SOFTWARE ENGINEERING DAILY CHALLENGE 12/10/2019

1. **Explain the following agile methodologies Scrum, Kanban, Extreme Programming.**

    Ans

Firstly, “Agile” refers to “agile software development,” the approach to development that follows agile principles. But what the heck are “agile principles?” Take a look at the Agile Manifesto and at the 12 principles of agile, which lay the foundations of agile development. From the Manifesto:

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

Agile principles encourage continuous delivery of functioning software, close communication among teams, and high adaptability to changing needs. If you follow these values and principles in your work, you can say that you are working in an agile environment. So, agile software development is not a methodology, it is just a set of different methodologies, frameworks, and techniques that follow the same principles. It can be said that “agile” is a framework for thinking and making decisions.

What is scrum? scrum is:

"A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value."

So it's a *framework*, and like any other framework it can be, and regularly is, used the wrong way. Using scrum effectively requires not merely adopting the structure set out by scrum, but having a deep understanding and appreciation for agile principles across the entire team.

Scrum consists of three roles: **Product Owner**, **Scrum Master**, and **Development Team**; four ceremonies: **Planning Meeting**, **Daily Scrum**, **Sprint Review**, and **Sprint Retrospective**; and three artifacts: **Product Backlog**, **Sprint Backlog**, and **Product Increment**. It is organized into regular time frames, which we call **sprints**. Sprints should last between one and four weeks.

The **Product Owner**, or PO, is responsible for guiding the project's direction. As new tasks and features are determined, the PO adds them to the **Product Backlog**. A **sprint** starts with a **Planning Meeting** where the **Development Team** selects the tasks from the Product Backlog to work on and plans how they will be implemented. That is followed by development, during which the Dev Team uses the **Sprint Backlog** to track progress and meets for the **Daily Scrum** in order to synchronize activities and adjust the plan, if needed. The result of development should be a **Product Increment**, something that can be applied to the product and released immediately. At the end of the sprint, the Product Increment is presented to the Product Owner at the **Sprint Review**, where the product backlog is augmented if further changes are needed.

Afterwards, the whole team attends the **Sprint Retrospective** (also known as the Pub Meeting) where they talk about the work process and how it can be improved.

KABAN

Kanban is a simple method that aims for just-in-time delivery while not overloading the team members. It is similar to scrum in that the goal is to deliver maximum value at the end, but it is much more flexible than scrum.

Kanban was not invented by the software development community. In fact, it has its origins in manufacturing processes at Toyota, and it has wide usage in other spheres. There are no strict procedures that you should follow, and no strict way you should implement and use kanban; it is, rather, a set of principles and practices, and you can choose from these practices to suit your needs. But there is one most-often used implementation of kanban in software development that includes the usage of a **kanban board**, consisting of columns representing stages of work, and tasks.

Columns represent the state of a task in the development process. The simplest example consists of three columns: “To Do,” “In Progress,” and “Done.” So, tasks are added to “To Do,” moved to “In Progress” when development starts, and considered “Done” when moved to the last column. But of course, it could be more complex:

Backlog → Defining Specification → Ready for Development → Development → Code Review → Testing → Deployed (→ No one really uses it → Completely Removed)

The Agile Process: Extreme Programming

Extreme Programming (XP) is a highly disciplined management method, which focuses on continually improving quality and speed of software delivery. The development team works closely with customers, continuously planning, testing and providing feedback to developers, to quickly deliver valuable software. Like Scrum, delivery is in iterations of 1-3 weeks.

The Extreme Programming method follows these lifecycle stages:

- **Planning** – setting goals for the entire project and specific iterative cycles. Planning is done together with the customer, who formulates a vision of the product and defines user stories. Developers estimate the stories and turn them into more granular tasks.
- **Designing** – developers are responsible for designing the main features in the next iteration. XP emphasizes simplicity, so design should be as simple as possible.
- **Coding** – throughout an iteration, developers constantly refactor the code to bring it down to the simplest, most elegant form possible. Pair programming is commonly used in XP projects to boost innovation and code quality.

- **Testing** – testing is done in tandem with writing the code, not afterwards as a separate development stage, typically using Test Driven Development (TDD).

2. Who are the members of an agile team and what are their roles?

a. Product Manager aka Product Owner
b. Program Manager
c. Architect
d. Engineering Manager
e. Product Developer
f. QA
g. UED
h. Operations

Product Manager aka Product Owner

Role definition

- "CEO" of the product
- Vertical – focus on long and short term product vision of a product line
- Represents customer's interest
- Represents the product to the outside world (Customer)

Responsibilities	*Challenges*
● Responsible for market, business case, and competitive analysis ● Responsible for long and short term product vision ● Responsible for ROI and Net Profit ● Prioritizes features for releases based upon expected ROI ● Writes Acceptance Criteria ● Writes user stories ● Makes tradeoff decisions between scope(value in Expected ROI) and schedule(higher operating expense in longer release cycles)	1. Resisting the temptation to “manage” the team. The team may not self-organize in the way you would expect it to. This is especially challenging if some team members request your intervention with issues the team should sort out for itself. 2. Resisting the temptation to add more work 3. Being willing to make hard choices during a planning meeting. 4. Balancing the interests of competing stakeholders.

Program Manager

Program Managers can perform both Project & Program Manager roles. On a Scrum team, Program Managers usually perform both Program Manager & Scrum Master roles unless another team member is assigned to be Scrum Master.

Role definition

- Horizontal (Across the product lines, projects, x-functional teams)
- A rational voice to drive the cross-functional core team to solve problems and make decisions
- Impartial – not biased to Product, nor to Engineering, nor any other group

Responsibilities

Program Management responsibilities

- Manages planning process
- Manages overall program schedule
- Drives multiple releases/projects
- Facilitates Release Planning & Retrospective
- Provides access to tools and people
- Owns all action items for the project until he/she finds the right owner
- Owns reporting on project status, to all directions
- Coordinates other release support
- Responsible for risk assessment & mitigation
- The Role is a peer to the Product Manager and the Engineering Manager on the release/project
- Educates/Enforces agreed upon

Scrum Master responsibilities

- Manages 1 sprint at a time
- Facilitates Sprint Planning, Review & Retrospective
- Finds and works to remove roadblocks
- Helps to motivate the team and keep them excited
- Protects team from outside distractions
- Facilitates communication between roles for every aspect of the project
- Responsible for keeping release/project information consolidated, organized and up to date
- Drives the cross-functional team at all levels
- Responsible for throughput (team velocity)
- Drives the execution of sprint items

processes & methodology rules
- Educates/Enforces roles and responsibilities

Architect

Role definition

- Leads the technical direction of overall system

Responsibilities

- Responsible for end-to-end cross functional system design and communication
- Works with the PM to group features based upon the Architectural Elements which support them, an influence on priorities
- Tests Architectural Elements with executable and testable design (abstract interfaces, aka the contract)
- Facilitates technical decision; incorporates feedback and emergent patterns from the team back in to the overall design
- Produces alternate Design Concepts & detailed approach
- Ensures the Design goals – Performance, Modularity, Reliability, Maintainability, Reusability, Internationalization and Accessibility – are met
- Ensures technical cohesion and helps write the technical contract in interfaces and other abstract objects and data entities

- Leads design review & provides feedback

Engineering Manager

Role definition

- Ensures the successful completion of Work-in-progress (WIP)
- Understands the process of product creation

Responsibilities

- Responsible for rate of production and lead time (what each engineer needs to fulfill their commitment)
- Responsible for initial high level sizing
- Works with the architect and the team to prove technical integrity
- Responsible for conducting forward leaning technology investigations (spikes)
- Negotiating with architect on technical approaches
- Removes most bottlenecks
- Enforces engineering best practices
- Ensures motivation of their team
- Assists career development of team members
- Conducts focals

Product Developer

Role definition

- Responsible for creation of product

Responsibilities

- Estimates size of backlog items
- Translation of backlog items into engineering design and logical units of work (tasks)
- Evaluation of technical feasibility
- Implementation of backlog items
- Writes unit tests first to the contract of the interface and other abstract classes
- Writes and verifies code which adheres to the acceptance criteria
- Application of product development best practices

QA

Role definition

- Prevents defects from entering the system, as opposed to finding them at the end
- Facilitates building integrity into the software product and development process

Responsibilities

- Writes test plans which enforce the acceptance criteria of features
- Keeps all test plans and cases updated to changing requirements
- Continually integrates the code base with automated builds and functional-level regression tests
- Notifies when production is blocked due to errors in development
- Measuring Quality
- Defining Quality
- Improving Quality
- Enforces QA Best Practices
- Provides visibility into end-to-end product quality

- Owns QA Health status of releases/projects in SM dashboard

UED

Role definition

- Integrating the human factor in to features

Responsibilities

- Responsible for overall UI consistency across interaction models and visual styles
- Upfront user research (to help inform features/direction)
- Interaction Design specs (wireframes, user flows) which conform to acceptance criteria of the features
- Works with the PM on analysis of features conform to user's needs and provide a consistent experience
- Ensures features are implemented as designed, and design modified to reflect implemented features
- Visual Design
- Prototyping
- Usability testing

Operations

Role definition

- Keep products running in production

Responsibilities

- Design hardware configurations for BCP, security, monitoring, availability during product design stage
- Responsible for getting the required hardware using the “HRC” process
- Set up staging, production environments (and in some cases the QA environment)
- Post production monitoring, and troubleshooting of product/services
- Owns OPS Health Status of releases/projects in SM dashboard

3. List 5 Project Management Tools

Ans

a. Trello
b. Ganttpro
c. Jira Software
d. Basecamp
e. Wrike

4. List the Phases of Software Development Life Cycle.

Ans

There are six phases of software development life cycle

a. planning
b . analysis
c. Design
d. Development & Implementation
e. Testing
f. Maintenance