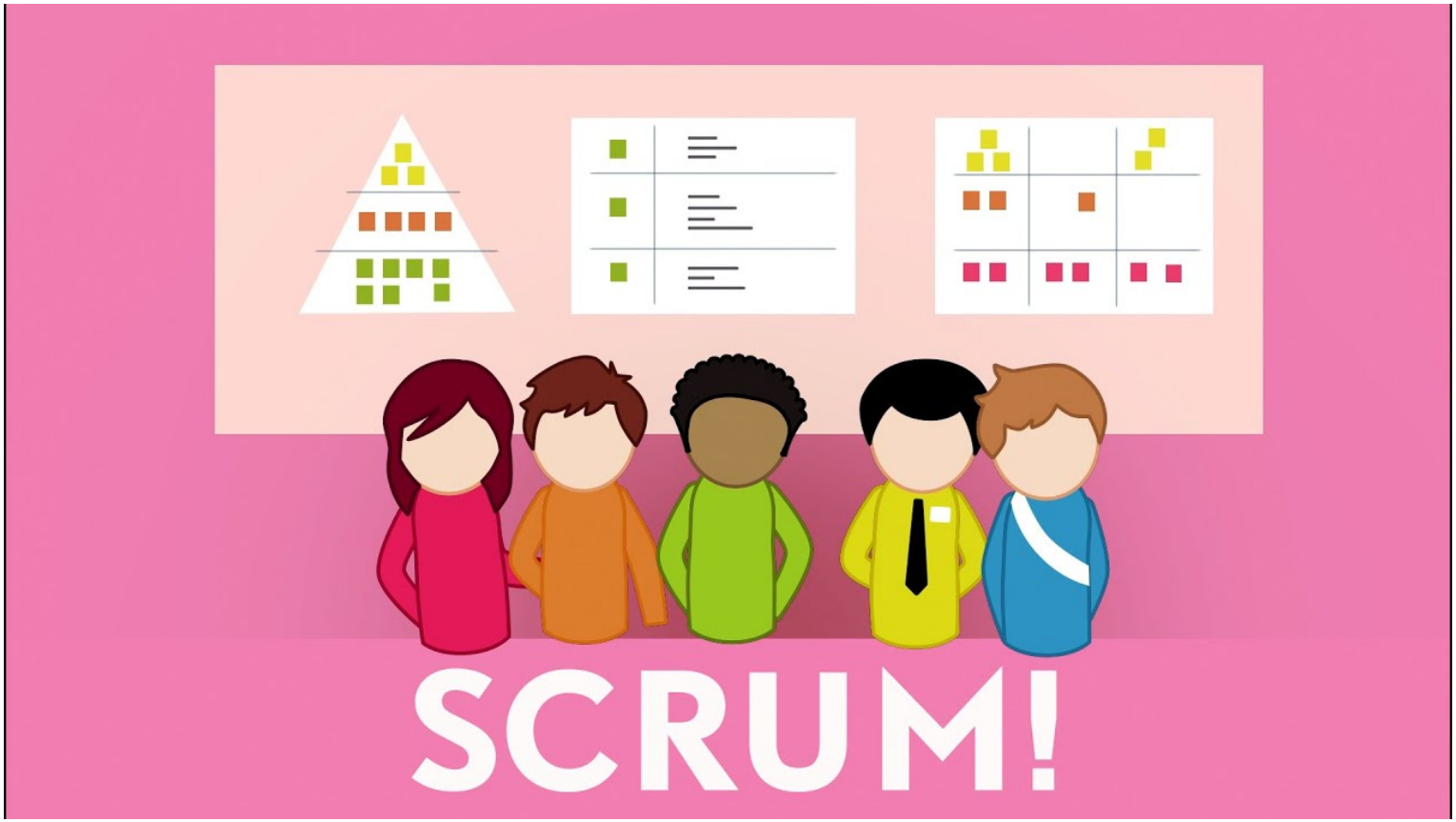


Cynthia Kop and Paul Frederiks

11 March, 2021

The agile manifesto

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

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

That is, while there is value in the items on
the right, we value the items on the left more.

Starting with Scrum

- No long preparations. It's a philosophy and a mindset you can adopt directly.
- Accept that functionality is flexible which can not be defined at forehand in detail. Accept that flexibility requires frequent feedback.
- Scrum is not only a change in the development process but also in the management of software development. It's a change in culture. A critical change is that the development team gets more of their own responsibility.
- Start working with Scrum in the beginning according to the book. Later on, when you have more experience, you can adapt it to your own situation. Watch out for Scrino; use an Agile/Scrum coach!

Characteristics

- Embrace change. (In contrast to traditional project management!)
- Short development cycle (2–4 weeks) with a fixed completion time, called a sprint.
- Development team is fully flexible for future sprints, but totally inflexible for the current sprint.
- Feedback is essential to develop the product. The earlier the better. Use of mock-ups is often helpful.
- In general: you should practice the things you find difficult. The same goes for releasing software: release as often as possible.

Characteristics (continued)

- Since a sprint is a fixed short planning cycle you will become better and better in planning the amount of work you can do. So you will keep your promises to the customer better.
- Choose per sprint the functionality that is most essential, delivering the most added value to the customer.
- The customer is the one who can tell what functionality delivers the most added value. Therefore, the customer should be available for contact (i.e. being the product owner).

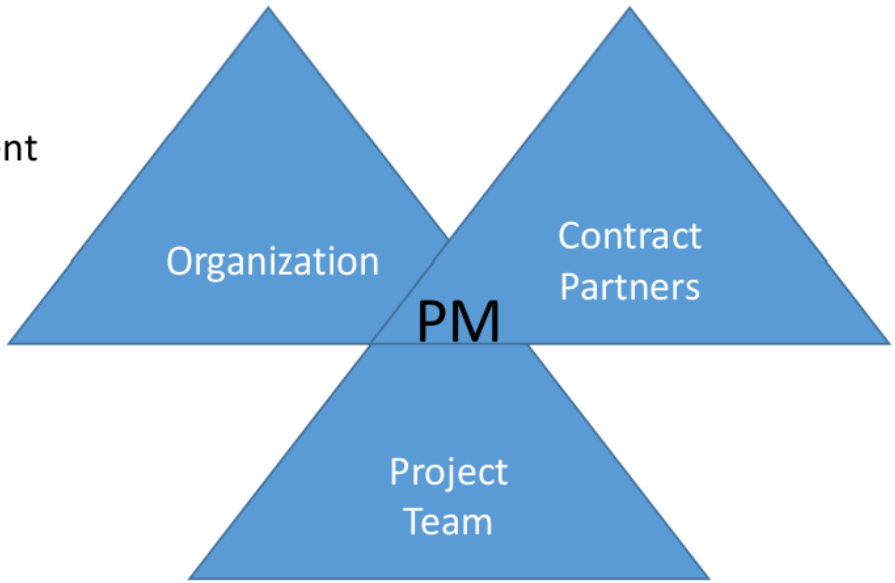
Roles and Responsibilities

- The development team is responsible for the result and therefore manages itself. The team choose their own working process and tools. The team decides how to make the product. The team is also responsible for planning, progress and reporting of a sprint.
- The product owner prioritises with all stakeholders what work the development team has to do in a sprint. Furthermore, the product owner maintains the product backlog (list of desired features). The product backlog is prioritised on added value.
- The Scrum Master is responsible for the sprint process (that it is efficient and effective). A Scrum Master is not the boss!
- The Scrum process itself does not require a project manager! A manager can be helpful to manage the external environment and to coordinate reports, steering committees, intakes and escalations.

The focus of a project manager in a traditional project

- The tocus of a project manager should not only be on the project execution but also on the environment.

- Board of Management
- Line management
- Staff
- Users

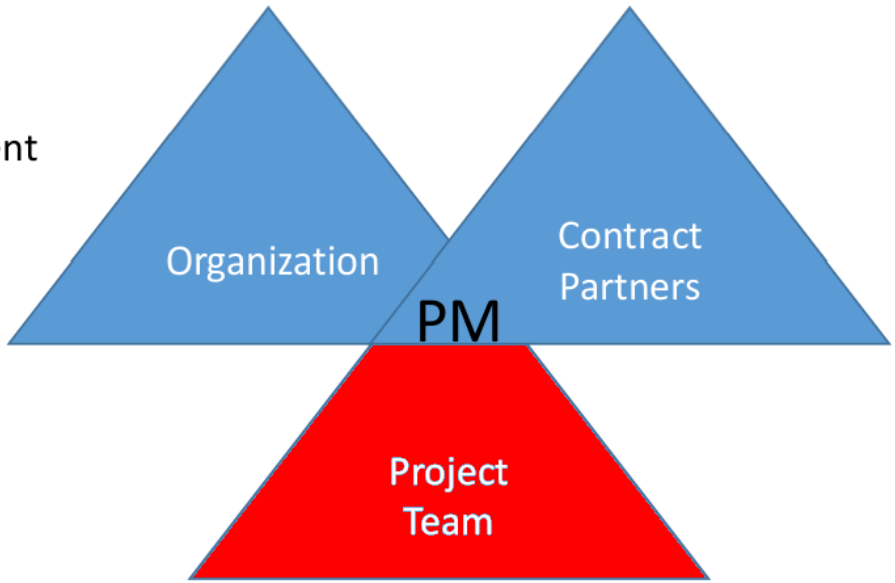


- Supplier
- Financial partners
- Consultants
- Unions

The focus of a project manager in a Scrum project

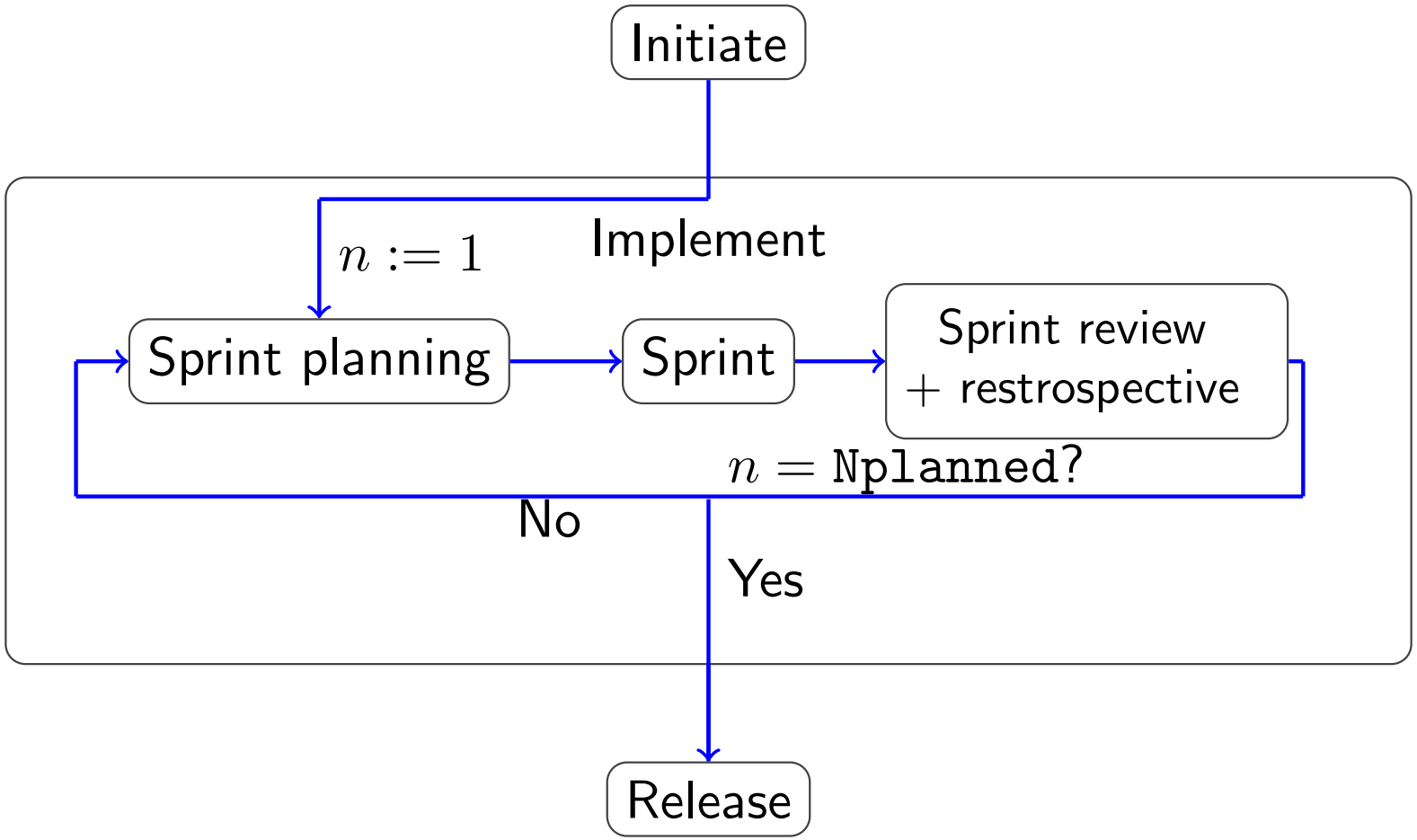
- The focus of a project manager should not ~~only~~ be on the project execution but ~~also~~ mostly on the environment.

- Board of Management
- Line management
- Staff
- Users



- Supplier
- Financial partners
- Consultants
- Unions

The Scrum lifecycle



The Scrum Process

- After each sprint a working product is released.
- A Scrum team consists of 5–9 people, so you can work in one room together.
- A sprint starts with discussing the sprint backlog (the work to be done in this sprint). The team members question the product owner until they have answers to all the questions they have, so they know exactly what is expected of them.
- In the sprint planning meeting the sprint backlog is split in separate tasks called work items. A work item describes a demonstrable result which can preferably be achieved in 2-4 hours but never more than 2 days.
  If it is hard to split work into smaller items it often means that it is not clear enough what should be achieved.

The Scrum Process (continued)

- The Scrum team divides the work and delivers an achievable planning. Each member of the team gives their commitment.
- The Scrum Master organises a daily stand up with three questions for each participant:
  1. What did you achieve yesterday?
  2. What will you achieve today?
  3. What is blocking you?
- An impediment is a blockade for the team to achieve the result of the sprint. The Scrum Master is responsible for removing this blockade before the next stand up.

Planning and adjustment in Scrum

- Velocity is the amount of work a team can do in one sprint. During each sprint the velocity is measured. Velocity is used to estimate if the desired work is achievable in a sprint.
- Planning Poker is an estimation technique in Scrum. It is based on the consensus within the group on the duration of time of an amount of work. Discussion during the planning poker is very important and the arguments of different team members give insight in the work to be done.

Planning and adjustment in Scrum

- By comparing the estimation time to the real development time, the team becomes more predictable in future sprints. The burn down chart shows the hours "burned" in relation to the amount of work done.
- The Scrum board is used for mutual adjustments. On the Scrum board you can easily see what work is ToDo, Ongoing and Done. Work items are put on the board from highest to lowest priority. The highest priority items are worked on first.

Definition of Done

The following is an example of a DoD:
A feature is finished if:

- the code is complete;
- the code is entirely unit-tested;
- the code has been reviewed and accepted by at least one team member;
- the feature has been acceptance-tested;
- the helpfile and user manual is adjusted;
- the software can be deployed directly.

Finishing the sprint

- The end of sprint demo or review meeting is a live presentation of the current status of the product. This is a demo of the real product, not a Powerpoint presentation or UI demo!
- The product owner has to give the team decharge of the sprint.
- As a scrum team you want to improve constantly. Therefore, the retrospective is a very important meeting in which the team evaluates the sprint, defining measures for what went wrong and keeping what went right. The retrospective leads to concrete actions leading to structural improvements in the development process. The retrospective should not be a ritual dance!

How to manage a Scrum team?

- Give the team the confidence and space to organize themselves. Provide them the boundaries within they should work.
- Allow the team their own learning process. Do not intervene!
- Make sure that all the results are transparent all the time.
- Manage on the value for the customer.
- Be an example for the team by working transparently on your backlog.
- Help the team by removing obstacles and roadblocks.

Pitfalls for working with Scrum

- Seeing mistakes and failures as something bad and wrong.
- Not acting after you have a painful learning experience.
- Not starting with something if you believe it is difficult.
- Being afraid to make mistakes or punishing making mistakes
- Hiding mistakes so nobody can learn form it.
- Using transparency and openness against people.

Pitfalls for working with Scrum (continued)

- Managing Scrum teams via command and control.
- Not measuring the benefits of Scrum because its seems difficult.
- Only measuring how "happy" everybody is.
- Implementing Scrum only half.
- Adjusting Scrum before using Scrum.
- Accepting that a piece of work can not be split in more details.
- Starting a sprint with too much work to end with an executable product.

Literature

- Agile for Project Managers – Denise Canty
- De kracht van Scrum – Rini van Solingen & Eelco Rustenburg (in Dutch)
- De Bijenherder – Rini van Solingen (in Dutch)
- Scrum voor managers – Rini van Solingen & Rob van Lanen (in Dutch)
- Agile: geschikt/ongeschikt – Fred Heemstra, Luuk Ketel & Erik van Daalen (in Dutch)

Advice (following presentations and project plans)

Make concrete requirements about testing and continuous integration. Examples:

- We will have X% code coverage on every build.
- Every function of at most X lines is unit-tested.
- When functions cannot be automatically tested, they will be tested using a checklist every time they are changed.
- When pair programming, one team member writes the code and the other the test.
- All unit tests are run when pushing to BRANCH using continuous integration.
- We will have a weekly/daily/constant build process.
- When a build fails, the person whose push caused it to fail is responsible for fixing it.

Advice (following presentations and project plans)

Make concrete requirements about code quality and reviews. For example:

- Every function has a single responsibility that is clear from the name.
- If a function has side effects this must be clear from the name and/or explicitly indicated in its comment.
- A function may have at most X lines of code. (Exceptions are allowed only if Y other engineers agree that this is needed.)
- Outdated code is removed instead of outcommented.
- If more than three lines of code are copied within the project, all engineers must agree that this is necessary.
- Use of linters or a tool like BetterCodeHub.

Add requirements as they are discovered!