rank-by-rank knowledge and responsibility.

Team Member is responsible for designing and implementing the plan. The members of the team will usually consist of programmers and QA. They are in-charge of planning, designing, developing, reviewing and implementing the product.

Product Owner (Scrum), On-Site Customer (XP), Active Stakeholder. Represents the customer's voice and is responsible for the backlog prioritization and maximizing investment return (ROI). Part of the responsibility for this position is to report user stories or project requirements.

Stakeholders represent a wide range of people who can be customers, product managers, production, help, portfolio managers, other related Agile teams, executive teams, stakeholders, and more. In addition to these common roles, Agile teams will sometimes have extended cast members who are called upon to provide technical or domain expertise for certain specialized skills that may not be present amongst the team members.

4. SELF-ORGANIZING AGILE TEAM ROLES

The self-organizing roles on ASD teams that exist. As general members of both Agile and Non-ASD teams fulfil the team's organizational roles. Developers, for instance, are responsible for several tasks such as growth, testing is done by researchers, business analysts were responsible for the analysis and requirements definition. Nonetheless, in Agile teams, the organizational roles are not strictly adhered to, and team participants often work outside their boundaries as they organize themselves.

These self-organizing Agile team roles are Mentor, Translator, Coordinator, Promoter, Champion, and Terminator, the Mentor Initially guides and supports the team. Mentor helps them to be confident in using Agile methods. Mentor ensures continued adherence to Agile methods and encourages the team to develop self-organizing practices “It’s more important that you get everything right at the start. Because the process itself is not that complicated [but] doing things along the lines of the process is a little bit harder than the process itself...So with [the Mentor] it was kind of to teach us how Agile works and shape our mindset and make sure everyone

knows how to work under the Agile umbrella" [24]. When working on Agile projects, development teams and their customer representatives use various languages, hence, there is the need of a Translator. While the development teams for Agile use a technical language that consists of technical terminology. Moreover, the customers use a business language consisting of terminology from the business domains thus, for that the translation between the two languages is needed.

5. CONCLUSION

The paper reviewed and introduced the ASD Methods briefly. It discussed the most important concepts of the ASD, Agile methodologies, ASD teams and self-organizing agile team roles. It discussed the identification of Agile trends and the discussion of their implications that may prove useful to software development educators, students, practitioners and researchers. The paper discusses, in a snapshot, the theoretical concepts of agile presented by Agile champions and special thanks in large part to the passionate advocacy of individual champions, those seem to be increasingly adopting Agile in organizations. Nonetheless, the continued realization of the promised benefits of agile growth will be the key to winning over both skeptics and resistors and encouraging the broader dispersion and use of Agile methods in the long run.

ACKNOWLEDGEMENT

We are highly obliged to Future University Khartoum, Sudan and Al-Neelain University Khartoum, Sudan in order to support this research.

REFERENCES

1. Costa, E.O., et al. *Modeling software reliability growth with genetic programming*. IEEE.
2. Kan, S.H., *Metrics and models in software quality engineering*. 2002: Addison-Wesley Longman Publishing Co., Inc.
3. Kaur, R. and J. Sengupta, *Software process models and analysis on failure of software development projects.* arXiv preprint arXiv:1306.1068, 2013.
4. Bauer, A., et al., *Automode-notations, methods, and tools for model-based development of automotive software*, 2005, SAE Technical Paper.
5. Baida, Z.S., *Software-aided service bundling: Intelligent methods and tools for graphical service modeling.* 2006.
6. Kelter, U., M. Monecke, and M. Schild. *Do We Need 'Agile'Software Development Tools?* in *Net. ObjectDays: International Conference on Object-Oriented and Internet-Based Technologies, Concepts, and Applications for a Networked World*. 2002. Springer.
7. Beck, K., et al., *Manifesto for agile software development.* 2001.
8. Erdem, S. and O. Demirörs. *An exploratory study on usage of process mining in agile software development*. in *International Conference on Software Process Improvement and Capability Determination*. 2017. Springer.
9. Curcio, K., et al., *Usability in agile software development: A tertiary study.* Computer Standards & Interfaces, 2019.
10. Aldahmash, A., A.M. Gravell, and Y. Howard. *A review on the critical success factors of agile software development*. in *European conference on software process improvement*. 2017. Springer.
11. Larman, C. and V.R. Basili, *Iterative and incremental developments. a brief history.* Computer, 2003. **36**(6): p. 47-56.
12. Fowler, M. and J. Highsmith, *The agile manifesto.* Software Development, 2001. **9**(8): p. 28-35.
13. Ambler, S., *Quality in an agile world.* Software Quality Professional, 2005. **7**(4): p. 34.
14. Kumar, G. and P.K. Bhatia, *Impact of agile methodology on software development process.* International Journal of Computer Technology and Electronics Engineering (IJCTEE), 2012. **2**(4): p. 46-50.
15. Ambler, S.W., *The agile scaling model (ASM): adapting agile methods for complex environments.* Environments, 2009: p. 1-35.
16. Stoica, M., M. Mircea, and B. Ghilic-Micu, *Software Development: Agile vs. Traditional.* Informatica Economica, 2013. **17**(4).
17. Ahmed, A., et al. *Agile software development: Impact on productivity and quality*. in *2010 IEEE International Conference on Management of Innovation & Technology*. 2010. IEEE.
18. Khmelevsky, Y., X. Li, and S. Madnick. *Software development using agile and scrum in distributed teams*. in *2017 Annual IEEE International Systems Conference (SysCon)*. 2017. IEEE.
19. Hoda, R., N. Salleh, and J. Grundy, *The rise and evolution of agile software development.* IEEE Software, 2018. **35**(5): p. 58-63.
20. Schwaber, K. and J.J.S.A. Sutherland, *The scrum guide.* 2011. **21**: p. 19.
21. Poppendieck, M. *Lean software development*. in *Companion to the proceedings of the 29th International Conference on Software Engineering*. 2007. IEEE Computer Society.

22. Sommerville, I., *Software engineering*. 2011: Addison-Wesley/Pearson.
23. Ahmad, M.O., J. Markkula, and M. Oivo. *Kanban in software development: A systematic literature review*. in *2013 39th Euromicro conference on software engineering and advanced applications*. 2013. IEEE.
24. Hoda, R., J. Noble, and S. Marshall. Organizing self-organizing teams. in 2010 ACM/IEEE 32nd International Conference on Software Engineering. 2010. IEEE.

AUTHORS PROFILE