Agile Software Development

Dr. Mohammad Moshirpour

Fall 2021

Outline

- Why Agile?
- Agile and Scrum Definition
- Scrum Roles
- Scrum Practices
- Project Breakdown and sizing
- Team Structure
- Summary

Why Agile?

- In the old days software project management was done using waterfall approach
- Development teams were disjoint, and communication was sequential
- The end-product was very different from what the customer was initially described
- Agile promises :
  - Iterative and faster software delivery
  - Iterative feedback from customer
  - Faster response to change
  - Better communication and collaboration between team members
  - More customer satisfaction

Agile and Scrum Definition

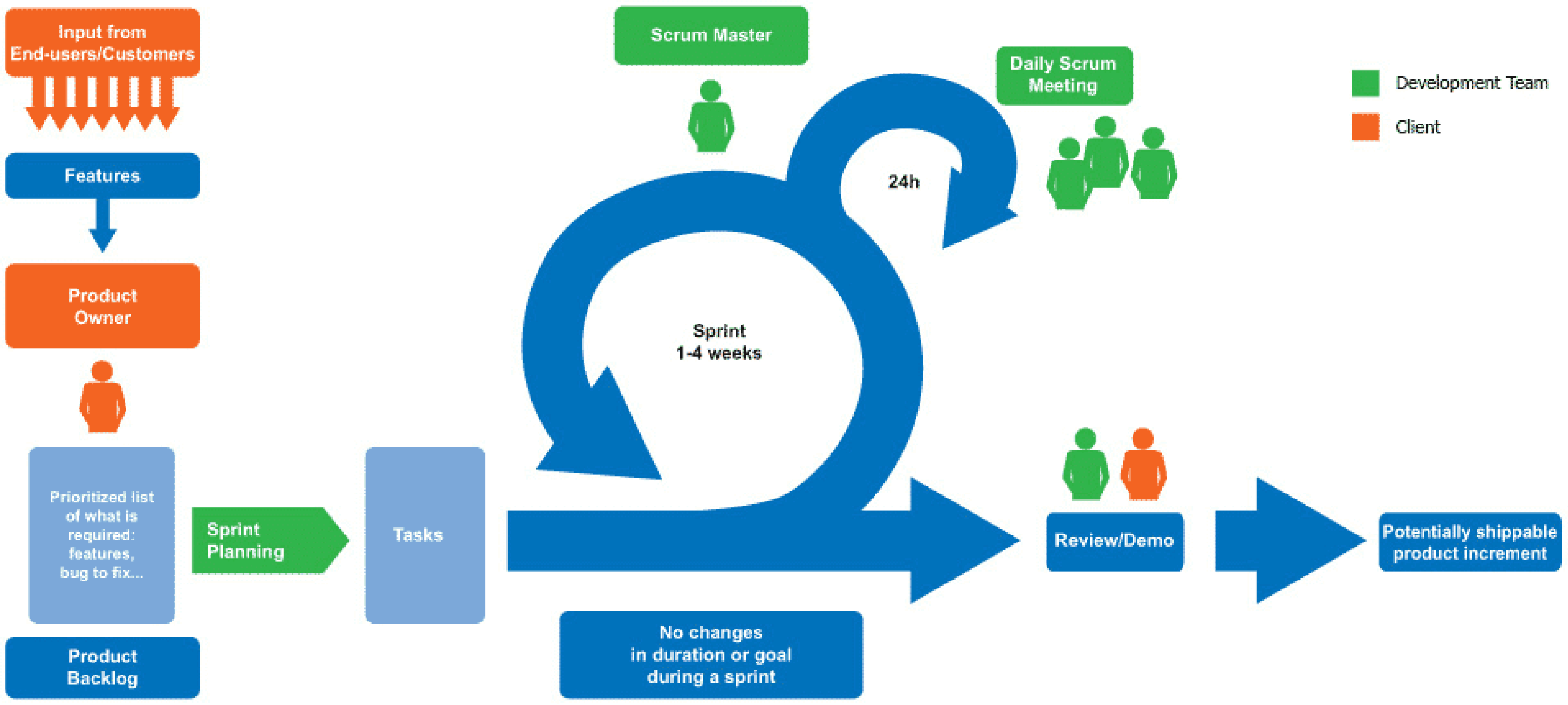
- Agile manifesto describes the basis of Agile software development
  https://agilemanifesto.org/

  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

- Scrum is one the frameworks that implements Agile ideas
  https://www.scrum.org/resources/what-is-scrum

In this presentation you will learn how scrum works

Scrum Overview


Input from End-users/Customers
Features
Product Owner
Prioritized list of what is required: features, bug to fix...
Product Backlog
Sprint Planning
Tasks
Scrum Master
Daily Scrum Meeting
24h
Sprint 1-4 weeks
No changes in duration or goal during a sprint
Review/Demo
Potentially shippable product increment
🟩 Development Team
🟧 Client

Outline

- Why Agile?
- Agile and Scrum Definition
- Scrum Roles
  - Product Owner
  - Scrum Master
  - Development Team
- Scrum Practices
- Project Breakdown and sizing
- Team Structure
- Summary

Scrum Roles

- Stakeholders
  - External Customers: Those who will eventually use this product
  - Internal Customers: Colleagues who collaborate with the team and depend on the team’s output
- Product Owner
  - Communicate with stakeholders and brings the items needed for a product to the development team
  - Communicate with development team and prioritizes the development items

Scrum Roles (Cont'd)

- Scrum Master
  - Facilitate the scrum meetings
  - Ensures that the development team understand the purpose of the meeting and the goal of the sprint

- Development Team
  - Not just software developers
  - Includes designers, testers and developers
  - Takes the ownership of software development based on the prioritized items
  - Attends scrum meetings
  - Communicates with the scrum master and the product owner

Outline

Scrum Practices

- Scrum runs in short iterations called sprint to deliver shippable product
- Each sprint consists of 5 ceremonies
  - It starts with **sprint planning**
  - It has **daily scrum** everyday of the sprint
  - It ends with **sprint review** and **retrospective**
  - If needed, a **refinement** during the sprint
- A sprint can be between a week or a month

Scrum Practices – Sprint Planning

- **Purpose**: The purpose of sprint planning is to plan what should be delivered in each sprint and how .
- **Time**: It is held on the first day of the sprint
- **Attendees**: The meeting is held with the presence of Product Owner, Scrum Master, and the entire development team
- **Responsibilities:**
  - **Product Owner:** Brings the prioritized work of the *product backlog* and clarifies the sprint goal
    - *We will talk about product backlog soon*
  - **Scrum Master:** Ensures that the discussion is effective, and everyone agreed on the sprint goal and understand the purpose of the sprint
  - **Team members:** Determine which product backlogs they will be able to complete during the sprint and how they will achieve it.

Scrum Practices – Daily Scrum

- **Purpose**: Team members inform each other of what is going on across the team. It is held to improve collaboration, tracking progress, removing impediments and planning the product backlog
- **Time**: Daily scrum is a quick meeting, usually five to fifteen minutes. It is held on everyday of a sprint.
- **Attendees**: Development team
- **Responsibilities:**
  - All team members should answer the following questions:
    - What have you done since the last meeting?
    - What will you do?
    - Did you face any difficulties? Why?
- Note that the team does not need to go through the details in this meeting

Scrum Practices – Sprint Review

- **Purpose**: To assess the works completed during the sprint against the sprint goal defined in sprint planning
- **Time**: At the end of the sprint
- **Attendees**: Product Owner, Scrum Master, Development team, stakeholders
- **Responsibilities**:
  - **Product Owner:** Invite the stakeholders to review the product and report the items in product backlog
  - **Stakeholders**: Give feedback to the team
  - **Scrum Master**: Capture the feedback
  - **Development Team**: Should give a demo of their completed work to attendees

Sprint Retrospective

- **Purpose:** The scrum team inspect itself and plan for improvement for the next sprint
- **Time:** After the sprint review and before the next sprint planning
- **Attendees:** The Development team, Scrum Master
- **Responsibilities :**
  - **Scrum Master :** facilitates the meeting
  - The **Development team** should answer three questions:
    - Three things that went well
    - Three things that need improvements
    - Does the previous issues that needed improvements addressed?

Sprint Practices - Refinement

- Refinement is very similar to the planning meeting
- Its purpose is to give more time to the development team for discussing and planning in more detail

Outline

- Why Agile?
- Agile and Scrum Definition
- Scrum Roles
- Scrum Practices
- Project Breakdown and Sizing
  - Backlog
  - User Story
  - Sizing User Stories
- Team Structure
- Summary

Project Breakdown - Backlog

- **Product backlog:**
  - product backlog is a list of prioritized works needed to be done for a product
  - Product owner is responsible for prioritizing works in the product backlog
  - Items in the product backlog can be broken into epics and stories.
- **Sprint Backlog:**
  - Contains the items that should be done during one sprint
  - The development team is responsible for identifying tasks that should be done during the sprint, pull the tasks from product backlog and break them down to user stories.

Project Breakdown – Epic and story

- **Epics** : Imagine epics as large stories which are needed to be broken down into smaller pieces of tasks
- **Story**
  - Story is the smallest piece of work
  - User story is a tool used to describe a software feature from end-user perspective
  - A user story clarifies three things:
    - What features to implement
    - The purpose of the feature being implemented
    - What value does the feature add to the product
  - A story is “Done” when the outlined tasks defined in the story is completed

User Story Example

- Imagine authentication feature to unlock phones as an example.
- The epic is as follows
  - As a user I want to authenticate myself to my phone to make sure that only I can access the phone.
- The stories are as follows
  - As a user, I want to use face recognition to authenticate so that I do not have to enter my username and password all the time.
  - As a user I want to use fingerprint authentication so that I do not have to enter my username and password all the time.
  - As a user I want to enter my username and password to enter the phone in case I can not use my face authentication or fingerprint.

Estimating the Difficulty of a Story

- Humans are bad at estimating time
- But they are good at estimating relative time
- We use a system called story points to give relative difficulty to story
- Story point is a measurable estimation of the difficulty of a task.
  - For example, a story can be easy, medium, hard, etc.
  - Or represent it as numbers: 3, 5, 8, etc.
- When sizing the stories you should put complexity, risk, effort, time needed to complete a story into consideration.

Sizing a Story

- Steps to size a story:
  - Find a story that is easy to estimate. Then, set it as the baseline
  - Compare all the other stories against the baseline story for pointing more or less
  - Discuss your pointing estimation with your teammates. Find the rational behind their estimation and reach a conclusion for pointing each story.
- We usually use the following scheme to size stories:

1:	trivial
3:	easy
5:	medium
8:	hard
spike:	unknown

Sizing Stories - Spike

- If we do not have enough information to give story a size, we put it in a time-box and research about it.
- These types of stories are called spike
- At the end of the time-box, we should have enough information to size the story

Outline

Team Structure

- Teams are organized for software delivery in two ways
  - Component Teams
  - Featured Teams

  https://www.agil8.com/blog/component-teams-vs-feature-teams/

- Each have their pros and cons and are useful in different situations
  - Duration of the project
  - Expertise of the team member
  - Dependency between stories

Component Team

- Contain several teams. Each team specialized in one or two domains and focuses on conducting one or two components of a project.
- Dependency between component teams exposes three problems:
  - **Long lead time:** Each teams waits for the previous phase output to adjust and complete its tasks.
  - **Difficult Recall and Fix:** One team might be done with a task and move to the next task, while another team wants to open the previous tasks and ask for some modifications.
  - **Inequality of the amount of work:** One team might finish with its work and have no other task in queue, while the other team starts with a large amount of work.

Feature Team

- All team members have multidisciplinary knowledge and are able to solve problems related to all layers of the system.
- Feature teams look at a multidisciplinary task which requires several skills as a user story. They break down the task and work on the most valuable items.
- The lead time in feature teams is manageable
- As the team work full stack, they are more confident for managing future software integration

Summary

- We learned about agile and scrum
- We learned how to manage software project using scrum framework
- We learned about different team structures