Emerging Programming Paradigm

ASSIGNMENT WORK FOR ASSESSMENT

Submitted to: Kanika Mam

Submitted by: Akhil Hooda

2017UCO1645

Question-> Write short notes on following->

1. **Scrum:**

Scrum works by breaking large products and services into small pieces that can be completed or released by a cross-functional team in a short timeframe. It works well for long-term, complex projects that require stakeholder feedback. A **scrum team** contains 5 to 9 people with the roles of the product owner, the scrum master and the development team. The **product owner** is a key stakeholder or a key user. The Product Owner is ultimately accountable for managing the product backlog and accepting completed increments of work. The **Scrum Master** is responsible for making sure the team is as productive as possible. The ScrumMaster protects the team by doing anything possible to help the team perform at the highest level. This may include removing impediments, facilitating meetings, and helping the Product Owner groom the backlog.The **development team** is a cross functional group consisting of the people who deliver the product increment inside a Sprint. The **product backlog** is a prioritised features list containing every desired feature or change to the product. A **sprint** is a time-boxed period of one month or less during which specific work (product increment) is completed and made ready for review. meeting: At the start of each sprint, a **sprint planning** meeting is held, during which the product owner presents the top items on the product backlog to the team. The Scrum team selects the work they can complete during the coming sprint. That work is then moved from the product backlog to a **sprint backlog**. Each day during the sprint, a brief meeting called the **daily scrum** is conducted which aims at keeping the team on track. At the end of each sprint, the team demonstrates the completed functionality at a **sprint review meeting**, during which, the team shows what they accomplished during the sprint. The **Retrospective** is the final team meeting in the Sprint which is attended by the team, product owner and the ScrumMaster to determine what went well, what didn't go well, and how the team can improve in the next Sprint.

2. Lean Development:

The Lean approach or The Minimum Viable Product (MVP) strategy is one in which a team releases a bare-minimum version of its product to the market, learns from users what they like, don’t like and want to be added, and then iterates based on this feedback. The practice is based on seven principles:

1.Eliminate waste. Waste is defined as anything that is capable of reducing code quality, hindering time and effort, or reducing delivered business value.
2.Amplify learning and create knowledge. Learning happens by implementing short iteration cycles and continuously gathering feedback to adjust the deliverables as the needs of the users become clearer.
3.Late Decision Making The last moment is defined as the moment you’ve learned enough about a decision to act on it.By waiting until the last minute to make decisions, the cost of change remains much less.
4.Build quality in. Integrity is built into the software to ensure architecture and that system components flow well together. Creating quality at the coding level so that instead of tracking and looking for defects, they can be prevented from the start.
5.Respect people. Respect in this context is defined as giving team members a voice and valuing their opinions. Respect extends to communication, conflict resolution, and encouraging healthy and productive discussions about business decisions.

6.Deliver fast. It starts by identifying the things that are slowing down the team and eliminating them. Delivering fast doesn't mean overworking until burnout to hit deadlines. it's about creating the most functional versions of solutions and then improving it over time using customer feedback.
7.View Applications as a Whole. Finding the components of the process that are dependent on one another and optimising all of them, instead of just one part of it.

3. Extreme programming (XP):

XP is a set of engineering practices. Developers have to go beyond their capabilities while performing these practices.XP has simple rules that are based on 5 values which are as following:communication, simplicity, feedback, courage, and respect.

- **Communication:** Teams work together on every part of the project, from gathering requirements to implementing code, and participate in daily standup meetings to keep all team members updated. Any concerns or problems are addressed immediately.XP stresses the importance of the appropriate kind of communication – face to face discussion with the aid of a white board or other drawing mechanism.
- **Simplicity:** It means keeping the design of the system as simple as possible so that it is easier to maintain, support, and revise. Avoid any waste and address only the requirements that you know about; don't try to predict the future.
- **Feedback:** The team should demonstrate their software early and often so they can gather feedback from the customer and make the necessary changes. Through constant feedback teams can identify areas for improvement and revise their practices.
- **Courage:** Kent Beck defined courage as "effective action in the face of fear". You need courage to raise organisational 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:** Each person on the team, regardless of hierarchy, is respected for their contributions. Every member of your team need to respect each other in order to communicate with each other.The team respects the opinions of the customers.

With this foundation Extreme Programmers are able to courageously respond to changing requirements and technology.XP follows some basic rules for each iteration which are:

1. **Planning**: On the basis of the User Stories provided by the customer feedback team creates a release schedule and divides the project into iterations.
2. **Managing**: Give the team a dedicated open work space and hold a stand up meeting each day. Everyone needs to work collaboratively and effectively communicate to avoid any slipups. Measuring project velocity and reacheduling work to avoid bottleneck. And in case Xp don't work properly change the rules.
3. **Designing**: Start with the simplest design and refactor often to keep your code clean and concise and don't add functionality early.Create spike solutions to reduce risk.
4. **Coding**: XP practices collective code ownership: Everyone reviews code and any developer can add functionality, fix bugs, or refactor.
5. **Testing** : The team performs unit tests and fixes bugs before the code can be released. They also run acceptance tests frequently.

4. Adaptive Software Development (ASD):

It grew out of rapid application development.It aims to enable teams to quickly and effectively adapt to changing requirements or market needs by evolving their products with lightweight planning and continuous learning. There is **no pre planned steps** in this process. The ASD approach encourages teams to develop according to a three-phase process: speculate, collaborate, learn.

1. **Speculate**: During this phase, coders attempt to understand the exact nature of the software and the requirements of the users. This phase relies on bug and user reports to guide the project. Speculate encourages exploration and experimentation. Iterations with short cycles are encouraged.
2. **Collaborate**: Collaborate would require the ability to work jointly to produce results, share knowledge or make decisions.Effective collaboration with customer is very important. Communication, teamwork, individual creativity is part of effective collaboration. More concern about collaboration and dealing with concurrency than about the details of designing, testing, and coding. Here the team members or sub teams are working concurrently to deliver the working components.
3. **Learn**: This phase consists of Quality review and Final Q/A release. During the learning phase, the newest version of the software is released to users. This generates the bug and user reports used during the first phase of the project, and the cycle repeats itself. During this phase components are implemented and tested.

This lead to an iterative approach.Sees the project continuously cycling from speculation to collaboration.Since it is focused on the end users, it can lead to better and more intuitive products an also increasing the transparency between the clients and the developers.

5. Feature Driven Development (FDD):

It is one of the agile processes that is not talked or written about very much. It is primarily used in those cases where we need to apply agile to larger projects and teams. FDD in Agile encourages status reporting at all levels, which helps to track progress and results. It uses a customer-centric, iterative, and incremental approach to deliver software. The general objective of FDD is to deliver concrete and flexible software in a short time. Its greatest advantage is that the process is scalable even for large teams.It helps reduce confusion and rework as FDD provides the team members with an opportunity to communicate more easily and provides the development team with the ability to break the entire problem into many smaller issues that they can cope with.It is designed to follow a five-step development process, built largely around discrete “feature” projects. These are :

1. **Developing an Overall Model** : The development team members cooperate together to propose a model for the domain area. After looking into the very teams model, one or a merge of models is chosen and it becomes the model created for that domain area.
2. **Building a Feature List** : After the development team built an object model features based on the user or client values are identified .These features are meant to be the building barriers of the project that help the group members to navigate the processes.
3. **Planning by the Feature** : The third stage is about managing and implementing the features.It’s essential to consider the team workload, risks, as well as other important aspects in order to prevent any kind of complex issues from arising.
4. **Designing by the Feature** : The chief programmer determines which features will be designed and built in a two-week iteration. This person also defines the feature priorities and determines who will be involved on the feature team. A design review needs to be completed by the whole team before moving forward.
5. **Building by the Feature** : The last step is to put all the necessary items into action in order to support the design. In other words, once your team developed, tested and inspected the code, it is time to start developing the software.