Software Engineering

Software Project Scrum

Prof. Dr. Mirko Sonntag

Hochschule Esslingen
University of Applied Sciences

Fakultät Informationstechnik

Hochschule Esslingen
Flandernstr. 101
73732 Esslingen
Germany

Phone +49 711 397 4160
Fax +49 711 397 4214
E-Mail mirko.sonntag
@hs-esslingen.de

Learning Objectives

- We will run two software projects in the lecture
  - 1 traditional/plan-driven project
  - 1 agile project

- Goals:
  - Get practical experience in running software projects
  - Get deeper knowledge about Waterfall and Scrum
  - Learn differences between plan-driven and agile development

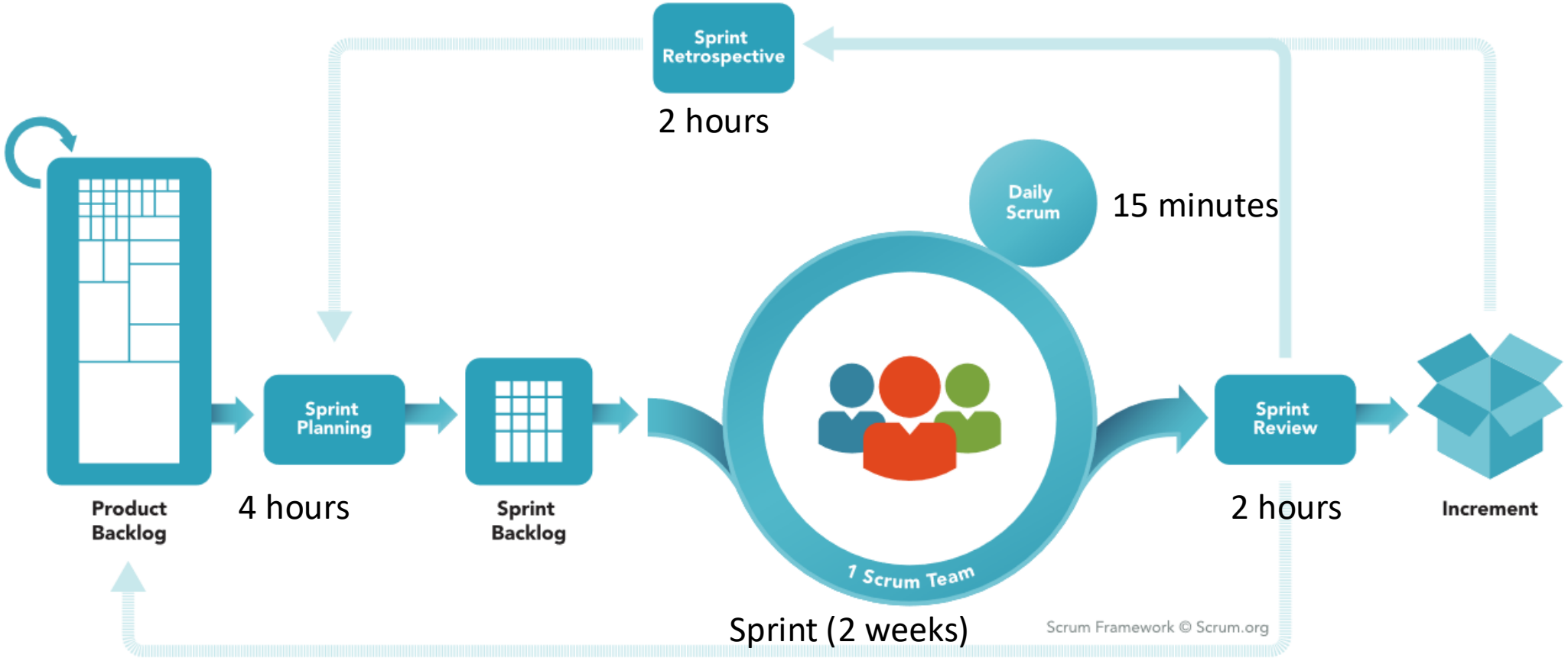
- Hints
  - Choose technology that you are familiar with
  - Not every task needs tools support
  - It's not the goal to run a perfect project!
    - Mistakes are welcome for learning
  - Possible free tool for UML: UMLet, http://www.umletino.com/, http://draw.io
  - Use version management from the very beginning (e.g., Gitlab)

Agile Software Project with Scrum

Roles

- 1 Person:
  - Product Owner
    - Has the last decision about the requirements
  - Scrum Master
    - Makes sure the Scrum process is lived:
      - A backlog is created and used (see Moodle for hints)
      - Sprint planning and sprint review are done
      - A retrospective after the sprint review
        - What was good? What went wrong? Can we improve anything for the next iteration?
      - Daily Standups
        - Planning for the next work day:
          - What did I achieve? What is my goal for the day? Do I have impediments?
    - Is timekeeper of the events
    - Removes impediments/blocking issues
- All other team members
  - Developers: They realize the project

Scrum Cycle in Micro Project



Scrum Process

1. Plan the process
   - Who is PO/SM?
   - Setup version control
   - Setup a collaboration board
   - Create a definition of done
2. Create the initial backlog
   - As a team collect specific requirements (user stories)
   - Put it all into the board
   - They should fill at least the first sprint (a little bit more is better)
3. Sprint planning
   - Discuss in the team which backlog items will be part of the first sprint
   - Add new user stories to the sprint backlog
   - Are there any risks that need your attention?
   - Are there any topics besides the pure software requirements?
     - E.g., get more knowledge about a technology

Scrum Process (2)

4. Sprint
   - If not yet done: decide about technology
   - Distribute the work (the sprint backlog items)
     - Either once per sprint or more often in a daily standup format
   - Work on the sprint backlog items
     - Design, code, test every sprint backlog item
   - Create the product increment
     - Integrate what is **DONE!!!** (see your definition of done)
5. Sprint review
   - Inspect the product increment together with the lecturer
   - Find new requirement and add them to the product backlog
6. Make a retrospective (SM moderates)
   - What was good? What went wrong? Are there any impediments? Can we improve anything for the next iteration? How can we improve the team work?
   - Methods for nice retrospectives: https://retromat.org
7. Go on with the next sprint planning (go to #3)

Project: Library

Requirements

- Goal: A library application where books are registered, can be borrowed and rated (0-5 stars)
- Statistics are tracked
  - how often is a book borrowed, how is the average rating
- Implement GUI or API
  - New books can be registered
    - Title, author, year, edition, publisher, number of pieces/books
  - A book can be borrowed
  - A book can be returned and rated
  - A book (or a number of pieces) can be removed
  - One or more pieces of books can be added
- Technology choice is up to you

Project: Parking Deck

Requirements

- Goal: A parking deck application that represents a real parking deck
- The number of parking lots can be configured for different types of parking lots:
  - Family, electric vehicle, small parking lots (for Smarts etc.), normal
- The prices can be configured:
  - First 15 minutes are free
  - Price for 1st hour
  - Price for every follow-up hour
  - Max price for the day
- The parking deck knows how many parking lots of the specific type are occupied
- The revenue is counted per year

Requirements (2)

- The parking deck can be configured (parking lots, prices)
- The current configuration can be read and changed
- A parking lot can be occupied
- A parking lot can be released
- The status of the parking deck is shown
- The revenue(s) are calculated
- A new fiscal year can be started

- Technology choice is up to you

Project: Hotel Booking

Requirements

- Goal: Implement a hotel booking system
- A hotel has a number of rooms
- Rooms can have different numbers of beds (single, double, …)
- Each room has a specific price per night
- A traveler can search a room by specifying the number of persons and the travel dates
- A list of available rooms is returned and the price is calculated
- A room can then be booked by the traveler
- Implement GUI or API
- No requirements regarding technology

Project: Latex Online Editor

Requirements

- Goal: A web app for writing documents with Latex
- Requirements
  - Latex editor as web application
  - Create a new document
  - Save, close, re-open a document
  - Generate a PDF from the document
  - Download the PDF
- Future:
  - User login (e.g., via Keycloak)
  - Collaborative working
  - Change tracking
- Technology choice is up to you
- You can get inspiration here:
  - https://www.overleaf.com

Sustainable Nutrition Tracking App

Requirements

- Goal: Develop a nutrition tracking application that encourages environmentally friendly eating habits by logging meals and evaluating their environmental impact
- Inspiration: For examle MyFitnessPal, but with a focus on the environmental aspects of food consumption.

Requirements

- Meal Registration and Logging:
  - Users should be able to register new meals with details such as:
    - Title, Calories, Carbohydrates, Fat, Protein, Contains meat (yes/no), Vegetarian (yes/no),
    - Vegan (yes/no)
  - Users can log their meals daily and upload an image for each meal entry.
- Meal Rating System:
  - Meals can be rated on a scale of 0-5 stars.
  - Develop a scoring system that rewards users for choosing meals with low environmental impact, such as vegetarian or vegan options.

Requirements

- Statistical Tracking:
  - Track and display statistics about:
    - Total calories consumed
    - Average meal rating
    - Number of meat-free meals
    - Estimated CO2 savings compared to average meals (provide a reference for what constitutes an average meal's CO2 footprint)
- Implement a graphical user interface (GUI)
- Technology Choice:
  - Students may choose the technology stack and tools they find most suitable for implementing the project (e.g., web-based, mobile app)

Wrap Up
after the project

Wrap Up – How did your team do?

- What went well?

- What can be improved?

Wrap Up – Assessment of the method

- What is your assessment of the method (plan-driven/agile)?
  - E.g., Did it fit to your way of working? Did it fit to the type of project? Any pros/cons regarding the method?