EPP Assignment

2017UCO1695 Mohammad Saif 24/07/2020

# **Scrum:**

It is defined as a software development methodology and is type of agile development management strategy. It is designed for small teams generally. It breaks the overall development into smaller cycles or iterations known as sprints. The sprints last for approximately two weeks or a month at most. The team generally holds 15-minute meetings everyday where they discuss what they did the previous day and what do they plan on doing that day. When a sprint ends, the team holds a sprint review meeting where they discuss about what they had achieved in that sprint, and also showcase demos about the work done so far.

Scrum is generally ideal for complex tasks where the major principle behind it being that the customer may change their requirements at any time. So instead of taking all of the requirements at once and proceeding forward, they do it incremental cycles instead, where each time they take a portion of the requirements and achieve it. This allows the customers to also be kept in loop and change the requirements if necessary. This way, the customers get the feeling that there is work being done on the project and know what stage it is currently on.

# **Lean Development:**

It is the translation of Lean manufacturing principles into software development. It is summarized by 7 principles.

1. Eliminate Wastage:  
   Anything that does not add value to what the customer wants is regarded as waste by lean development. This includes things like: building of the wrong feature, reworking things, unnecessarily complex solutions, extra features, waiting, multitasking, etc. One should be able to recognize the waste and remove it. This waste removal is done incrementally alongside the project until only the necessary features remain.
2. Amplification of Learning:  
   Software Development is about learning from what went wrong and finding solutions to the given problems. So, in order to speed up that learning, we must test as fast as possible. Hence, we run tests and integration test as soon as we are done with the code for that iteration. We also provide screens to the end user to take their inputs. We accumulate the errors and see what went wrong and attempt to fix it in that iteration.
3. Decide as late as possible:  
   Instead of allowing potential solutions become a headache, we delay such decisions for later, until it cannot be delayed any later. This allows us to get rid of as much assumptions or hypothesis as possible. This allows the customer to understand their requirements better, and then they can give a better requirement description, thus allowing us to get rid of more potential solutions.
4. Deliver as fast as possible:  
   In the current world, the faster wins, not the bigger. We need to deliver products as fast as possible, while also maintaining as much quality as possible. As quality matters a lot to the customers, but if the product takes too long, the customer also loses money, and will eventually get tired of waiting. And lean development allows for fast deliveries alongside maintaining quality.
5. Team empowerment:  
   The team manager must listen to their team and use their expertise to manage the project. This allows them to be able suggest improvements and can explain better why an action was chosen. They must absolutely not micro manage the team as it just lowers the morale. The people must not be treated as resources, but instead be motivated and appreciated for the work they put in.
6. Build integrity in:  
   The customer needs to also go through the experience of the system while its being developed. This is also known as integrity. The information between the developers and the customer needs to flow in both directions in chunks and not all at once. The method to achieving this, is to test as much as possible, the code must be as simple as possible, repetitions in code to be avoided as much as possible, thoroughly integrated tested, and check for the quality standards, and cross checked to make sure that it does exactly what the customer needs.
7. Optimization of the whole:  
   The system needs each of its parts to work with each other. This also includes their interactions with each other. Thus, during the development, the number of errors accumulate, and because we develop in iterations, we can find the root cause of the problem. Thus, when we are able to successfully optimize all the parts, we can then easily optimize their interactions, and thus optimize the whole system.

# **Extreme Programming:**

It aims at producing a high-quality product while also maintaining a high quality of life for the developers. It consists of 5 values:

1. Communication:  
   This is a very integral part of software development as developers rely on good communication to transfer knowledge about the problem and requirements with one another. A good communication means the problem can be easily understood, resulting in a better planning and also better quality of products.
2. Simplicity:  
   We would like to keep the wastage to a minimum. In order to do that, we need to keep the system design as simple as possible. If we are able to do that, we can easily revise, update, and debug our code without producing much wastage. This also means to only address requirements that we know about and not try to add new features by predicting the future.
3. Feedback:  
   This means that every time something is pushed out, we get a feedback from the customers, and we are able to improve the product further, by acting upon the feedback. This allows the quality of the product to be improved per iteration, and allows the product development to also be dynamic, in case the customer decides to add or remove some requirements.
4. Courage:  
   One needs to show courage to speak up. To speak up about actions or decisions that could be detrimental for the team. If achieved, the team’s productivity can be increased. Instead of trying to make something bad work, one could come up with a new idea, and make things easier, faster, and make the product better. One needs to accept feedback, as in the end, the customer is the one who will be using the product and paying for them.
5. Respect:  
   Every member in the team, must respect each other in order to communicate effectively, and keep the productivity of the team at its highest. They must understand and accept the feedback from their fellow members and work together to achieve the best possible result.

# **Adaptive Software Development:**

It is cyclic like the evolutionary model, but is meant to accommodate change and is adaptable to change in a turbulent environment for products and evolves from a little amount of planning and learning. It focuses mainly on results and not tasks, and the results are the features of the product. It consists of 3 phases:

1. Speculate:  
   Planning is replaced by Speculate, because the word planning is too deterministic. It sounds like we know about the future, and know the shortcomings. But in reality, we don’t. So, we speculate about the future, take note of as many unaccounted situations as possible, and try to make a plan. Thus, this allows for short cycles to be the choice of development while we experiment and explore the given problem.
2. Collaborate:  
   A complex application cannot be built at once; it evolves over time by analyzing large amounts of data and applied to the problem. This means that a diverse amount of knowledge is required. And that’s where collaboration comes into the picture. It requires the ability to produce results jointly, share results and make decisions.
3. Learn:  
   It is all about learning from the reviews, feedbacks, mistakes, and applying it to the product in the cycles to further refine it into the product the customer wants. Teams can use technical reviews, project retrospectives, and customer focus groups. This allows the team to understand the requirements better. And if the cycles are short, the mistakes can remain small.

# **Feature Driven Development:**

It is an agile method for software development that is lightweight and is iterative in nature. It consists of 5 stages, out of which the last three stages define the iterative part of the development method.

1. Develop overall model:  
   This part consists of walking through the scope of the system at a high level. Then detailed domain models are presented by small groups for peer reviews. After choosing one model per domain, the chosen models are then merged into a single overall model.
2. Build feature list:  
   The knowledge that was gathered during the overall model development is used to make a list of features by functionally decomposing each domain into subject areas, which consist of features. Features are small in size, and generally should not take too long to complete, or else they will be broken down into smaller individual features.
3. Plan by feature:  
   Now each feature or feature set is planned down in the form of a development plan and assigned to programmers in the form of classes.
4. Design by feature:  
   Design packages are designed for features. A team leader will select a set of features that are to be developed in a small period of time. Now alongside the programmers, the leader will detail out each and every feature and refine it further. After the classes are written, a design inspection.
5. Build by feature:  
   After a successful design inspection, the class owners develop codes for their classes, which are then integrated into the main build after they pass all the unit and integration tests.

Feature completion progress is monitored in the form of milestones. Each feature consists of 6 milestones that are to be completed sequentially. The first 3 are done in the Design by Feature part, and the last 3 are done in the Build by Feature part.