Write short notes on following

Scrum

· Lean Development

· Extreme programming (XP)

· Adaptive Software Development (ASD)

· Feature Driven Development

Answers starting from next page.

Submitted by:

Abhyarchi Harsh

2017UCO1686

**Scrum**

Scrum is a process of developing software that helps developers achieve their goal methodically. It is a subset of Agile development.

Scrum is basically a strategy by which all the developers plan and then cooperate to work towards a common problem systematically.

It is best suited when a team with different functional areas is working on the same product and the work can be split in 2-4 weeks duration.

It starts with the team meeting to understand the product backlog - take notes of what needs to be completed and how much time it’ll take.

Scrum also uses the agile methodology Sprints - a period of time when development actually takes place. It lasts anywhere from a week to a month and the goal is to produce a saleable product. There are sprint reviews after every completed sprint where the entire team reviews the result of the sprint along with the stakeholders, so that any feedback and comments can be integrated in future sprints. Sprint retrospective needs to be performed after the review so that the next sprint backlog can be defined.

There are daily standups - or daily scrums - where the team meets to discuss what’s been done, and what’s on the table to be done for the day.

Scrum works on values like commitment, focus, openness and respect. Each member is expected to have or learn these values during the process for a successful scrum.

Principles such as transparency, inspection and adaptation are very crucial to any scrum team, and work as its backbone.

Scrum Lifecycle:

1. Scrum Planning (Product Backlogs)
2. Sprint Planning - how to complete product backlogs
3. Sprint Development - developing of the features decided in sprint planning
4. Daily Scrum meets - to be aware of things done, and to be done
5. Sprint Review - to review requirements supposed to be met vs what did; and take feedback
6. Sprint Retrospective - use feedback to address features of next sprint plan
7. Repeat steps 2-5 until all product requirements are met.

**Lean Development**

Lean Software Development (LSD) is an agile framework which focuses on what is the most important thing that needs to be covered, and based on optimizing development time and resources, and eliminating waste it delivers what the product needs. It is oftentimes also called the Minimal Viable Product(MVP) strategy, in which a team releases a very basic version of its product, gets user feedback on what could be added, improved and then works upon this feedback.

The main motives here remain optimisation of resources and reduce waste - but also covering all needs.

The advantages include a very streamlined approach with a clear idea of what needs to be worked upon. It gets rid of unnecessary activity which can ultimately reduce expense. It also empowers the development team to make its decisions, not relying on the managers to instruct them , thus boosting morale.

Some disadvantages could be the lack of scalability - since it depends a lot on the team working on it that it might not be possible to reproduce the process if another team starts working on it. It also depends on very clearly defined requirements, and if those aren’t correctly identified, could be a major loss.

The one major benefit of lean development would be that since it works on continuous feedback, the requirements are originated straight from the source, thus does not include any unwanted developments.

It is based on 7 major principles:

1. Elimination of waste and unwanted features.
2. Amplifying learning
3. Dealing with Late Decision Making
4. Ensuring Fast Delivery
5. Encouraging Team Empowerment
6. Building trust and integrity
7. Viewing the application as a whole - not in parts.

**Extreme Programming**

It is a very specific agile methodology which focuses on creating higher quality products that can adapt with changing requirements while maintaining high quality life of the development team. It is very specific regarding the appropriate engineering practices for developing software.

It is built on 5 fundamental values:

1. Simplicity: do what and how much needs to be done.
2. Communication: every smallest requirement and code to be communicated - do it together.
3. Feedback: Keep in mind every small iteration, work on it. Demo early, take feedback, work on it.
4. Respect: respect each and every input, and every person giving it.
5. Courage: no false excuses, owning failures and working on them.

Works on some extreme rules and extreme practices that ensure high quality software to be produced. Extreme Programming assumes that the cost of changing a program can be held constant over time.

It involves:

* Unit tests written before programming and keeping all of the tests running at all times. These are automated and recognize defects early in development.
* Simple design meeting requirements first, but redesign should be feasible as and when required.
* Pair Programming - two programmers at one screen. Taking turns, one of them is at the keyboard, the other constantly reviews and provides inputs.
* Testing happens throughout the day.
* Shipping out minimal working product early, then upgrading it very frequently keeping in mind needs and feedback.

Benefits include rapid development, working quickly on feedback, low defect rates, high scalability, reduced costs, high customer satisfaction and also, high work satisfaction.

**Adaptive Software Development**

It is a method that focuses on building very complex softwares and systems. Works on human collaboration and self organisation.

ASD Life Cycle:

1. Speculation
2. Collaboration
3. Learning

Speculation aims at collecting project needs and planning on how to meet those needs with every release - what features to work on.

Collaboration is the most crucial part where individuals as a team collaborate and work on the product requirements. Trusting fellow teammates and being open to criticism are very important qualities that every team member should possess.

Learning is a way to make sure that the developers working on the product should have the appropriate and up to date knowledge of the skillset they would be using. Learning can happen in ways like focus groups, technical reviews and project postmortem.

These practices accommodate change and are adaptable in frequently changing environments with products being developed with minimal planning and learning.

**Feature Driven Development**

It is an Agile framework focusing on planning software development by making progress on given features to work on. They may not necessarily be product features, but can be compared to user stories in scrum.

Project lifecycle in a FDD framework is:

1. Develop an overall model.
2. Make a list of features.
3. Plan according to a feature.
4. Design around the feature.
5. Build the feature.

Advantages of this include faster development, with a defined 5-step lifecycle. Can move forward with a product with continuous successful integrations.

Disadvantages include high dependence on software developers, not effective for small projects and no very clear documentation - can lead to confusion.

Should be used with large scale products which are being developed by highly skilled developers who can handle all the use-cases of the product themselves and deliver products with high expertise.