Lecture 3 – Scrum II

Software Engineering & Project

Faculty of SET – School of Computer Science

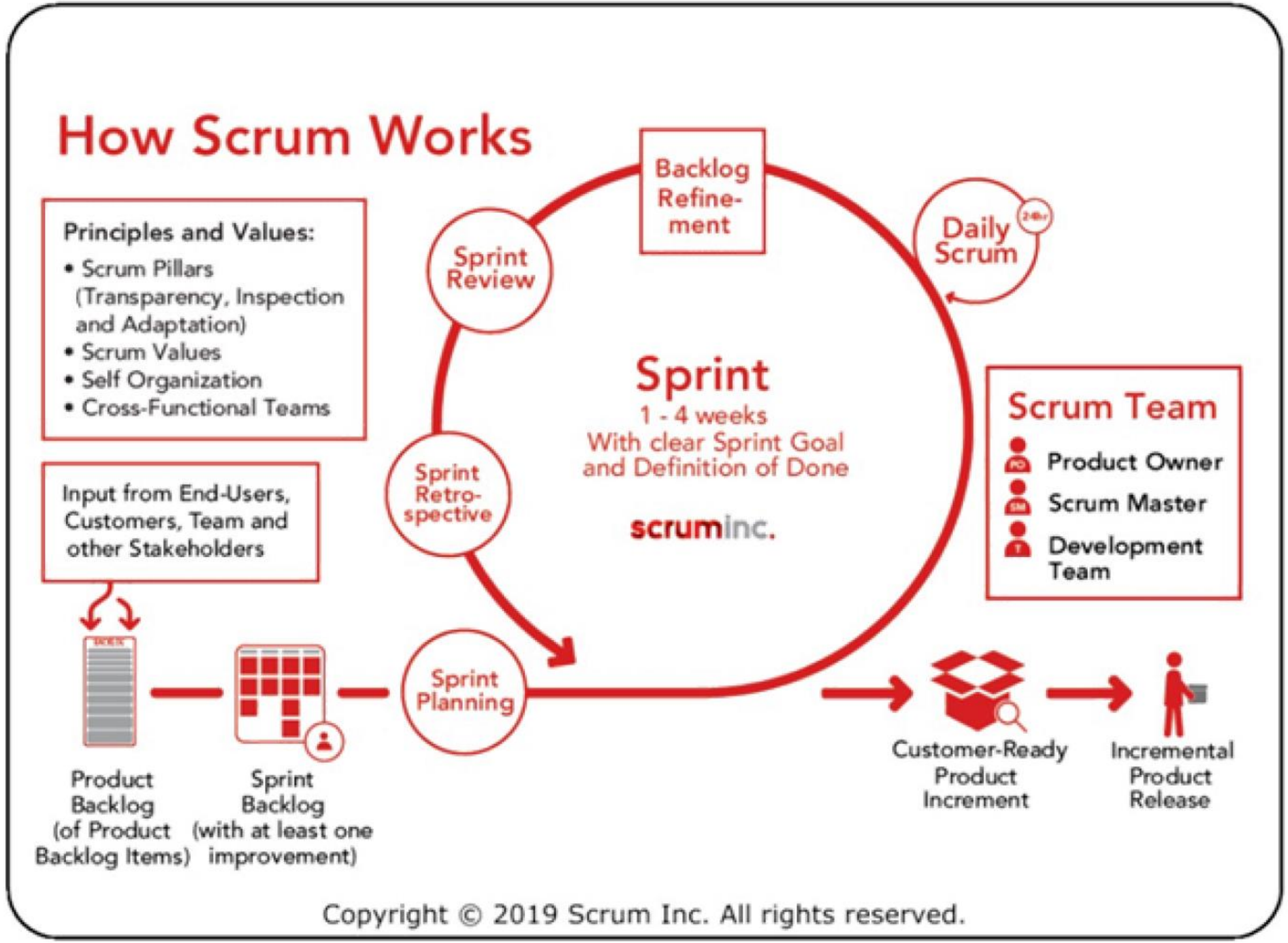
make history.

THE UNIVERSITY of ADELAIDE

Overview

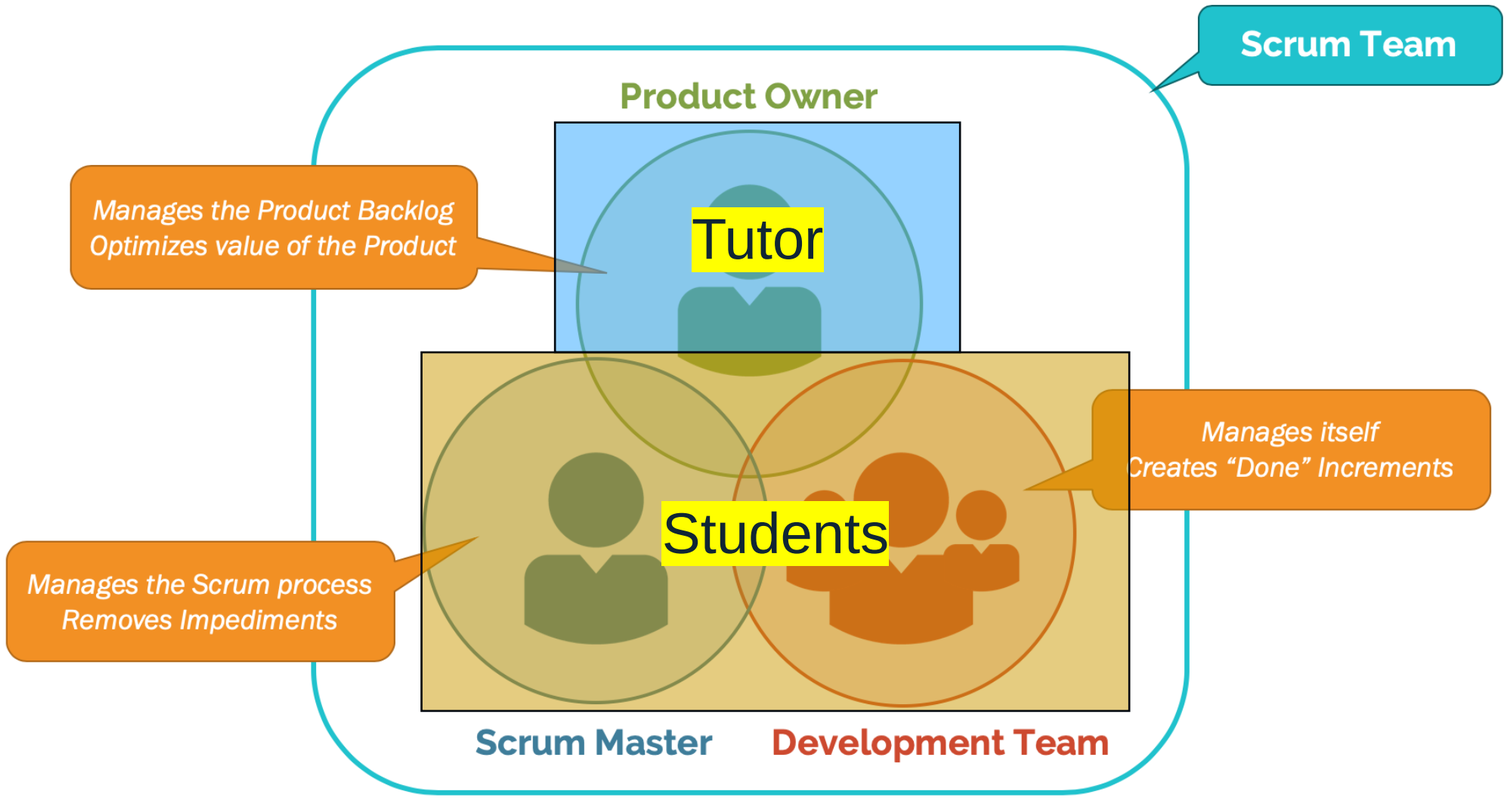
1. **Recap**
2. **Scrum Process**
    1. Daily scrum
    2. Sprint review
    3. Sprint retrospective
3. **Scrum Definitions**
    1. Project vision
    2. Definition of Done (DoD)
4. **Scrum Techniques**
    1. Refactoring
    2. Test-driven development (TDD)
    3. Behaviour-driven development (BDD)
5. **Scrum Tool**
    1. Github Projects

THE UNIVERSITY of ADELAIDE



Copyright © 2019 Scrum Inc. All rights reserved.

Scrum Team Roles



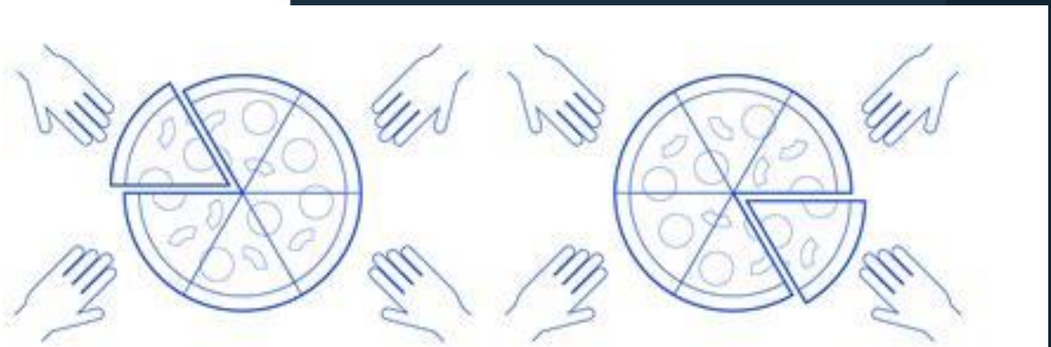
Concept Refresh

THE UNIVERSITY of ADELAIDE

Concept Refresh

Development Team (DT)

- Flexible / cross-functional: no fixed roles (e.g. backend / frontend)
- Self-organised: minimum management
- **Accountable** for their work
- Owns (manages) sprint backlog
- 3-9 people – 2 pizzas!



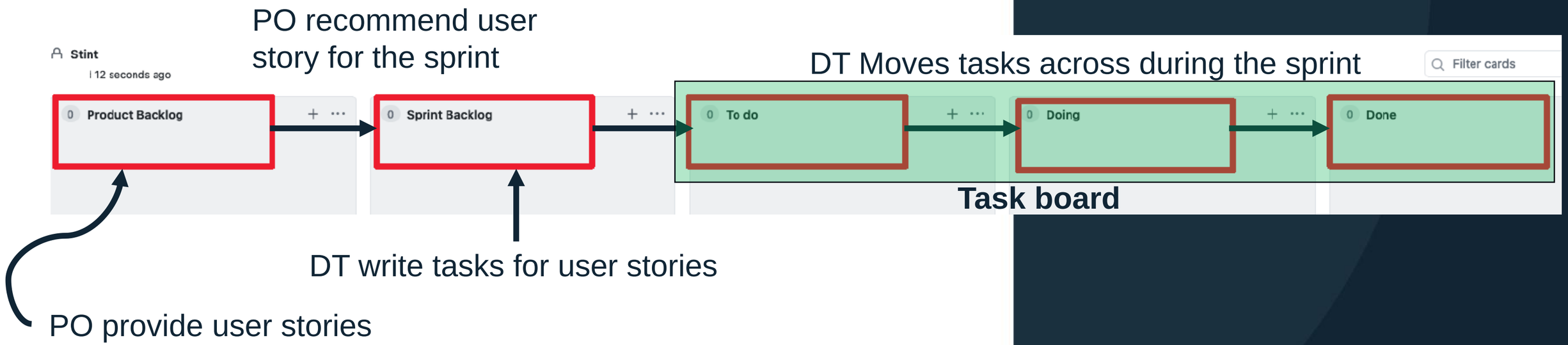
Task:

- Produce potentially deliverable software per sprint
- Communicate with PO to maintain product backlog
- Estimate work complexity
- Self-improvement!

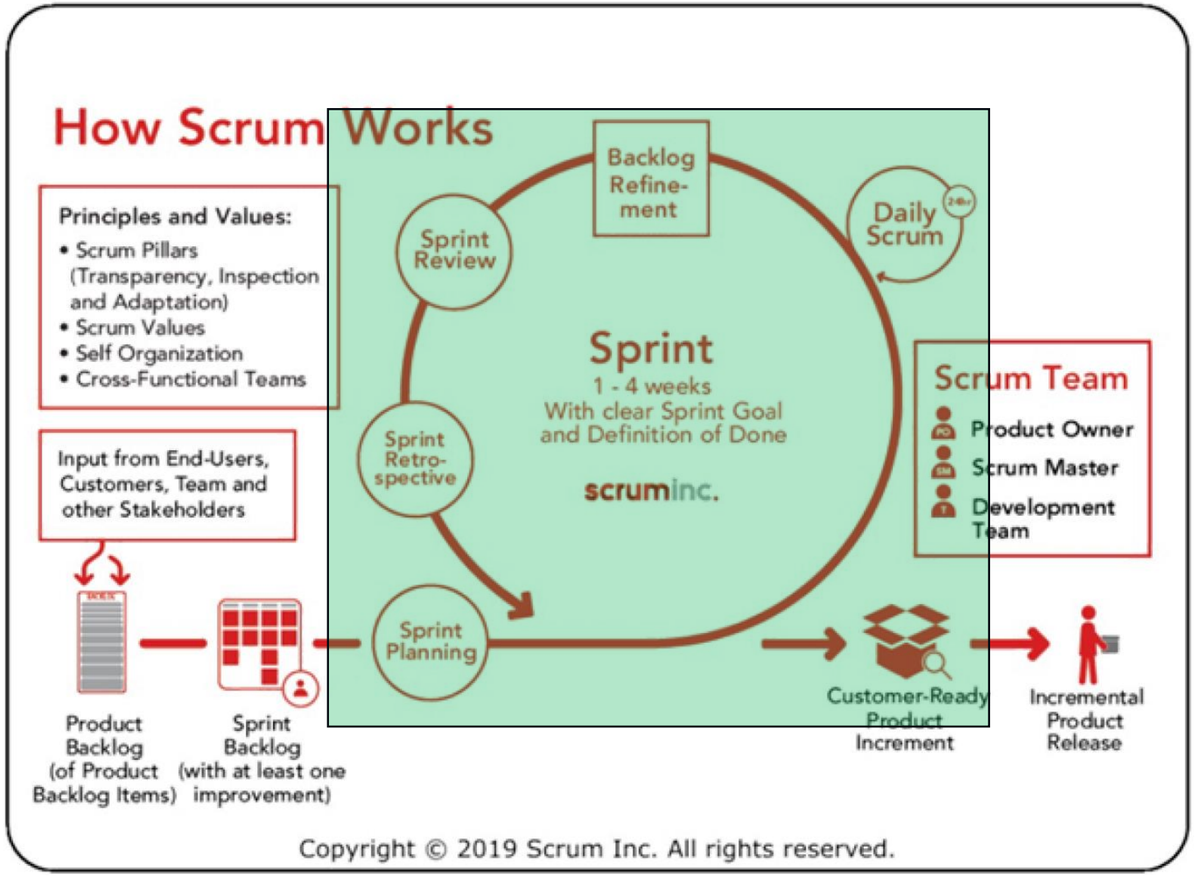
THE UNIVERSITY of ADELAIDE

Concept Refresh

Task board



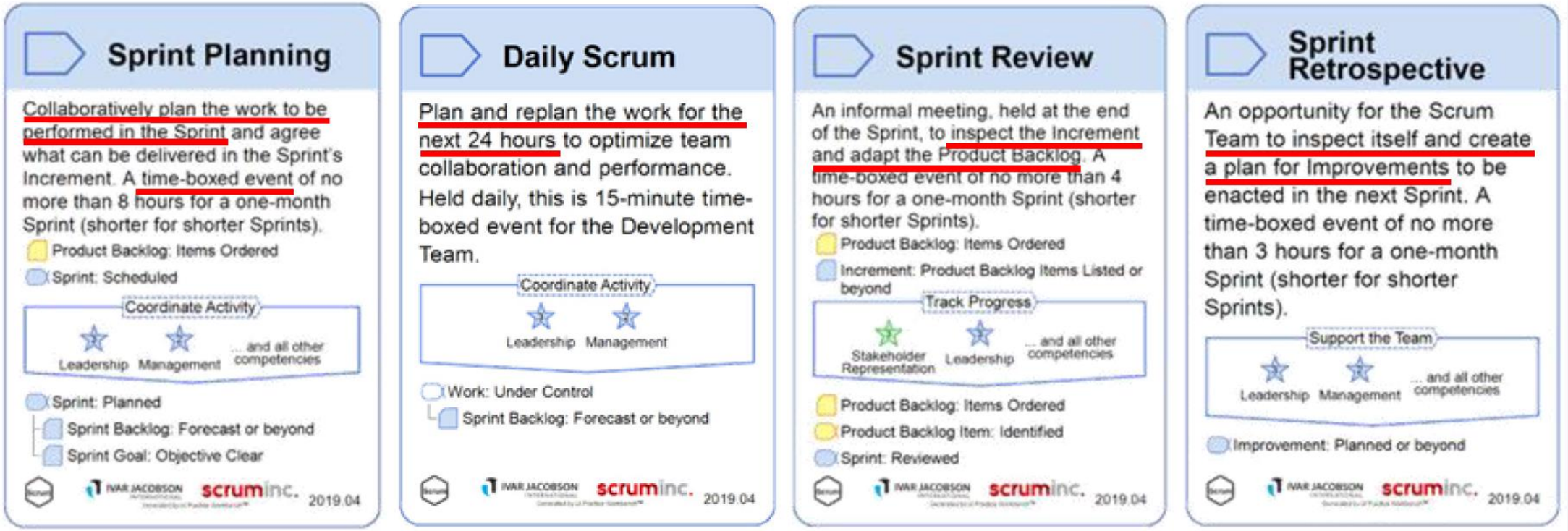
Scrum Events



- When are scrum roles determined?
- When are tasks assigned?
- How is progress assessed?
- SM should facilitate all these meetings – find time, keep meeting on track, etc.

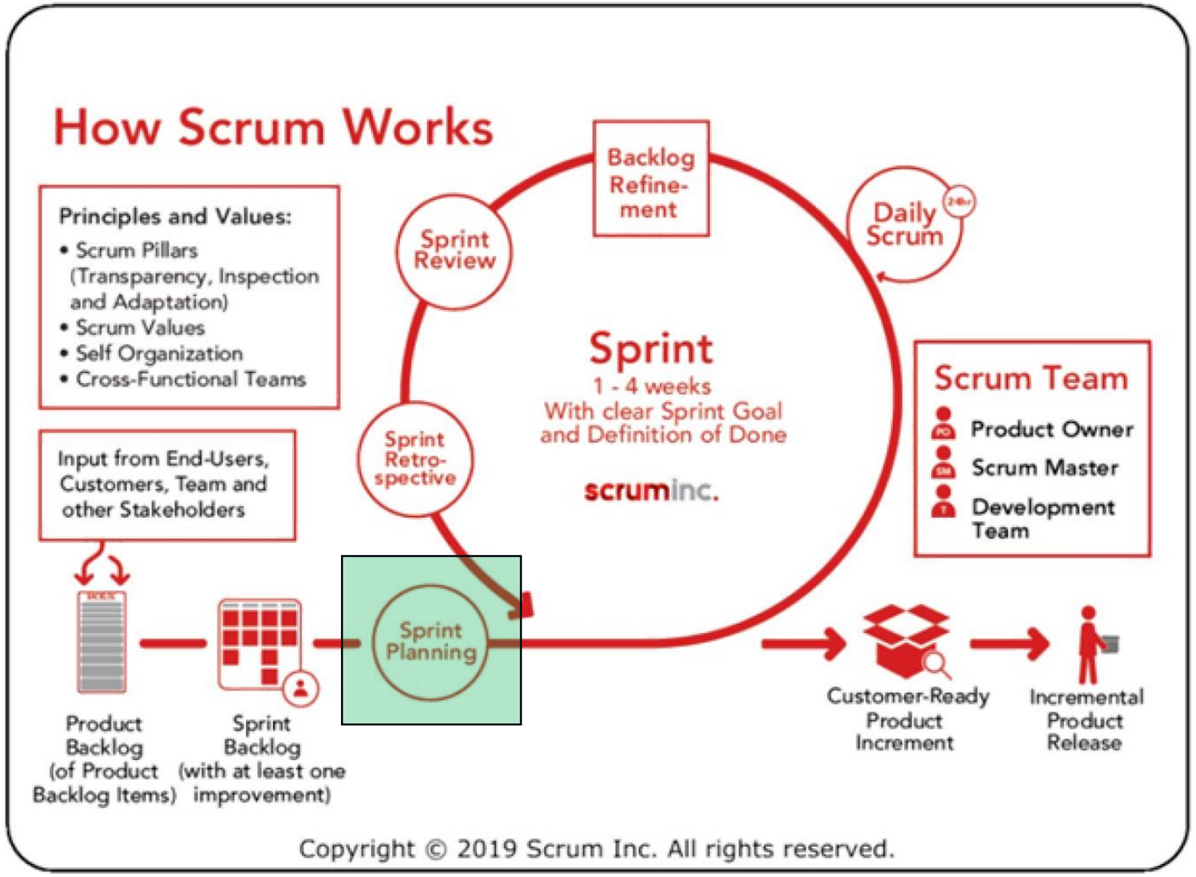
THE UNIVERSITY of ADELAIDE

Scrum Events Overview



THE UNIVERSITY of ADELAIDE

Recap: Sprint planning / kickoff



Tutor will plan first kickoff

Future planning should be done by DT & SM

When?

- Before every sprint, <8hr for 1-month sprint

Who?

- PO (tutor), DT, SM

What?

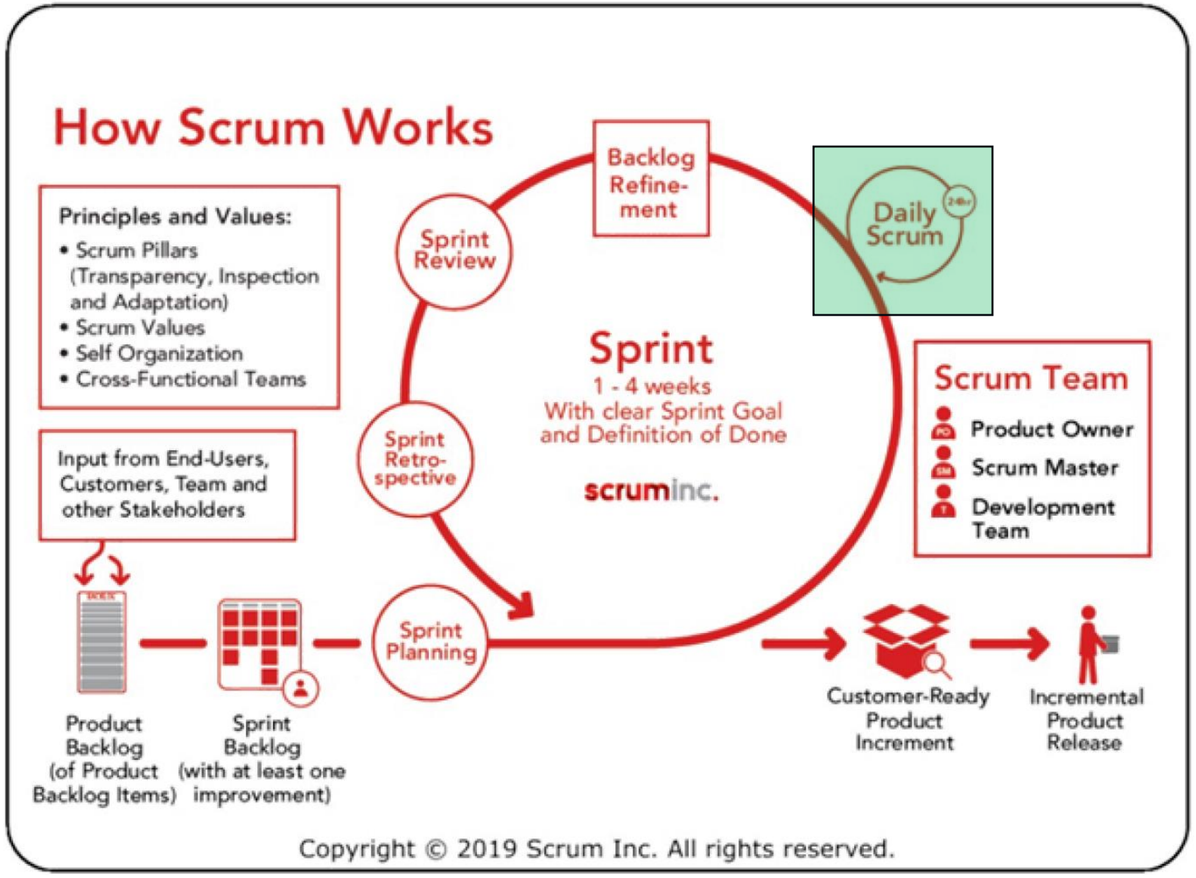
- PO propose user stories for the sprint
- DT decide on final focus
- Populate sprint backlog with tasks

Why?

- Decide sprint goal, define what can be delivered in this sprint

THE UNIVERSITY of ADELAIDE

Recap: Daily Scrum / Stand-up



When?

- Every day at the same time (ideally), ~15min

Who?

- DT, SM

What?

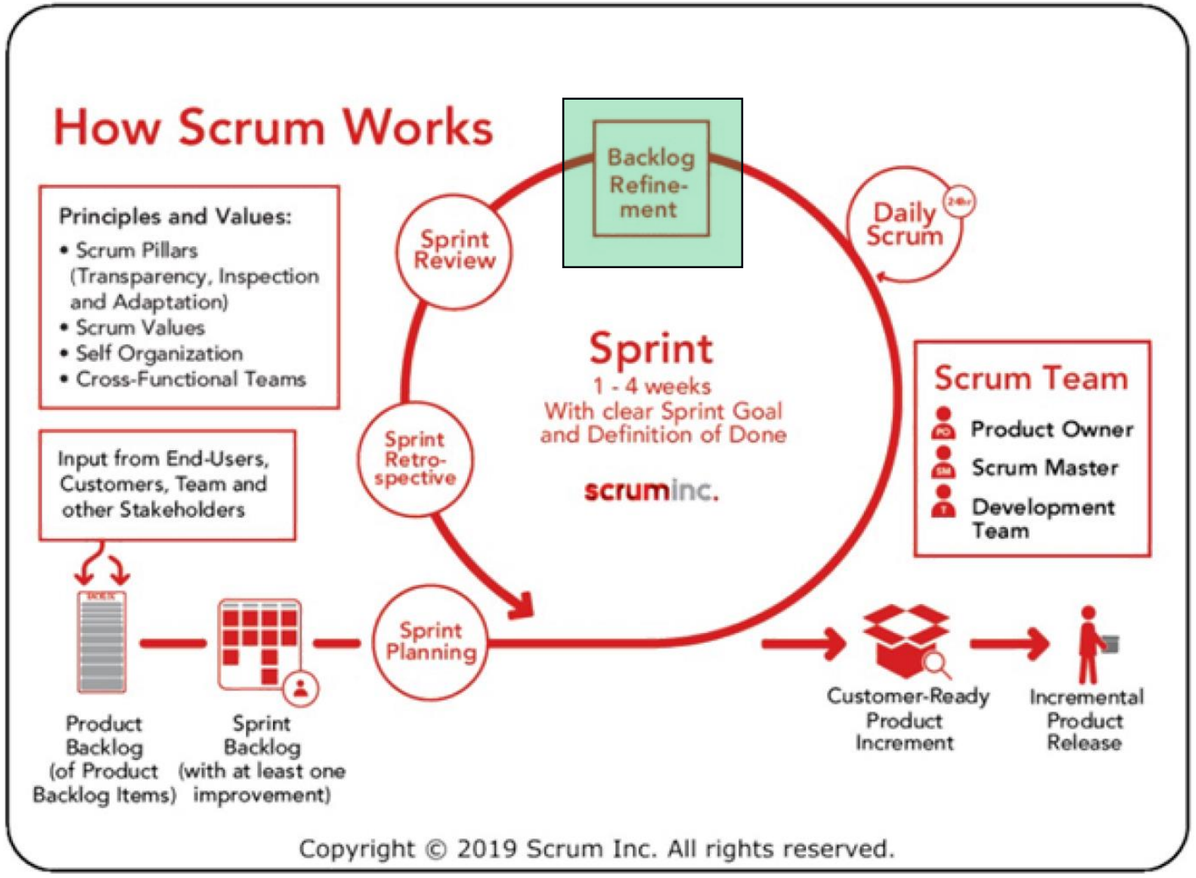
- What have you done yesterday?
- What do you plan to do today?
- Are there any challenges / impediments?
- Update sprint backlog and task board

Why?

- Team pulse check, remove blocks, ensure velocity, clear confusion

THE UNIVERSITY of ADELAIDE

Backlog Refinement



When?

- Ongoing / No fixed time – recommended 1hr/wk

Who?

- DT, SM, PO (not in SEP version due to time)

What?

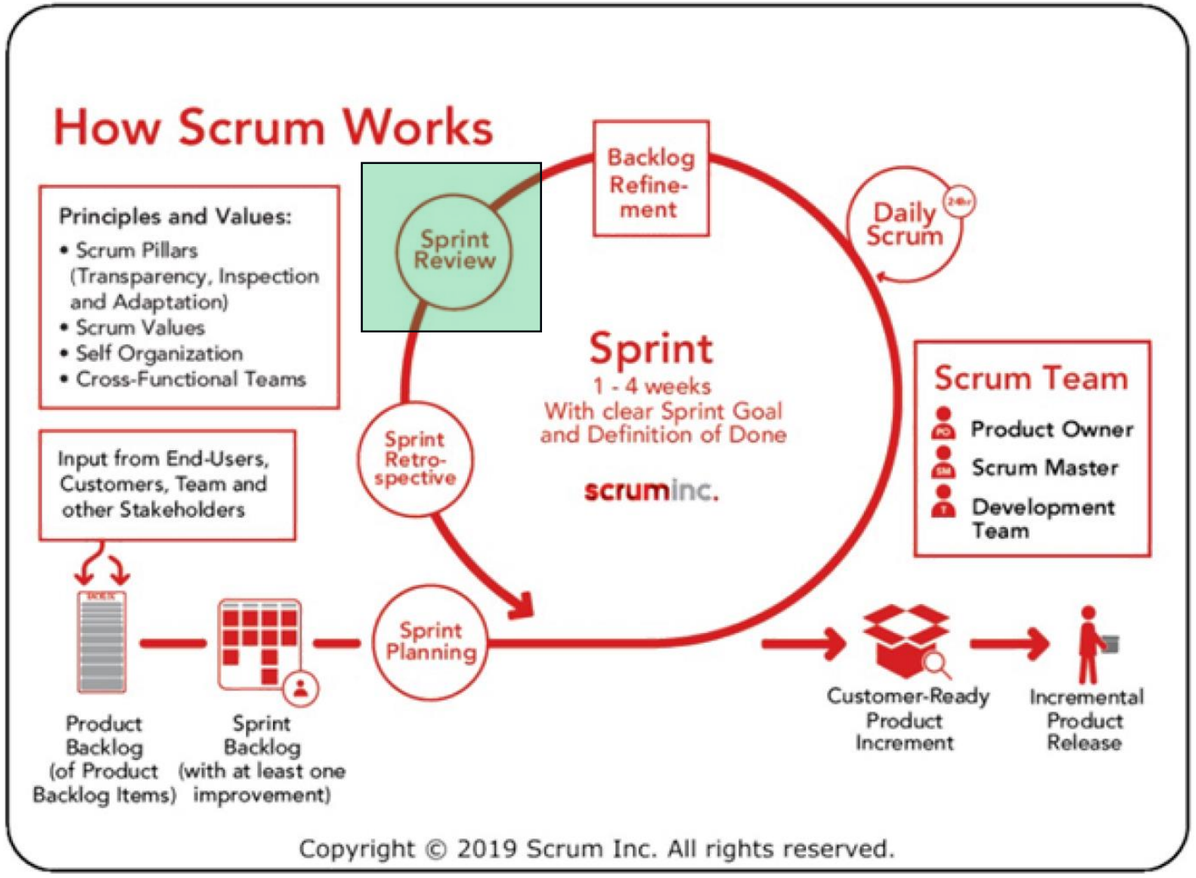
- Reviews future user stories to ensure relevance
- Add details necessary to describe the work needed for a user story
- Estimate user story complexity and refactor

Why?

- Ensure backlog stays relevant and concise

THE UNIVERSITY of ADELAIDE

Sprint Review



When?

- Immediately after sprint (depends on tutor availability), <4hr for 1-month sprint

Who?

- DT, SM, PO

What?

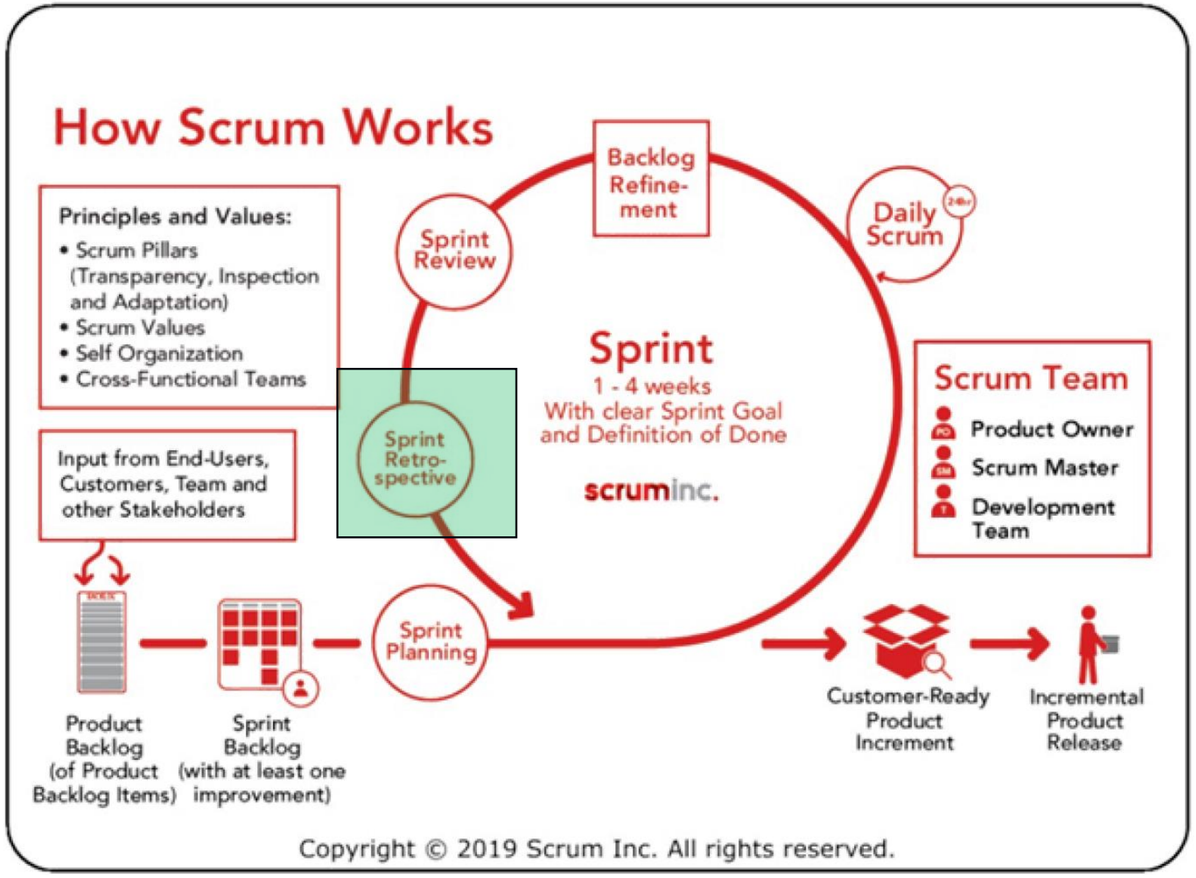
- PO (DT/SM in SEP) summarise sprint progress, include what's done and what's not
- DT explains what went well, problems & solutions
- DT demo product so far, Q&A
- PO review product backlog
- Informal and interactive

Why?

- Review and reflect **product creation**, inspect increment of sprint, adapt product backlog

THE UNIVERSITY of ADELAIDE

Sprint Retrospective



…and back to sprint planning!

When?

- After sprint review, before sprint planning, <3hr for 1-month sprint

Who?

- DT, SM, PO (not in SEP version)

What?

- What went well in the sprint?
- What could be improved?
- What will the team commit to improve next sprint?
- Reflect definition of done

Why?

- Review and reflect **scrum process**, inspect the process, adapt for next sprint
- Good for sprint retrospective assignment

THE UNIVERSITY of ADELAIDE

Scrum Rule of Thumbs in SEP

- Sprints are always 2-week long
  - 5 sprints this semester
- Except the first kickoff meeting, all sprint planning & sprint review are held together for 25min
  - PO (tutor) will only be in this meeting
- SMs are expected to lead the meetings

5	21 August	Mo: Architecture (Slides ↓) Th: - Fr: Real Time Software Design (Slides ↓)	Sprint 1 Review and Sprint 2 Planning (25min) Sprint 2	Retrospective Sprint 1 Quiz 5 Snapshot 2.1

THE UNIVERSITY of ADELAIDE

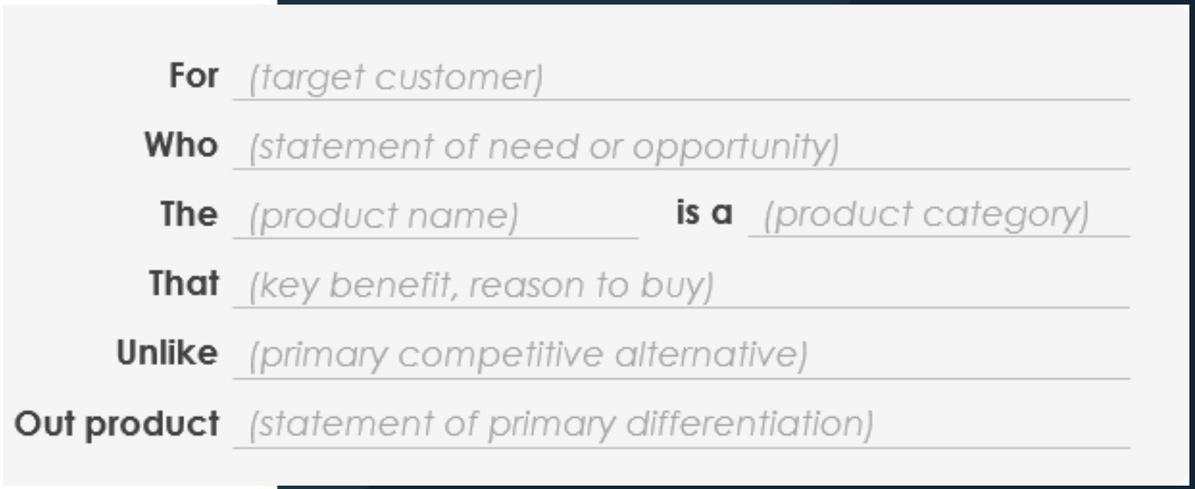
Scrum Definitions

What needs to be decided?

Project Vision

- Usually provided by PO – but write your own for initial report
- Key components:
  - What is the goal and purpose of this project?
  - What are the benefits for the users?
  - What are the key attributes of the project to meet user's needs?
  - How will the team achieve these attributes?
- Concise, Unambiguous, not technical

Example framework:

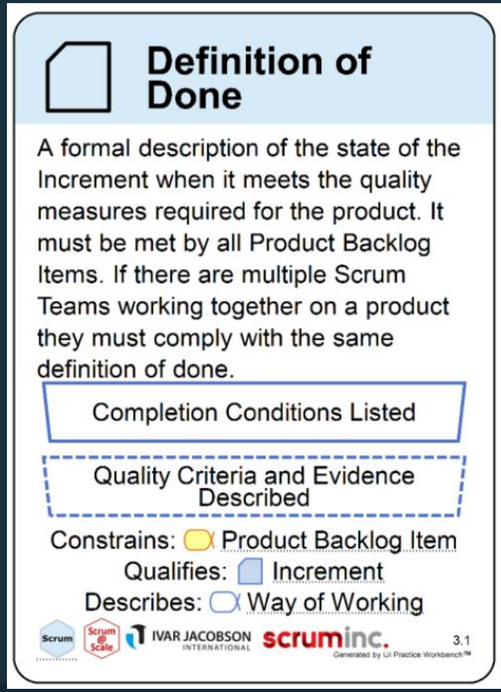


Definition of Done

- Should be defined as soon as possible – ideally in / after first kickoff meeting
- Share across all teams (not in SEP to give each team more experience in creating their own)
- Criteria for quality control, must be **met by all product backlog items**
  - Therefore, should be general enough to apply to all
- Examples:
  - Coding standards are used
  - Code is reviewed and has no bug
  - Sufficient documentations are provided
  - Tests written and passed
  - Built successfully on build server
  - Non-functional requirements met (performance, availability, etc.)
  - Fulfils all acceptance criteria

Scrum Definitions

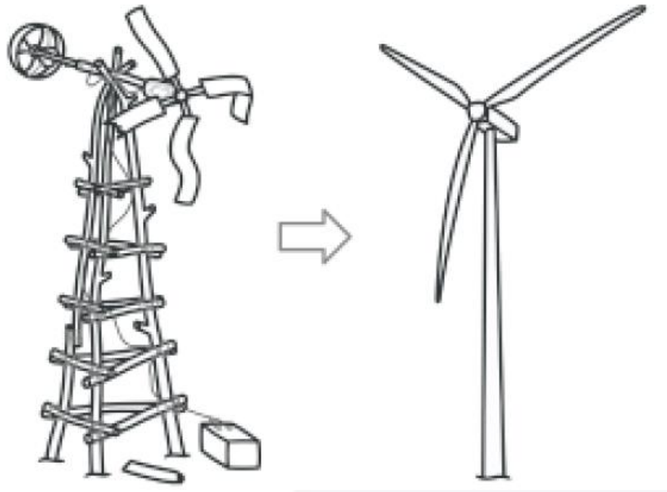
What needs to be decided?



THE UNIVERSITY of ADELAIDE

Refactoring

- = process of restructuring existing source code without changing its external behaviour
  - Reduces complexity, improves flexibility
  - Make the code easier to maintain and extend
- Agile =/= no consideration of software architecture!
- Could be a product backlog item, or (indirectly) incorporated in definition of done



THE UNIVERSITY of ADELAIDE

Refactoring Could be…

- Detect similar code used in multiple functions/methods/classes and move to a commonly used function/method/class
- Detect and remove unused code -> good documentation!
- Renaming variable / class / methods
- Extract parts of code into a new method
- IDEs (e.g. Visual Studio) generally supports common refactoring methods

How to make sure refactoring doesn't change external behaviour?

- Tests & test automation

THE UNIVERSITY of ADELAIDE

Test-Driven Development

- **Idea**: develop **test cases** before implementing a feature
- **Process:**
  1. Write a list of test cases
  2. Before implementing feature, run test cases and **make sure they fail**
  3. Implement feature
  4. Run test cases and see if any still fail – benchmarking
  5. Refactor code
- “Test cases” = unit tests normally
  - acceptance test-driven development – focus on user & acceptance criteria over functionality

THE UNIVERSITY of ADELAIDE

Behaviour-Driven Development

- **Idea:** develop **expected behaviour** before implementing / creating a feature in a form that both customers and developers understand
- User-story-driven: reuse behaviours defined in user stories & acceptance criteria
- Recap on acceptance criteria & user story:

Given<precondition(s)>
When <some user action(s)>
Then<expected result>

THE UNIVERSITY of ADELAIDE

Behaviour-Driven Development Example

```
Given a 5 by 5 game
When I toggle the cell at (3, 2)
Then the grid should look like
.....
.....
.....
..X..
.....
When I toggle the cell at (3, 1)
Then the grid should look like
.....
.....
.....
..X..
..X..
When I toggle the cell at (3, 2)
Then the grid should look like
.....
.....
.....
.....
..X..
```

Framework helps developer implementing the corresponding code

```
private Game game;
private StringRenderer renderer;

@Given("a $width by $height game")
public void theGameIsRunning(int width, int height) {
    game = new Game(width, height);
    renderer = new StringRenderer();
    game.setObserver(renderer);
}

@When("I toggle the cell at ($column, $row)")
public void iToggleTheCellAt(int column, int row) {
    game.toggleCellAt(column, row);
}

@Then("the grid should look like $grid")
public void theGridShouldLookLike(String grid) {
    assertThat(renderer.asString(), equalTo(grid));
}
```

Example: BDD for Game of Life with JBehave
(https://en.wikipedia.org/wiki/Behavior-driven_development#Tooling_examples)

jbehave

THE UNIVERSITY of ADELAIDE

Scrum Tool – Github Project

Once the projects are released, you will be invited to a github repository

- This makes it easier for tutors to check your progress
- You will be using the project board in the repository for backlog & task board
- Make sure to sign in using https://github.cs.adelaide.edu.au/login so we can add you

Demo time!

THE UNIVERSITY of ADELAIDE

Takeaways

1. Sprint planning -> daily scrums -> sprint review -> sprint retrospective
2. Concept of project vision and Definition of Done (DoD)
3. Refactoring, Test-Driven, Behaviour-Driven as scrum techniques to lead development

THE UNIVERSITY of ADELAIDE