Software Management Process

Thierry Sans

Agile Principles

- Short development cycles
  for fast **working product releases**
- **Self-organizing** teams (more than strict processes)
- Rapid **response to changes**
- Perpetual **client collaboration**

Product Backlog (so far)

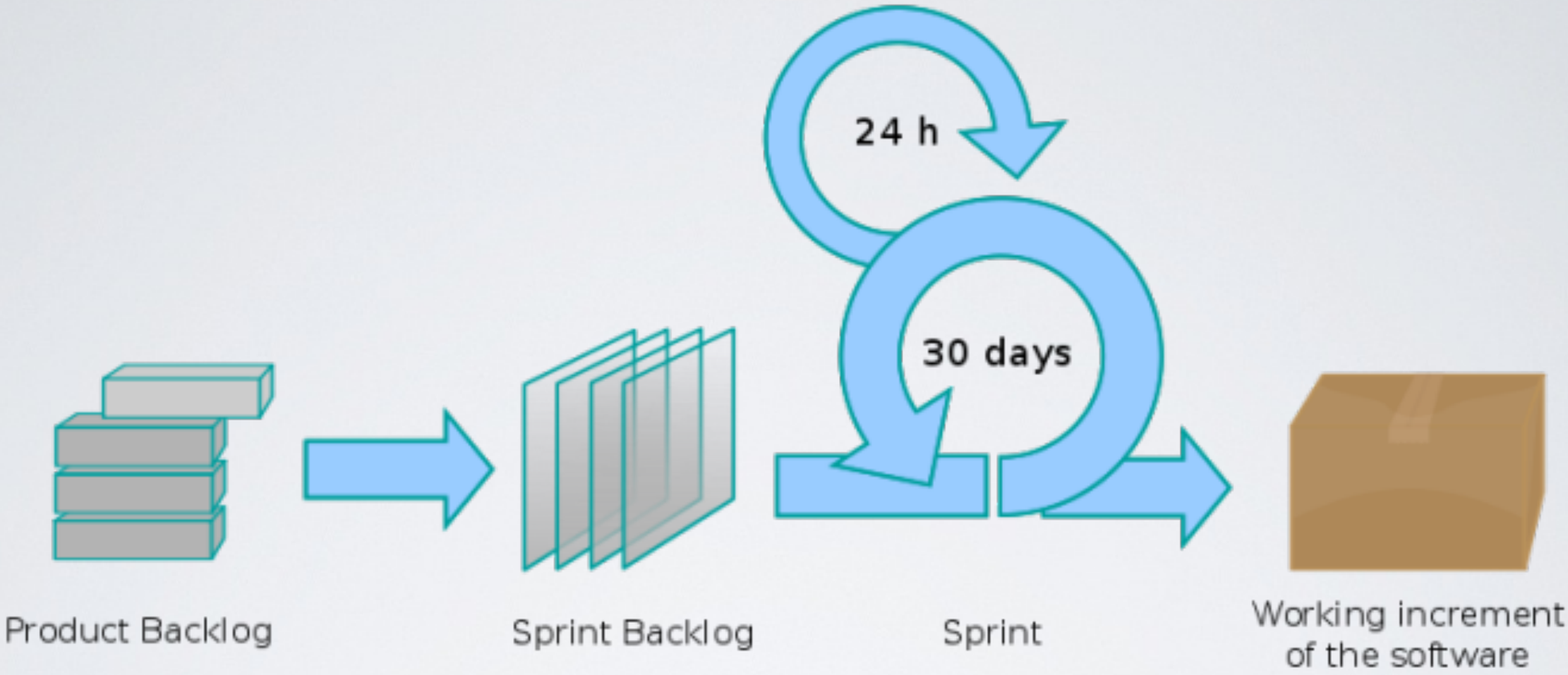
- **Personas**
  knowing who the users are in terms of behaviours and goals towards the software

- **User Stories**
  identifying the software features that will match the users' behaviours and goals

This week's new concepts

- **Sprints** : defining development cycles
- **Scrums** : defining and managing the workflow
- **Sprint backlog :** documentation about the project management

Sprint


24 h
30 days
Product Backlog
Sprint Backlog
Sprint
Working increment
of the software

Sprint = Development Cycle

- Before - **Sprint Plan**
  as a team, decompose user stories into tasks, and assign tasks to developers (see later in the presentation)

- During - **Sprint Execution**
  individually, developers executes the tasks

- After - **Sprint Delivery**
  as a team, deliver a product release for the completed user stories

Sprint Backlog

Users stories, tasks, story points and dependencies

User story 1 : 8 story points

As a user, I want to create my user's profile

- **Task 1 :** 1 story point
  setup a database and create a database schema for storing user's profile
  dependency : none
- **Task 2 :** 2 story points
  implement an API function to add a user's profile
  dependency : T1
- **Task 3 :** 3 story points
  design UI form to enter user's profile
  dependency : none
- **Task 4 :** 2 story points
  connect UI with the API
  dependency : T2 & T3

Sprint Length, Unit and Velocity

- **Sprint Length**
  duration of a sprint (usually 1- 4 weeks)

- **Sprint Unit**
  number of story points targeted to be completed per developer hour (usually 1 story points = 1 developer hour)

- **Sprint Velocity**
  number of story points targeted to be completed for the entire sprint

Recommendations for your project

- **Sprint length**
  1 week (maximum)

- **Sprint unit**
  1 story point = 1 h

- **Sprint velocity**
  4/5 dev * 6-8 h/week = 24-40 story points / week

Scrum

Scrum = Designing / Planning / Tracking Progress

Scrum Players

- **Team**
  The implementers of the product
- **Scrum Master**
  Behaves somewhat like a team lead or project manager, in working to resolve issues blocking team progress
- **Product Owner**
  Speaks for the customer
- **Users**
  customers or consumers of the product
- **Stakeholders**
  Project sponsors – may be customers or company management or both
- **Managers**
  Keep the team's organization running smoothly

Sprint Scrum = problem solving and organization

Review previous sprint

- Measure progress made
- Review the team work and capture the lessons learned
- Review the product and assess overall progress towards completion

Planning next sprint

- Re-work the product backlog
- Design the product and defines tasks
- Build the sprint backlog

(15 minutes) Daily Scrum = progress report

Each member should answer these three questions:

1. What have you completed (relative to the Backlog) since the last daily scrum meeting?
2. What got (or is) in your way to completing this work?
3. What will you do between now and the next scrum meeting?

➡ **Commitment in front of peers**

➡ **Attempt to remove barriers**

Recommendations for your project

- **Sprint Scrum**
  At least 1h once week

- **Daily Scrum**
  At least 15 minutes every 2 days

Sprint Planning

Step 1 - Fleshing out user stories

For each user story

- Improve the description by adding specific details
- Draw screen sketches (*wireframing*)
- Make technological choices
- Design the structure of the code
- Define testing and validation strategy

Step 2 - Building the sprint backlog

1. Decompose user stories into tasks
2. For each task,
   - describe it with a title and as many details as possible
   - estimate the number of story points
     either 1, 2, 3, 5, 8, 13, 21, 34, 55 story points
   - identify dependencies

Step 3 - Task allocation

1. Allocate task to developers
2. Build the provisional burndown chart

Run, Review, Reflect, Repeat

Our software development process so far

Product backlog

- Personas
- User Stories
- Tasks (with brief description, story points, dependencies and details)

Sprint backlogs - for each sprint

- Sprint Plan
- Provisional burndown chart

Tracking progress

Sprint backlogs - for each sprint

- Sprint Plan
- Provisional Burndown Chart

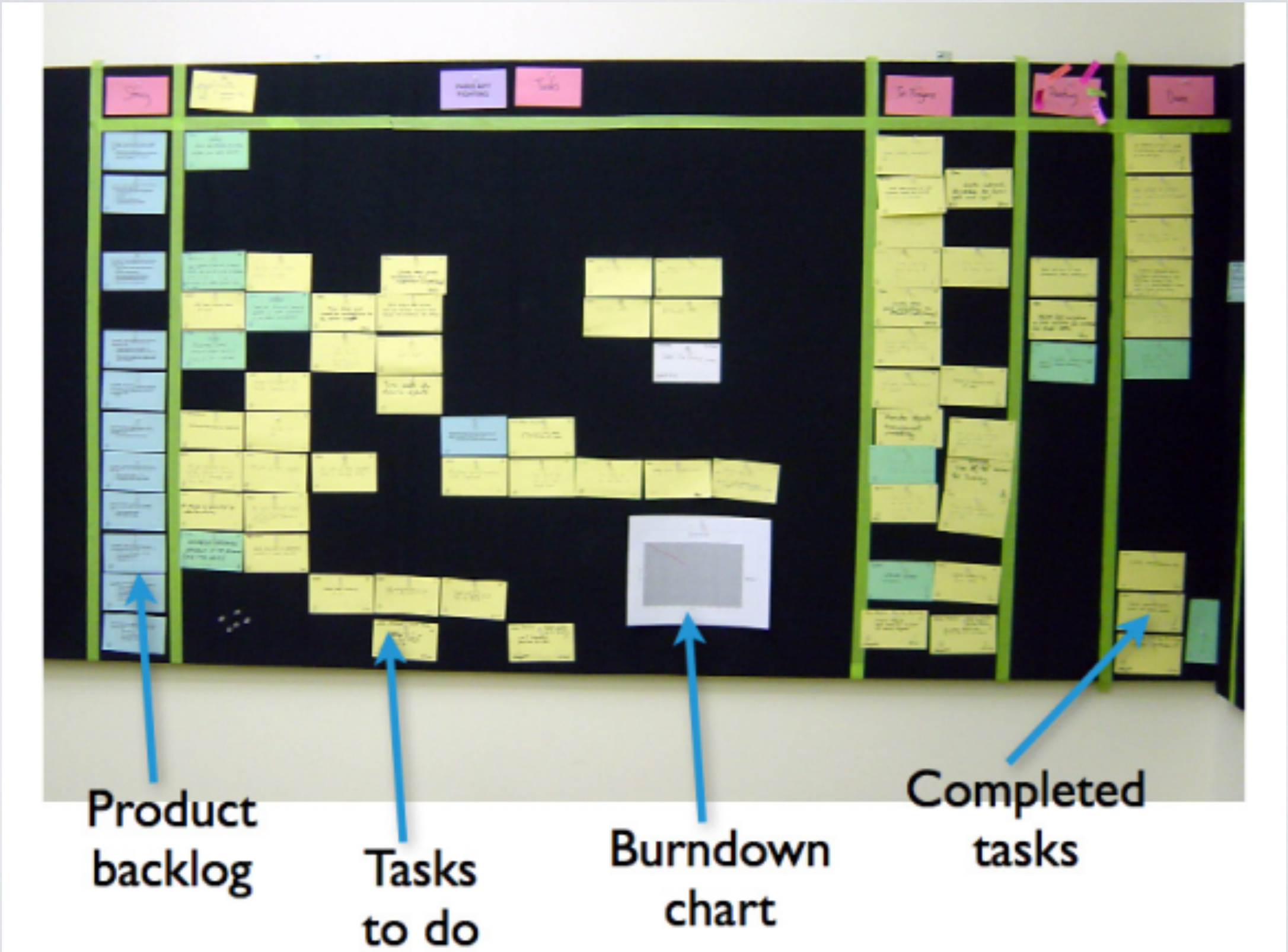
Planning **before** the sprint

- **Taskboard (a.k.a the *Kanban*)**
- **Sprint Execution Report**
- **Actual Burndown chart**

Tracking progress **during** the sprint and **report** after the sprint

Run

The taskboard (a.k.a the *Kanban*)



Sprint report and burndown chart

- See **`execution-exercice.pdf`**

Review and Reflect

Review

- What data do we look at?
- What do we learn from these data?

Reflect

- How to assess what we did well?
- How to assess what we not do well?
- How to identify what we will do better in our next print?

Outcome

- Revaluate tasks that have not been completed
- Review and possibly amend the product backlog
- Plan the next sprint

and finally repeat!