Scrum vs Science?

...

2021-09-21

What is Scrum?*

Scrum** is a lightweight framework that helps people, teams, and organizations generate value through adaptive solutions for complex problems.

- It's simple ("a non-framework")
- You can't improve what you can't measure ("Empiricism")
- Trust people over processes ("self-managing", "cross-functional team")
- Incremental ("time-boxing")
- Transparent ("fail early", "lean")

Values: Commitment, Focus, Openness, Respect, and Courage

[*]: according to the Scrum-Guide at https://scrumguides.org/
[**]: term borrowed from Rugby as an emphasis on teamwork

Historic context (1990s)

- Era of impactful 1-person software projects ended in the 1980s
  - Collaborations between individuals and organizations required
  - Increased complexity and project size
  - Specialization of developers
- Project-Management adapted from other engineering often fail
  - Software is more dynamic than a building (V-model, Spiral, Waterfall model)
  - Less formal training of software engineers (self-taught, lateral hires)
- Lean manufacturing becomes more prevalent (success in car manufacturing)
  - Switch to object-oriented programming paradigms
  - Best practices become (over)-formalized with “Extreme Programming” (embrace change, simplicity, communication), Feature Driven Development (FDD), and Unified Process (*UP)
- One of several “Agile Software Development” practices

Scrum assumptions

- End product cannot be fully defined at the beginning of a project
  - Requirements change
  - Challenges are unpredictable
- Specialists perform best in their speciality
  - Customers know what they can sell
  - Engineers know how to solve technical problems
  - Designers know user interfaces
  - Managers know how to organize work
- Complex tasks require team effort
- Minimize organizational overhead, maximize time on task

Theory

How to Scrum

1. “Product Owner” (re)defines “product backlog”
2. “Team” turns backlog into work units for current “Sprint”
3. “Team” and “Stakeholder” inspect results
4. Go back to 1.

Product Owner: accountable for value of the product, owns backlog
Team: No hierarchies with 3 roles: Developer, Scrum Master, Product Owner
Sprint: one increment of a development cycle, fixed time period (eg 2...6 weeks)
Product backlog: Goal and results of the project with work items

Artifacts

Product backlog

- Only Product Owner can modify
- Ordered list of work items
- Defined by Product Goal

Sprint backlog

- Planned by and for the team
- Actionable plan of work items
- Defined by sprint goal
- Real-time representation of the work

Definition of Done

- Formal description of quality measure for work item
- Sprint specific agreement in the team

Work item / Increment

- Improvements toward Product Goal
- Work units that meet “Definition of Done” become “increment”

Events

Sprint

- Increment to the product
- defined by sprint goal
- Fixed time period

1) **Sprint Planning** (team, 1st day, a few hours)

- Product Owner suggests what to improve, team defined Sprint Goal
- Team selects items from Backlog and commit to it
- Developer break down items into work units

2) **“Daily” Scrum** (developers, daily, 15min)

- discuss plan and roadblocks of the day

3) **Sprint Review** (team + stakeholders, 2nd to last day, few hours)

- Discuss sprint progress towards product goal

4) Sprint Retrospective (team, last day, few hours)

- Review and improve the process and tools

Practice

Translations for Scrum in Science

- Team ← Lab(?)
- Product Owner ← Principal Investigator(?)
- Developer ← Researcher(?), Co-Authors(?)
- Scrum Master ← Scrum Facilitator(?), **Scrummer**(!)
- Stakeholder ← Experts(?), Collaborator(?), Co-Authors(?)
- Product ← Paper(?)
- Sprint
- Definition of Done

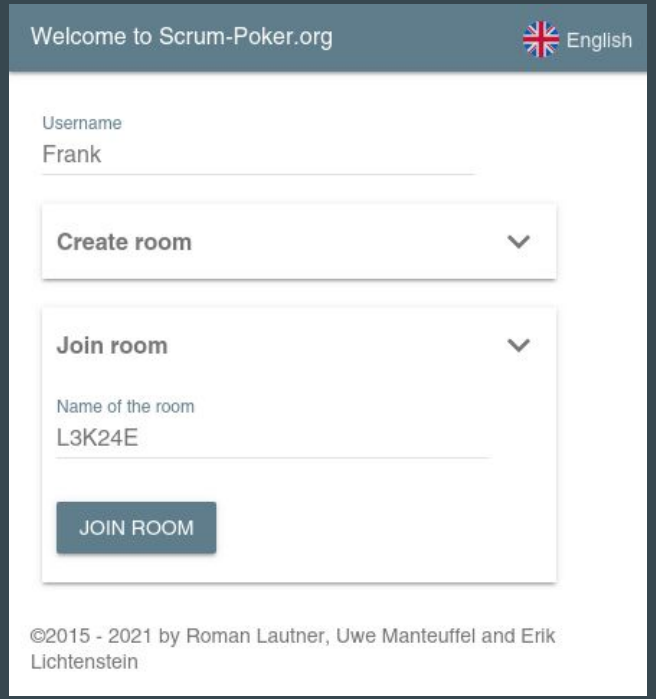
Planning Poker

- Useful feedback for Product Owner
- Increases Transparency
- Uses crowd intelligence to estimate workload

How?

1. Product Owner introduces task
2. Team asks questions
3. Everyone estimates “Task Points”
4. Highest and lowest points explain their view
5. repeat from 2 until consensus

https://scrum-poker.org/L3K24E



Poker Test

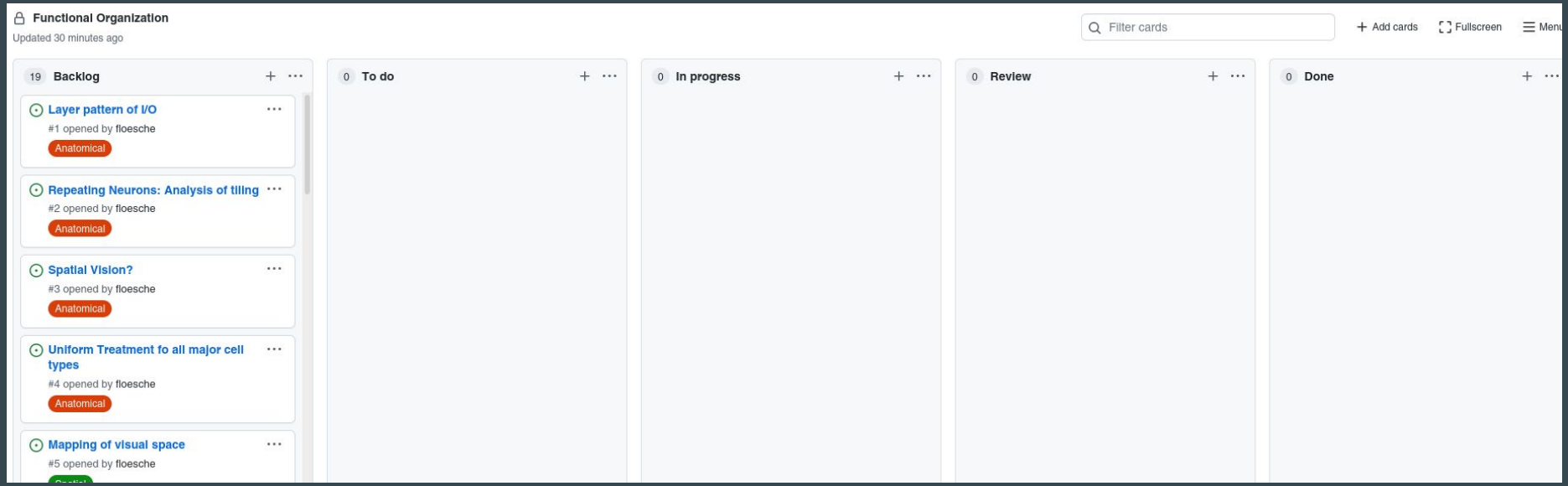
Task 1:

Prepare a presentation for the next lab meeting.

Task 2:

Set up a new flight arena.

Scrum board (Single Source of Truth)



Suggestions how to succeed and Recipe for disaster

- Getting used to Scrum takes time (> 1...2 sprints)
- Not a step-by-step guide to success, but a space to negotiate collaboration
- Works best with stable team
- Scrum-board as “single source of truth”
- Time-boxing! (fixed times and locations)

- One or more team members working outside the agreement
- Not adapting the explicit descriptions to the real world
- Not respecting the roles and agreements
- Blindly following rules

Commitment, Focus, Openness, Respect, and Courage

Quick review of Scrum in Science

- LabScrum [3]:
  Neuroscience lab at Oregon → “organic” growth throughout the institute
  Largely positive observations
  Implementation close to Scrum Guide [4] (PI=PO, small teams, scrum of scrum)
- Nature paper [2]:
  Mostly besides the point, misrepresentation of Scrum [4] by violating fundamental principles (Focus on Standup instead of board, Lab head as Scrum Master, focus on tools (slack) instead of people...)
- Feedback from interviews [1]
  More balanced discussion of pro and con
  Interesting case studies

Summary

- I* think it's possible to use the tools from Scrum for the project
- Quick turnarounds and increments during a sprint allows fast improvements
- With transparent discussions in sprint planning and review, the resulting paper could be a smoother read

[*] Full Disclosure: I am a certified "Professional Scrum Master I" :-)

Software engineers vs scientists

- “Don’t tell me how to do my work”
- “I don’t know at the beginning what the software will look like”
- “Meetings take away time from coding”
- “I have been successfully solving most issues when I encountered them”
- “I have never seen a good project management”
- “You need to rewrite code 4 times before you can code a (good) library the 5th time”
- “I only believe what I can measure”
- “There is no management framework that caters to my unique coding skills.”

- “Don’t tell me how to do science”
- “I can’t plan for my next steps before completing a set of experiments”
- “Meetings take away time from experiments”
- “I have successfully solved most problems in the past”
- “I have never seen a project management and that’s good”
- “Science is 80% failures anyway, but the other 20% are publishable”
- “Quantification is fundamental to science”
- “I have it in writing that my skills are unique; no management framework can acknowledge that.”

Conclusion: There might also be differences :-)

References

[1] Adapting the scrum framework for agile project management in science: case study of a distributed research initiative ***doi:*** *10.1016/j.heliyon.2019.e01447*
[2] A project-management tool from the tech industry could benefit your lab
doi: *10.1038/d41586-019-02620-6*
[3] LabScrum: A Case Study For Agility in Academic Research Labs
doi: *10.31234/osf.io/zg4ub*
[4] The Scrum Guide ***url:*** *https://scrumguides.org*