Harshdeep Singh 2017UCO1640

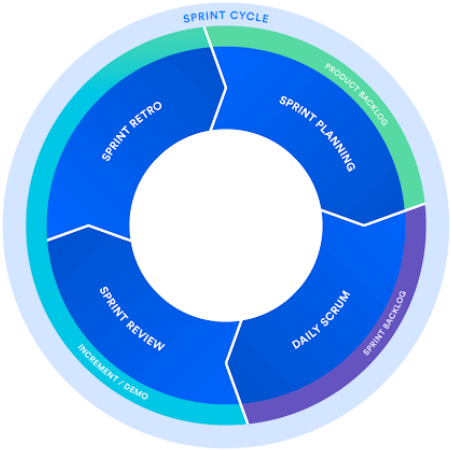
EPP Assignment

Scrum

Scrum is a framework that closely follows the principles of Agile development. It can be used in all types of work which requires teamwork but usually it is utilised in software development. Scrum enables the team to develop products effectively and precisely along with a retrospection at end that further reflects on their wins and losses to boast continuous improvement.

It all starts with a list of **product backlogs** which is basically a todo list for the project, this list may include new features or improvements and even bug fixes for the product. The product manager along with the development team sets the tasks that need to be done in the next few weeks by taking into account the financial and competitive edge completion would provide. The tasks that are chosen from the backlog are called as **Sprint backlogs**. This whole process is termed as **Sprint planning**. After successful completion of planning a sprint starts, sprint is the actual time period when the scrum team works together to finish an increment. Two weeks is a pretty typical length for a sprint, however it's not strict and can be changed according to needs. During the sprint a **Daily Scrum or Standup** is organised which is a short meeting that happens at the same time (usually mornings) and place to keep it simple. This is the place where one can raise their concerns or any blockers that they may face. Many teams try to complete the meeting in 15 minutes, but that's just a guideline. This meeting is also called a 'daily stand-up' emphasising that it needs to be a quick one. The goal of the daily scrum is for everyone on the team to be on the same page, aligned with the sprint goal, and to get a plan out for the next 24 hours.

At the end of the sprint, the team gets together for a review where they discuss their sprint results and showcases it to the product owner/manager and teammates for feedback. The product owner can decide whether or not to release the increment, although in most cases the increment is released. This is also the stage where the stakeholders plan on for the next sprint based on the results of the current sprint.



Lastly, a **Sprint retrospective** is held where the team comes together to discuss what worked and what didn't. It encourages the team to be open about various aspects such as relationships, tools, ideas, etc. This helps to improve the team as a whole for the upcoming sprint.

Extreme programming

Extreme programming (XP) is one of the most important software development techniques in the Agile models. It is most suitable for projects which have continuously changing requirements. The main characteristics of XP include dynamically changing software requirements; using a small, collocated extended development team; and leveraging technology that facilitates automated unit and functional tests.

XP is based on the frequent iteration through which the developers implement **User Stories.** User stories are requirements that are asked by the customer/manager from the developer. It is an informal statement rather than being a full SRS. This gives a vague idea of the product to be developed by the developer. On capturing user stories, the team moves on to develop a high level model of how the system will work (also know as **metaphor**). Afterwards the development team may decide to build a Spike for some feature. A **Spike** is a very simple program that is constructed to explore the suitability of a solution being proposed. It can be considered similar to a prototype.

XP is mainly helpful in small projects consisting of small team as it makes it easier for the team to discuss problems. It has been also proved effective for research projects which require changing of requirements swiftly.

Lean Development

Lean development is yet another agile method for software development. It originated from the principle of production line used in manufacturing goods. Similar to a production line, lean development follows a repeatable process and guarantees quality checks for the final product. Also it relies on the collaboration of one's coworkers or specialised workers. Lean Development is based on 7 underlying principles that are:

- Eliminate waste
- Build quality in
- Create knowledge
- Defer commitment
- Deliver fast
- Respect people
- Optimise the whole

Eliminate Waste: This process requires the team to get rid of any task that doesn't add value to the product or somehow hinders the development of the project. Some things that falls under this category are: manufacturing things that are not required now, use complicated tools that could have been done by simpler means, unnecessary code or functionality.

Build quality in: This phase focuses on improving the quality of the product without overdoing the testing i.e- coming up with quality issues that aren't even actual issues. To enable a good quality there are certain mechanisms already in place:

- Pair programming: Getting two developers instead of one to work on a single unit of product
- Test-driven development: Writing code that meets the given requirement and meeting the requirement being sole focus when writing code.
- Automation: Eliminate repetitive work done by developers and replacing it with automation

Create Knowledge: This step requires developers to create documentation for the work they have been doing to reduce knowledge gaps and train fellow teammates.

Defer commitment: This Lean principle encourages team to demonstrate responsibility by keeping their options open and continuously collecting information, rather than making decisions without the necessary data.
Deliver fast: It focuses on delivering the product as fast as possible to the customer for feedback. This does not imply that developers work more hours but rather work more efficiently and customer centric.

Respect people: The Lean principle of Respect for People is often one of the most neglected, especially in the fast-paced, burnout-ridden world of software development. It applies to every aspect of the way Lean teams operate, from how they communicate, handle conflict, hire and onboard new team members, deal with process improvement, and more.

Adaptive Software Development (ASD)

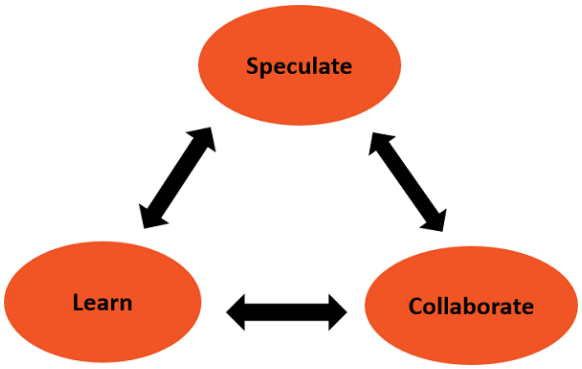
Adaptive Software Development is a method to build complex software and system. ASD focuses on human collaboration and self-organisation. ASD **"life cycle"** incorporates three phases namely:

1. Speculation

During this phase project is initiated and planning is conducted. The project plan uses project initiation information like project requirements, user needs, customer mission statement etc, to define set of release cycles that the project wants.

2. Collaboration
It is the difficult part of ASD as it needs the workers to be motivated. It collaborates communication and teamwork but emphasises individualism as individual creativity plays a major role in creative thinking. People working together must trust each others to: criticise without animosity, assist without resentment, work as hard as possible, possession of skill set, communicate problems to find effective solution.

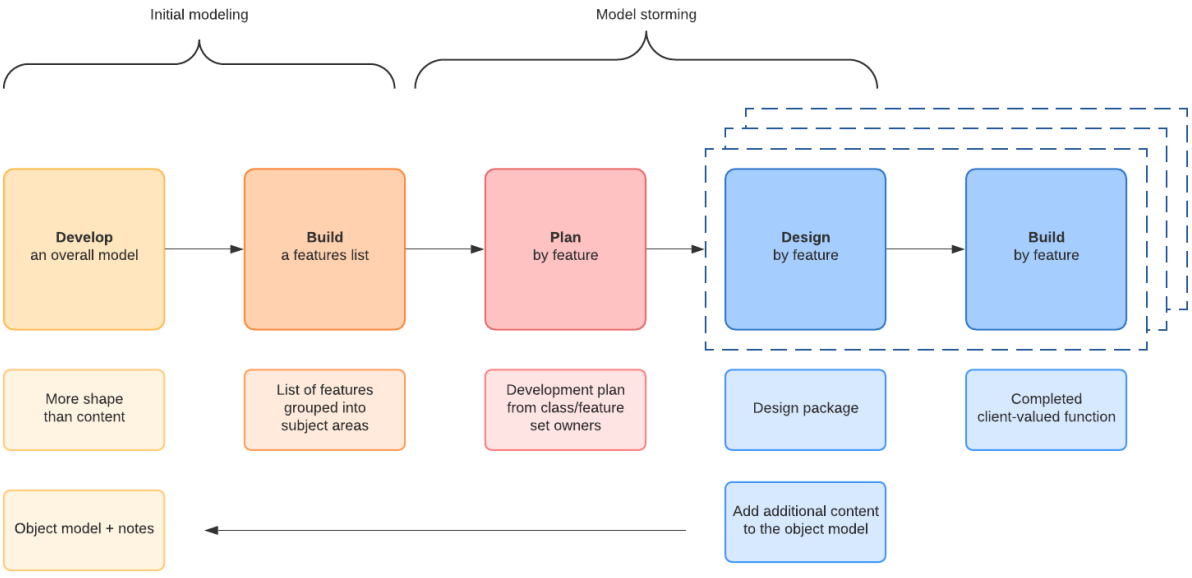
3. Learning:
The workers may have an overestimate of their own understanding of the technology which may not lead to the desired result. Learning helps the workers to increase their level of understanding over the project.



Feature Driven Development

Feature Driven Development (FDD) is an agile framework that organises software development around making progress on features. It conducts up-front planning, design and documentation and relies very heavily upon domain modelling. FDD also uses feature teams with chief architects/ programmers and traditional code-ownership and code-review, as opposed to pair-programming and refactoring. In contrast to scrum, FDD focuses to develop small feature sets in one to two weeks iterations. Like other agile methods this method has its own set of best practices as given below:

- Drawing a high level diagram of how the feature is to be implemented saves time by helping you uncover what function to add for each feature.
- If a feature is more time consuming i.e more than 2 weeks, it can be broken down into small pieces to accommodate 2 weeks timeline.
- Instead of pair programming, each class or group of code is given to a single owner.
- Periodic inspections are held within the team to uncover faults/bugs.
- Regular check for successful builds i.e- ensure that the project is building fine after each alteration to the project.
- Project managers should provide frequent progress reports of completed work.



Advantages of feature-driven development:

- Gives the team a very good understanding of the project's scope and context.
- Uses documentation to communicate thus eliminating the need of daily meetings as in scrum.
- Uses a user-centric approach. With scrum, the product manager is usually considered the end user. With FDD, the client is the end user.
- Breaks feature sets into smaller chunks and regular iterative releases, which makes it easier to track and fix coding errors, reduces risk, and allows you to make a quick turnaround to meet your client's needs.