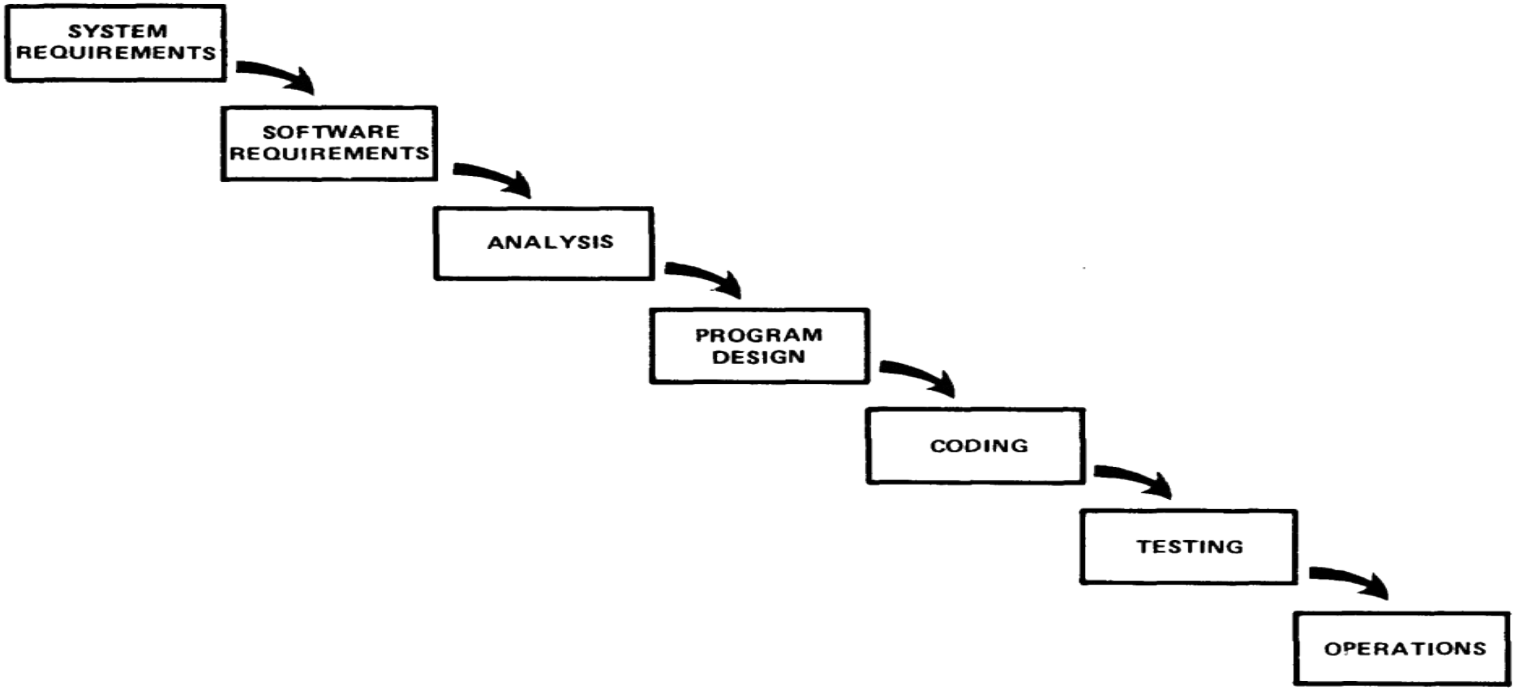

Scrum
In theory & practice

WHO?

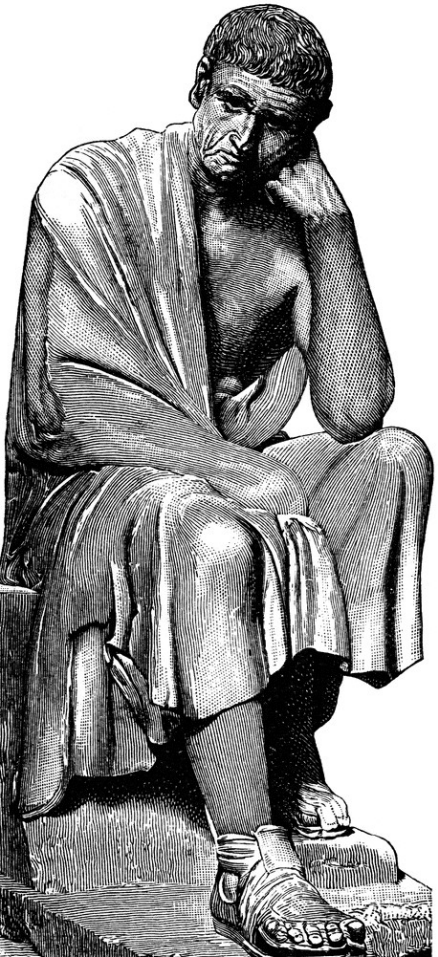
- Visa Parviainen
- B.Sc. Metropolia 2010
- CSM 2010
- Scrum projects last 4 years
- Project manager
- Eficode
- Software company
- Software and software development tools & services


Scrum is not for just software projects. You can use it for **everything.**

Waterfall...



Picture credit: Dr. Winston Royce



Waterfall?

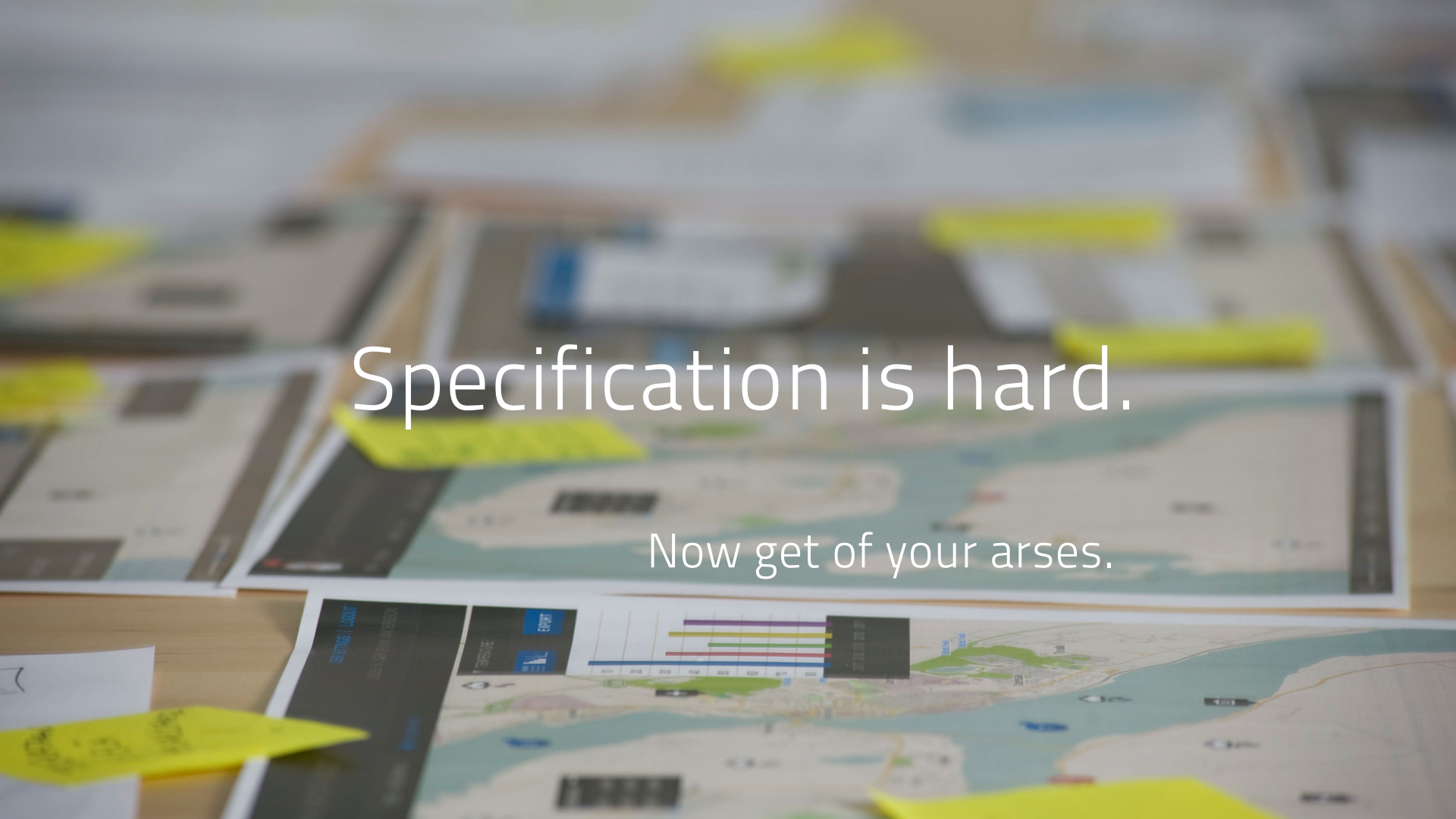
*"the implementation described above is risky and **invites failure**"*

Dr. Winston. W. Royce in *Managing the Development of Large Software Systems*

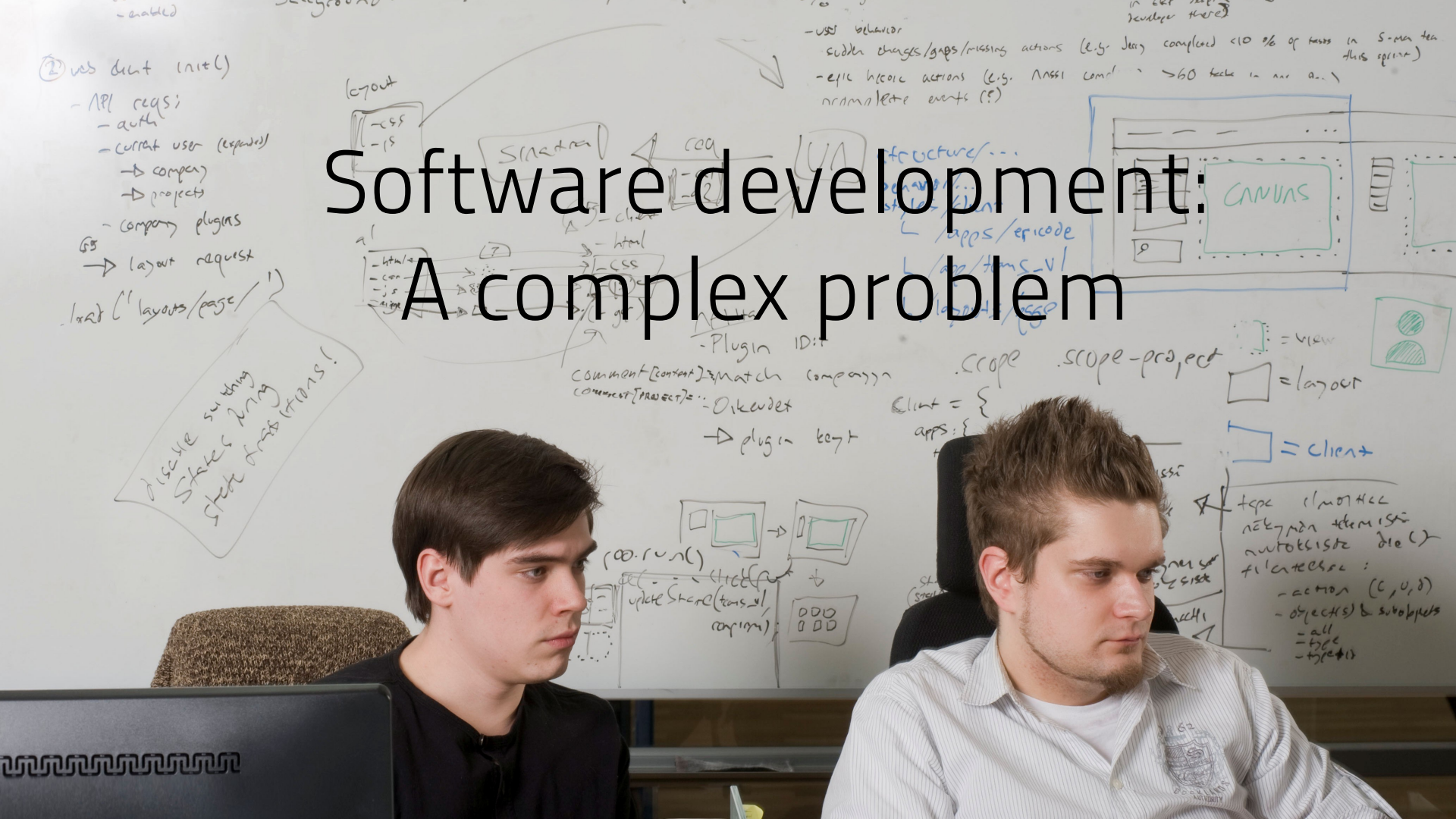
Who *is* this Winston fella?

*"He was **the first** who described the Waterfall model for software development"*

Wikipedia: Winston Royce


Specification is hard.
Now get of your arses.

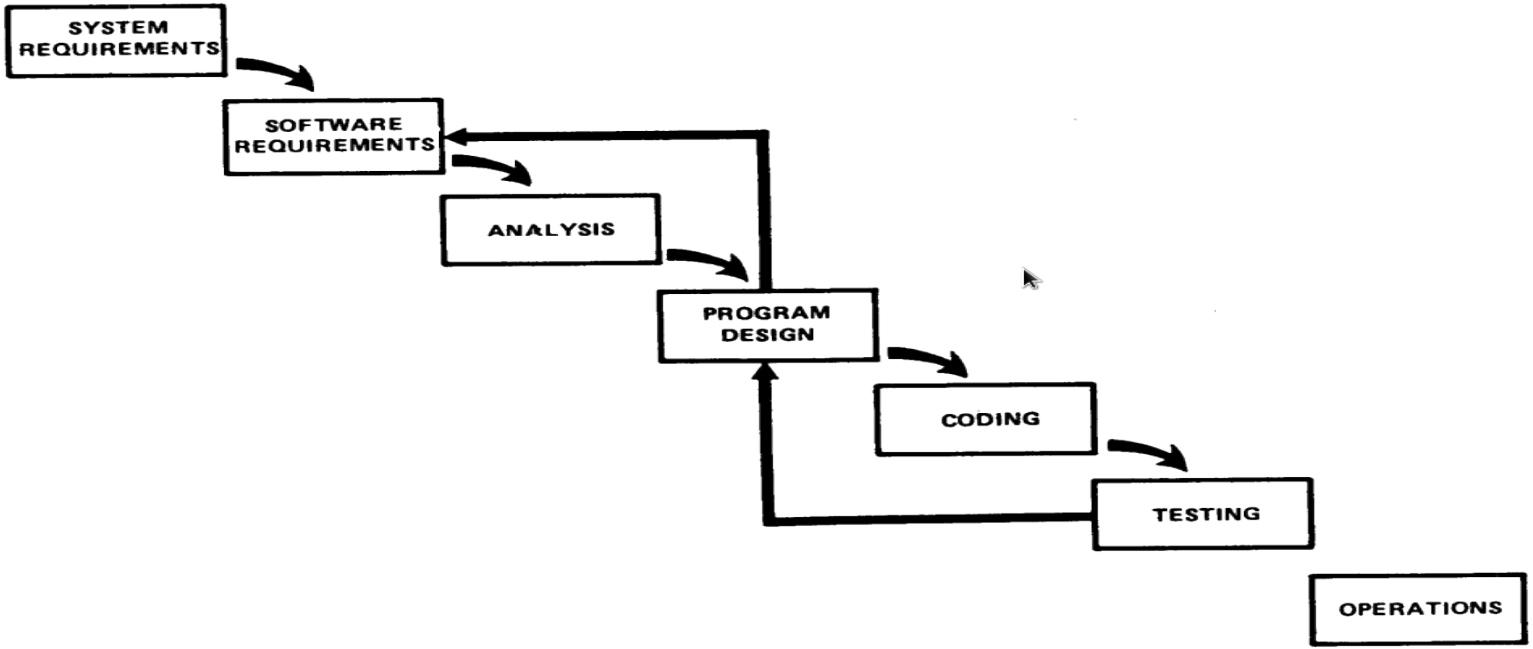

There are two kinds of problems in the world.
Simple and complex.
Chili con Carne vs. bouncer


Software development:
A complex problem

Iterations

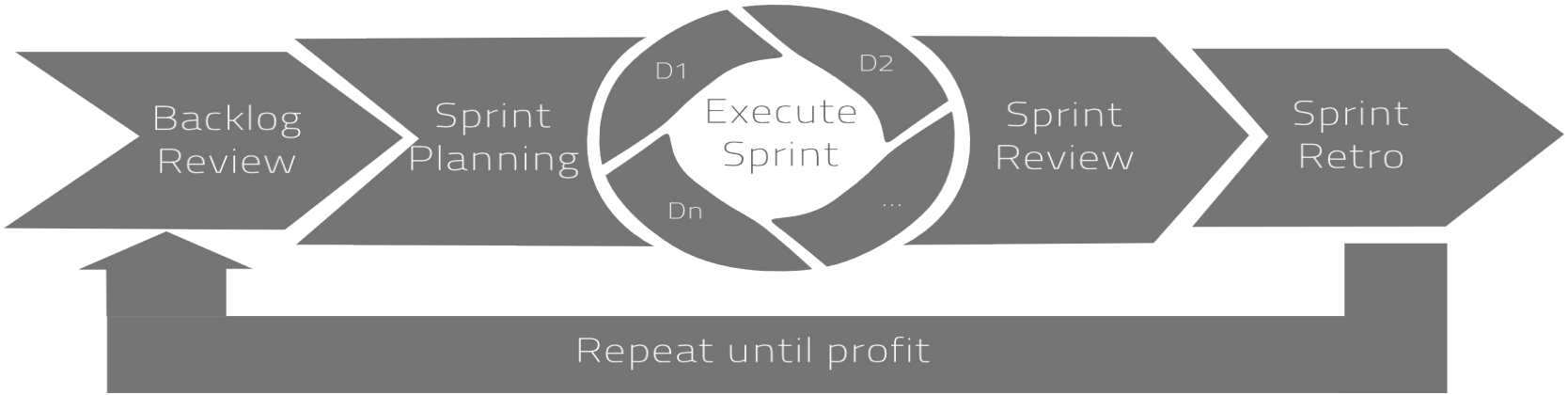
FTW!!!

Looping



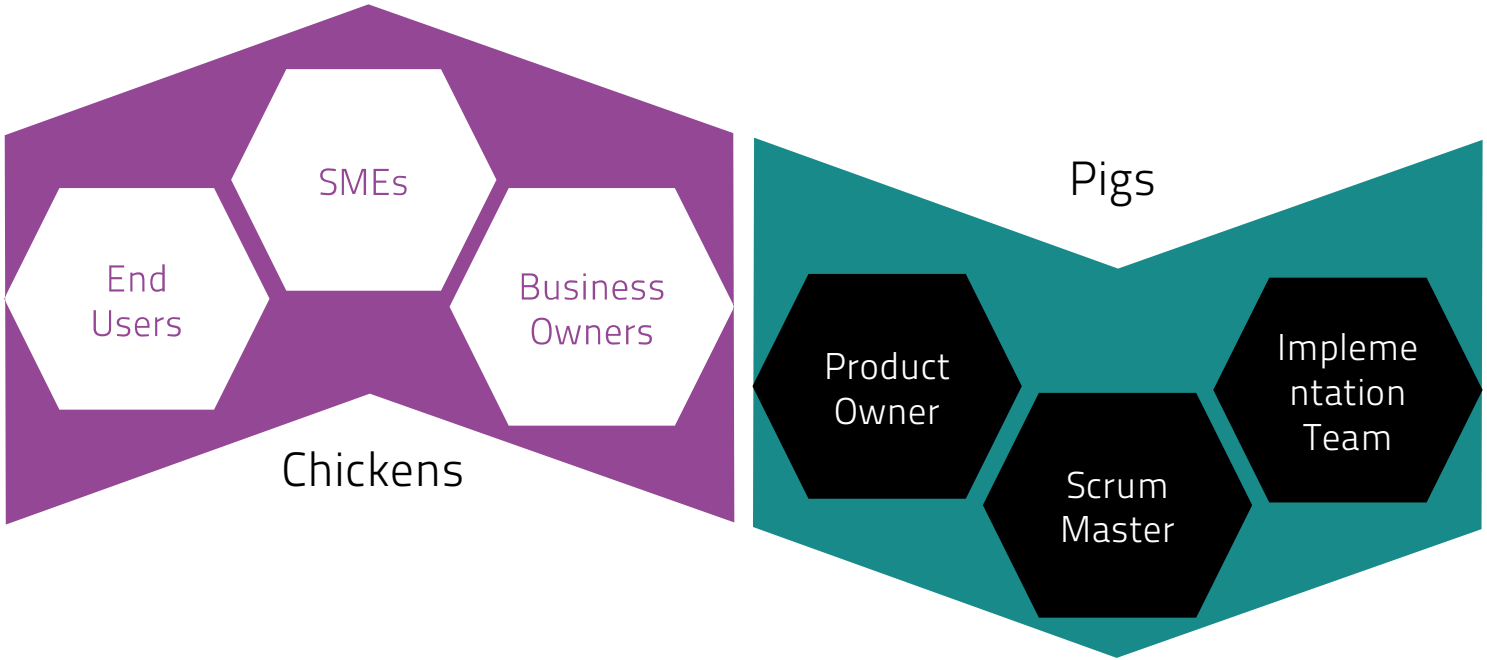
Picture credit: Dr. Winston Royce

Sprint = Short iteration



What you need...

Those Animals!




User story


Backlog = List of work to be done


Wall

Let's talk ready

When can we start working on a task?

What's done is done.

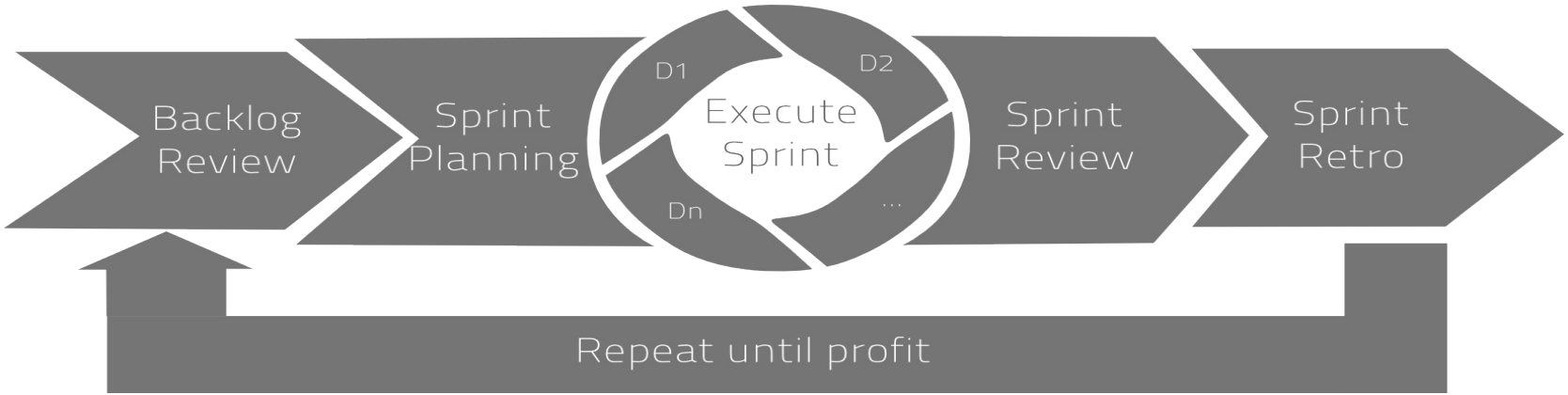
But how do you know it is?

Tools

- Version control (Git ftw. (SVN?))
  - Agree on working methods!!!
- Development environment
- Testing & demonstration environment
- CI-environment
- Workstations
- Information sharing: Wikis, issue management systems etc.

Meetings **prior** to sprint

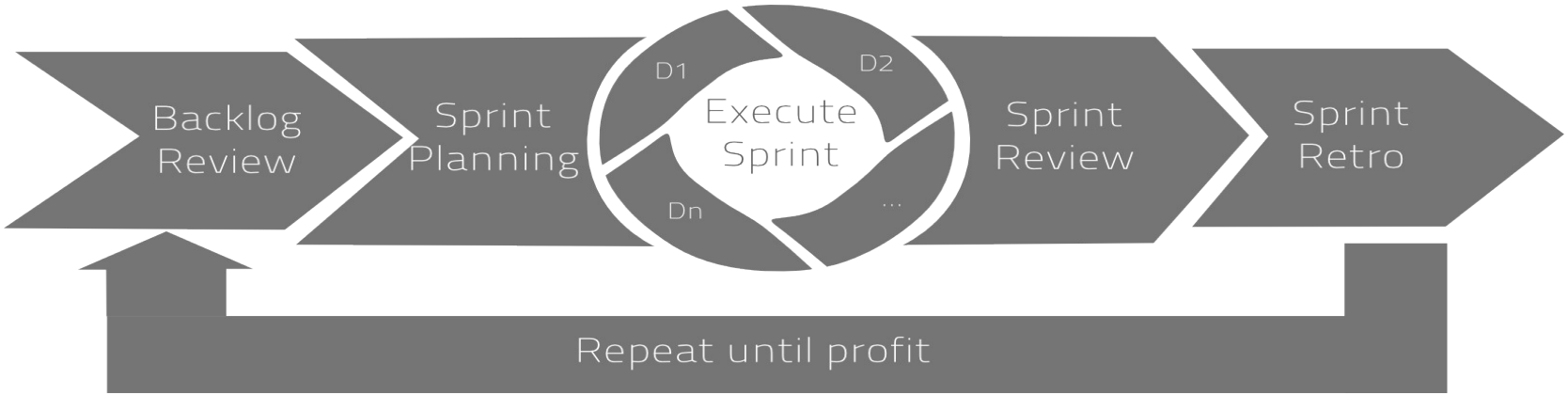
Short Iterations



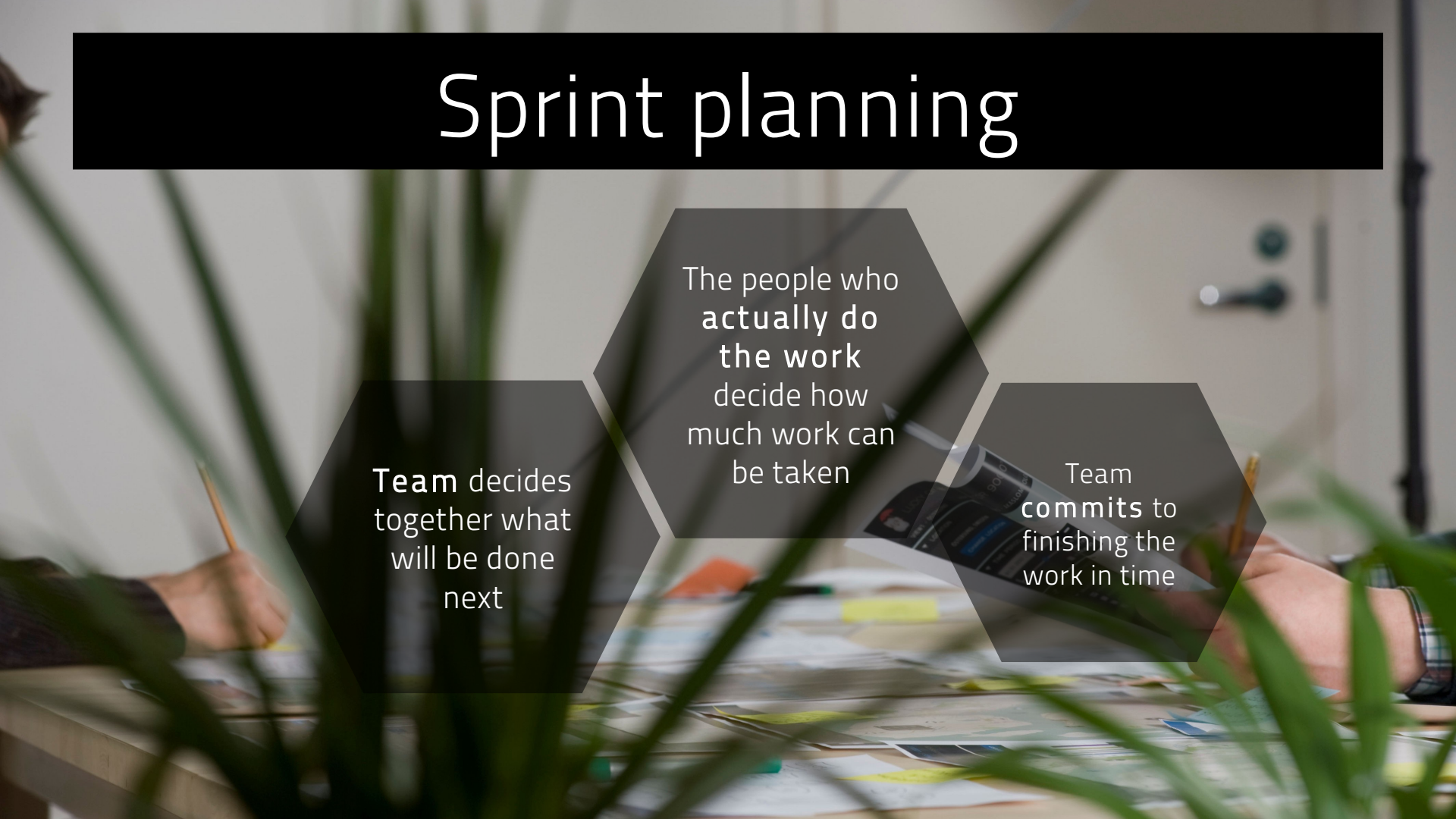
Backlog Review

- Product owner's own show
- Vision of product into user stories
  - Forming acceptance criteria
- Prioritization of existing user stories
  - Determines what happens next

Short Iterations



Sprint planning




Team
Declines to take tasks that are:
too vague
contain too many unknowns
too large to be done in one sprint

Meetings **during** the sprint

Daily Scrum

- Pigs speak, Chickens listen
- What did I do?
- What will I do next?
- Did I run into any trouble or impediments?
- Do I need any help?

Other useful meetings

- AD-HOC chats
  - Good way to create architecture or principles in agile projects
- Workshops
  - Let's brake the product (safety)
  - UI workshops (common agreement on functionality)
  - Architecture and refactoring workshops

Remember: Decide what meetings you want to have **before the sprint** and stick with it!

Meetings **after** the sprint

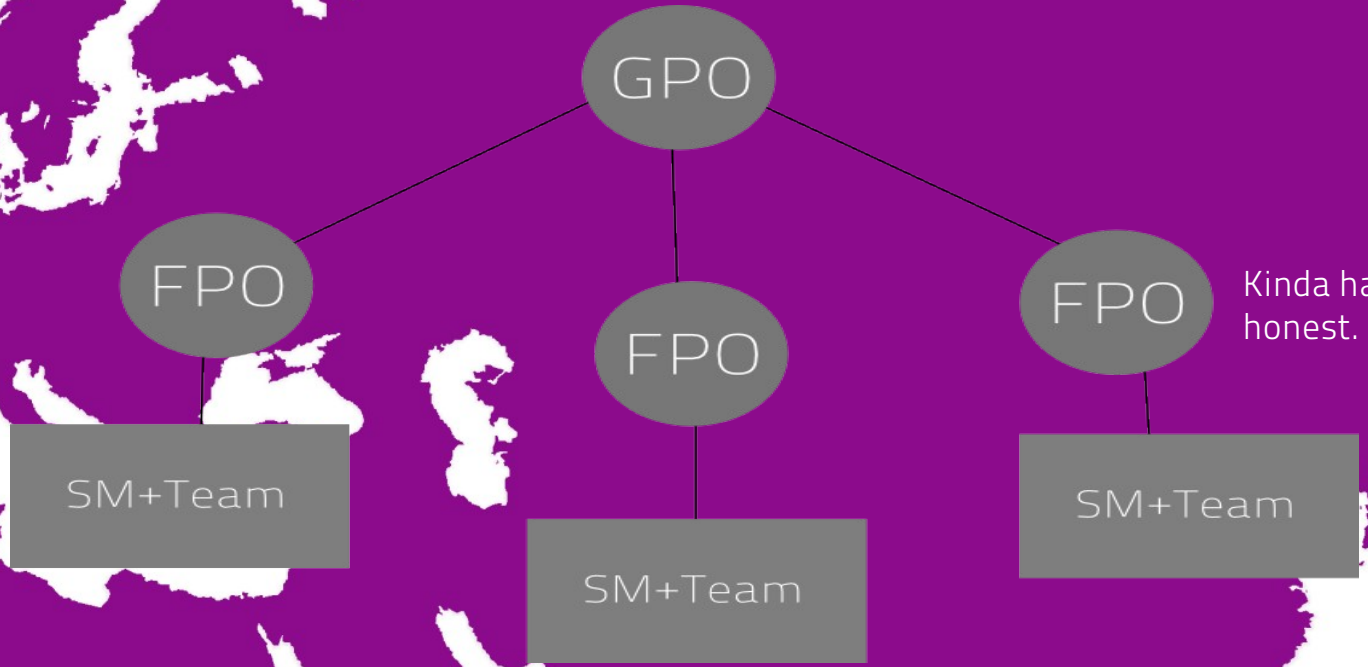
Sprint Review

- Show **for** the Product Owner/Business owners
  - What have we managed to complete
    - Team presents functionality to PO
  - How did we do?
    - Are they done right?
    - PO accepts/rejects the work done, or issues additional changes as new user stories.

Sprint Retrospective

- Team's own show
- What went well?
- What went badly?
- What could be improved?
- From last retrospective, which issues are still current?
  - Have all of them been worked on, if not, why?

Scaling



Kinda hard if I'm honest.

Practical tips

- Sprint 0 > 0!
- Estimations through story points = yes.
  - SM makes statistics after each sprint
  - Use statistics to estimate how much work for next sprint
- Using your sense in planning = allowed
  - Not too many overlapping stories
  - No stories with too many unmet dependencies

Practical tips 2

- TDD FTW!
  - Feels useless until you refactor the first time (and you will)
  - Good method for learning:
    - Senior writes test, junior codes, senior checks and gives feedback
- … Which reminds me: Code review FTW!
  - Implement-critique-fix -cycle


eficode


eficode


eficode


eficode