Lecture Notes – Software Development Lifecycle

Agile and Extreme Programming

Introduction to Agile

Let's revisit the topics learnt in the introduction to Agile segment.
In this segment, you learnt, in early 2000's how fast-paced and changing environment of software development led developers to come up with suggestions such as Agile methodology. Agile methodology has 4 foundation values as well as 12 principles around those values. The foundation values are:

1. Individuals and interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Customer collaboration over contract negotiation.
4. Responding to change over following a plan.

Agile Manifesto: Principles

You learnt the 12 principles of Agile are:

1. Customer satisfaction — Customer satisfaction is a need in Agile and it is implemented through releasing the software often with frequent feedback taken from the customer.
2. Welcome change — In Agile, we welcome change at every iteration.
3. Working software — We release the working software at each iteration.
4. Collaboration — We encourage frequent discussion with business team.
5. Motivation — We encourage motivation in the team.
6. Face-to-face — We encourage face-to-face interaction with among team.
7. Measure the progress through working software — We measure progress of the development through no of functionalities the last release has given.
8. Constant pace — In Agile constant pace of development is promoted which leaves the chance of last minute burnout.
9. Continuous attention — We encourage continuous attention.
10. Simplicity — Simplicity in code is preferred in Agile because it leaves minimal complications.
11. Self-organized teams — Team should be encouraged to have self-organized and can take necessary steps to meet the responsibilities and deadlines.

Review the work regularly — Reviewing the work regularly gives opportunity to improve in every iteration.

What Is Extreme Programming?

You learned in this segment, Extreme programming (XP) is a software development methodology that focuses on changing user needs, along with the development of software, while maintaining efficiency within teams. Extreme Programming advocated to start testing first and then write the bare minimum code to meet the required functionality without any bug. You learned how old preferred process stretched to practices of 'extreme' programming.
It is an implementation of Agile methodology just like SCRUM.

Values and Practices of Extreme Programming

In this segment, we explored the value and practices of extreme programming:
Values of extreme programming:

1. Communication — It values effective communication which enhances productivity.
2. Simplicity — Bare minimum simple code is written completing the required functionalities.
3. Feedback — At every iteration feedback is given so that suggested improvements don't pile up.
4. Courage — Focus only on what matters, accepting feedback, telling the truth about progress, gives a programmer courage.
5. Respect — Extreme programming teaches respecting other team members so that you remain conscious and consider saving others from reworking.

Practices of extreme programming:

1. User stories — Requirement are broken down into short and crisp user stories.
2. Planning game — User stories are then converted into tasks.
3. Short releases — Short releases are planned so that we get constant feedback after each iteration.
4. Metaphor — Software architectures are discussed in simple terms with help of metaphors.
5. Simple design — Design and development are kept simple with writing just enough code.
6. Testing — Along with development constant testing is done leaving the chance of post-production bugs.
7. Refactoring — Constant refactoring is in order to simplify the code.
8. Pair programming — A quality engineer and software engineer or a two software engineer sit together constantly reviewing and writing the code.
9. Collective ownership — Entire team is considered to be accountable for the delivery.
10. Continuous integration — Code is committed to the central database every few hours so that development efforts don't remain fragmented.

11. 40-hour week — Team should spend 40-hour a week on the project to ensure the good productivity.
12. On-site customer — Customer should sit with the developer teams for faster feedback and resolution cycle.
13. Coding standard — Team should maintain a good coding standard.

Advantages of Extreme Programming

Advantages of extreme programming, you learnt, are:

1. No slipped schedules
2. No cancelled projects
3. No cost incurred due to changes
4. No production and post-delivery defects
5. No misunderstanding of the business requirements
6. No change aversion
7. No staff turnover

User Stories

A user story is short and crisp description of requirements in the format of —
“As a <type of user>, I want <some goal> so that <some reason>”
E.g. As a user, I should be able to print all the listed documents with their details. Before printing them I should be able to see the print preview in the browser only so that I can decide the format in which I want to print the document.
While anyone can write user stories, primarily it’s the job of product owner. Later on theses user stories are prioritized for creating the order of development.

Comprehensive User Stories and Backlog Documents

Goal of a user story is to understand the requirement correctly and, engineering team should be able to estimate the time required to develop it. Sometimes user stories, called EPICS, needs to be broken into smaller user stories. Then conditions of satisfaction is added as a criteria of completion for the user stories. Conditions of satisfaction should conform to INVEST criteria which is a short form of independent, negotiable, valuable, estimable, small and testable. A backlog document is the list of user stories created from breaking requirements. It keeps track of the development through various parameter. A grooming session is kept periodically to add, delete or edit the user stories from backlog document.

Estimation and Planning

There are various estimation techniques used to quantify the efforts required to develop the user story. Some of them are:

1. Story points
2. T-shirt sizing
3. Largest estimate
4. Relative mass

And planning techniques are the process through which we come to a estimation. Various planning techniques are:

1. Planning poker
2. Affinity planning
3. Bucket system
4. Big/uncertain/small
5. Dot voting

Planning Poker

Planning poker technique is used to determine the estimation that will take to develop the user story. In this technique, each participant chooses to give an estimation, typically from the Fibonacci series. The largest and the smallest estimators explain the reasons for an estimation. Discussion and reconsideration is carried out. After few round either team comes to an agreement or average of all the estimations is taken as the final one. Planning poker ends when all the user stories are discussed.