**Assignment**

**Made By : Karanpal Singh**

**Roll No : 2017UCO1643**

**Q.** Write short notes on following:

* Scrum
* Lean Development
* Extreme programming (XP)
* Adaptive Software Development (ASD)
* Feature Driven Develop

**Ans.**

* **Scrum:** Scrum is a lightweight agile framework that focuses on delivering products in a short amount of time (or in short cycles).Scrums consist of a **product backlog, sprint backlog, development team, scrum master, product owner and increment.**

Scrum works on the concept of **sprints.** A sprint can be thought of as a duration(1 or 2 or maybe 4 weeks) in which some new set of features are added into the existing application and delivered to the customers. Scrums are headed by a scrum master whose job is to establish an environment where the team can be effective, address the different obstacles that the team might encounter, maintain a healthy relationship with the clients, motivate the developers and protect them from outside distractions. This helps in increasing productivity and as a result, the entire process becomes smoother and flexible. The risks are minimized and the waste is reduced.

The product owner describes the vision of the product that needs to be developed and organizes the backlog. Sprint planning is done in which the teams decide the duration of each sprint, the number of features that need to be implemented, etc. Each feature is prioritized and given a priority, and the scrum master decides in which sprint that feature will be implemented. Complex features are broken down into simpler modules and then conquered individually and finally merged together to form the overall feature. Scrum teams are cross functional and self organising, that is, they have the capability to implement complex features without outside interference. The members of the development team constantly train each other so that none of them becomes a bottleneck in the long run. Scrum meetings are conducted by the scrum master wherein the code is reviewed, feedback is taken from the product owner, discussion on what all new requirements are to be added are discussed. What were the obstacles faced in the last sprint, what all work had been done yesterday and what all needs to be completed today are also discussed.

* **Lean Development:** As the name suggests, Lean Development is an agile framework that stresses on eliminating waste, reducing costs, budgets and providing only those functionalities that a user asks for. Lean Development focuses on shorter iteration cycles. The team develops a bare minimum of the product, and releases the product to the customers. After using the product for a certain period of time, the customers/users of the product can provide valuable feedback to the developers regarding what all features can be added to the product, what all can be taken down, etc. These new set of requirements can be taken care of in the next iteration. It also takes relatively lesser time to deliver the product and also to add new features.

The Lean Development strategy comprises 7 principles. They are:

1. **Eliminating Waste -** Any factor that can lead to inefficient product or delayed delivery should be avoided and eliminated. Examples: useless code, slow communication, etc. Regular scrums need to be conducted to eliminate such waste.
2. **Build Quality In -** Quality Software should be the main objective of the project and thus constant feedback, regular testing, and various other strategies like pair programming should be considered to produce good quality software.
3. **Creating and Amplifying Knowledge -** Regular meetings, code reviews, documentation and training sessions should be conducted to ensure that knowledge is not accumulated by just a single person who is writing the code.
4. **Defer Commitment -** Till the time, the members don't have a clear understanding of the project, decisions that can impact the product drastically should be deferred and considered later. Actively collect more data to analyze the system under construction so that better decision making can be made.
5. **Delivering fast -** Start with a simple model, include only a bare minimum of functionality and deliver the product to the customer. Through continuous iterations and feedback, evolve and improve the product over time.
6. **Respect the team -** Constantly motivate and encourage other team members to do their best, consider their viewpoint and decisions and integrate them if the team finds it appropriate, appealing and useful.
7. **Optimize the whole -** Tighter deadlines may force the team to write sloppy and messy code full of defects that might put more pressure on the testers and therefore quality of the code gets destroyed. Instead of optimizing individual sections of a project, the team should focus on optimizing the overall value streams. Make sure there is efficient communication between the team members so that delays are minimized.

* **Extreme Programming:** Extreme Programming is an agile development process that focuses on better software development and shorter iteration cycles. This allows the developers to accommodate the changing requirements and evolve the product over time. Extreme programming is useful when the requirements frequently change, the customers are not sure what the final product will look like, the number of programmers required are small etc.

Extreme Programming consists of a number of phases that enhance the overall software development life cycle.

1. **Simplicity:** The developers try to create a model that is simple to implement, easy to visualize and can be accomplished with the specified timeframe, instead of creating a complex model so that the entire team can evaluate whether we can perform some tasks more efficiently and easily.
2. **Testing:** Unit tests are written before programming and they are run at all times in order to follow TDD(Test Driven Development). This helps in eliminating bugs and defects early.
3. **Feedback:** The customers are involved at every stage of the development cycle and as a result continuous feedback from both the developers and customers help in simplifying the design process, improving the overall quality of the software and adapting to changing requirements.
4. **Refactoring:** Refactoring is performed daily in order to simplify the tasks, code reviews are conducted daily and this makes the code clean and concise. Code that is not required anymore is flushed out.
5. **Pair Programming:** Two people work together and share the same system. This helps them in discussing new ideas, better decision making happens and they in turn take the command of the system one after another.
6. **Early Releases:** As other iterative methodologies, this framework focuses on interacting with the customers early on, so that they can get a sense of what the final product will look like and what all changes need to be made in the current build of the product.

* **Adaptive Software Development:** Adaptive Software Development(ASD) is a direct outgrowth of Rapid Application Development(RAD). The development of the product starts with little planning, but with continuous learning, this methodology allows us to adapt to the changing needs by accommodating the newer ones, and as a result the product evolves over time. John Highsmith and Sam Bayer invented the ASD approach in the early 1990’s. ASD helps in developing better products since it is more user focussed and there’s more liquidity.

ASD is a 3 phase and a cyclic process. The 3 phases are:

1. **Speculate -** The team figures out the project requirements, the aim of the project, the customer needs and initiates the project. ASD prefers iterations with shorter cycles.
2. **Collaborate -** The team must be able to collaborate with each other in order to produce products with high quality and performance. The team members must be able to discuss their ideas, criticize each other’s viewpoints, provide valuable feedback, assist other team members, share knowledge.
3. **Learn -** The members must continuously learn and enhance their knowledge, domain expertise in order to develop a sustainable and successful product. Feedback must be provided after each iteration, both from the customer’s viewpoint as well as the developers' viewpoint so that appropriate decisions and changes can be made in the next iteration.

* **Feature Driven Development:**  Feature Driven Development(FDD) is an agile framework. It is an incremental and iterative process that tries to integrate a whole lot of practices into a more sophisticated and cohesive unit. FDD focuses on making progress on features. Features are of a similar character to user stories in a scrum. Features can be visualised as subparts of a bigger complete unit. Examples of features include
  + Polishing the UI of the application
  + Completing the validation process, etc.

FDD was first introduced in 1999 via the book [Java Modeling In Color with UML](http://www.amazon.com/exec/obidos/ASIN/013011510X/ambysoftinc/) and later refined by **Jeff De Luca.**

FDD is a five step process:

1. **Developing an overall model -** An overall model is created by the chief architect or the leader of the project that depicts the overall architecture, and in some sense, defines the context of the application. This helps in letting the other members know what the project is all about and thus provides a better understanding of the model.
2. **Design a list of features -** Though brainstorming, the developers decide which all features need to be included in the application. Each feature has a timeline of around 2 weeks. So the feature must be such that it can be implemented within this timeframe.
3. **Plan out each feature -** Each feature is given a priority to see how important it is from the application’s and the client’s context. Developers are broken down into teams and a feature is then assigned to them on which they can work.
4. **Design each feature -** Each feature is further analyzed, class and flow diagrams are drawn and a methodical approach is discussed to follow the design process which is then inspected and finalized.
5. **Build each feature -** After the inspection process is over, the features are implemented by the respective teams, tested in isolation and added to the final product if approved by the project leader.