

CSE 403

Software Engineering

Winter 2023

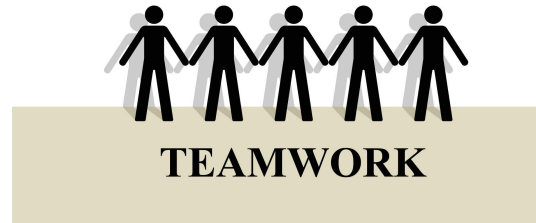
Scrum and Teams

Today

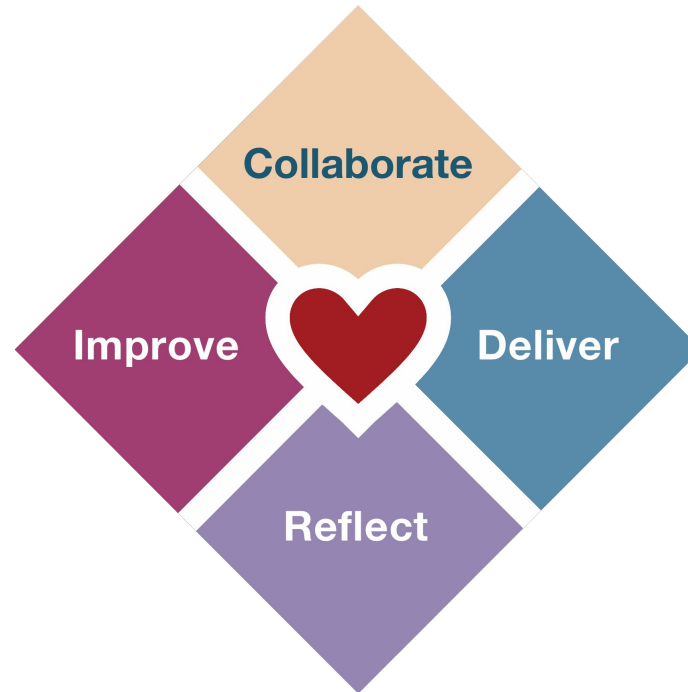
- Scrum



- Working in Teams

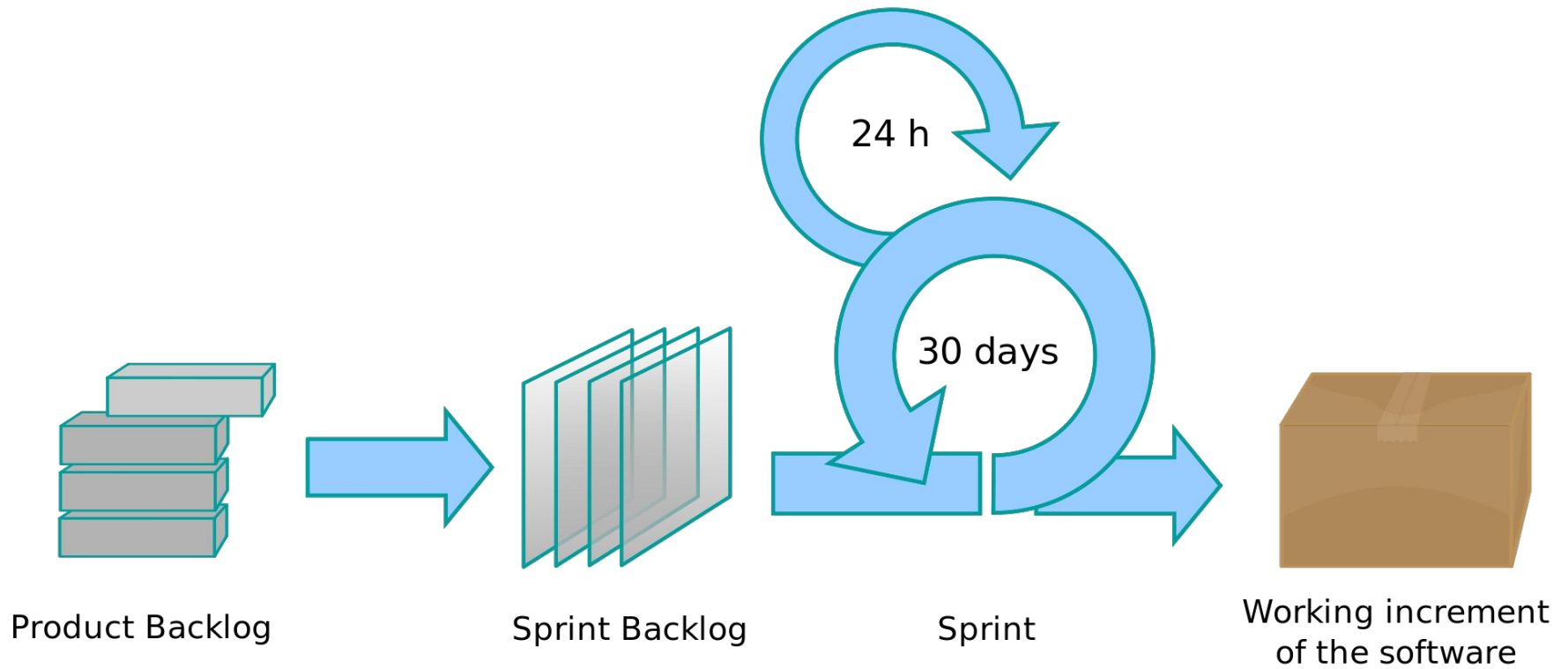


Scrum: overview



Heart of agile [Cockburn]

Scrum: overview



Scrum: overview

Small number of team members: 6 (+/- 2)

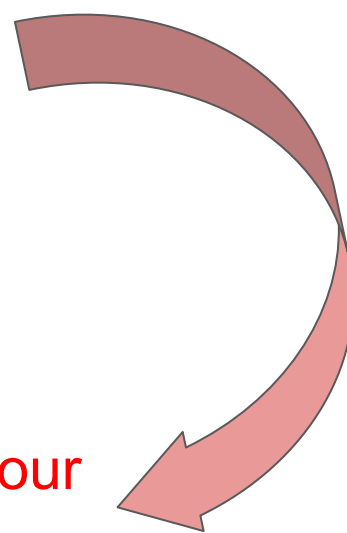
A time-boxed model:

- Each Sprint (time box): max 30 days
- Fixed number of tasks for each Sprint
- Daily Scrum meeting: 15 min max
- Each sprint results in a
 - Sprint review (product demo): 0.5-1 hour
 - Sprint retrospective (post-mortem): 1-3 hours

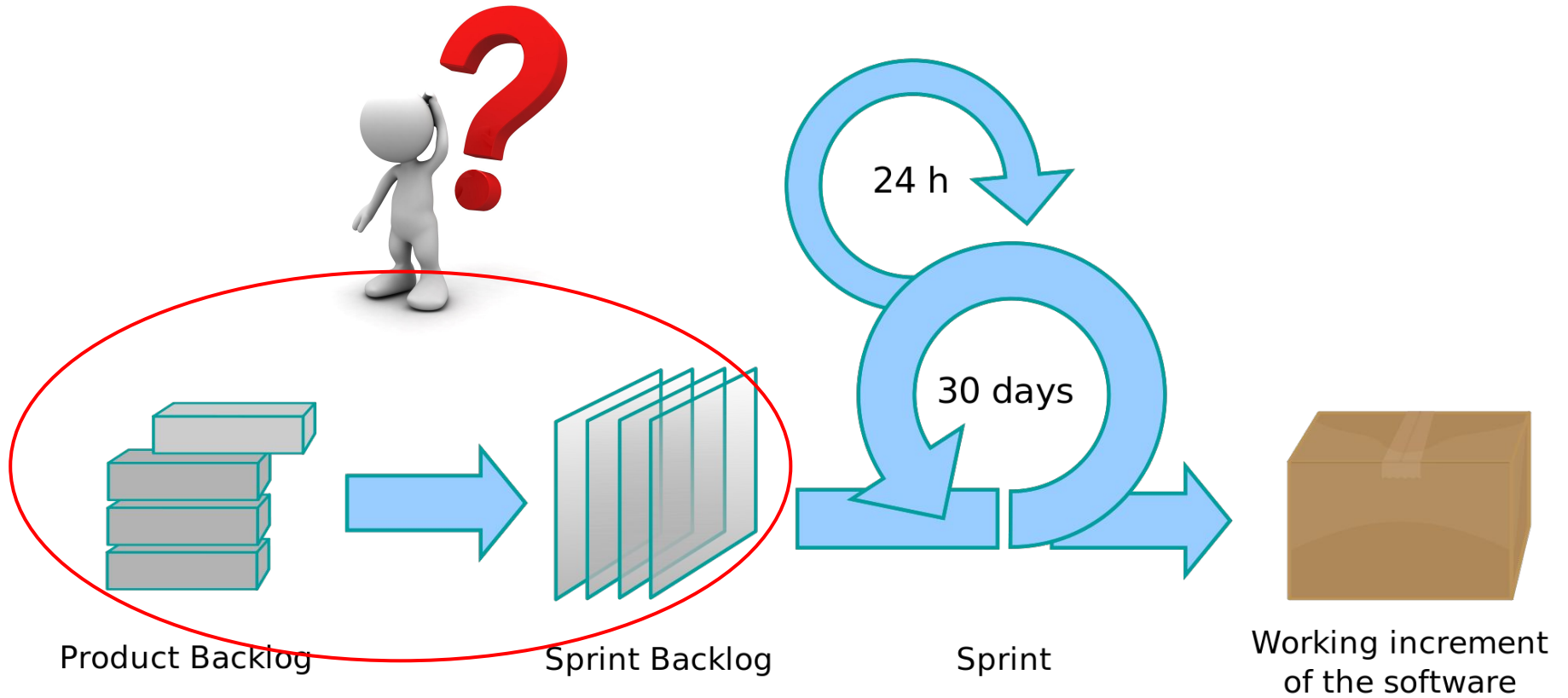
Scrum: overview

Small number of team members: 6 (+/- 2)

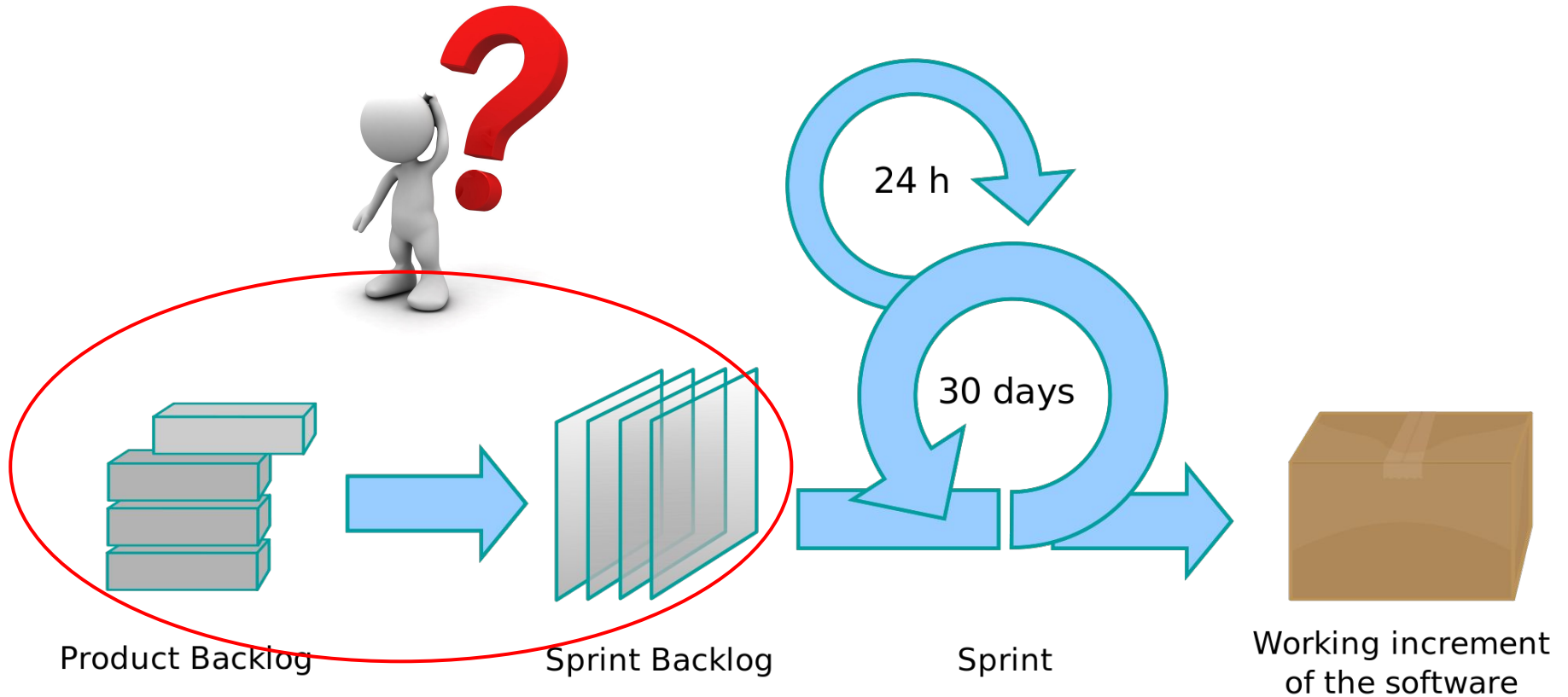
A time-boxed model:

- Each Sprint (time box): **max 30 days**
 - Fixed number of tasks for each Sprint
 - Daily Scrum meeting: 15 min max
 - Each sprint results in a
 - Sprint review (product demo): **0.5-1 hour**
 - Sprint retrospective (post-mortem): **1-3 hours**
- 

Scrum: overview



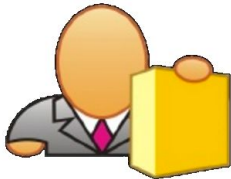
Scrum: overview



Prioritization:

- **Must have vs. Should have vs. Could have vs. Won't have**

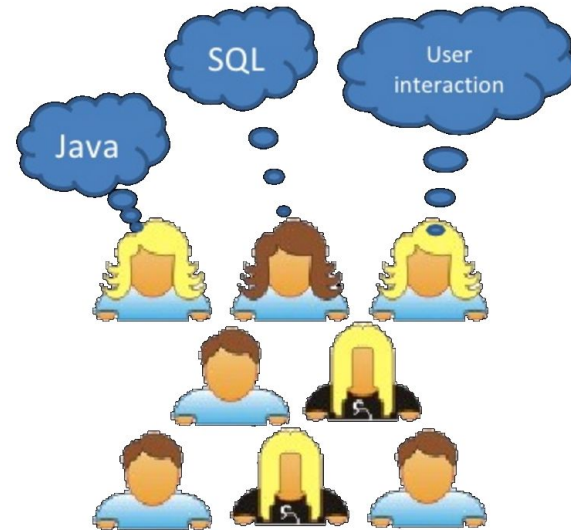
Scrum: roles



Product owner
(Customer)

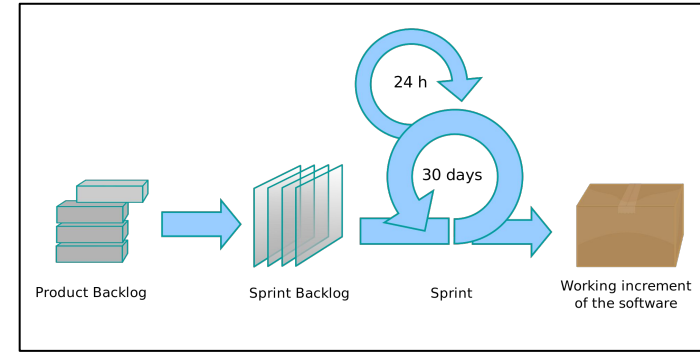
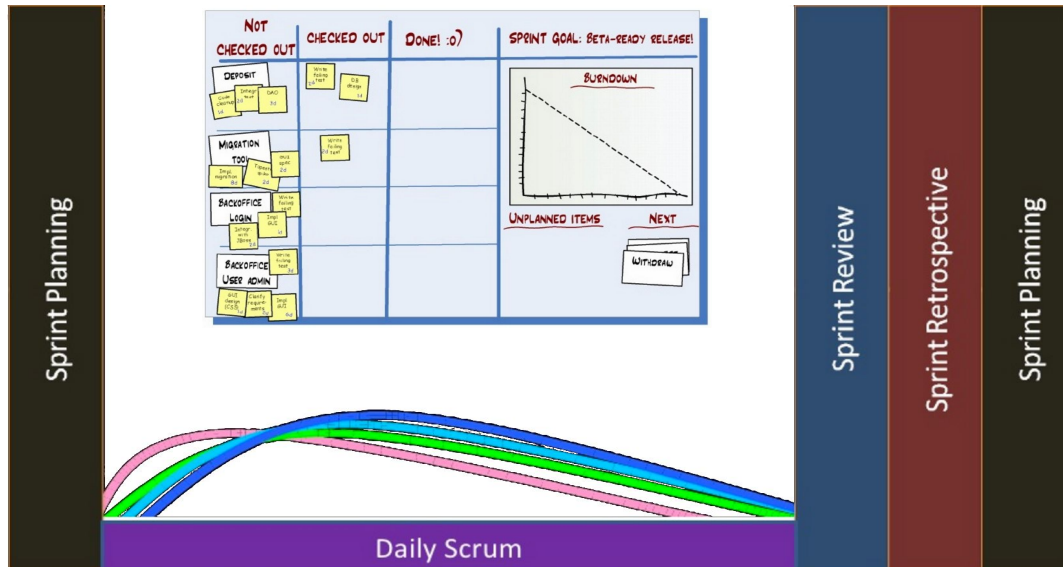


Scrum master
(Manager/Moderator)



Scrum team
(Tech experts)

Scrum: activities and planning



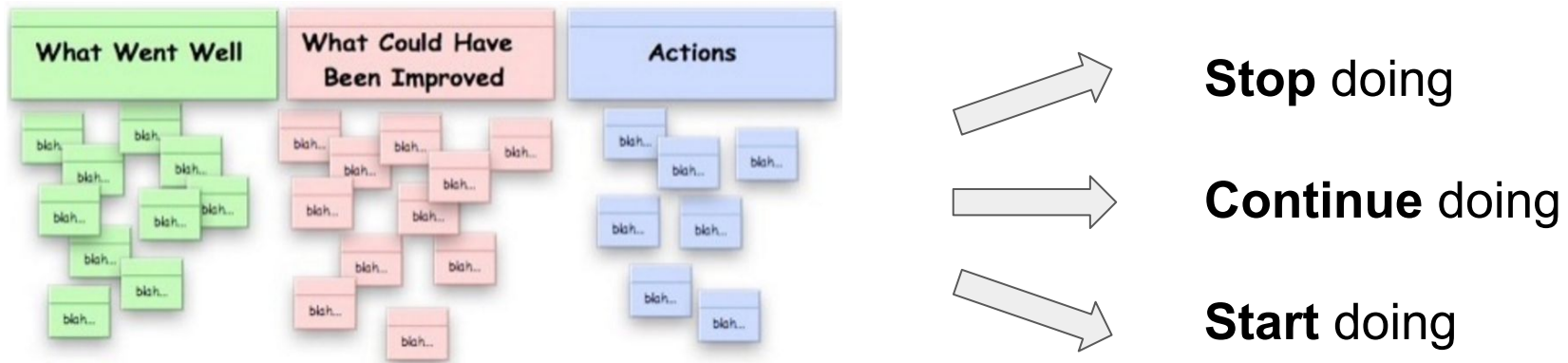
Daily scrum meeting (15min):

- What did I do since the last meeting?
- Any obstacles or blocking issues?
- What will I do until the next meeting?

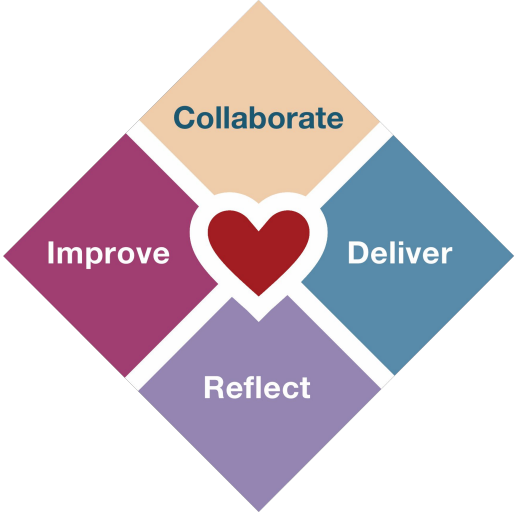
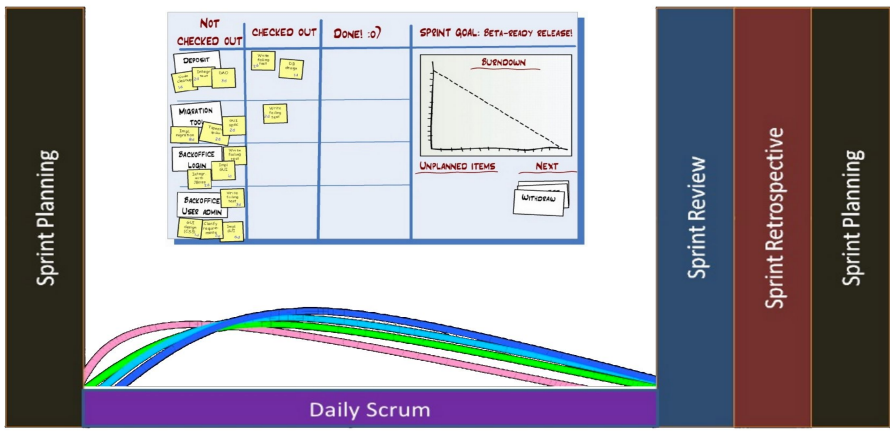
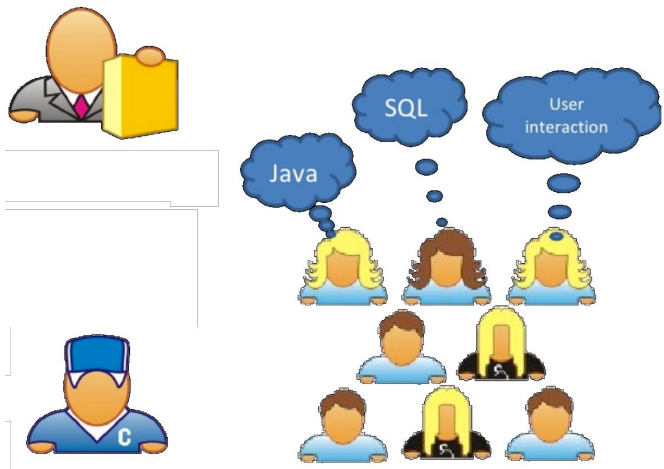
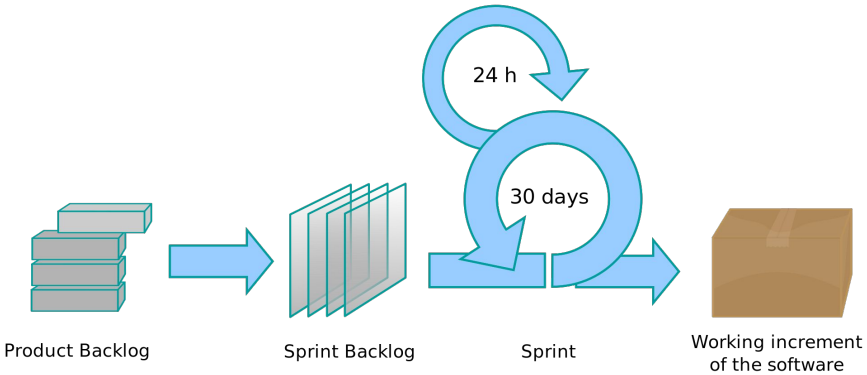
Scrum: sprint retrospective

Who and what?

- Product owner, scrum master, and scrum team.
- Reflect, change, improve



Scrum: summary





Scrum: discussion

Will you use **Scrum** for your project?

- **Yes** (describe why)
- **A variant** (describe your variant)
- **Need more infos** (state 1-3 specific questions)
- **No** (describe why not)

Working in Teams

Working in teams is great



Seriously, working in teams can be great!

Benefits

- Attack bigger problems in a short period of time
- Utilize the collective experience of everyone

Risks

- Communication and coordination issues
- Lack of planning, reflection, improvement
- Conflict or mistrust between team members

Big questions

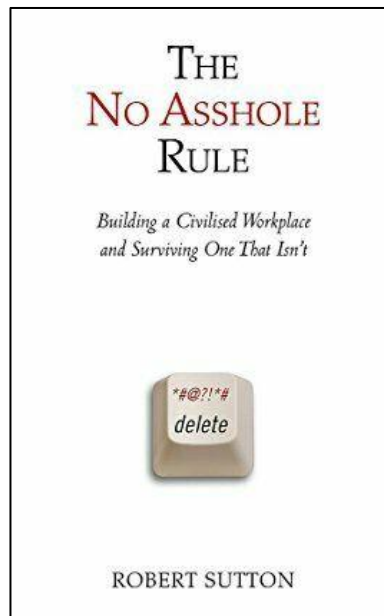
- **Communication:** How will everyone communicate?
- **Decisions:** How will your team make decisions?
- **Structure:** How do you **divide** your team **into subgroups**?

Big questions

- **Communication:** How will everyone communicate?
- **Decisions:** How will your team make decisions?
- **Structure:** How do you **divide** your team **into subgroups**?

Communication: powerful but maybe costly

- **Communication** requirements increase with increasing numbers of people (**everybody to everybody: quadratic cost**)
- Every attempt to communicate is a **chance to miscommunicate**
- **Not communicating will guarantee miscommunication**



Communication: example

"Hey X, I was wondering whether you finished the Y feature you were assigned? Since we were late on some features last time, I thought I'd check. When you have time, can you please tell me when Y is done. Thanks, Z."

What do you think about this email?

Big questions

- **Communication:** How will everyone communicate?
- **Decisions:** How will your team make decisions?
- **Structure:** How do you **divide** your team **into subgroups**?

Leadership and high-impact decisions

Who makes important product-wide decisions?

- One person?
- All by unanimous consent?
- Other options?
- Is this an **unspoken** or an **explicit** agreement?

Making decisions

- Delegate to subteams when possible
- Let everyone give their input (even if some is off-track)
- Write down pros/cons of alternatives
 - Evaluate cost/benefit/risks
 - How long will it take? How much to learn? etc.
- Have a clear procedure for resolving disagreement
 - Strive for consensus, but if it cannot be achieved, ...
 - Majority vote and PM decides on a tie, etc.
- Pareto: find 20% of work that solves 80% of a problem
 - Know what the real problem is!
- Document, Plan, Prioritize

Most importantly: compromise, compromise, compromise

Big questions

- **Communication:** How will everyone communicate?
- **Decisions:** How will your team make decisions?
- **Structure:** How do you **divide** your team **into subgroups**?

Common SW team responsibilities

These following could be all different team members, or some members could span multiple roles:

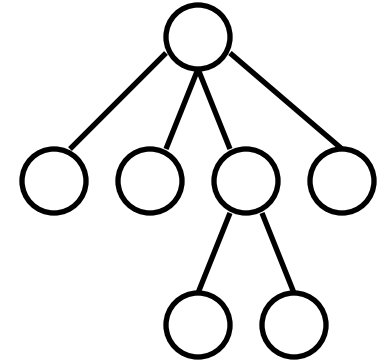
- Project management
- Functional management
- Designers/architects
- Developers: programmers, testers, integrators
- Lead developer (“tech lead”)

Key: Identify and stress roles and responsibilities

Team structure models

Dominion model

- Pros:
 - clear chain of responsibility
 - people are used to it
- Cons:
 - single point of failure at the top
 - little or no sense of ownership by everyone



Communion model

- Pros:
 - a community of leaders, each in their own domain
 - inherent sense of ownership
- Cons:
 - miscommunication, competing visions, dropped responsibilities
 - many points of partial failure

