COMP1531

2.3 - Agile & Basic Project Management

What is agile?

Is a hotdog a sandwich?

agilemanifesto.org

https://agilemanifesto.org/

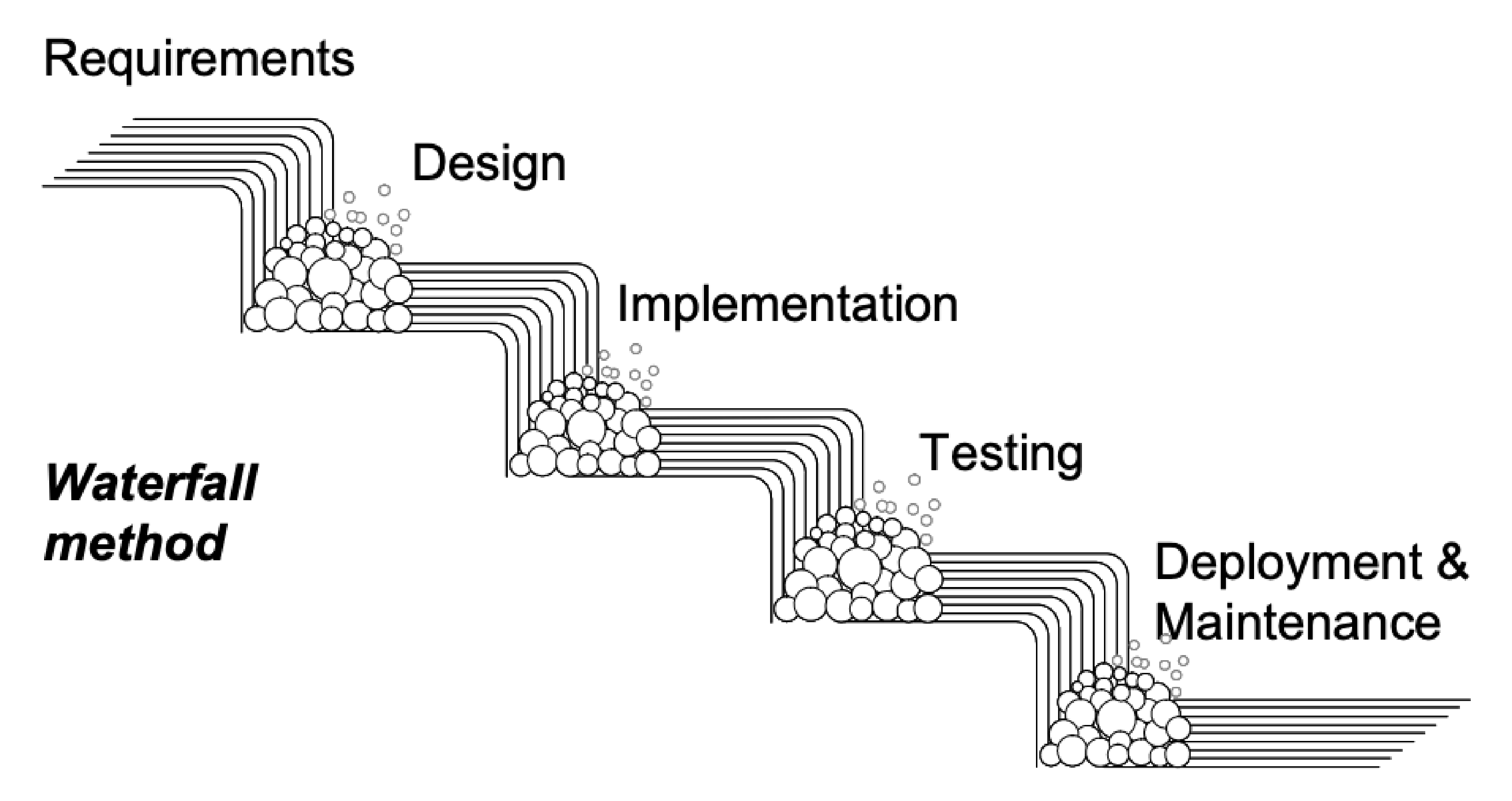
agilemanifesto.org/principles.html

https://agilemanifesto.org/principles.html

Yeah, but what is it really?

- Philosophy
- Practices
- Processes
- A cultural movement?

A brief history lesson



History is a lie

- "Waterfall" has never been proposed as a viable software methodology
- Reference: http://www.idinews.com/waterfall.html

Defining features (that people usually agree on)

- Iterative and incremental
- Quick turnover
- Light on documentation

So what is agile good for?

- Your resume?
- Changing requirements
- Delivering software on time
- Your project?

Agile Practices

- Practices today, processes later on
- We will focus on the ones you will find most helpful in your project

Standups

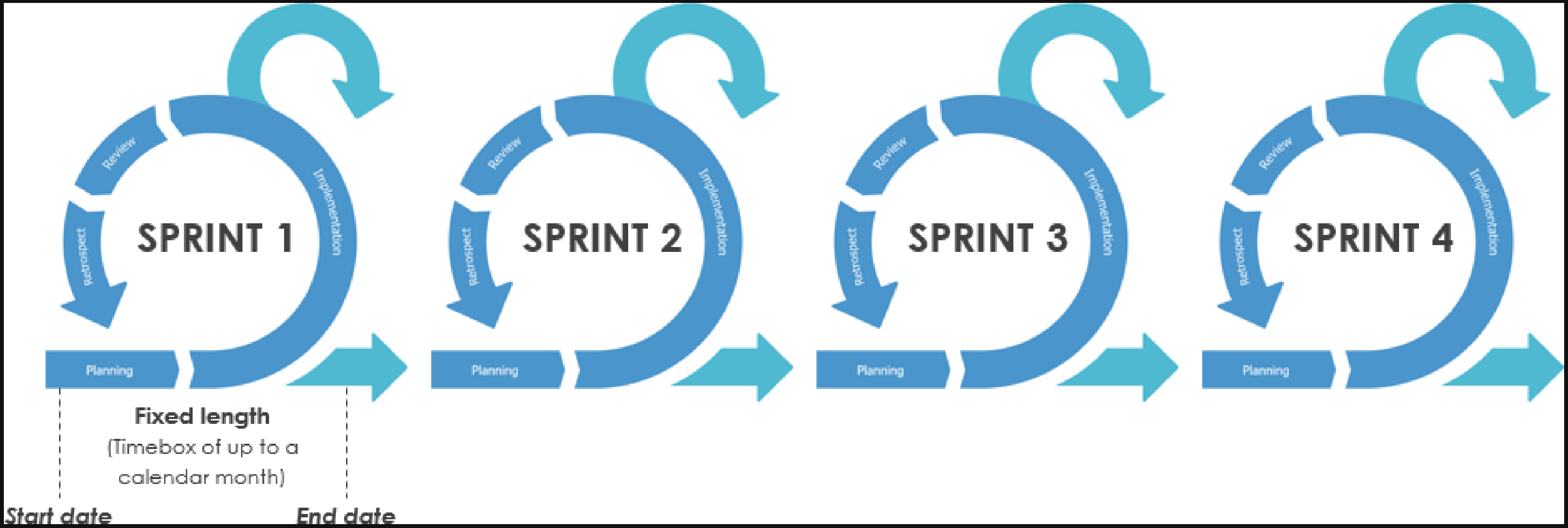
- Frequent (often daily) **short** progress update meetings
- Traditionally, everyone stands up
- Answer 3 key questions
    - What did I do?
    - What problems did I face?
    - What am I going to do?

Asynchronous Stand-ups

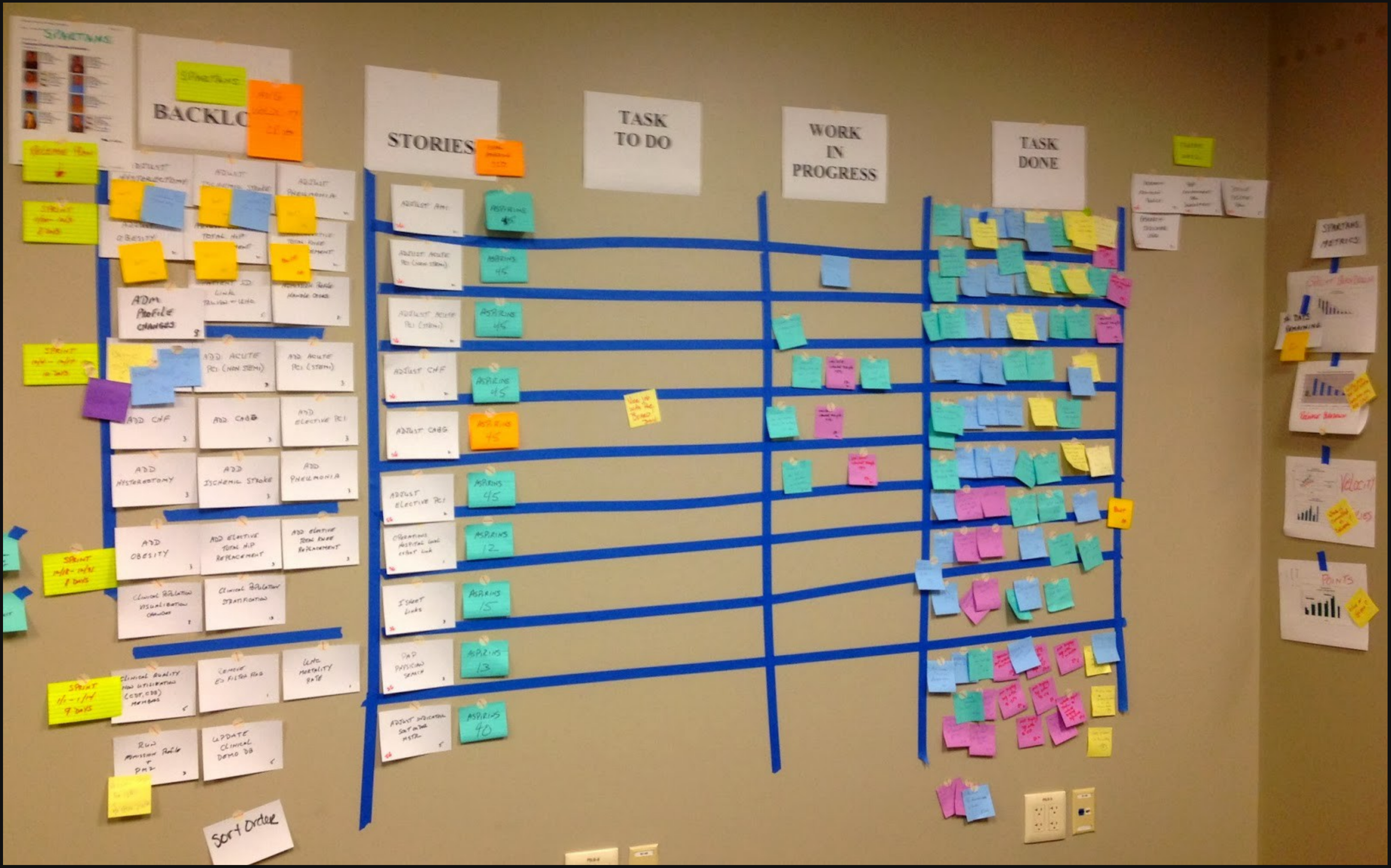
- A somewhat controversial topic
- Advantages
  - No need to find a suitable time for everyone
  - May work better for big teams
- Disadvantages
  - "Blockers" take longer to be addressed
  - Easy to forget to give an update
  - Less personal
  - Updates from others can be missed

Sprints/iterations

- Time is fixed, scope is flexible
- Plan only for the next sprint
- Typically have a release at the end of each sprint



Task Boards


BACKLOG
STORIES
TASK TO DO
WORK IN PROGRESS
TASK DONE
ADD CHF
ADD CABG
ADD HYSTERECTOMY
ADD ISCHEMIC STROKE
ADD PNEUMONIA
ADD OBESITY
ADM PROFILE CHANGES
ASPIRINS 45
Sort Order
Velocity
POINTS

Taskboards

- Available in GitLab
- Use them to store and track your progress on user stories
- You don't need many columns. E.g.
  - Backlog
  - Todo
  - Doing
  - Testing?
  - Closed/Done

Pair programming

- Two programmers, one computer, one keyboard
- Take it in turns to write code, but discuss it as they go
- Can result in better code quality
- Good for helping less experienced programmers learn *micro-techniques* from more experienced programmers

Test-Driven-Development (TDD)

- Writing tests *before* the implementation
- Write only enough code to make the next test pass
- Takes some practice
- We'll come back to this next week