Software Testing

Software Development Life Cycle

Agenda

- SDLC Overview
- SDLC Phases
- SDLC Models
- Practice session – The Scrum Game

Software Development Life Cycle - Overview

- SDLC contains following phases: **Requirement analyses, Design, Implementation, Testing, Release/Maintenance**
- There are many different SDLC methodologies; choosing which to apply and how, depends on the goals and specifics of the project/company
- Effectiveness of the project/company largely depends on the effectiveness of the implemented SDLC process

Requirement analyses phase

- Everything starts from an **Idea**
  "It would be good to have these functionalities on our Web Site"

- Ideas come from:
  - Marketing or product management
  - From end users/customers to anyone inside the company
  - The competition

- Ideas (if accepted) are formalized as **Requirements**
  "These are the functionalities that we are going to implement on our Web Site"

- Documented requirements are also known as **Specification**.

Design phase

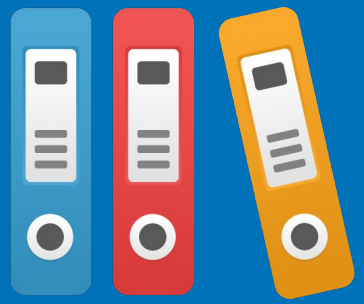
During the design phase, business and technical specification is created.

Types of Software Specification:

- **BRD (Business Requirements Document)** - high level document, providing list of requirements which are demanded by the client and should be part of the proposed system.
- **SRS (System Requirements Specification)** - describes entire system flows; contains both functional (often in the form of use cases) and non-functional requirements
- **FRS (Functional Requirements Specification)** - includes detailed information on how requirements are going to be implemented in the system

Examples for technical specification documents:

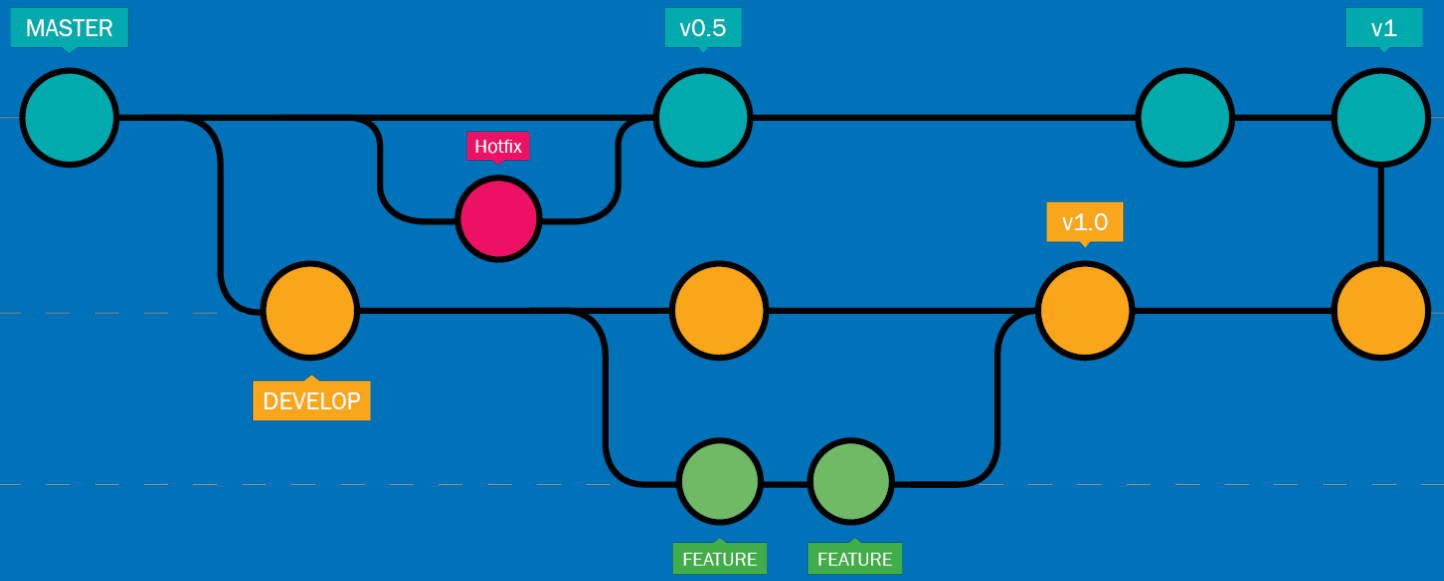
- HLD (High Level Design)
- DDD (Detailed Design)



Documentation formats, types and terminology may vary, depending on each company's workflow and processes.

Implementation phase

- Realization of the architecture & design documents (coding phase)
- Developers use special software (IDE - Integrated development environments) to write and maintain the code (Visual Studio, Eclipse, IntelliJ IDEA, etc.)
- Code changes are managed by Version control systems (Git, Mercurial, SVN, etc.)
- Decision on code branching strategy



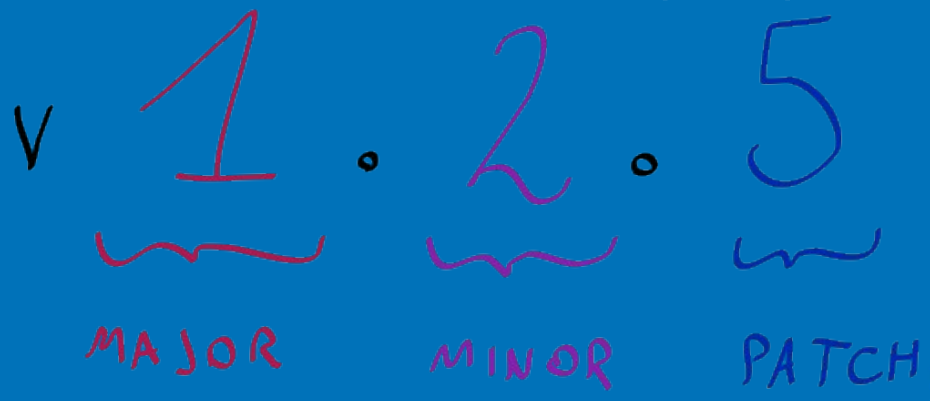
Testing phase

- When does the testing phase of a project start depends largely on the maturity of the team/company and the defined testing process
- Many companies implement “DevOps” approach, where testing is integral part of each activity on the project (from requirement to release)
- Following the “early testing” principle, testing can start as soon as there are requirements

- Once a code change is implemented and **new build** is provided, test engineer ensures:
  - Build scope (build contains required code change)
  - Correct build is deployed on a **testing environment**
  - Build is not broken (by performing a **smoke test**)
- After that test engineer performs detailed tests on the functionality
- Any bugs are reported, fixed and re-tested
- **Regression tests** are done to ensure new code changes are not ‘breaking’ old functionality
- Once release scope implemented and test completed, it is time for **Release** phase

Release & Maintenance phase

- Final build package provided to the clients/users is related to an official version (for example: v1.2.0)
- Releases can contain new features (**major releases**) and/or bug fixes (**maintenance releases**)
- Huge release can have **downtime**
- Testing activities continue throughout the release (deployment) and after it (production monitoring)
- If defects are found on Production environment after release, this may require urgent actions (**Rollbacks/Patches/Hot-fixes**)

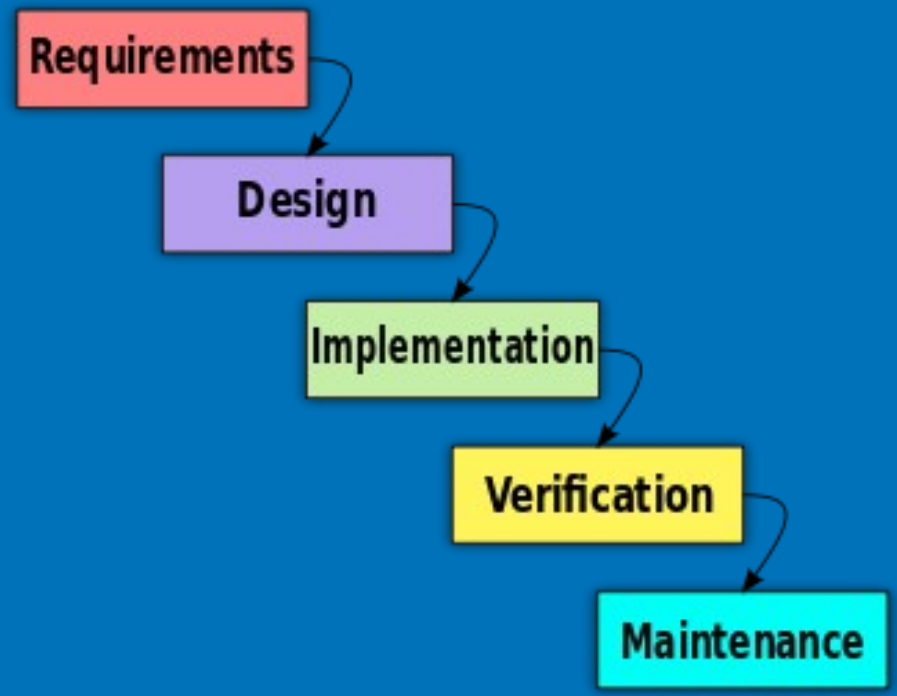


Release & Maintenance phase

- Often multiple versions of the software product/application need to be supported
- Test strategy should take version support into consideration (example: backwards compatibility)
- Bug fixes may need to be applied to and tested in more than one version
- If defects are found in production, it is important to update the test suite with relevant scenarios to avoid missing such bugs in the future

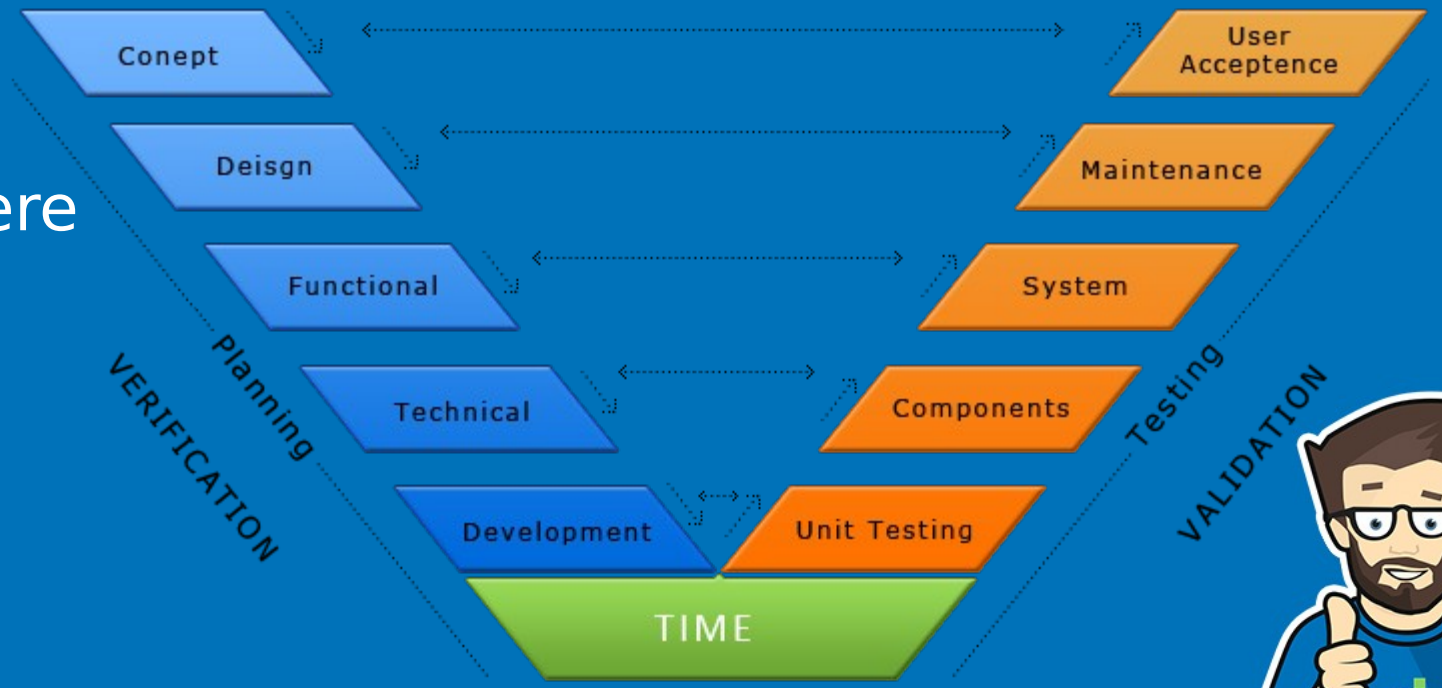
SDLC Models - Waterfall

- A stage starts after previous one is completed
- Requires big initial analysis and planning
- Simple to understand
- Time overrun
- Cost overrun
- Not adaptable to real needs



SDLC Models – V Model

- Also known as **Verification and Validation** model
- V-Model is an extension of the waterfall model - next phase starts only after completion of the previous phase
- For every single phase in the development cycle, there is a **directly associated testing phase**

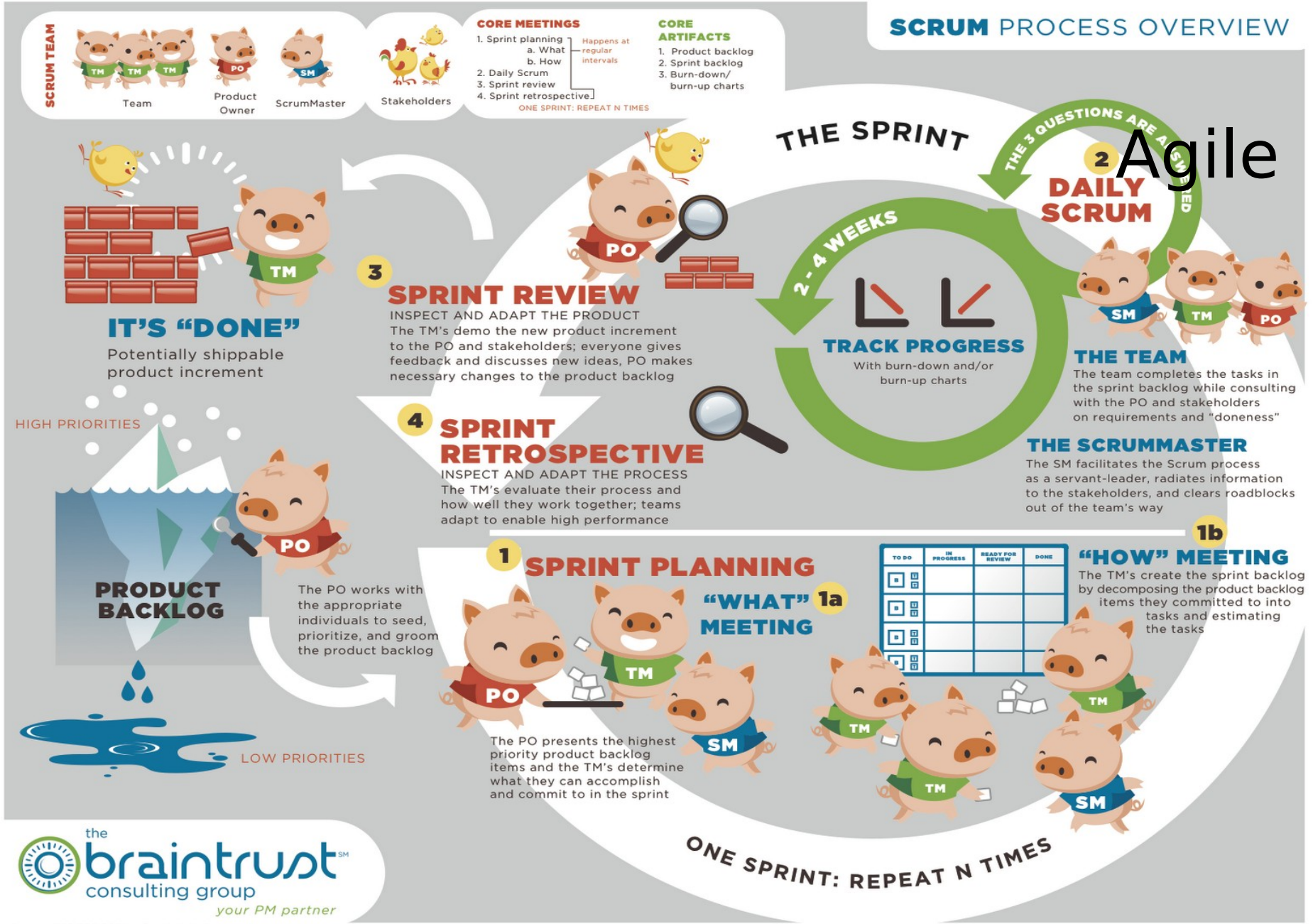


SDLC Modes - Agile

THE AGILE 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


SCRUM PROCESS OVERVIEW
SCRUM TEAM
Team
Product Owner
ScrumMaster
Stakeholders
CORE MEETINGS
1. Sprint planning
a. What
b. How
2. Daily Scrum
3. Sprint review
4. Sprint retrospective
Happens at regular intervals
ONE SPRINT: REPEAT N TIMES
CORE ARTIFACTS
1. Product backlog
2. Sprint backlog
3. Burn-down/ burn-up charts
THE SPRINT
IT'S "DONE"
Potentially shippable product increment
3 SPRINT REVIEW
INSPECT AND ADAPT THE PRODUCT
The TM's demo the new product increment to the PO and stakeholders; everyone gives feedback and discusses new ideas, PO makes necessary changes to the product backlog
4 SPRINT RETROSPECTIVE
INSPECT AND ADAPT THE PROCESS
The TM's evaluate their process and how well they work together; teams adapt to enable high performance
2 - 4 WEEKS
TRACK PROGRESS
With burn-down and/or burn-up charts
THE 3 QUESTIONS ARE ASKED
2 DAILY SCRUM
THE TEAM
The team completes the tasks in the sprint backlog while consulting with the PO and stakeholders on requirements and "doneness"
THE SCRUMMASTER
The SM facilitates the Scrum process as a servant-leader, radiates information to the stakeholders, and clears roadblocks out of the team's way
HIGH PRIORITIES
PRODUCT BACKLOG
LOW PRIORITIES
The PO works with the appropriate individuals to seed, prioritize, and groom the product backlog
1 SPRINT PLANNING
1a "WHAT" MEETING
The PO presents the highest priority product backlog items and the TM's determine what they can accomplish and commit to in the sprint
1b "HOW" MEETING
The TM's create the sprint backlog by decomposing the product backlog items they committed to into tasks and estimating the tasks
TO DO
IN PROGRESS
READY FOR REVIEW
DONE
ONE SPRINT: REPEAT N TIMES
the braintrust consulting group
your PM partner
Phone: 205.329.3794 • Email: info@your-pm-partner.com • www.your-pm-partner.com
© 2012 The Braintrust Consulting Group (Please feel free to share and distribute)

Scrum - Terminology

- Roles
  - Product owner (PO)
  - Scrum master (SM)
  - Development team
- Ceremonies
  - Sprint planning
  - Sprint review/demo
  - Sprint retrospective
  - Daily scrum meeting
- Artifacts
  - Product backlog
  - Sprint backlog
  - Burn-down charts

Product owner

- Defines the features of the product
- Decides on release date and content
- Responsible for the profitability of the product (ROI)
- Prioritizes features according to market value
- Adjusts features and priority every iteration
- Accepts or rejects work results

Scrum master

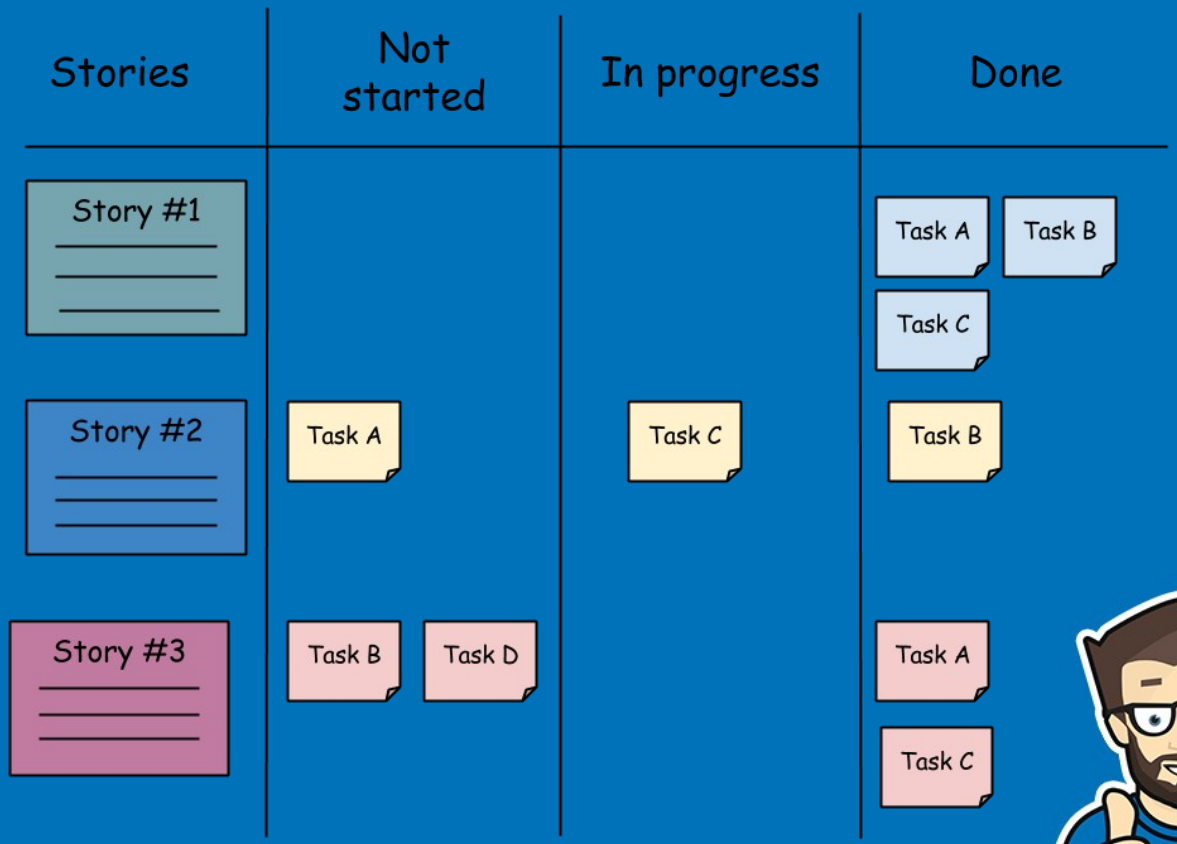
- Represents management to the project
- Responsible for enacting Scrum values and practices
- Removes impediments
- Ensures the team is fully functional and productive
- Enables close cooperation across all roles and functions

Dev Team

- Self-organized (no manager)
- 5-9 full time members
- Cross-functional (Front-end, Backend, QA, DevOps)
- Has the expertise to deliver a working piece of functionality on its own

Product Backlog

- The list of functionality, technology and issues
- Managed and prioritized by Product owner
- Issues are first placeholders, later turned into work
- One list for multiple teams on product
- Anyone can add items
- Keep visible



User Story

- Created by Business Analyst
- Major building block of Product backlog
- Unit of scope – Who, What, Why
- Describes customer view of value
- High level acceptance criteria
- Story differs from Use case

  - Story tells only What but not How. The Use case is more detailed.

Example of User Story:

As a user, I want to search for my customers by their first and last names.

Definition of Ready & Definition of Done

- **Definition of Ready (DoR)** – agreement between the Scrum team and the PO on the criteria for a user story to enter in the sprint

  Example:

  - acceptance criteria clearly defined

  - story estimated by the team

  - team knows how to demo the story
- **Definition of Done (DoD)** – quality criteria for the user story to be considered done

  Example:

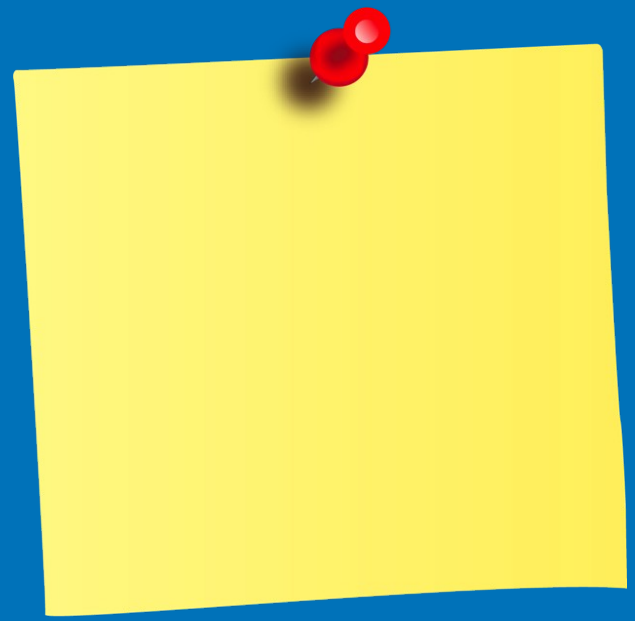
  - acceptance criteria met

  - all tests executed

  - no major defects pending resolution
- Ideally, each Scrum team should have DoR and DoD clearly defined and communicated with stakeholders

Sprint Planning

- Team picks items from product backlog that they can commit to complete
- Sprint backlog is created
  - List of tasks necessary to achieve the work
  - Task are identified and each item is estimated
  - Scrum master does NOT decide for the team
- Team self-organizes to meet the goal
  - Tasks are NOT assigned by manager
- High-level design is considered



Sprint Backlog

- List of tasks the team needs to address during the sprint
- Tasks are estimated by the team
- Team members sign up for tasks, they are not assigned
- Estimated work remaining is updated daily
- Only team can change it

Daily Stand-up

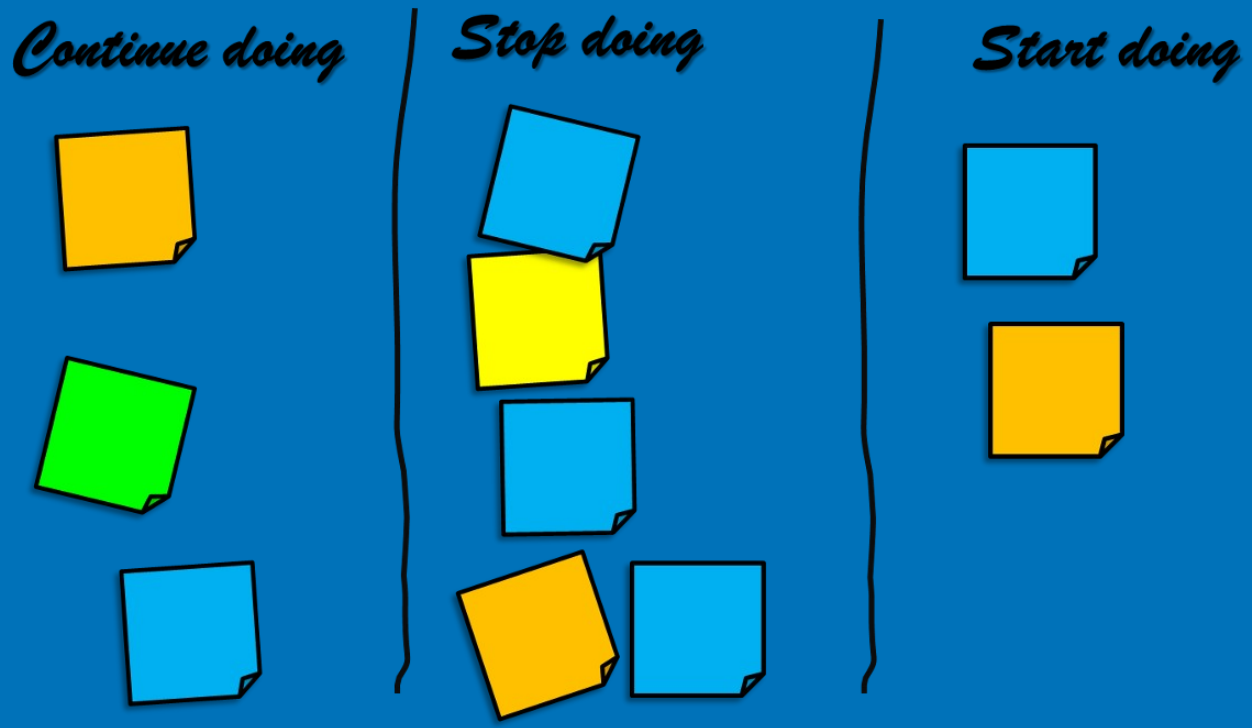
- Happens everyday at a fixed time
- 15 minutes long stand up meeting
- 3 questions are answered by every team member
  - What did I do yesterday?
  - What do I plan to do today?
  -Do I have some blocker?
- Only one team member can speak at a time
- Specific issues are resolved offline

Sprint Demo

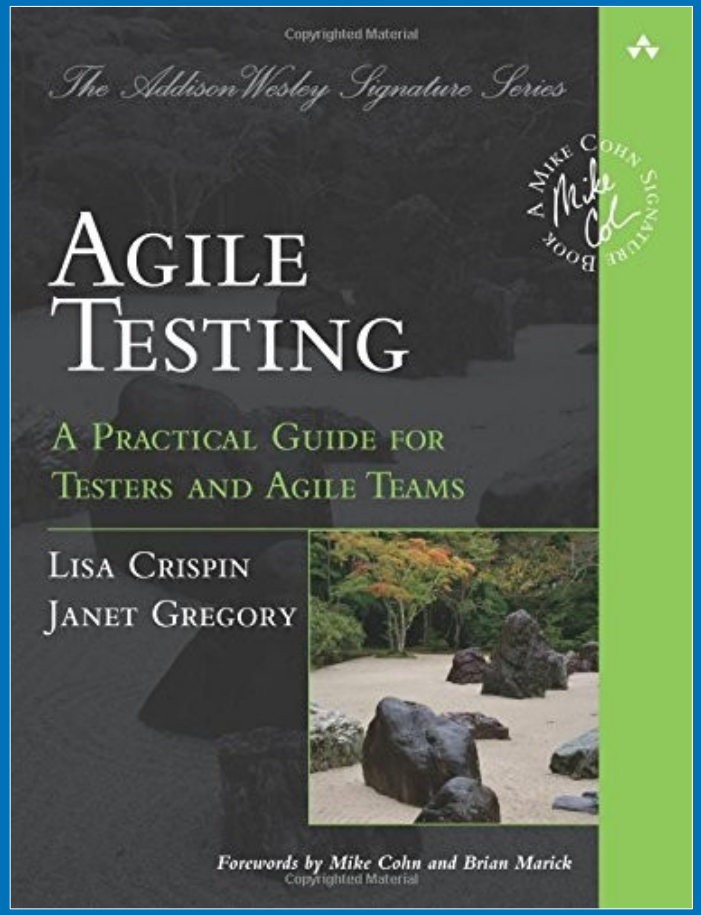
- Team presents what is done during the sprint
- Typically takes the form of a demo of new features or underlying architecture
- Informal
  - 2-hour prep time rule maximum
  - No slides
- Whole team participates
- Product owner(s) signs off stories

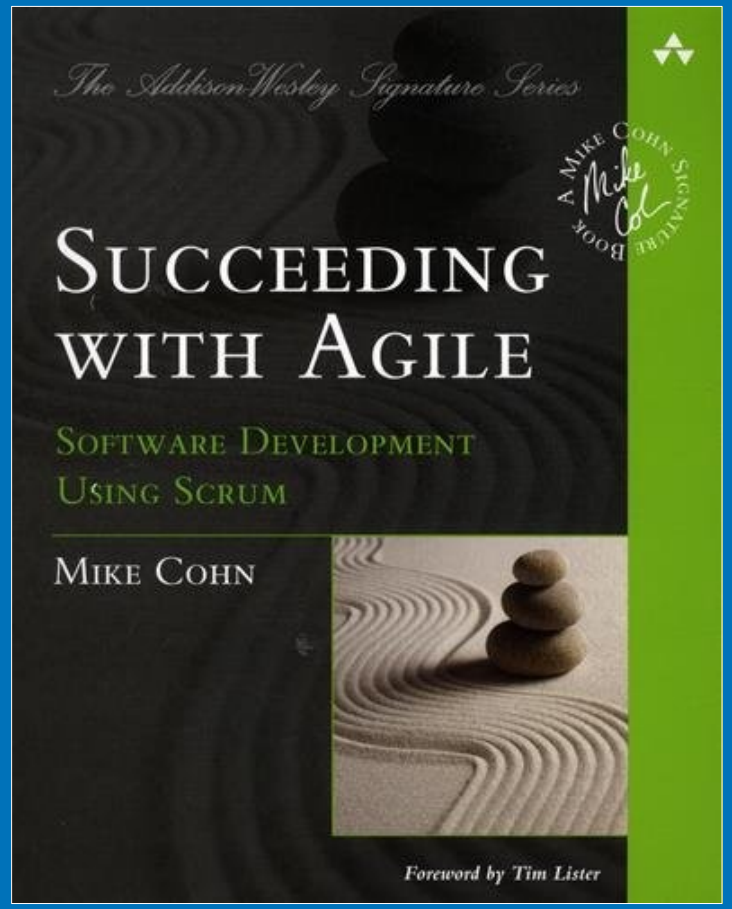
Sprint Retrospective

- Review at what is and is not working for the team
- Issues must be acted upon
- Typically an hour or so
- Done after every sprint
- Whole team participates
  - Scrum Master
  - Product owner
  - Team



Further reading






Q
A