Agile Methodologies

Article · December 2021

CITATION
1

READS
6,752

1 author:

Devharsh Trivedi
Stevens Institute of Technology
35 PUBLICATIONS **12** CITATIONS

SEE PROFILE

International Journal of Computer Science & Communication (ISSN: 0973-7391)
Volume 12 • Issue 2 pp. 91-100 April 2021 - Sept 2021 www.csjournals.com

Agile Methodologies

Devharsh Trivedi
M.Tech. Student, Nirma University, Ahmedabad, India
devharsh@live.in

Abstract: The dictionary meaning of Agile is to be able to move quickly and easily. Agile can be sketched as the ability to adapt and respond to change. It is about dealing with, and succeeding in, a turbulent environment. It is about understanding what is going on in the environment, identifying the uncertainty you face, and figuring out how to adapt. The Agile Manifesto authors chose "Agile" because it represented the adaptiveness and response to change, crucial to their approach.
Keywords:Agile, Agile Methodologies, Agile Principles, Agile Values, Scrum, Sprint.

Introduction

Agile software development is more than Scrum, Feature-Driven Development (FDD), or Extreme Programming (XP). It is more than practices such as Pair Programming (PP), Test-Driven Development (TDD), stand-ups, sprints, and planning sessions. Agile software development can be sketched as an umbrella term for a set of practices and frameworks based on the values and principles narrated in the Agile Manifesto. Therefore, when you approach software development, it is good to live by these principles and values to use them to figure out the right things to do given your context. Something that separates Agile from other software development approaches focuses on working together. As a result, solutions evolve through collaboration between self-organizing cross-functional teams utilizing the appropriate practices for their context.

Agile software development has a big focus on collaboration and the self-organizing team. However, that does not mean that there are no managers. Instead, teams can figure out how they will approach things independently. It means that those teams are cross-functional. Those teams do not mandate specific roles involved so much as that when you get the development team together, you make sure that you have all the right skill sets on the team. There is a requirement for managers who make sure team members have, or obtain, the right skill sets. Managers also provide the environment that allows the team to be successful. For example, managers let their teams figure out how to deliver products and step in to resolve issues. When organizations start doing Agile development, they focus on the practices that help with collaboration and organization, which is excellent. However, another essential set of practices that are not as frequently followed are specific technical practices that deal with developing software in a way that helps the team deal with uncertainty.

Waterfall Methodology

Waterfall methodology follows a step-by-step linear process. It is the most straightforward version of the Software Development Life Cycle (SDLC) for software engineering and information technology. The Waterfall is often planned-out using a chart showing start and end dates for each task in the process. The development team can only move on to the next task after completing, reviewing, and approving the client. Due to its linear nature, it is impossible to go back or jump forward without starting the entire process again. The waterfall process involves each step obligated to the previous one, building upon preceding foundations to fabricate a flow headed in a single direction. The Waterfall is best suited to projects with a fixed budget, deadline, and scope.

A. Advantages of the Waterfall method

Due to the rigid and linear nature of the Waterfall method, it is best suited to unchanging and straightforward projects. It also makes it a reasonably straightforward process to follow and document as it follows the same sequential pattern for each project. Each phase in this process has specific goals and deliverables, including the review processes, and thus it is an elementary process to manage and control. Due to its intended simplicity, development teams require little to no onboarding before starting a project.

Because each step contains a predetermined start, end, and review date, it naturally lends itself to a highly disciplined process. All project requirements are set to shrink the risk of missing deadline deliverables before written code. This means that clients and stakeholders can chart and review progress at each step, allowing transparency and communication throughout the software development project. In addition, this appropriates that every step of the entire project is documented, so everyone is aware of every detail, which helps enormously for posterity purposes.

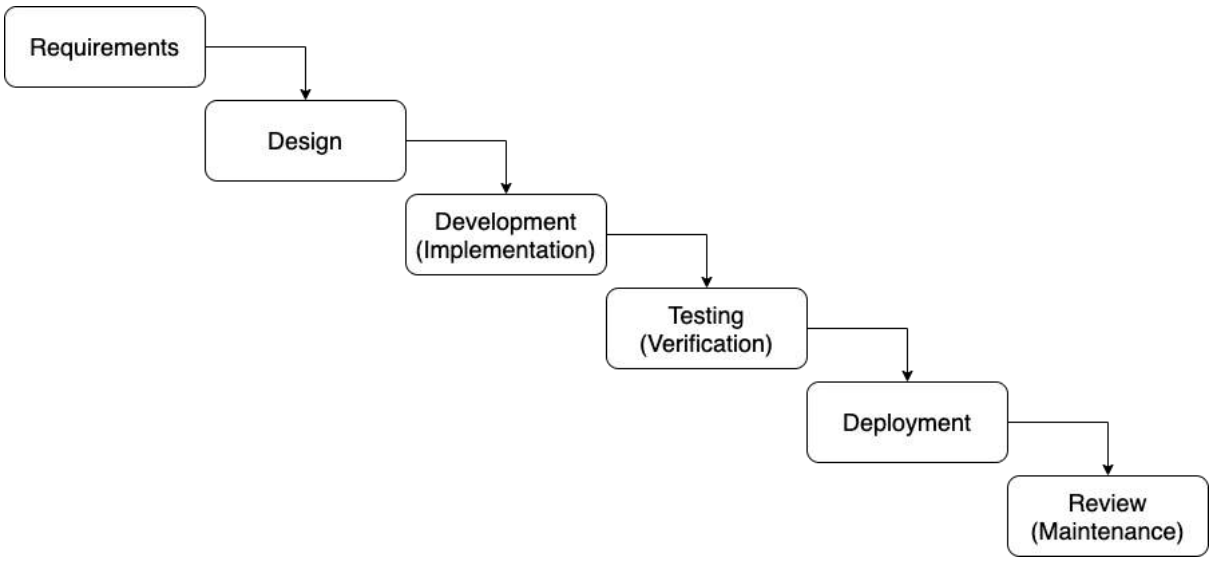

Figure 1. Waterfall Methodology

B. Disadvantages of the Waterfall method

The most blazing problem facing the Waterfall method is its inability to attune to change. It is practically impossible to go back a step or jump between them when issues arise. Instead, you are supposed to start from the beginning as each phase impacts the next stage in the waterfall model. E.g., code can only begin to be written once two or three phases of the project have already been completed. This means that stakeholders can only review software very late once much work has already been concluded. Naturally, problems can arise due to this delay. E.g., clients may begin a project without clarity on what is needed and usually identify issues and requirements as the project approaches the middle of its life cycle. In addition, if clients realize specific changes need to be made, these changes are challenging to perform because of the sequential and rigid nature of the Waterfall method.

C. Advantages of Agile

The most eloquent aspect of the Agile method is that it wholeheartedly embraces flexibility, speed, and above all else, continuous improvement. Because pre-project planning is lightweight, teams are not beholden to a rigid system based on predefined restrictions and are free to adjust and change when necessary. This flexibility at any stage propagates creativity and freedom within processes. In addition, development teams can refine and re-prioritize the backlog, meaning ad-hoc changes can be implemented quickly.

Moreover, because of Agile's lightweight pre-project planning, it is not uncommon for development teams to begin a project without a predefined end goal. This would appear counter-intuitive at first, but it means that teams are given a fair amount of freedom and flexibility to structure the project in new directions according to work already done. If teams encounter a great idea, they are encouraged to pursue it. In addition, project elements are encouraged to adapt quickly to new goals, which means the possibilities for each project are almost endless.

Agile involves breaking the project into more accessible, more feasible units, or iterations, leading to greater quality output across all teams, including improved and elaborate development, collaboration, and testing. In addition, with testing being conducted at each iteration, bugs and inconsistencies can quickly and easily be identified and resolved. Thus, iteration consistency is fundamental to improving the quality of the project output.

Agile relies heavily on communication between stakeholders and development teams, and it breeds a strong work ethic, both individually and collectively. Therefore, we do not need to explain how stronger teams produce more significant project results.

Because of the continuous review process at each stage of the project, clients remain up to date with the current developments and are encouraged to provide feedback to the development team to get the best results possible for each project. The team considers the feedback and incorporates it at each stage of the project, ensuring that the clients' input is recognized and realized throughout its life cycle. This translates to continuous improvements to the project, resulting in a far more detailed and sophisticated project being released.

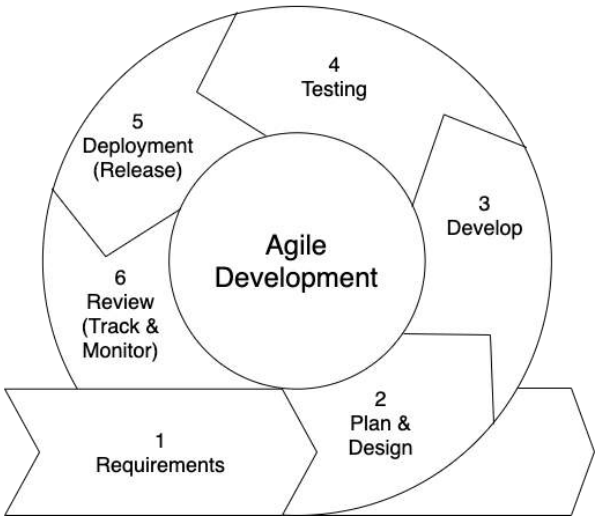

Figure 2. Agile Methodology

D. Disadvantages of Agile

The cons of Agile may not be seen as cons as such if you prefer flexibility because they are just the downside of having freedom and flexibility. The cons can be easily avoided if you keep your affairs in moderate order.

Planning the project before commencing with any actual work is less in-depth than the Waterfall method, so specific issues of keeping sight of the end goal may arise. Some see this as an advantage, others as a disadvantage. If you are among those who prefer concrete deadlines and timetables, then you would intuitively see the fluid nature of Agile as a disadvantage. A project using Agile may change course, pivot in direction, and jump from one idea to another, which can cause delays in deliveries and schedules. It makes the project manager's job trickier. It also means that the result may differ drastically from the initial goal. However, if you want to bake a cake, you must break a few eggs.

Because of the small size of Agile development teams, individuals need at the very least to be very knowledgeable and at best highly skilled to work on various tasks. Of course, having a highly skilled and bright team is good, but having teams filled with Jacks-of-all-trades is arduous and extravagant.

Time allocation to an Agile project can usually become taxing. The nature of Agile involves continuous iterations and improvements. Agile is often cumbersome to keep each team member dedicated to one specific project component.

The Four Agile Values

There are four agile values and 12 agile principles outlined in The Agile Manifesto. These principles for agile software development help establish the tenets of the agile mindset. They are no rules for practicing agile but a handful of principles to help instill agile thinking.

A. Individuals and interactions over processes and tools

The first value described in the Agile Manifesto is "Individuals and interactions over processes and tools." Valuing people highly than processes or tools is easy to understand because they respond to business needs and drive development. Conversely, if the tools or the process drive development, the team is less responsive to change and less likely to meet customer needs. Communication is an illustration of the difference between valuing individuals versus processes. For the case of process, communication is scheduled and requires specific content. For the case of individuals, communication is fluid and happens when a need arises.

B. Working software over comprehensive documentation

Historically, mammoth amounts of time were spent documenting the product for development and terminal delivery. Technical requirements, specifications, prospectus, interface design documents, test plans, documentation plans, and approvals are required. The list was extensive and was a reason for the long delays in development. Agile does not eradicate documentation, but it streamlines it in a form that gives the development team what is needed to do the work without getting bogged down in technicalities. In Agile,

requirements are documented as user stories, which are sufficient for a software developer or a development team to begin the task of building a new function.

C. Customer collaboration over contract negotiation

Negotiation is when the product manager and the customer work out the delivery details, with points along the way and where the details may be reconsidered. Collaboration is a different story entirely. With software development models such as Waterfall, customers negotiate the requirements for the product, often in detail, before any work starts. This meant the customer was involved in development before development began and completed, but not during the process. A customer in the Agile Manifesto is described as engaged and collaborates throughout the development process, making it easier for developers to meet the customer's needs. Agile methods may include rare demos, but a project could just as quickly have an end-user as a part of the team and attend all daily meetings, ensuring the product meets the customer's business needs.

D. Responding to change over following a plan

Traditional software development regarded change as an expense, so it was to be avoided. Instead, the intention was to develop elaborately detailed plans with a defined set of features and with everything, generally, having as inflated a priority as everything else, and with a vast number of many dependencies on delivering in a particular order so that the team can work on the next piece of the puzzle.

The Twelve Agile Principles

A. The highest priority in Agile is to satisfy the customer through early and continuous delivery of valuable software.

B. Agile processes welcome changing requirements, even late in development, and harness change for the customer's competitive advantage.

C. Deliver working software frequently, with a preference for the shorter timescale from a couple of weeks to a couple of months.

D. Businesspeople and developers are mandated to work together throughout the project every day.

E. Build projects around motivated individuals. Please give them the environment and support that they need. Trust individuals to get things done.

F. The most effective and efficient method to convey information to a development team is face-to-face conversation.

G. In Agile, the working software is one of the primary measures of progress.

H. Agile processes promote sustainable development. Therefore, the sponsors, developers, and users should maintain a constant pace indefinitely.

I. In Agile methodologies, continuous attention to technical excellence and good design enhances the project's agility.

J. The simplistic art of maximizing the amount of work not done is essential.

K. In Agile, the best requirements, architectures, and designs emerge from self-organizing teams.

L. Agile teams reflect on becoming more effective at regular intervals, then tune and adjust their behavior accordingly.

Agile Methodologies

Agile development is an umbrella term for iterative development methodologies focused on adaptive planning, faster delivery, continuous improvement, and the ability to evolve. Agile is a framework with several specific methods for the Agile movement. You may think of these as different flavors of Agile.These iterative methodologies might have a unique approach individually, but they share a single vision and core values. The standard operating principle of these practices is that the feedback from continuous iteration is used to refine the product until its tested and integrated for sophisticated software delivery. Agile methodology is compatible, collaborative, and empowering, making it a popular choice in business development circles. As per VersionOne's State of Agile Report of 2017, 94% of organizations practice agile methodology in one way or the other. However, despite being widespread, it is hardly ever exercised in its entirety.

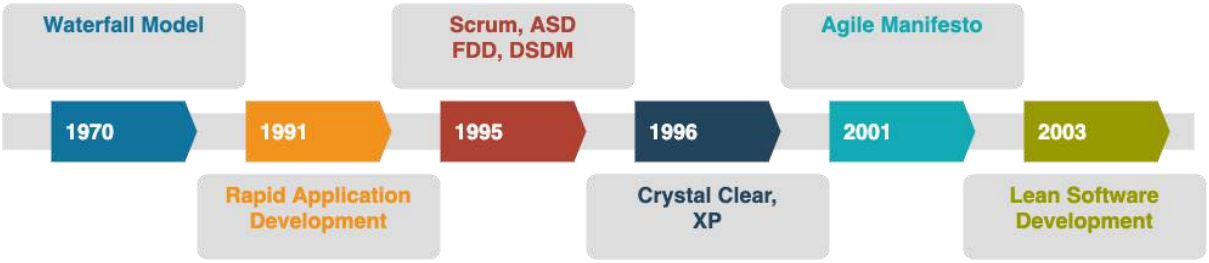

Figure 3. History of Agile Development

A. Lean Software Development (LSD)

Lean Software Development (LSD) takes Lean manufacturing and Lean IT principles to software development. The seven principles about Lean Software Development are as follows:

a. Eliminate Waste
b. Build Quality In
c. Create Knowledge
d. Defer Commitment
e. Deliver Quickly
f. Respect People
g. Optimize the Whole

B. Feature Driven Development (FDD)

Feature Driven Development (FDD) is an iterative and incremental software development process that blends industry best practices into one approach. FDD has five primary activities:

a. Develop an overall model
b. Build feature list
c. Plan by feature
d. Design by feature
e. Build by feature

C. Adaptive System Development (ASD)

Adaptive system development (ASD) represents the notion that projects should always be in a state of continuous adaptation. Adaptive system development (ASD) has a cycle of three repeating series:

a. Speculate
b. Collaborate
c. Learn

D. Dynamic Systems Development Method (DSDM)

The Dynamic Systems Development Method framework is used to develop IT and non-IT solutions. DSDM addresses the most common failures of IT projects, such as missing deadlines, going over budget, and lacking user involvement. The eight principles of DSDM are:

a. Focus on the business need

b. Deliver on time

c. Collaborate

d. Never compromise quality

e. Build incrementally from firm foundations

f. Develop iteratively

g. Communicate continuously and clearly

h. Demonstrate control

E. Crystal Clear

Crystal Clear development is part of the Crystal family of methodologies. Crystal Clear can be used with six to eight developers. It focuses on the people rather than processes or artifacts. Crystal Clear requires the following: frequent delivery of usable code to users, reflective improvement, and osmotic communication, preferably by being co-located.

F. Scrum

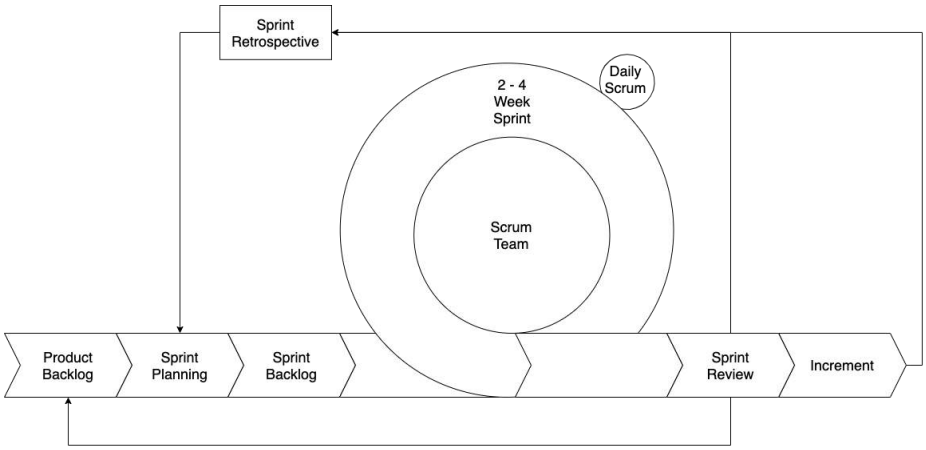

Figure 4. Scrum Methodology

Scrum is one of the most fashionable ways to instrument Agile. Scrum is an iterative development model which follows a set of roles, responsibilities, and meetings which hardly change. Usually, sprints last one or two weeks and allow the team to deliver software regularly. Scrum is a very well-known Agile methodology due to its simplicity, productivity, and the ability to act as a framework for all the various practices promoted by Agile methodologies. Scrum relies on time-bound sprints like other agile methodologies. However, Scrum is a bit more prescriptive instructuring your sprints. Each sprint in Scrum features four "ceremonies" that help teams move forward:

1. Sprint planning is a team meeting to decide what to include in the current sprint. Once it is decided what to include in the sprint, nothing else can be added except the team.
2. Sprint demo is a sharing meeting where the team shows off what they have shipped.
3. Daily Stand-up is a regular 10-15 meeting to sync up and talk about progress and roadblocks.
4. A retrospective is a review of the results of the previous sprint to tweak the process if required.

Along with these four ceremonies, teams will use a dedicated "Scrum board" to mirror this process. E.g., during the sprint planning meeting, the team will move any operational issues to the board. The issues will move from To Do to In Progress, Code Review, and Done throughout the workflow, or however a team organizes the scrum board. It is a powerful tool for adding transparency to the project management process. Scrum is a framework that brings regularity to the project utilizing its various processes or phases. Any organization can realize a scrum framework without altering its rules and regulation.

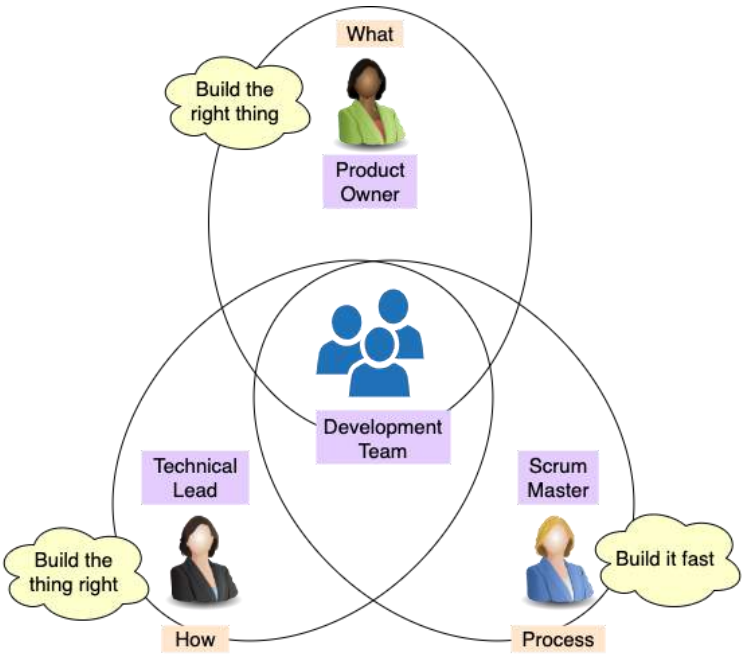

Figure 5. Scrum Roles

1. The Product Owner (PO) plays the liaison between customers and the development team. PO is the champion for their product. PO ensures that the expectation for the completed product has been communicated and agreed upon accordingly.
2. The Development Team (DT) works together to develop and test incremental releases of the final product. DT members are the champions for sustainable development practices. Scrum teams usually are five or seven members, tight-knit and co-located.
3. Scrum Master (SM) is the champion of Scrum within the team. It is the primary responsibility of SM to ensure the best scrum practices in the project. In addition, SM schedules the needed resources for sprint planning, stand-up, sprint review, and the sprint retrospective.

- The Product Backlog Creation

  The product owner (PO) takes part in product backlog creation. It is the primary responsibility of the PO to take a constant view of market and customer needs, which is why the PO must drive the product by the product vision. The PO makes a list based on customers' feedback, and the development team helps prioritize the list to work at any given time. This is an ongoing process, and the changes or new emergence are accepted as new opportunities rather than obstacles.

- Sprint Planning Meeting and Backlog Creation

  All the development team (DT) members must participate in the sprint planning meetings at the beginning of a project. A sprint planning meeting is led by a scrum master (SM). In this meeting, the team members and the product owner decide the goal of the sprint. The sprint duration lasts only two weeks, but it is essential to maintain cooperation between stakeholders and DT members. The product owner (PO) regulates the importance of a good user story, and the scrum team can choose a powerful story from the product backlog. Every user story should always be associated with a goal. Every scrum participant must have a concise idea of what can be achieved in the ongoing sprint and how the increments will be delivered at the end of the sprint planning meeting.

  The sprint backlog lists items, user stories, or bug fixes chosen by the development team for the application in the current sprint cycle. The team decides what item or items they will choose from the product backlog at the beginning of a sprint. The sprint backlog is highly flexible as it is developed during the sprint. However, the ultimate sprint goal must be fixed.

- Working on the sprint

  The development team begins work to deliver an increment while the goal of the sprint is fixed at the sprint planning meeting. Generally, a task board is used to track the current working progress.

- The Impediment List

  When the working process initiates, each team member may add blockers or impediments to a list of what the member faces during working on the preassigned task. As soon as the development team members announce or add the blockers to the list, the scrum master (SM) takes the initiative to eliminate those barriers. When the blocker arises, it should be conveyed to other development team members in daily scrum meetings and recorded in the impediment list.

- The Daily Scrum Meeting

  The daily scrum meeting generally co-occurs to discuss the necessary task to reach the sprint goal. This informal meeting usually lasts 10-15 minutes, and the scrum master (SM) must take the initiative to make it happen daily. It is not an extensive discussion meeting, and it is also called 'daily stand-up.' The motive of the daily scrum meeting is to keep each member of the scrum team in exact alignment. A work plan is made for the next 24 hours without compromising the goal.

  The team members usually keep focusing on the following three questions for the sprint goal in this meeting -

  1. What did I do yesterday?
  2. What do I plan to do today?
  3. Are there any obstacles that may avert meeting the goal?

- Sprint Review and Sprint Retrospective

  At the end of the sprint, all the members meet to demonstrate the backlog items to the teammates, stakeholders, and product owner (PO) for feedback. After this, the product owner (PO) can decide whether the increment will be released or not.

  In the sprint retrospective, the team members share their experiences about the last sprint and what they have gained during the sprint, if anything. What went well and what needs improvement in the next iteration are discussed.

G. Kanban

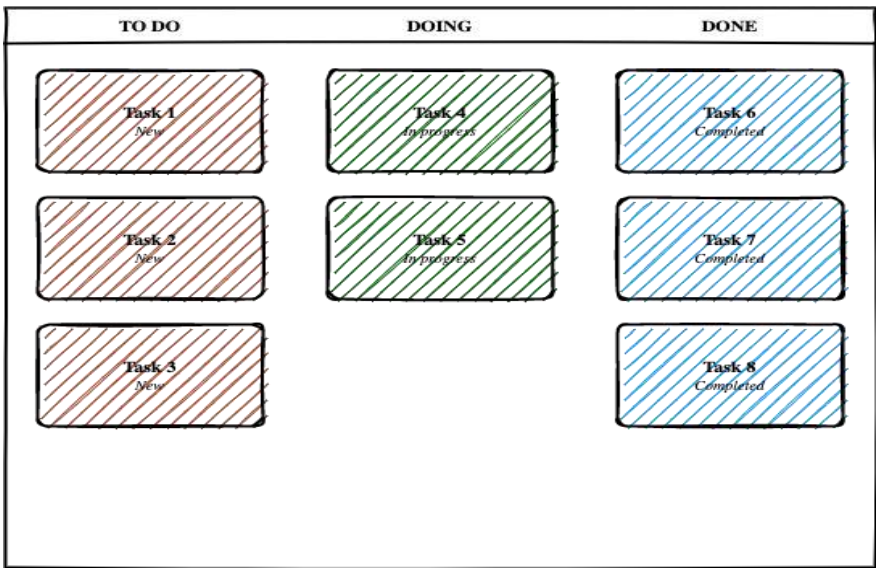

Figure 6. Kanban Dashboard

A. Extreme Programming (XP)

Extreme Programming (XP) is a software development intended to improve responsiveness and quality to evolving customer requirements. Extreme Programming (XP) principles include feedback, assuming simplicity, and embracing change. Extreme Programming (XP) is not extreme in the Mountain Dew sense, but it is a bit of a departure from the other methodologies we have discussed. Extreme Programming is a more disciplined approach to Agile project management that involves rapid feedback loops, high customer involvement, continuous testing and planning, and close teamwork to deliver working software quickly. A typical XP Sprint lasts only about 1–3 weeks.

The original XP "recipe" described by software engineer Kent Beck was based around four values: simplicity, communication, feedback, and courage, with 12 supporting practices. As a result, it is more complex than other methodologies. For example, there are tight feedback loops in XP where the customer works closely with the development team to prioritize and define granular goals called "User Stories." The development team then estimates, prioritizes, and plans the delivery of these stories, getting more feedback from the customer until it is ready for release.

Kanban means "visual sign" or "card" in Japanese. Kanban is a visual framework to implement Agile. Kanban promotes small, continuous changes to the current system. Kanban principles include visualizing the workflow, limiting work in progress, manage and enhancing the flow, making policies explicit, and continuously improving. Kanban is an Agile methodology, just like Scrum. However, Kanban is built around continual delivery while keeping things simple and less burdening for the development team. Development teams in Kanban organize around a limited number of Work in Progress tasks and can release them at any time when they are ready rather than use tightly time-bound sprints. There are a few core principles that help differentiate Kanban from Scrum:

- Visualize workflow on a board.

  While the Scrum board is a good tool, Kanban relies on a visual board to keep all tasks visible and show progress. Kanban boards are great tools for project managers to manage resources and set priorities.

- Keep limited tasks as Work-in-progress (WIP).

  While in Scrum, the backlog is defined before the sprint, and nothing can be added except by the development team, Kanban sets a limit on the tasks added to the WIP board. However, Kanban teams do not have a dedicated product backlog in most cases but rather keep those tasks in a "to-do" column on their board.

- Release when ready.

  Kanban teams are not stuck to the rigid schedule, unlike Scrum ceremonies and sprints. In Kanban, the development team can release whenever they work through enough WIP tasks. This adds extra flexibility and means you need to build your structure so that releases take too long.

Abbreviations

FDD	Feature Driven Development
XP	extreme programming
PP	Pair Programming
SDLC	Software Development Life Cycle
ASD	Adaptive Software Development
DT	Development Team
TDD	Test Driven Development
DSDM	Dynamic Systems Development Method
PO	Product Owner
SM	Scrum Master

Conclusion

Agile methodologies are the standard practices for modern-day industries. This paper reviews Agile software development and how it is practiced. Different flavors or implementations of Agile like Lean software development (LSD), Feature-driven Development (FDD), Adaptive system development (ASD), Dynamic system development method (DSDM), Crystal clear, Scrum, Extreme programming (XP), and Kanban are discussed in detail.

References

[1]. Agile 101. https://www.agilealliance.org/agile101/.
[2]. Eby, Kate. 2016.Comprehensive guide to the agile manifesto.https://www.smartsheet.com/comprehensive-guide-values-principles-agile-manifesto.
[3]. Eby, Kate. 2017.What is the difference? agile vs.Scrum vs.Waterfall vs.Kanban.https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban.
[4]. MacKay, Jory. 2018.The ultimate guide to implementing agile project managementand Scrum.https://plan.io/blog/what-is-agile-project-management/.
[5]. Principles behind the Agile Manifesto. 2001.https://agilemanifesto.org/principles.html.
[6]. Thakur, Madhuri. Scrum process.https://www.educba.com/scrum-process.
[7]. Trivedi, Devharsh. 2021.Agile methodologies.DOI:10.13140/RG.2.2.33950.25928.
[8]. What is it they call Agile Software Development?2018.https://devbatch.com/what-they-call-agile-software-development/.
[9]. Wolhuter, Samantha. 2020.Waterfall vs. agile vs.Scrum: what is the difference?https://www.wearebrain.com/blog/software-development/waterfall-vs-agile-vs-scrum/.