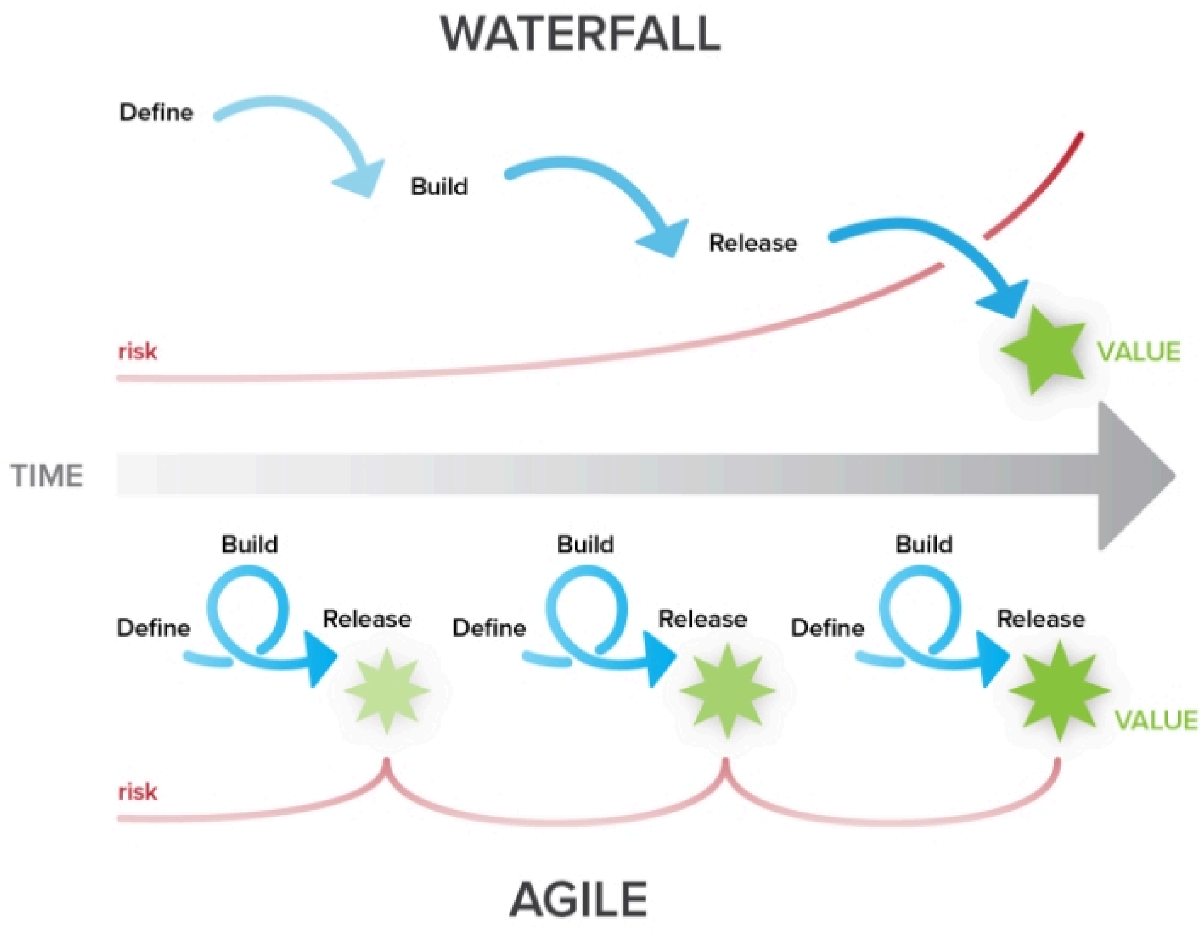
Scrum

Learn scrum

In this learning we will see what is **Agile Development**. Also, We'll see in detail how to manage a project using the Scrum techique which is a form or the flavor of Agile.

What is Agile Development?

Agile is a different approach to software development which replaces the old methology **Waterfall software development process**.



In **Waterfall model** one would consider the whole project at the beginning and do the following step by step: Define/Analyse, Build and release.
Each of these steps should be completely done before proceeding to the next stage. But the problem with this process and the model is that when a change happens at the middle of the project then you will face huge problems, as you didn't have any plans for it!

In Agile development you first define different pieces for your project and then do all the steps mentioned in the Waterfall process for each piece.
Agile methods use an iterative approach to a project with tight feedback loops that enable rapid adaptation when a new information or requirement for the project becomes available.

In simple terms instead of predicting the whole project requirements right at the beginning, like how you do it in the Waterfall process, you are an adapter and accept the fact that change may happen! So as an adapter you adapt yourself along the way as the project is moving forward.

What is Scrum?

Scrum is one of the most and widely followed Agile techniques.

Why do we need scrum?

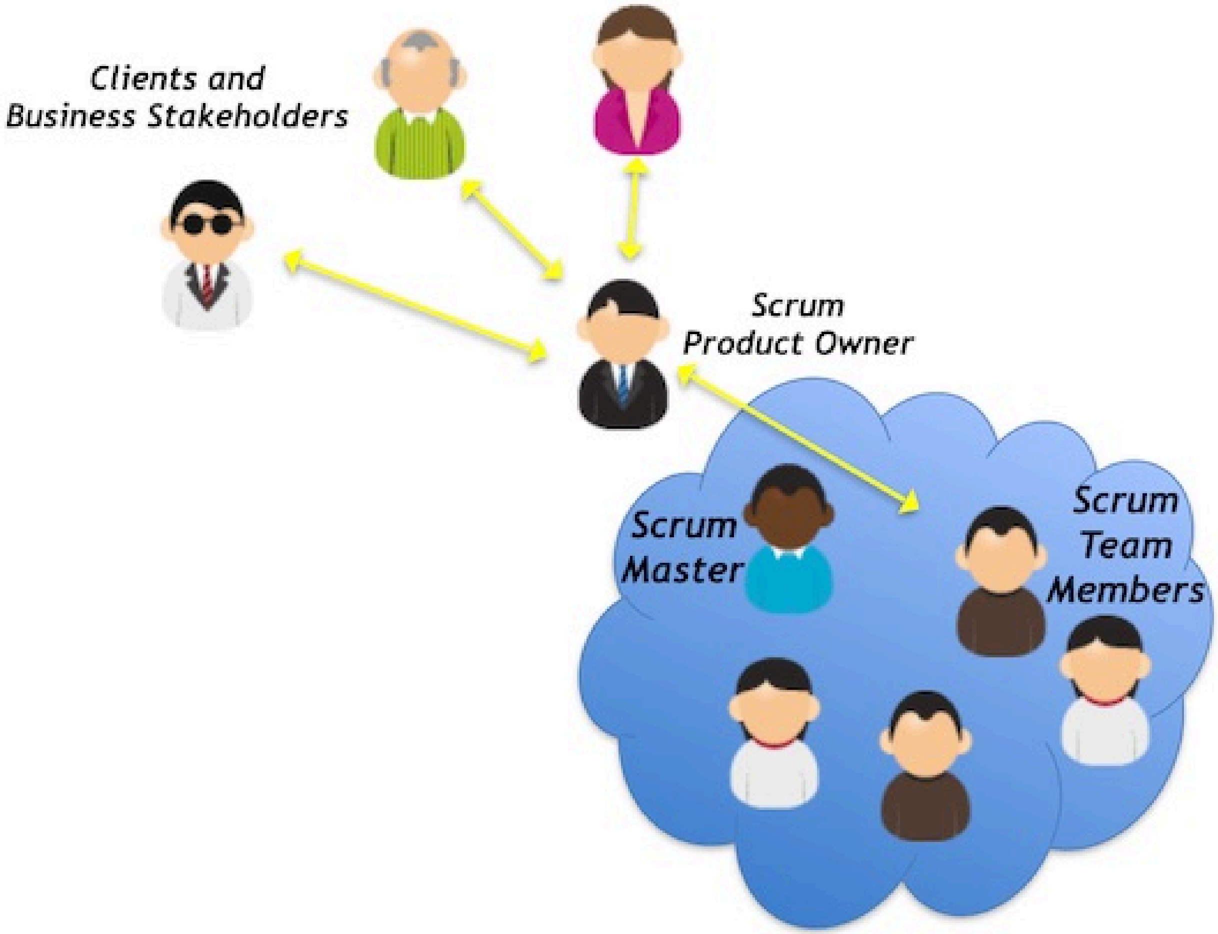
First let's just mention some of the problems that the traditional Waterfall project development process has:

- **Teams work completely independent from each other** and each sub-team completes its own activities before the other one starts its own. So what is the problem? An issue on one team, blocks the whole production line!
- **Everything is decided up-front!** So what is the problem? Well, although the client has mentioned what she needs at the beginning of the project, but in reality what she wants may change over the course of the project, and it's not her fault! It's just how the things are... No one can completely predict the project requirements and insist on that! Things may change and new decisions must be made.
- **Weak estimations!** As the tasks increase, the accuracy of how long each task takes decreases... Because unforeseen issues always arise, like bugs or client requirement changes

The scrum tried resolving the above mentioned problems:

- Inspecting the project's progress
- Evaluating the project's progress
- Adapting to the requirement of the project's in the pathway
- More importantly, the transparency is maintained through out the project

The scrum team and the people involved



- **Scrum product owner** This person is responsible for the overall project. They tells the features to be included and the main focus areas that team should be concerned about. They contact the stakeholders and are the point of communication to the enhance the value and significane of the project.
- **Scrum team members** This usually involves scrum developers of the project. They are responsible for coding and testing
- **Scrum master** They stand as a medium of communication between the Scrum product owner and the Scrum team members. The master of the Scrum should be sound technically and keep an eye on entire development process. Should always work with the product owners to keep the left over tasks(**Product backlogs**), and with the team to maximize the work. This master is repsonsible for **sprint handlings** and focusing on unplanned works to get it done.

Scrum process

Just like in the Waterfall model, team don't work on the project from start to finish, rather they divide the project into **pieces** according to the product backlog and work on pieces available in a short period of time known as **sprint**.

We would have a Planning Board for each sprint and developers aim to complete all of the **user stories** for that sprint on the board by evaluating how long each story would take to be done before the sprint ends.

So in a nutshell, each sprint has a defined number of tasks and they happen in a very short interval or time. So in this way any barriers will be highlighted very quickly and be fixed, team members also remain focused on small number of tasks and know exactly what they need to do. Any unplanned tasks or requirements can easily be planned into the next sprint. This whole process helps developers to have a constant speed of development, do better team work, provide early feedbacks to product owner and eventually build a better product.

Note on product backlog

- It's a list of tasks that should be worked so that it can be delivered.
- It's the Scrum Product owner who maintains them. Usually, it gets added on, removed if task is done as the project progresses.

User Story

- As per the definition, it's very short and one liners. It's a definition of the feature.
- The perspectives and the perseptions are mentioned here of a user, hence it's a user story.
- This person can be anyone (**Manager** was responsible in my earlier company)

Sprint

- As per the definition of the books, it's the basic unit of development in the Scrum methodology
- Usually, it will be be more than a day long and less than a month. It's commonly 2 weeks long (I learnt from my previous company). And the product should be in a stable state after every sprint.

Estimation of work in sprints?

Before starting a new sprint, we should decide which stories need to be done in the upcoming sprint.(**Important and complicated task**)

Scrum product owner will have few stories that team needs to work on. This list of stories contains more work than the team can achieve in a sprint, specially at first days of the project. The Scrum master needs to convince the product owner, how much work can be done in a sprint, and over time the product owner learns the approximate velocity of the team and also the team's productivity increases.

Now in order to understand our velocity, estimate how much time each story may take to be done and in overall how many stories can be done in a sprint, there are a couple of ways which help us in this process.

- **Story points:** It's one of the most common methods for estimating stories. A story point is actually the simplest story to be done and developers refer to it as the story of having one point. In some projects it can be only a ten minutes work, such as fixing an UI element; whereas, in some more complex projects it can be a two hours work which includes more than one person to complete. There are several point techniques to better help estimate stories:

    - Use Fibonacci Numbers: 1,2,3,5,8 (and maybe 13, but a 13-sized story seems too big).

- **Semaphores:** we can associate stories with colors instead of numbers to defines the complexity level of a story. So for example green defines easy tasks that can be done in the next sprint for sure, yellow is for a little complex story that may involve more than one person but still can be completed in the next sprint, red is for the story that is extremely difficult and cannot be finished in a single sprint.

- **T-Shirt Sizes:** We can use sizes like S, M, L, XL, XXL for our stories. This technique simply gives up on comparing story complexity with time of completion.

Estimation is a really important thing that we need to consider. Product owner and stakeholders need to know what to expect at the end of the sprint, they should be able to know when to release the product and what features are ready.

Let's understand planning board better in an example

- **Product Backlog:** All the project stories reside here. (Stories in the picture)

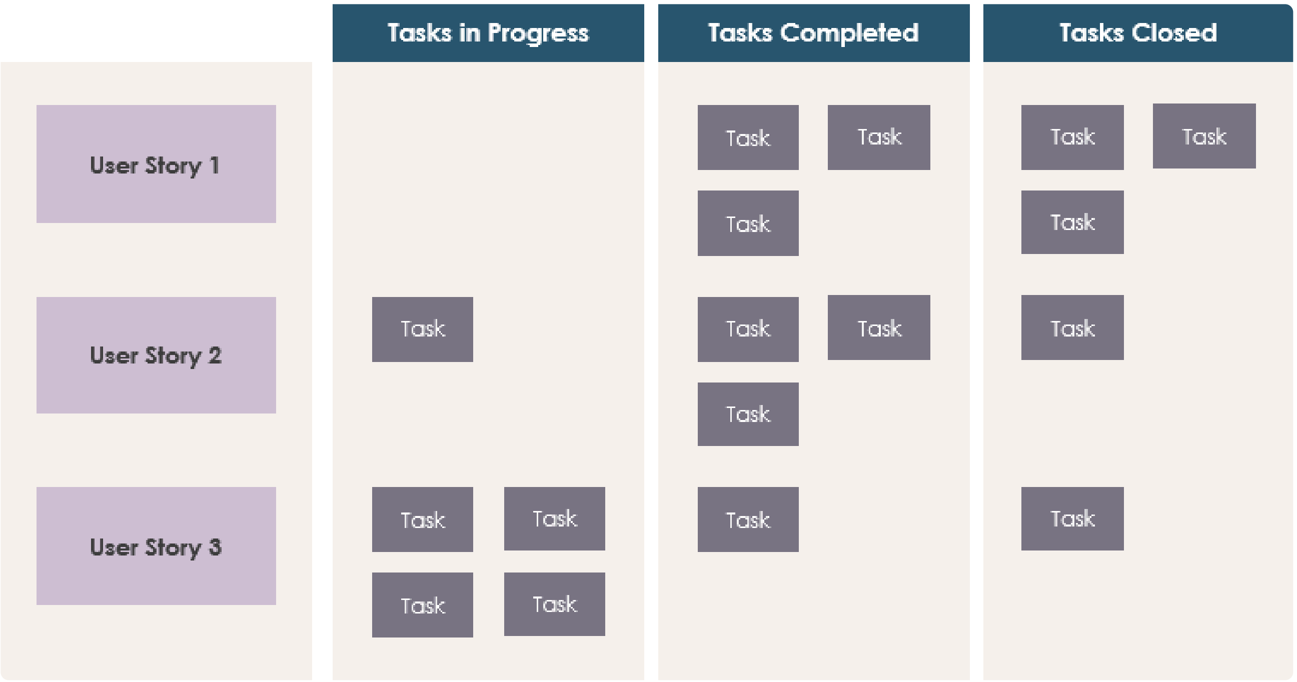
- **Sprint Backlog:** The stories that should be done in the current sprint are here and the team focus to finish them all by the end of the sprint. (Stories in the picture)

- **Tasks in progess:** Team members take a story from the 'Sprint Backlog' column and put them here to show others what are they currently working on. This is needed for transparency and improving productivity.

- **Task is completed:** This column is the place where all completed stories go. Ideally, all of the stories in 'Sprint Backlog' should be here by the end of the sprint.

- **Task is closed:** This column is the place where all done stories go. (**Managers do this job usually**)

As each sprint starts, some stories will be moved to 'Sprint Backlog' from the 'Product Backlog' and all of those in the 'Sprint Backlog' should be moved to the 'Task completed' column.



Sprint working style

So as we know, after each sprint, our product should be in a stable state. But how is that possible? Well, we need some rules and meetings which are called sprint rituals.

Sprint planning

Occurs at the start of the sprint. Teams review the product backlog, pick up some of the high priority tasks and try to deliver them by the end of the sprint.

How long does it take? This meeting can take a whole day, but shorter sprints only need a couple of hours or so.

What are its results? After the meeting we will have sprint backlog. It includes some tasks that need to be done in the current sprint from the product backlog.

Stand up (Daily Scrum):

Occurs early in the day at an exact hour and each team member needs to answer to the following three questions:

1. What has been done yesterday?
2. What are we going to do today?
3. Are there any problems blocking you?

Scrum master should help to solve the blocking problems and impediments. Sometimes the problems can be solved in a timely manner but other times, there might be a serious issue that solving it itself would take a couple of weeks. Resolving any blocking issue that is preventing the development process should be the team's high priority.

How long does it take? Stand up should be fairly quick, each team member should only speak for a couple of minutes. That's why all team members stand up and speak, so after a little while if you have seen that someone is looking for a chair to sit, it's probable that the meeting has gone too far!

What are its results? Transparency about what each team member is working on and identifying the impediments.

Sprint review:

Occurs at the end of the sprint. Developers, scrum master and product owner discuss what has been achieved during the sprint. And a Demo should be prepared to show to the product owner the new features that has been added.

How long does it take? Well not more than an hour or so. It depends on the number of features and how you like to showcase your demo.

What are its results? Based on the demo, product owner can decide whether to publish a new version of the application or not. Sometimes although the added feature is not completed yet, a release will help to gain users' feedbacks early in order to decide what to modify to satisfy more users as much as possible.

Sprint retrospective:

Occurs at the end of the sprint. In this meeting developers provide feedback about what went well in the passed sprint and what went badly. There are no right or wrong answers here, developers simply give their own opinions. They should put aside their own personal feelings and speak about what is bothering them.

How long does it take? It's a short meeting that wouldn't take more than an hour if there were no serious issues that is bothering team members.

What are its results? The most important thing is to propose a solution for any problem and let everyone a couple of minutes to think about it so that we can avoid it for the next sprint.

Some tools to look for handling Scrum

- nTask
- Jira
- Targetprocess

Note:

- I personally used Jira for the projects.

Signing off:

Akshay Prasad

#Developer #TechEnthusiast #Agile #Agilelearning #scrum #scrumlearning #projectmanagement #software