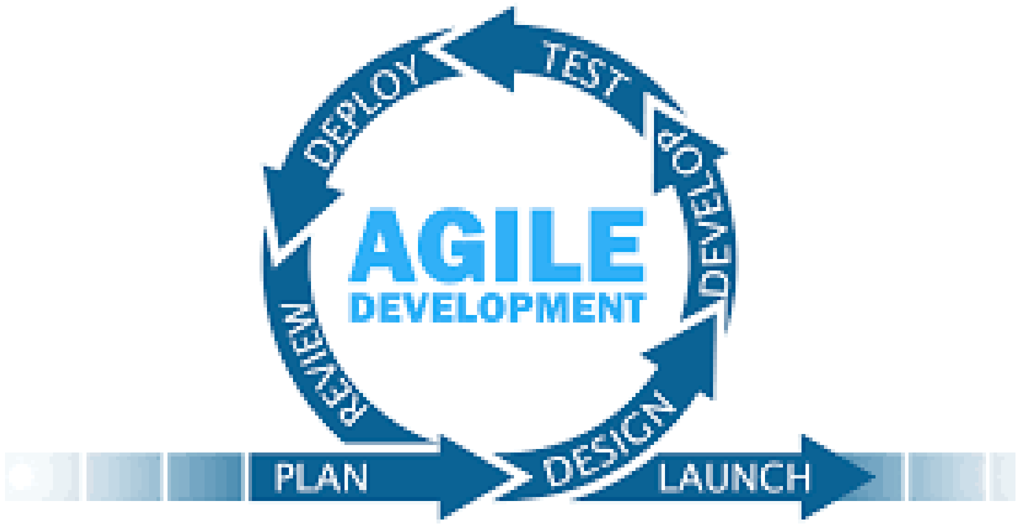
Agile Research



Carlo van Kessel
S-DB-IP3 and S-DB-GP3
S3-DB03
Hans van Heumen, Marc van Grootel
11-01-2023
V3

Version control

Version	Changes
1	Initial version.
2	Added XP. Added Kanban.
3	Added Personal Agile use.

Table of Contents

1. Personal Agile use

1.1 Backlog

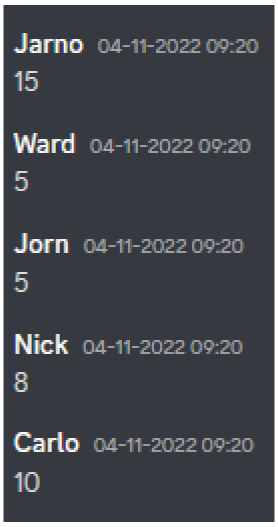
For the groups project we have created a backlog. In this backlog we have cards with user stories and cards with things to do. These are assigned to one or multiple people. For my individual project I have done the exact same. The scrum boards can be found here:
Groups project
Individual project

1.2 Daily standups

For our groups project every day we did a stand up. In this stand up we talked about what everyone was going to do that day and if they had any problems. We documented this and can be found here:
Group stand ups

1.3 Sprint poker

In our group project when we started a new sprint, we did a game called sprint poker. In this game you pick the user stories or requirements you want to make/do that sprint. Then you all at the same time choose a number on how many hours you think that task is going to take. So, it looks like this:



Jarno 04-11-2022 09:20
15

Ward 04-11-2022 09:20
5

Jorn 04-11-2022 09:20
5

Nick 04-11-2022 09:20
8

Carlo 04-11-2022 09:20
10

Here you can see that we each put a number down and most of the time you meet in the middle of all the numbers but when someone has a really big difference you can discuss this on why someone has a big or small number. Then when common ground is found you go to the next.

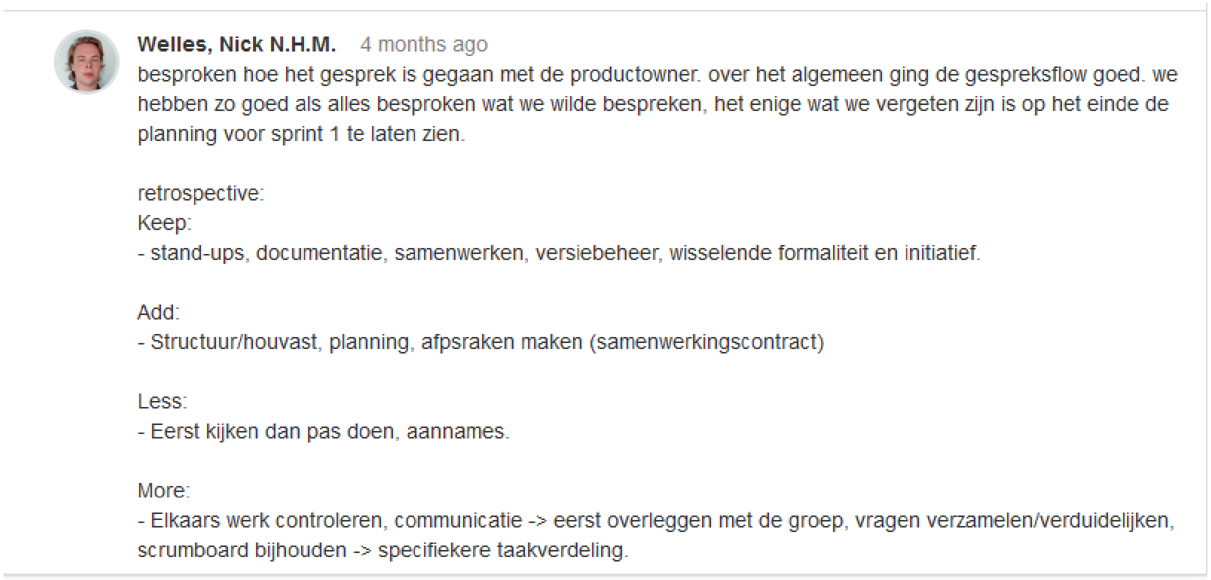
1.4 Sprint review

For each sprint we have done a sprint review with the costumer. We have done 2 different meetings: one where we as group where only in the room with the costumer and one where all the groups are around the meeting and can listen and see the meeting. I found that the one with all the groups is a bit less pleasant because of all the eyes on you. But it has also some benefits: you can see the progress of the other groups and get inspiration from it.

In these meeting we would always have a planning. Most of the time we would start with showing what we have done: so first the frontend then backend then functions. We would ask some questions we had and then ask if the costumer had some questions. I think the customer found this kind of structure good and was happy with the products.

1.5 Retrospective

After each sprint review, we did a retrospective. Here are the first and last retrospective:
First:



Welles, Nick N.H.M. 4 months ago
besproken hoe het gesprek is gegaan met de productowner. over het algemeen ging de gespreksflow goed. we hebben zo goed als alles besproken wat we wilde bespreken, het enige wat we vergeten zijn is op het einde de planning voor sprint 1 te laten zien.

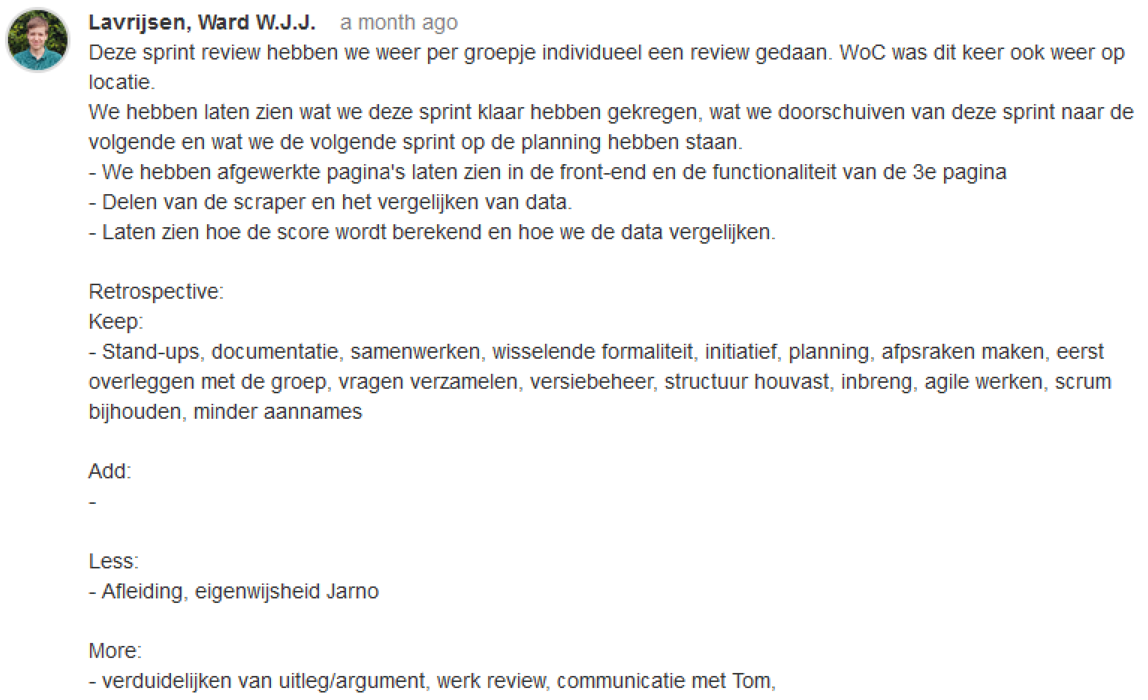
retrospective:
Keep:
- stand-ups, documentatie, samenwerken, versiebeheer, wisselende formaliteit en initiatief.

Add:
- Structuur/houvast, planning, afpsraken maken (samenwerkingscontract)

Less:
- Eerst kijken dan pas doen, aannames.

More:
- Elkaars werk controleren, communicatie -> eerst overleggen met de groep, vragen verzamelen/verduidelijken, scrumboard bijhouden -> specifiekere taakverdeling.

Last:



Lavrijsen, Ward W.J.J. a month ago
Deze sprint review hebben we weer per groepje individueel een review gedaan. WoC was dit keer ook weer op locatie.
We hebben laten zien wat we deze sprint klaar hebben gekregen, wat we doorschuiven van deze sprint naar de volgende en wat we de volgende sprint op de planning hebben staan.
- We hebben afgewerkte pagina's laten zien in de front-end en de functionaliteit van de 3e pagina
- Delen van de scraper en het vergelijken van data.
- Laten zien hoe de score wordt berekend en hoe we de data vergelijken.

Retrospective:
Keep:
- Stand-ups, documentatie, samenwerken, wisselende formaliteit, initiatief, planning, afpsraken maken, eerst overleggen met de groep, vragen verzamelen, versiebeheer, structuur houvast, inbreng, agile werken, scrum bijhouden, minder aannames

Add:
-

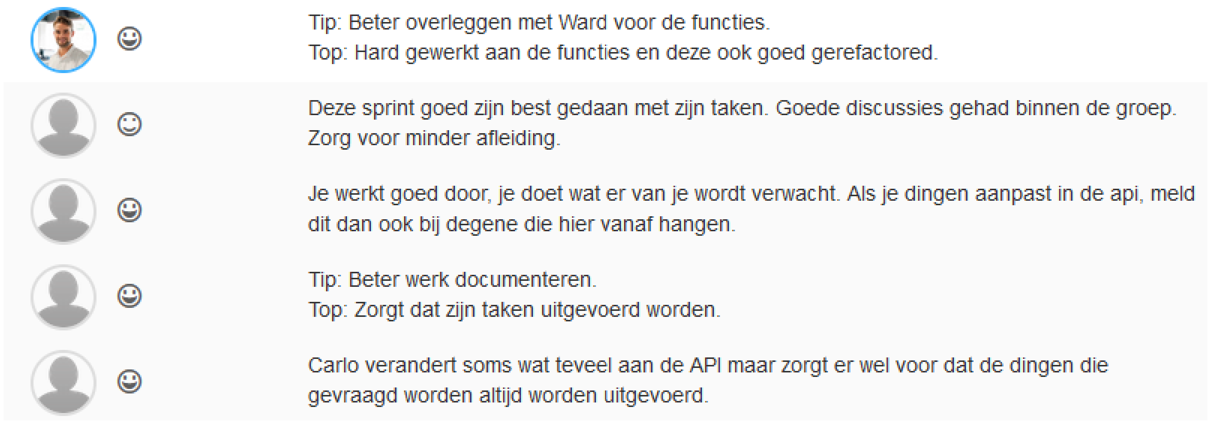
Less:
- Afleiding, eigenwijsheid Jarno

More:
- verduidelijken van uitleg/argument, werk review, communicatie met Tom,

As you can see in the last retrospective there are a lot more points in the keep sections and fewer in the more section. This means we progressed as a group.

1.6 Peer review

Each sprint we would also do a peer review. This is where you give feedback to your group members:



	☺	Tip: Beter overleggen met Ward voor de functies. Top: Hard gewerkt aan de functies en deze ook goed gerefactored.
	☺	Deze sprint goed zijn best gedaan met zijn taken. Goede discussies gehad binnen de groep. Zorg voor minder afleiding.
	☺	Je werkt goed door, je doet wat er van je wordt verwacht. Als je dingen aanpast in de api, meld dit dan ook bij degene die hier vanaf hangen.
	☺	Tip: Beter werk documenteren. Top: Zorgt dat zijn taken uitgevoerd worden.
	☺	Carlo verandert soms wat teveel aan de API maar zorgt er wel voor dat de dingen die gevraagd worden altijd worden uitgevoerd.

Here you can see the feedback. We tried to give some feedback on what they can do better and what was already going well. So, everyone knows what they are doing well and what they can improve on.

1.7 The difference between semesters

In the second semester I used the waterfall method. After using this scrum method fully as it should be used, I learned that this is much better communication wise and the program itself is more what the customer wants. I really recommend this way of working because you always know what group members are doing, you have good communication, and the customer can always step in and tell us if something needs to change.

2. Research

1.1 What is Agile?

Agile inside the universe of software development characterizes a bunch of structures and practices that endeavor to further develop the way software developers can work in an unsure and fierce climate. It depends on the Agile Manifesto which characterizes 12 principles that structure the basic beliefs of Agile software development. It gives apparatuses that permit developers to self-coordinate themselves in cross-functional teams.

Key Agile concepts are:

- User Stories
- Daily Standups and Standdowns
- Team
- Incremental Development
- Iterative Development
- Milestone Retrospective

1.2 What Agile methods exist?

There are a lot of methods/frameworks to work Agile. A few of them I have listed below, the top three I have explored in this report:

1. Scrum
2. Extreme Programming (XP)
3. Kanban
4. Adaptive software development (ASD)
5. Agile modeling
6. Agile unified process (AUP)
7. Disciplined agile delivery
8. Dynamic systems development method (DSDM)
9. Feature-driven development (FDD)
10. Lean software development
11. Lean startup
12. Rapid application development (RAD)
13. Scrumban
14. Scaled agile framework - SAFe

1.3 Methods

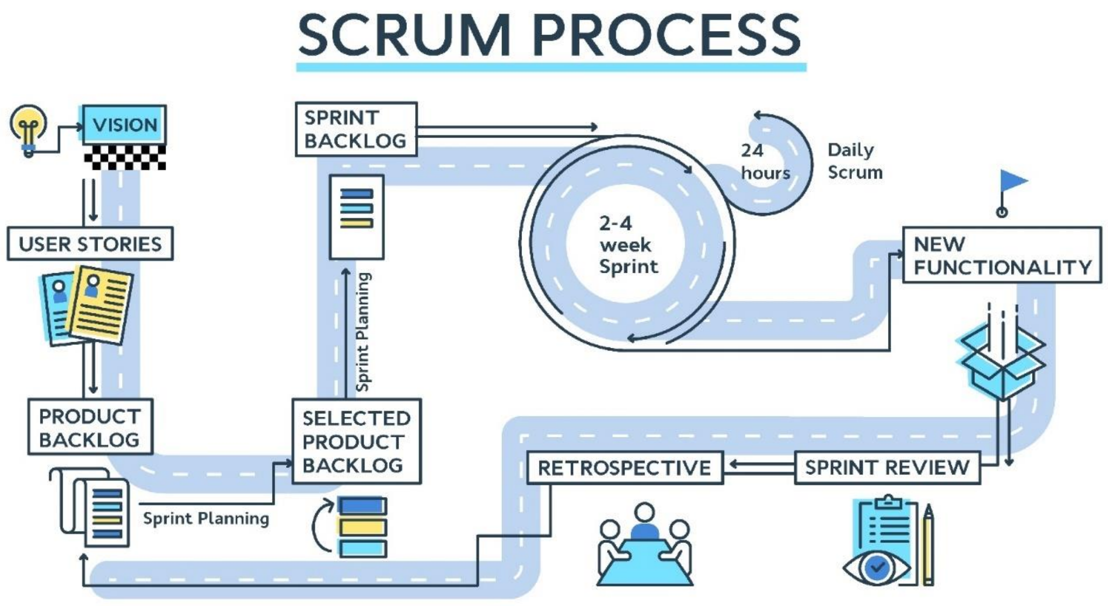
1.3.1 Scrum

Scrum is the most popular Agile method and the one that I personally have the most experience with. It characterizes a particular progression of work by which groups work in time-boxed iterations called sprints. These sprints are most of the times two to four weeks long.

1.3.1.1 Process

Scrum first starts with making roles. The scrum master needs to be selected. The scrum master guides the scrum process. This person ensures that the team can perform at its best. The team works with a product owner. The product owner ensures that his requirements and wishes are met and passed on to the team. These wishes are called user stories. User stories are the first step in the scrum process. Then these will be stored in a backlog where the developers will make a calculation of how many hours it will take to make these user stories.

At the start of a sprint a sprint planning is created. During this meeting various user stories are chosen to be worked on during the sprint and these user stories are then stored in the sprint backlog. During the sprint a stand-up is performed each day at a set time and location. This is a short session of about 15 minutes , facilitated by an appointed Scrum master, where team members report on their progress, any problems or issues encountered, and the task(s) they intend to work on during the day. At the end of a sprint a sprint review is held where the completed work gets presented to the stakeholders/product owner. his provides an opportunity for the development team to receive feedback, discuss impact of incomplete work and to receive suggestions for upcoming work from the stakeholders/product owner. This can also be a good moment to review the user stories in the product backlog and amend, update or re-order them if needed. The last part of a sprint is the sprint retrospective. During this session the team reflects on the past sprint: what went well and what could be improved in the next sprint. If many points for improvement are found, a selection should be made based on what the team determines are the points with the highest priority. After this step the project is either finished or enters the cycle anew and starts with a new sprint.

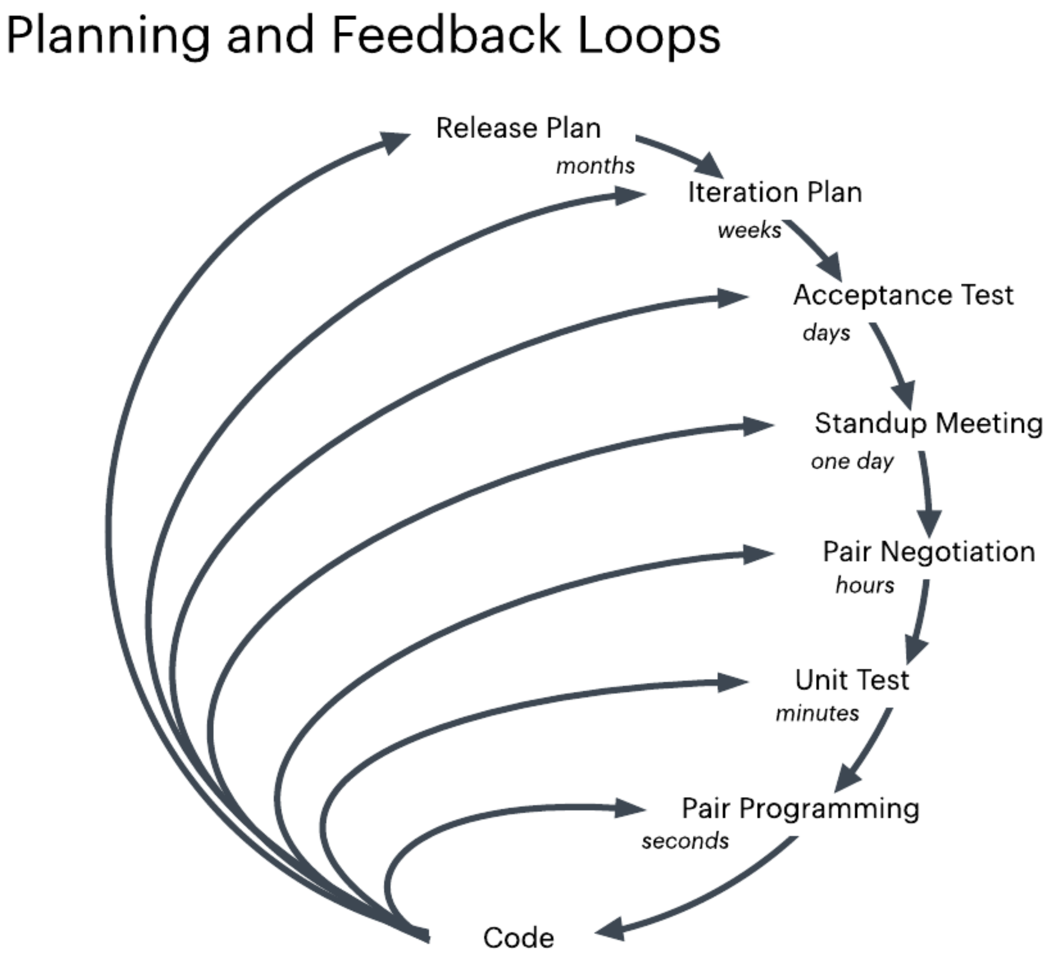


1.3.1.2 Sources

- https://www.scrum.org/resources/what-is-scrum
- https://agilescrumgroup.nl/wat-is-scrum-methode/

1.3.2 XP

Extreme Programming (XP) is another well-known agile method with a particular focus on extensive (unit) testing, pair programming, adding functionality only when needed, simplicity, and frequent communication with the customer. The name comes from the idea that practices that are considered beneficial when developing software are taken to an extreme level, such as frequent code review and rigorous testing.



When compared to Scrum, Extreme Programming contrasts in that it endorses more specific software development practices and strategies and is therefore just helpful for programmers. Scrum's emphasis lies more on project management and is therefore applicable to manage results of various sorts. Another distinction is that Extreme Programming can be executed in part and/or changed to suit the needs of the team. There is even a rule for that: "Fix XP when it breaks." Incorporating Scrum then again requires your team to completely embrace it and without modification, otherwise — at least according to the official Scrum Guide — you are not utilizing Scrum appropriately.

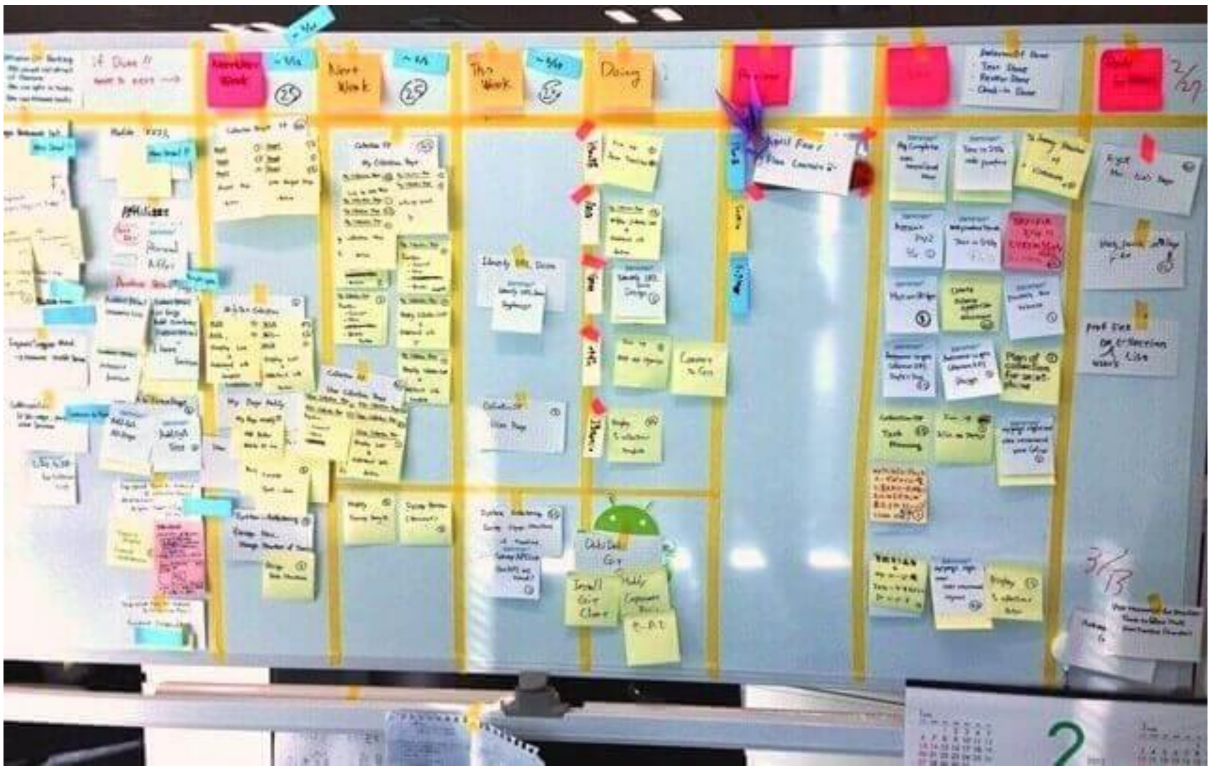
1.3.2.1 Sources

- https://www.stillgeek.com/extreme-programming-xp-an-agile-framework/

1.3.3 Kanban

Kanban, like Scrum, is a project management system designed to improve work across human systems. It places an emphasis on visualizing the workflow in a project. Central to this system is the Kanban board—Kanban being the Japanese word for signboard, billboard, or visual signal—which shows a clear overview of work in all its stages represented by cards, and work in progress limits for each stage. The overview provides complete transparency within the team—who is working on what, which tasks are available and which tasks are completed. The stage limits make sure that no stage becomes overwhelmed with too many tasks. If a stage has reached its task limit, an existing task must be completed first before a new one can be put in its place.

Overview of a Kanban board:



When compared to Scrum, Kanban varies in its ceaseless progression of work, whereas with Scrum work is partitioned into sprints. There are also no roles required within teams working with Kanban. Changes, (for example, to-do items or estimates on item/client story duration) can also be presented at any time whereas with Scrum these changes can be made during a sprint planning.

1.4 Sources

- https://en.wikipedia.org/wiki/Agile_software_development
- https://en.wikipedia.org/wiki/Extreme_programming
- https://en.wikipedia.org/wiki/Kanban_(development)
- https://en.wikipedia.org/wiki/Scrum_(software_development)