Write short notes on following

Scrum

· Lean Development

· Extreme programming (XP)

· Adaptive Software Development (ASD)

· Feature Driven Development

**EPP FINAL ASSIGNEMENT**

SUBMITTED BY **–**

**Harsh Yadav**

**2017UCO1597**

**COE-2**

**Q1. SCRUM**

Scrum is an agile project management methodology or framework used primarily for software development projects. The Scrum methodology follows the values and principles of agile, but is more enhanced in terms of specifications, especially related to some software development practices.

Scrum tries to make information regarding a project as transparent as possible so that the development team can improve and adapt on the current requirements and feedback. This allows teams to address the common pitfalls of a waterfall development process: chaos resulting from constantly changing requirements; miscalculation of time, resources and cost; probably poor software quality; and inaccurate progress reporting.

Transparency of common terms and standards is required in Scrum development to ensure that what is being delivered is what was expected. Timely meetings and inspections, detect variances early on so that adjustments can be made quickly.

**Components of Scrum**:-

· The three roles: Scrum Teams are self-organizing and cross-functional. Self-organizing teams decide on their own how they are going to achieve their goals, rather than being instructed by others outside the team. Cross-functional teams have all requirements needed to achive the work without depending on others. The team model in Scrum is designed to optimize flexibility, creativity, and productivity.

· A prioritized Backlog containing the end user requirements

· Sprints

· Scrum Events: Sprint Planning answers the following: What are the goals, or deliverables in the Increment resulting from the upcoming Sprint? How the above planned goals be achieved which includes all the technicalities?, Daily Scrum Meeting, Sprint Review Meeting, Sprints Retrospective Meeting dealing with past events and situations.

The Scrum Framework tries to deal with the fact that the requirements are likely to change quickly or are not completely known at the start of the project.

**Q2. LEAN DEVELOPMENT**

Lean Software Development (LSD) is an agile framework which focuses on improving development time and optimize the usage of resources, reducing or eliminating waste, and finally delivering only what the product needs. The Lean Software Development is also often known as the Minimum Viable Product (MVP) approach, in which a team releases a bare-minimum version of its product to the market, gets feedback from the users, and then iterates based on this feedback.

Lean development reduces or eliminates unnecessary activity, and as a result can optimize the usage of resources and reduce costs of development. Empowers the development team to make decisions, which can also boost morale.

This approach greatly depends on the team involved in the process, which makes it not, as scalable, as other frameworks. It also depends on strong documentation, and failure to do so can result in development mistakes.

Principles of Lean Development :-

· **Reducing Hindrances and Wastes** - Unnecessary features and code, More tasks in log than can be completed, Delays in the engineering process, Inefficient communication, Issues with quality, Data duplications. These are some types of wastes. To identify and eliminate waste, after each short iteration, timely meetings are held by Project Managers.

· **Quality of the Product** - Quality management is also a one of the important principles in LSD, as problems in this domain can lead to different types of waste. Repetitive testing of the code, problems in logging and their resolution takes considerable time and therefore lead costs of development higher; LSD strives to identify and resolve such nuances before they even happen.

· **Postponing Commitement** - It always leaves scope for optimizations and improvements by delaying decisions which can’t be modified or undone in future, until all the needed experimentation is done and as much information as possible is collcted.

· **Delivering Fast** - approach allows to enhance a piece of software incrementally, based on the feedback collected from real customers, and ditch everything that is of no value.

**Q3. EXTREME PROGRAMMING**

Extreme programming (XP) is a software development methodology which focuses to improve software quality and responsiveness to changing customer requirements. As a type of agile software development, is suggests more frequent releases in short development cycles, where the intention is to improve productivity and introduce checkpoints at which new customer requirements can be adopted.

It is applicable when there are Dynamically changing software requirements, risks caused by projects being completed in fixed time and using new technology, small and co-located extended development team, rhe technology used allowing for automated unit and functional tests.

The five values of XP are :-

**Communication** - Software development is fundamentally a process requiring a team of people, and it is necessary to communicate among them to share knowledge of resources and requirements so that every member is aware of the product requirements and deadlines, and the methodology used.

**Simplification of the Design and process -** means identifying the most simplest solution which will fulfill the current needs. The purpose of this is to avoid waste and do only what is necessary and nothing extra such as keep the design of the system as simple as possible so that it is easier to maintain, support, and modify.

**Feedback of the users** - Through constant feedback about their previous efforts, teams can identify areas for improvement and revise their practices. Feedback also supports simple design. Your team builds something, gathers feedback on your design and implementation, and then adjust your product going forward.

**Courage and Confidence** - You need courage to raise organizational issues that reduce your team’s effectiveness. You need courage to stop doing something that doesn’t work and try something else. You need courage to accept and act on feedback, even when it’s difficult to accept.

**Respect** - The members of your team need to respect each other in order to communicate with each other, provide and accept feedback that honors your relationship, and to work together to identify simple designs and solutions.

**Q4 ADAPTIVE SOFTWARE DEVELOPMENT**

Adaptive Software Development (ASD) is an extension of an earlier agile framework, Rapid Application Development (RAD). It focuses on enabling teams to rapidly and effectively adapt to changing requirements or market needs by improvising their products with planning and continuous learning.

Strengths shown by ASD are: Focused on the end users, which can lead to better and more intuitive products, Allows for on-time and even early delivery, Encourages more transparency between developers and clients.

The ASD follows a three-phase process: speculate, collaborate, learn.

**Speculate** :- The pre-decided goals and features, bound the manager and his ability to modify the project in a more innovative path. In ASD, planning is replaced by speculation. While speculating, the team does not abandon planning, but it identifies the uncertainty in complex projects. Speculation encourages experimentation. Iterations with short cycles are encouraged.

**Colaborate** :- Complex applications require that a large volume of information be collected, analyzed, and applied to the problem. This results in diverse Knowledge requirements that can only be handled by team collaboration. This needs teams members to work jointly, share knowledge of resources, optimized solutions, and clarifications of changing requirements.

**Learn** :- Team has to enhance their knowledge constantly, using practices such as − Technical Reviews, Project Retrospectives, Customer Focus Groups. Reviews should be done after each iteration. The team learns − About product changes, More fundamental changes in underlying assumptions about how the products are being developed, The iterations need to be short, so that the team can learn from small rather than large mistakes.

**Q5 FEATURE DRIVEN DEVELOPMENT**

Feature Driven Development (FDD) is an agile framework which moves forward by tracking the progress on features being built. Features in terms of FDD are not necessarily product features. They are, instead more related to user stories in Scrum. It is an iterative and incremental software development process.

FDD in suggests reporting of progress status at all vertical levels which helps all the people working on the project have a track of progress and results, enabling them to decide speed and deadlines of the subtasks.

FDD allows teams to update the project regularly and identify errors quickly. Clients or stakeholders can be given status reports on the development process anytime. FDD is a favorite method among development teams because it helps reduce confusion and rework in terms of the development world.

Typically used in large-scale development projects, five basic activities exist during FDD:

**Develop overall model** - The overall model is created by identifying the scope and context of the system. This is done by the product manager or chief architect of the team.

**Make a list of Features** - Developers list down potential items that would be useful to users and could be completed along a set timeline for release. Each feature should be manageable within a timeframe

**Feature wise Planning** - Features are organized by how long they take to create and how important they are to the client. Responsibility of each feature is also assigned to different members of the team.

**Feature wise Designing** - The actual details of each feature are produced, inspected and finalized.

**Build by feature** - After the design is improved, the completed feature is added to the official build for delivery to the client.