IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Review of Computer Engineering Studies
Vol. 5, No. 1, March, 2018, pp. 12-16
Journal homepage: http://iieta.org/Journals/RCES

Agile software development: Implementation perspective

Asim Ismail*, Syed M. Ali

Shaheed Zulfikar Ali Bhutto Institute of Science & Technology Dubai Campus 345004, United Arab Emirates

Corresponding Author Email: asimismail7@gmail.com

https://doi.org/10.18280/rces.050103

Received: 18 January 2018
Accepted: 28 March 2018

Keywords:
software process, software development methodology, agile, scrum, kanban, XP, SAFe.

ABSTRACT

In today's world, the degree of technological and requirements change and market dynamics need fast responsiveness to sustain a competitive advantage. The ability of an enterprise to succeed in such a competitive environment is highly dependent on their capacity in developing software with faster speed. To work with such a pace enterprises are creating increasingly sophisticated software with requirements for business with efficiency. The practices required to develop these systems must conform to the increasing degree of innovation and also the evolving needs of organizations. Agile helps the enterprises to produce value more efficiently and continuously on a periodic and predictable schedule. This study provides a summary of the Agile from implementation perspective by discussing multiple agile methods, including its strategies, principles, practices and values for implementation. According to this study and observations, agile methodologies can provide excellent benefits for medium and small scaled projects. Also, explains how one can scale Agile for large projects using SAFe 4.0.

1. INTRODUCTION

Agile is a collection of systematic approaches and methods that improve to imagine more productively, work more efficiently, and make better commitments. Such sort of practices addresses all of the fields of traditional software engineering, including software design, project management, architecture, and process improvement [1].

Each of such methodologies consists of practices that are highly optimized to make them as easy as possible to embrace. Agile is also a mindset as the right mindset can make a big difference in how efficiently a team implements the practices. This attitude encourages people to share knowledge with each other so that they can make critical project decisions together instead of having a manager who makes all of those decisions all alone [2]. This mindset is about opening up design, planning, and process improvement.

2. AGILE APPROACH

Agile, as a campaign is different from any approach to software development because it started with values, ideas and principles that embody a mindset and attitude. It is through these ideas that you can start to become agiler as a practitioner, and more relevant as a member of one's project team. Agile is revolutionizing the world of software development. Teams that adopt agile have consistently reported improvements sometimes huge leaps in their ability to build excellent software. Teams that successfully adopt agile develop better; higher quality software products and they do it faster than the past. Our IT industry is at a transforming point with agile. Agile has progressed from being the underdog to maturing an institution. For the initial years of agile, people doing the implementation struggled to convince their companies and teams that it worked and that it was worth doing. As of now, there is little question that agile development is a highly powerful way to build software [2]. In fact, in 2008, a critical survey found that more than half of all software teams surveyed were using agile methodologies, practices, or principles. Agile teams are increasingly going beyond the problem of how to be agile themselves and are starting to Figure out how to spread agile development throughout their companies. But it wasn't always like this. Traditionally, companies have used a waterfall process when running their software projects, in which the team defines the requirements up front, plans the project in its entirety, designs the software, and then builds the code and tests the product. Plenty of great software, but also lousy software has been developed this way over the years. But as the decades went on, different teams in different companies kept running into the same kinds of problems, and some people started to suspect that a major source of project failure might be the waterfall process itself. The story of agile began when a small group of innovative people got together to try to discover a different way of thinking about these problems [1]. The first thing they did was come up with a set of four core values that are common to successful teams and their projects, which they called the Agile Manifesto.

2.1 Waterfall Vs. agile

The differences between Waterfall versus Agile can be summed up in two words: rigid versus flexible [3]. Waterfall is a much rigid, stricter process whereas Agile is flexible and continuously evolving. Here's more on their differences:

i. Waterfall is a structured, systematic process, where you cannot begin a new phase until the previous one has been finished. Agile is a flexible process allowing to progress through the project as one like.

ii. As Waterfall is sequential does not enforce a linear process.

iii. Waterfall projects usually have defined requirements in advance, whereas requirements are anticipated to change and evolve in agile projects.

iv. In Waterfall projects, one cannot make a change that was done in previous stages, whereas Agile is very adaptive to changes.

The key differences and similarities are shown in the table 1.

Table 1. Differences and similarities between waterfall and agile

Features	Waterfall	Agile
Sequential		
Flexible		
Accommodates change		
Defined requirements		
Deliver quality products		
Continually evolving		
Rigid process		

3. OBJECTIVES OF THE STUDY

The scope of the study is limited to the three agile methods namely Scrum, Extreme Programming (XP) and Kanban.

i. To discuss the core values, processes and practices of agile methodologies

ii. To understand Agile implementation features, benefits and limitations

iii. To address the developer challenges for agile implementation.

Agile has become one of the most well-known and widely accepted software development approaches across the globe. At the end of the study, the reader will be able to implement any of the three agile methods in its organization keeping in view the benefits and challenges that may experience in the journey.

4. MATERIAL AND METHODS

This study is qualitative in nature. To correctly understand the topic of agile software development from implementation perspective thorough literature reviews of articles, research papers, blogs and other informational pages on the internet have been done. Different books have been examined on the topic and its relevant areas.

Several Agile methods are proposed in the last 20-25 years. Agile manifesto helps in providing a solution to requirements change. All agile methods strictly follow agile manifesto [4]. The fundamental concept is to provide first working software in short iterations, modify the next version as per the response received from the customer, satisfying as per the client feedback and develop the complete system through several iterations. The Agile methods that are discussed in this study include Scrum, Extreme Programming (XP) and Kanban [4].

4.1 Scrum

Scrum is one of the most well-known methods for implementing Agile. It follows an iterative model used to develop sophisticated software and do product development. Iterations of fixed period called sprints lasting one to two weeks long permit the team to dispatch software on a regular basis [5].

When each sprint ends, team members and stakeholders meet up to plan next moves. Scrum owns a set of roles, responsibilities, and meetings that never vary. For instance, it calls for four ceremonies that give structure to each sprint: sprint planning, daily standup, sprint retrospective and demo. Task boards or burndown charts are used during each sprint to display progress and receive incremental feedback from the client [6]. The scrum process is shown in the Figure 1.

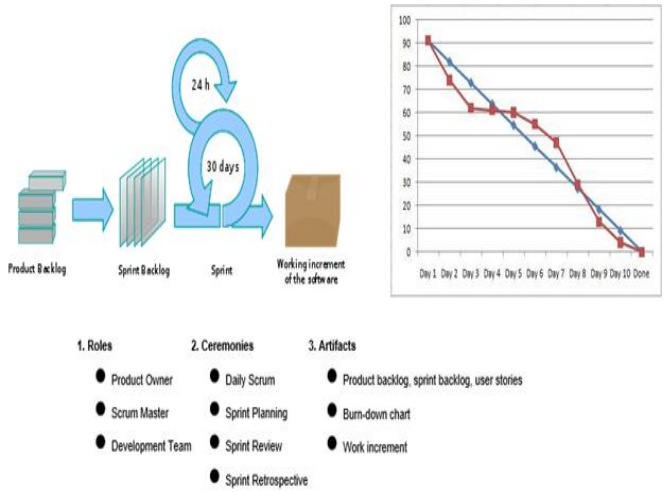


Figure 1. SCRUM Process [6]

4.2 Extreme Programming (XP)

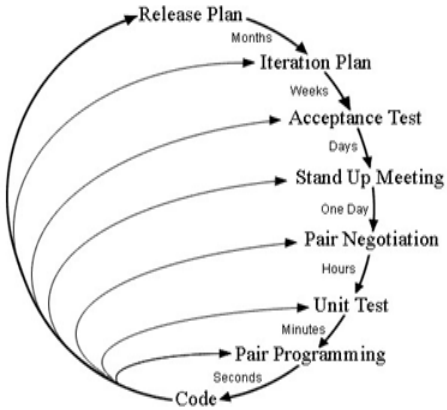


Figure 2. XP process – planning/feedback loops [6]

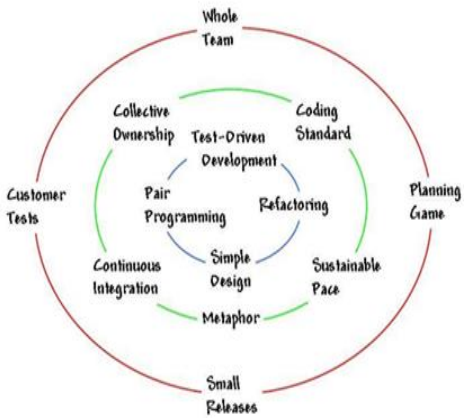


Figure 3. XP process – core practices [6]

XP stands for Extreme Programming, an agile method that focuses heavily on assuring the quality of developed software and that commands engineering solutions till the end. The engineering team engages in Iteration Planning and Release Planning. They work together in very short cycles so that changes requests brought by the client can be incorporated as soon as possible [5]. Through more than a dozen core practices including Customer Testing, Test Driven Development, Pair Programming, Small Releases and Continuous Integration, it works towards a regularly improving, high-quality product which can respond to requirement changes in the shortest possible time.

The main concept is to react quickly and make efforts to improve quality continuously to the ongoing changing customer requirements [6].

i. Periodic builds with short iteration cycles

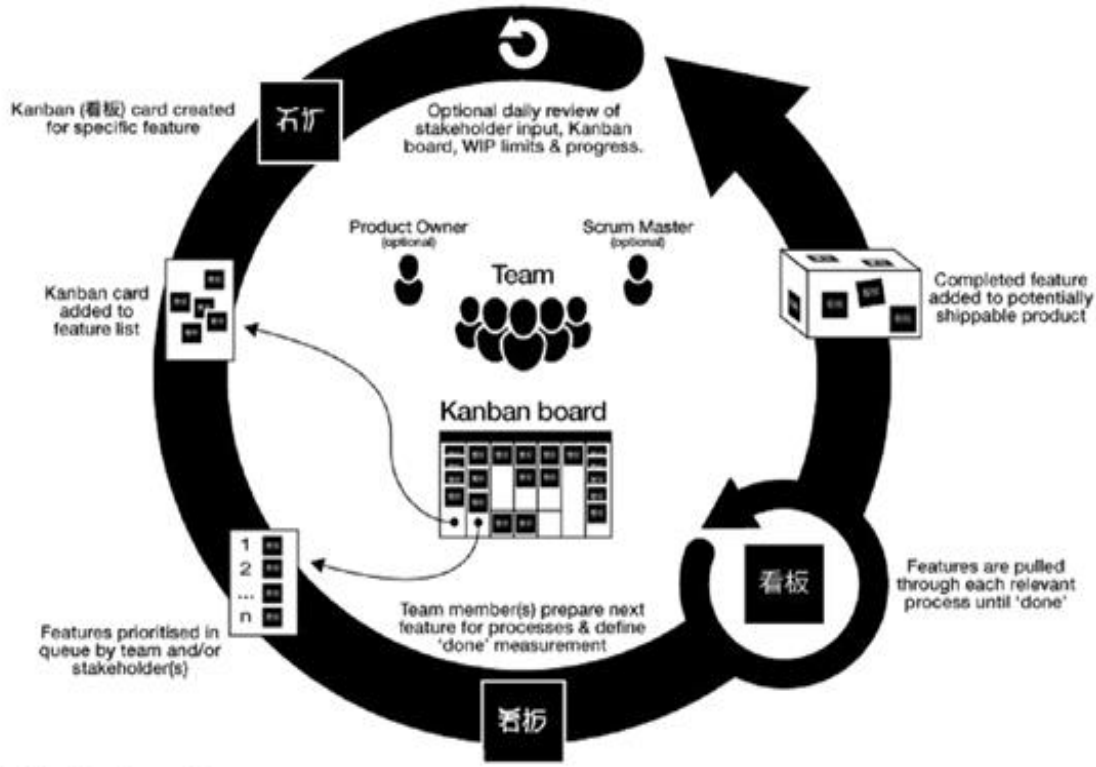
ii. Increase productivity on the spot and regularly implement checkpoints with the client

iii. Paired Programming Technique

It states that if developers in pair work collectively on a particular system, then they could react well to the customer requirements. Such strategy can develop high quality software artifact as shown in the Figure 2.

XP process core activities are stated in the Figure 3.

4.3 Kanban



1. Visualize the workflow
Kanban means "signboard" or "billboard."

2. Limit Work In Process (WIP)
Use a pull system (establish and respect your ideal capacity)

3. Manage Flow
Monitor, measure and report the flow of work through each state

4. Make Process Policies Explicit
Describe the process accurately so that can be improved

5. Improve Collaboratively
Using models & the scientific method (observed) to implement incremental, continuous, and evolutionary changes

Figure 4. KANBAN process [6]

Kanban is Japanese term for "visual sign" or "card." It is a visual method used to implement Agile that displays: What? When? And How? to produce it. It supports small, incremental changes to the current system and does not require a particular setup or procedure (meaning, you could overlap Kanban on top of other existing workflows) [5]. In this context, development work-in-progress (WIP) takes the place of inventory, and new work can only be included when there is an "empty space" on the team's visual Kanban board. Kanban matches the amount of WIP to the team's capacity, transparency, improving flexibility, and output [6] as illustrated in the Figure 4.

4.4 Visualizing the agile methods

Work stream for the 3 agile methods can be visualized as shown in Figure 5.

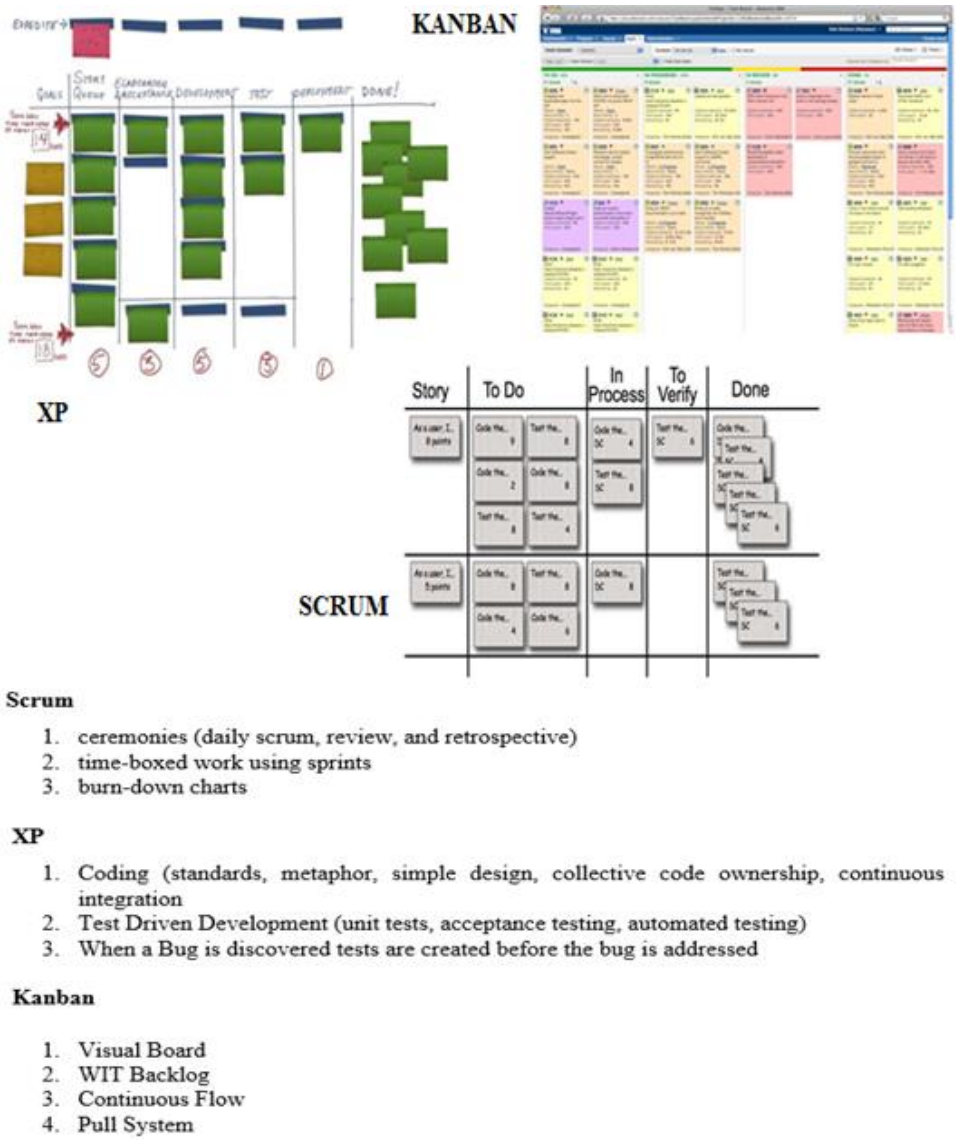


Figure 5. Visualize work - scrum, XP & Kanban [5]

4.5 XP Vs Scrum Vs Kanban

When comparing XP versus Kanban versus Scrum, there is no definitive winner. The best framework depends on one's project, team, and their goals. Because XP, Kanban, and Scrum are flexible Agile methodologies, one can quickly take principles from each and apply them as deems fit per the project requirements. It is important to remember that pure Scrum is a much bigger shift than Kanban [3]. The team will have to spend time in learning about the ceremonies, the iterations, and the specific roles. On the other hand, XP and Kanban encourage incremental improvements. One can apply Kanban principles to any process you already have in place, even XP or Scrum. There is nothing much needs to change significantly to get started with Kanban. As a general rule of thumb, if one's organization is stuck and requires a significant shift, Scrum may be more suitable. If one already has a process implemented that one is happy with but needs to implement some minor changes, Kanban might be a better choice. If product quality is supreme, then XP might be a better choice. The same is represented in the table 2.

Table 2. Differences and similarities among agile methods

Features	XP	Scrum	Kanban
Specific roles	■	■	
Time boxed iterations	■	■	
Accommodates change	■		■
Estimation	■	■	
Empirical	■	■	■
Lean and agile	■	■	■
Limits WIP	■	■	■
Work can be done simultaneously	■	■	■
Board is continuously used	■		■
Teams must be cross functional	■	■	
Pull scheduling	■	■	■
Transparency	■	■	■
Deliver software early and often	■	■	■

5. RESULT AND FINDINGS

It is quite clear that XP focus towards engineering practices whereas Scrum focus is more towards productivity.

The value that XP could add though is unquestionable and many organizations that use Scrum adopt Test Driven Development, Refactoring, and Pair Programming as practices which improve quality, fasten the release process and minimize the need to revisit work due to technical debt [5] alongside shorter iterations. Some other important things which differentiate XP from Scrum are:

XP team operates in a strict priority order whereas a Scrum team may not necessarily tackle each item in priority order.

XP team bring new items of work into an iteration and shift out items of similar size if the customer decides on a new priority order.

Regarding similarities, the role of the Client in XP is very analogous to that of the Product Owner in Scrum; in which they support to write user stories, provides prioritization order and are always accessible to the developer. The daily stand-up meeting is mandatory in both XP and Scrum [5]. Both of them stresses the significance of co-location, only XP makes it deal-breaker. Hence, XP is superior from engineering and quality perspectives.

5.1 Issues and challenges

Our software industry is at a turning point with agile. For the first initial years of agile, people embracing it struggled to convince their organizations that it worked. Now, there is little question that agile development is a highly efficient way to build software. In fact, in 2008, a critical survey found that more than half of all software development teams surveyed were using Agile principles and agile has only grown since then [4].

The following are some of the limitations of agile process:

Not much support for third party vendors as the contract is not in a clear form.

The development of safety-critical software (as having critical and complex architecture) is challenging to enhance or change in an agile process due to limited support. Also, the cost of enhancements in these systems is very exponentially high.

Systems having complex architecture, the components are tightly coupled. It is very complicated and complex to construct software for such systems using agile practices [4]. However, this is debatable and there are agile frameworks available today in the market that targets Scaling Agile for the enterprise.

6. CONCLUSIONS

The development of Agile project involves several iterations. All agile methods can accept change request at any stage of development time. In this study, we have focused on the overall agile process including its practices, methods, challenges and issues related to it. The critics of Agile are mostly concerned about Scaling of Agile [7].

Agile does not scale up to the requirements of the systems build by larger enterprises. An another way of working is needed that applies the power of Agile, but also leverages the more extensive knowledge supplies of lean product development and systems thinking. The Scaled Agile Framework (SAFe) is one of the best such approaches [7]. SAFe provides freely revealed, online, comprehensive and detailed guidance for achieving the benefits of scale Agile development. It supports enterprises to deliver more efficiently and value continuously on a periodic and predictable schedule, making them more Agile in the marketplace and more competitive in their industry. It is a knowledge base of integrated and industry best practices proven patterns for enterprise scale Lean-Agile development. It is modular and scalable and allows each organization to apply it in a form that produces better business outcomes and happier more engaged employees. It synchronizes alignment, collaboration, and delivery for a large number of Agile teams. It supports both systems development and software, from the modest scale of well under 100 practitioners to the most superior software solutions. It supports three primary bodies of knowledge: Agile development, Lean product development, and systems thinking [7].

SAFe supports those building large, integrated solutions that typically require hundreds or more practitioners to develop and maintain. SAFe is improving business outcomes for government agencies and companies of all sizes across the globe, resulting in dramatic increases in employee engagement, improved economics, and workplaces that are more productive, engaging, and fun. Benefits from documented case studies include:

20-50% increase in productivity
30-75% faster time to market
50%+ defect reduction.

REFERENCES

[1] Stellman A, Greene J. (2014). Learning Agile. O'Reilly Media, Inc.

[2] Cohn M. (2009). Succeeding with agile, Addison-Wesley Professional.

[3] Smartsheet. https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban, accessed on Feb. 20, 2018.

[4] Amir Mkkakamnk. (2013). An appraisal of agile software development process. International Journal of Advanced Science & Technology 58: 20.

[5] Bowes J. (2015). MANIFESTO. https://manifesto.co.uk/kanban-vs-scrum-vs-xp-an-agile-comparison/, accessed on Jan. 14, 2018.

[6] Ponomareff D. "SlideShare," TORAK. http://www.slideshare.net/dimka5/introducing-agile-scrum-xp-and-kanban, accessed on Jan. 25, 2018.

[7] SAFe. http://www.scaledagileframework.com/, accessed on Feb. 28, 2018.