Software Engineering Primer for the Capstone Project

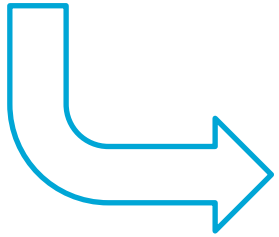
Carolin Brandt

TU Delft

Slides at:
https://carolin-brandt.de/publications/RE+PMforCapstone.pdf

Lecture Goal

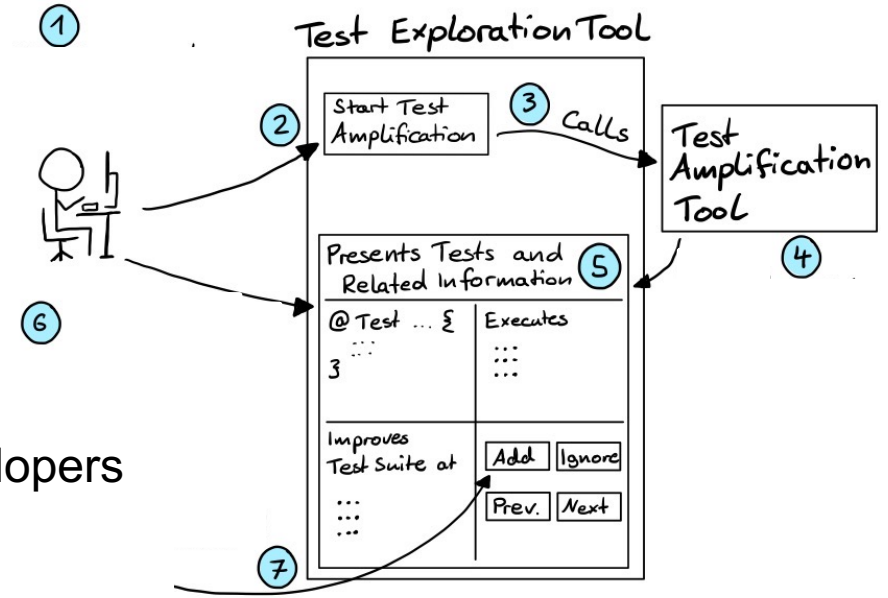
- Help you succeed in the capstone project
- From a team-that-writes-code-together perspective
- Based on scientific insights on **collaborative, large scale software development**



  - Software Engineering
  - Requirements Engineering
  - Project Management

Who am I

- Carolin Brandt
- last year PhD Student, Software Engineering
- Automated testing tools collaborating with software developers
- Passion: Software Quality



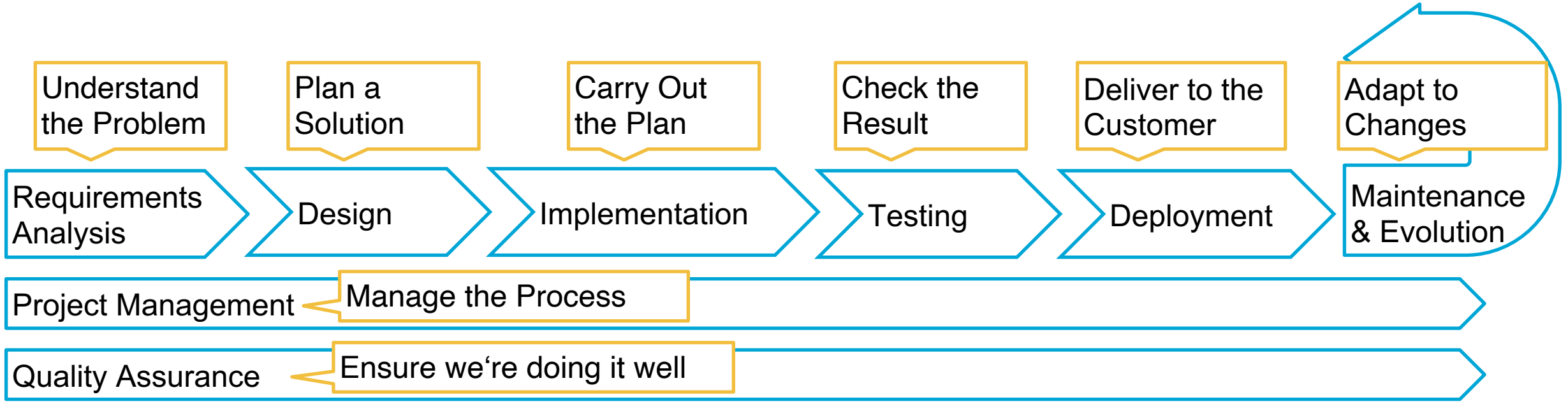
TU Delft

- https://carolin-brandt.de
- c.e.brandt@tudelft.nl

Simplified

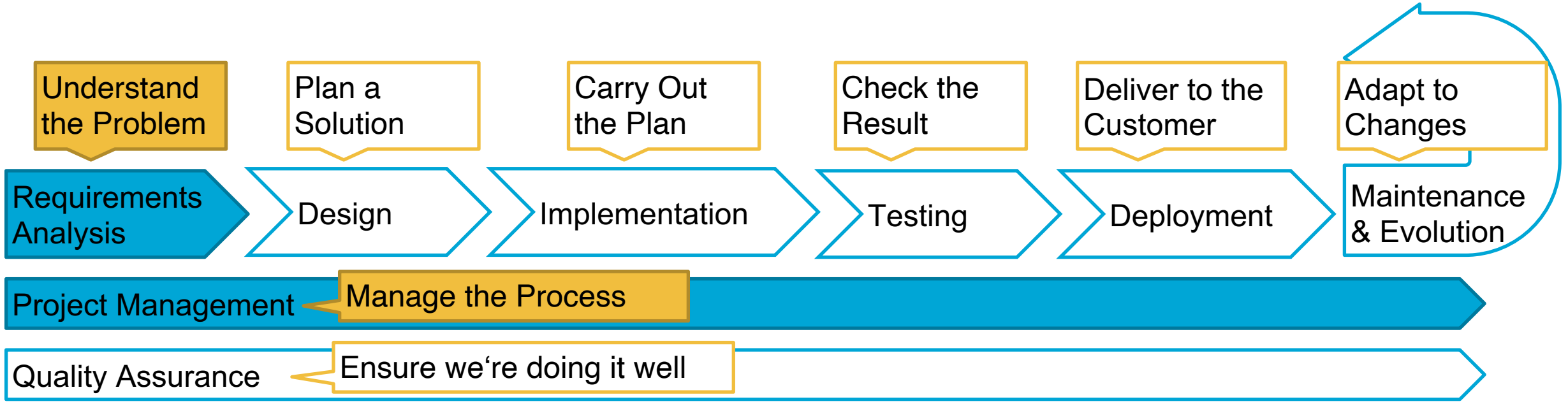
Software Engineering

- Build complex software systems in a timely manner and with high quality. [Pre05]



TU Delft

Focus for Today



TU Delft

Lecture Structure

First Hour

Requirements Engineering

Negotiation
Documentation & Quality

Q&A

Project Management

Scrum
How to have a meeting

Q&A

Second Hour

In Project Groups

Review of Initial Requirements
Preparation for Client Meeting

Q&A

TU Delft

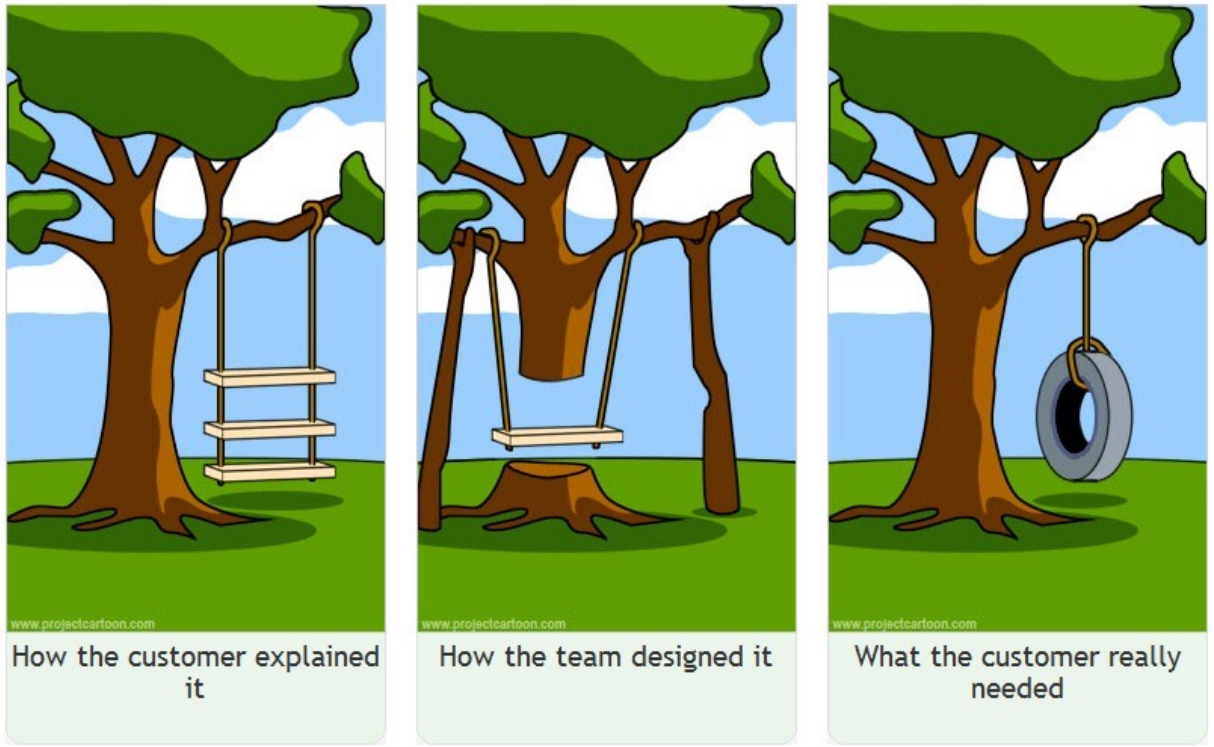
Requirements Engineering

Make sure you‘re solving the right problem

TU Delft

Why do we need requirements (engineering)?

- Agree on what will be done in the project
- Software: 1.000.000 possible solutions
- Create a shared understanding



TU Delft

Definition: Requirement

- Description on what the system should do:

  the services that it provides, → Functional

  and the constraints on its operation. → Non-Functional

  Product or Process: Usability, Performance, Ethics, Report

Capstone Examples:

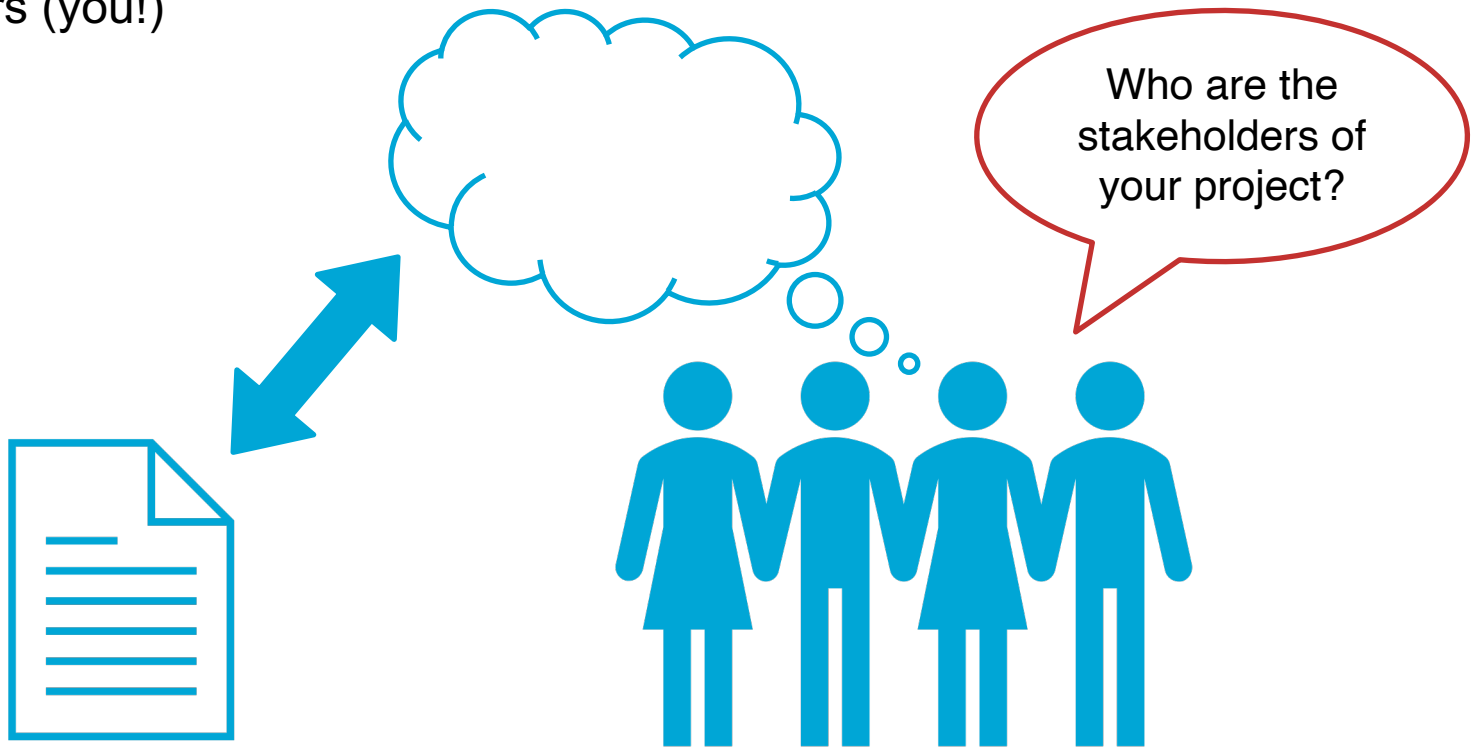
- Extract the social network structure in each city of the set
- Evaluate the performance of the clustering and classification algorithms

- Train the DECODE architecture on the TU Delft HPC cluster
- Get familiar with handling medical data (CT images, 3D dose distributions).

Requirements come from Stakeholders

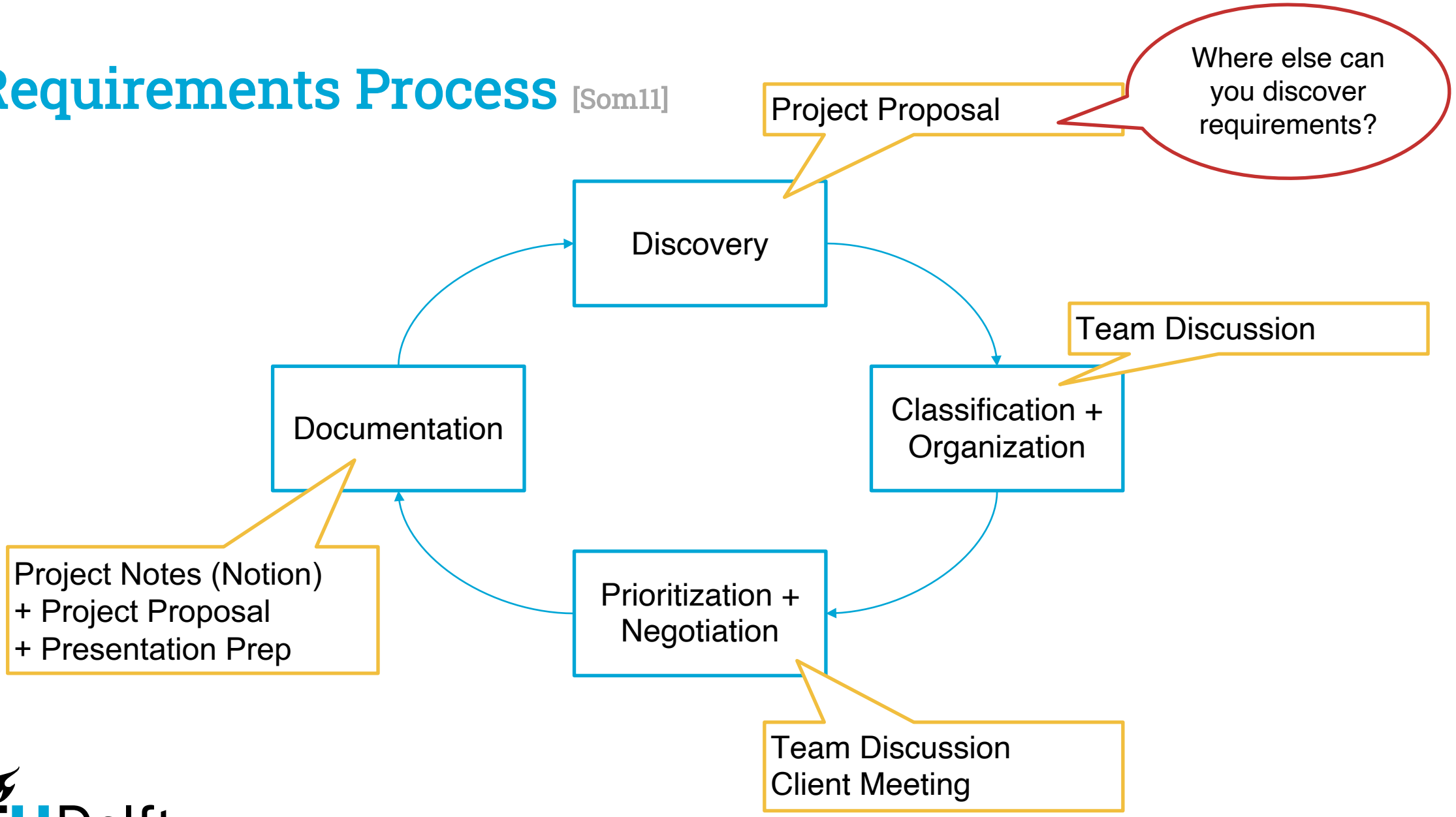
- Anyone who has a direct interest in or benefits from the system that is developed. [Pre05]
- Client, Users, Supervisors, Developers (you!)

- Talk to / Consider each one
- Be aware of conflicting needs



TU Delft

Requirements Process [Som11]



TU Delft

Documenting Requirements

- Make a template
- Specify the actor: “The system ...“ “The students ...“ “The client ...“

- Don‘t be afraid to write requirements on multiple levels / steps!

  “Evaluate performance...“
  → “Calculate metric x“
  → “Measure time needed for y“
  → “For algorithms a, b and c“

Note down
as much as possible
- for yourself
- as a group

Capstone Examples:

- Extract the social network structure in each city of the set
- Evaluate the performance of the clustering and classification algorithms

What questions
come up when
you read this?

TU Delft

Prioritizing Requirements

- MoSCoW Model
- **Must** have
  must be satisfied for project to not fail
- **Should** have
  important and valuable, but not essential
- **Could** have
  interesting enhancement, but only to be worked on if others are done
- **Won‘t** have
  explicitly excluded from this project (but might be interesting for the future)

How many % per category should you fulfill?

How do you decide the priority?

Why do we make could and won‘t explicit?

TU Delft

What is a Good Requirement?

ISO 29148

- Unambiguous
- Feasible
- Verifiable
- Necessary
- Traceable

The requirements should be ideally be SMART:
Specific,
Measureable,
Attainable,
Realistic,
Timely.

- All together:
  - Consistent
  - Complete

What could be improved in these requirements?

Capstone Examples:

- Must take clinical imaging data from various modalities as input (e.g. CT, MRI, echo)

- Must annotate meeting transcripts to create a training dataset

- Must reliably align two complete recordings corresponding to the same written character

TU Delft

Common Problems

- Missunderstanding: written requirements are the main result
- → **Information exchange and shared understanding** between developers and with client are the real value
- What the client says they want vs. what they need
- Stakeholder domain and vocabulary

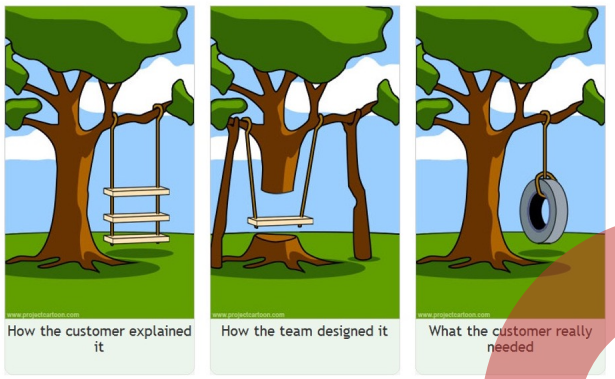
Negoitiation

- Client wishes vs. what you can deliver
- Initial agreement
- Potential re-negotiation

TU Delft

Why do we need requirements (engineering)?

- Agree on what will be done in the project
- Software: 1.000.000 possible solutions
- Create a shared understanding

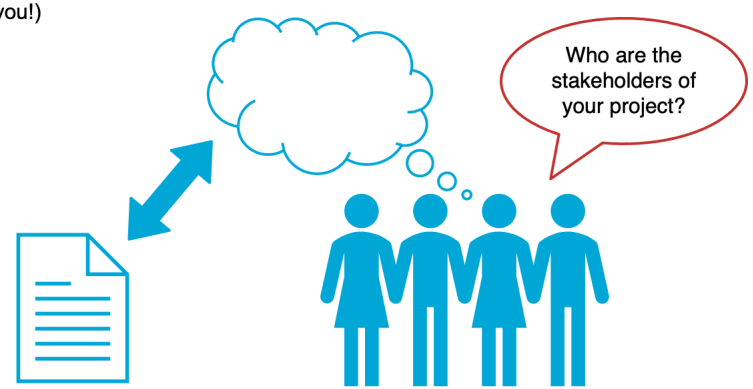


TUDelft

Requirements come from Stakeholders

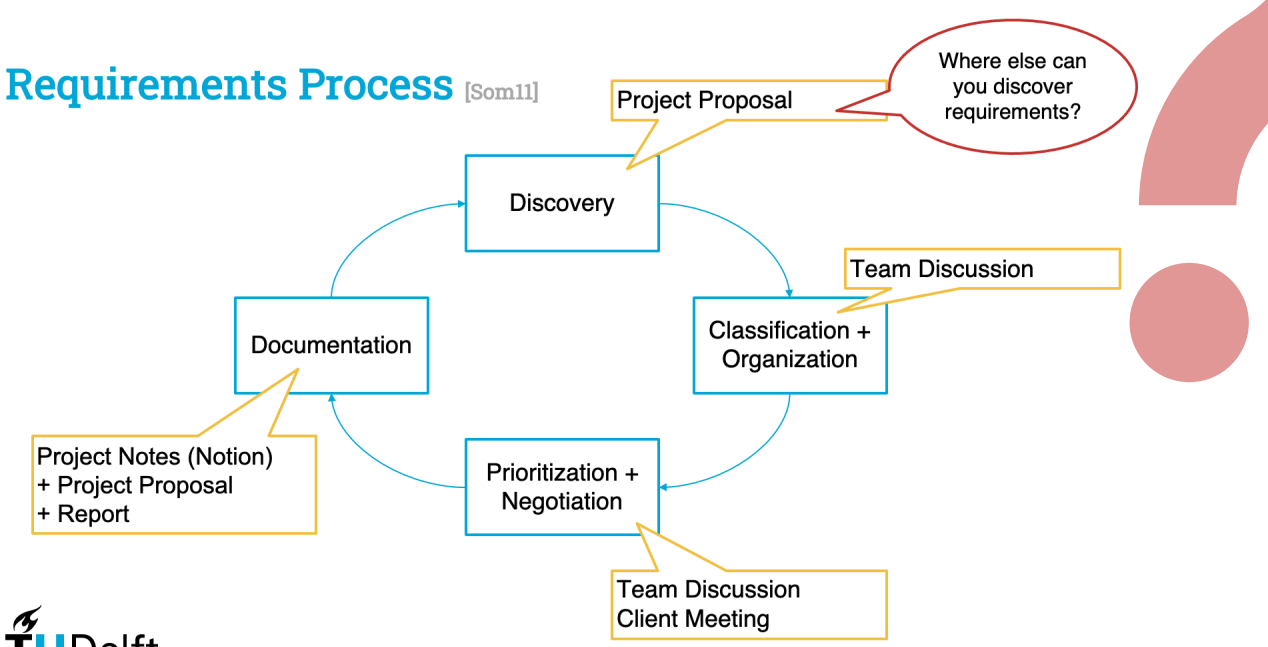
- Anyone who has a direct interest in or benefits from the system that is developed. [Pre05]
- Client, Users, Supervisors, Developers (you!)

- Talk to / Consider each one
- Be aware of conflicting needs



TUDelft

Requirements Process [Som11]



TUDelft

What is a Good Requirement?

ISO 29148

- Unambiguous
- Feasible
- Verifiable
- Necessary
- Traceable

The requirements should be ideally be SMART:
Specific,
Measureable,
Attainable,
Realistic,
Timely.

- All together:
  - Consistent
  - Complete

What could be improved in these requirements?

Capstone Examples:
- Must take clinical imaging data from various modalities as input (e.g. CT, MRI, echo)

- Must annotate meeting transcripts to create a training dataset

- Must reliably align two complete recordings corresponding to the same written character

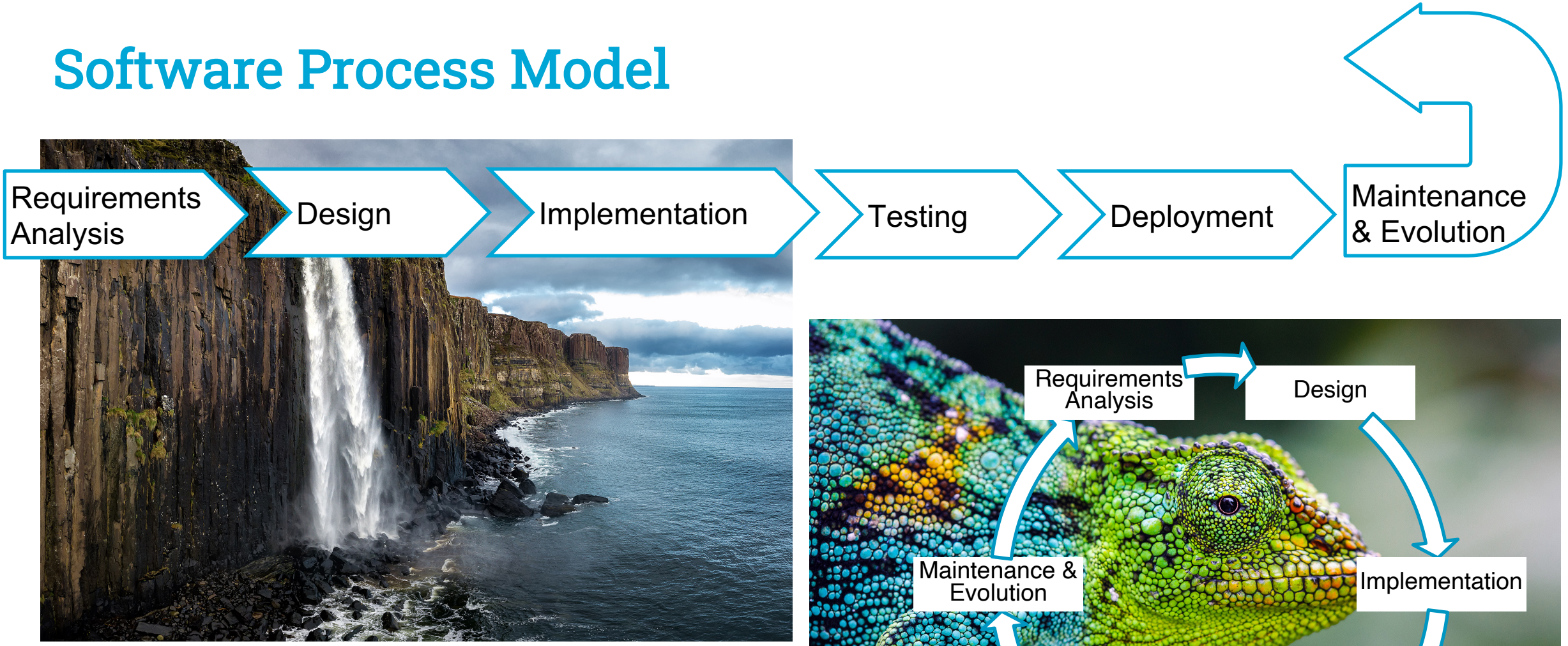
TUDelft

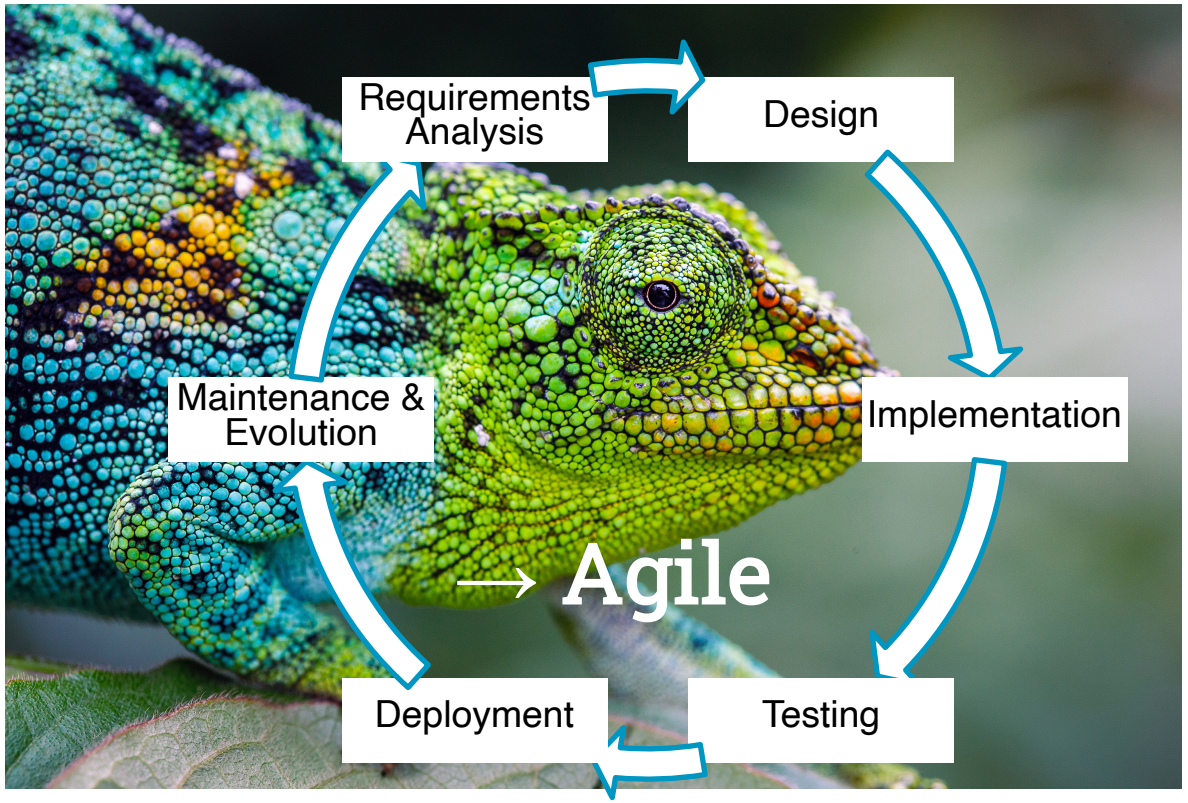
Project Management

How to organize developing together

TU Delft

Software Process Model



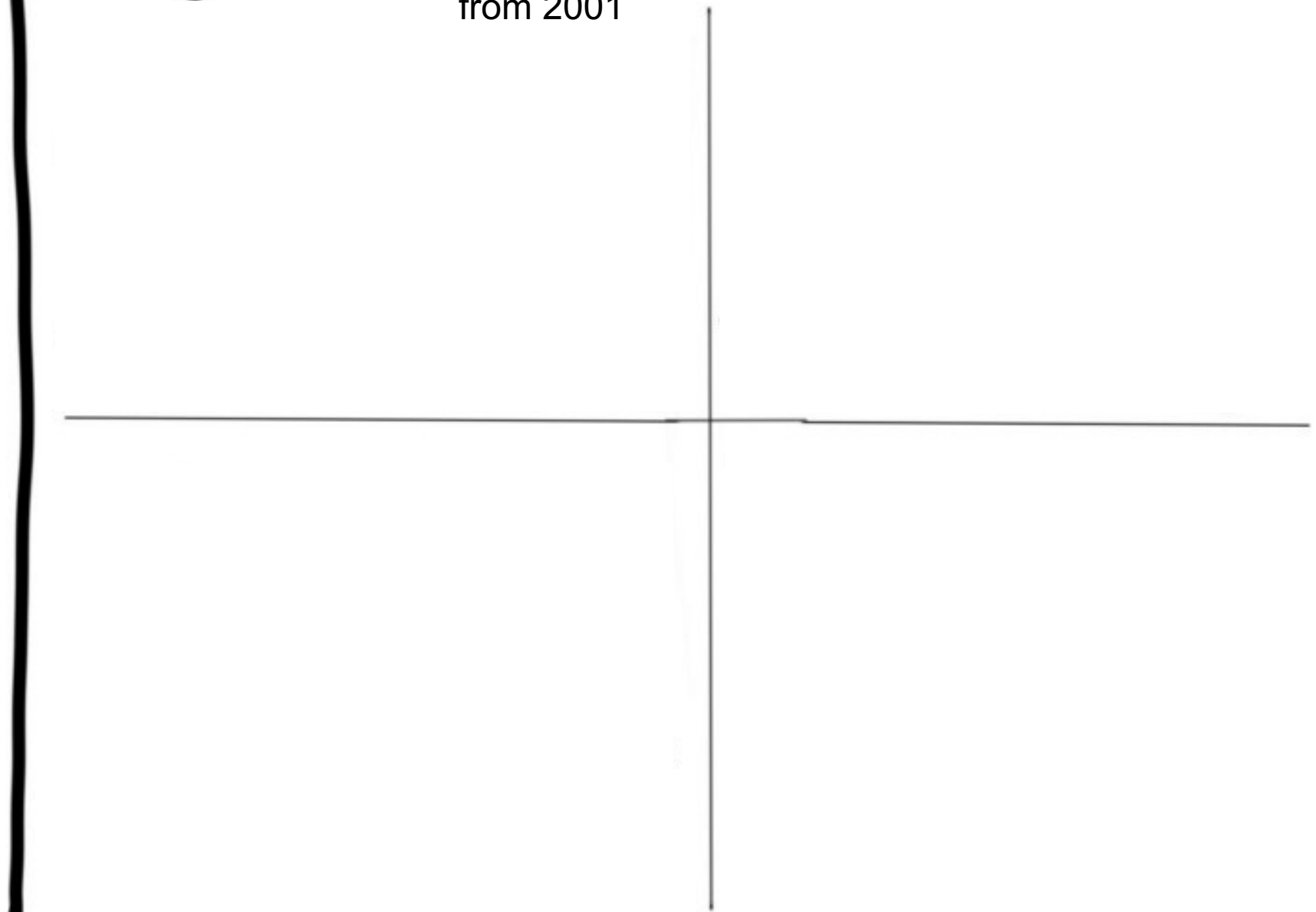


TU Delft

Agile Manifesto

from 2001

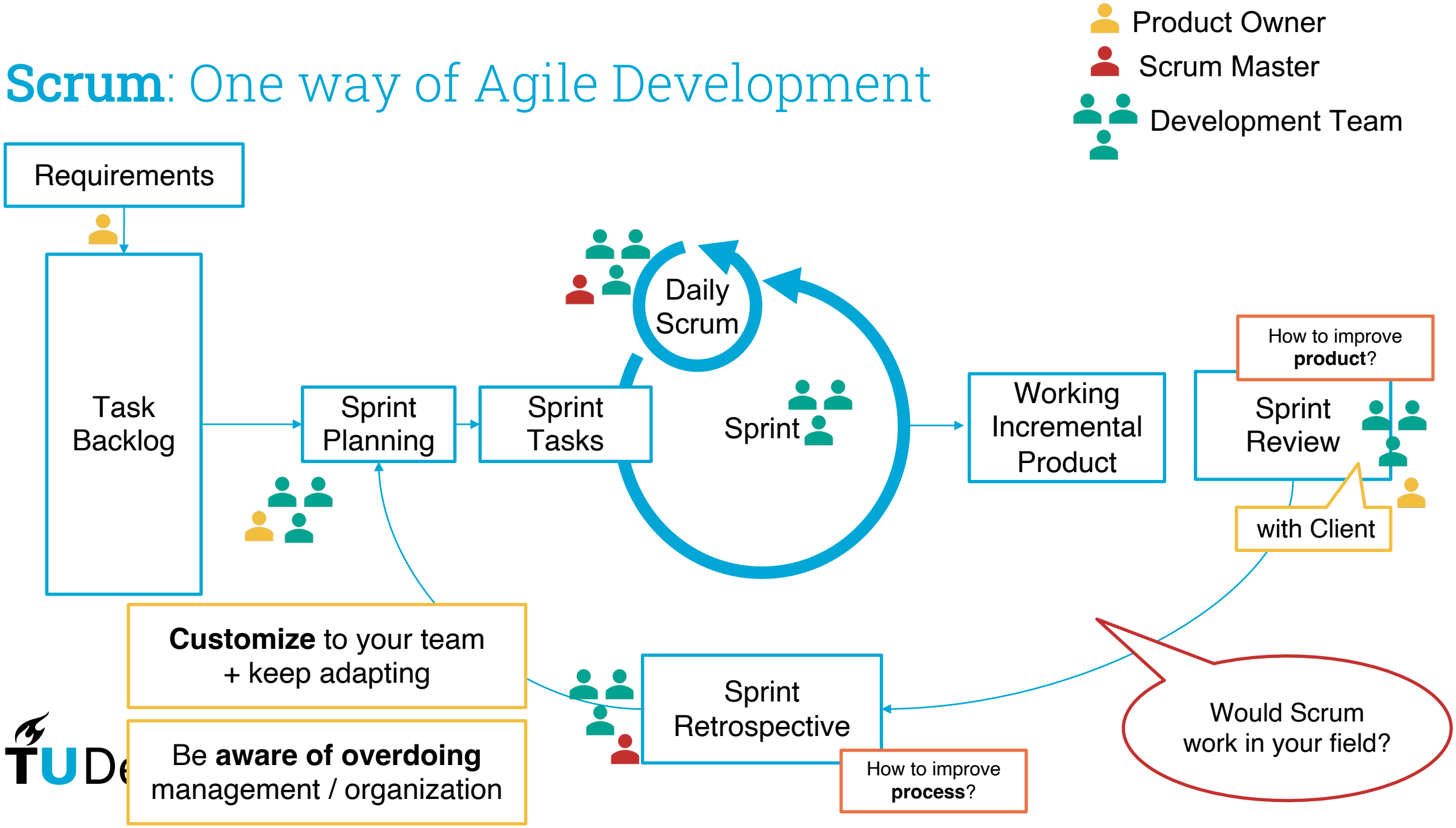
by:
Information Artist



Scrum: One way of Agile Development

TU Delft

Scrum: One way of Agile Development



How to have a (Client) Meeting

- **Pre-Meeting:** Prepare with the whole team beforehand (possibly with TA)
  - Agenda
  - Questions
  - Desired outcome

1-2 Days

- Have **the meeting**
  - Designated note-taker
  - Time keeper / Meeting leader

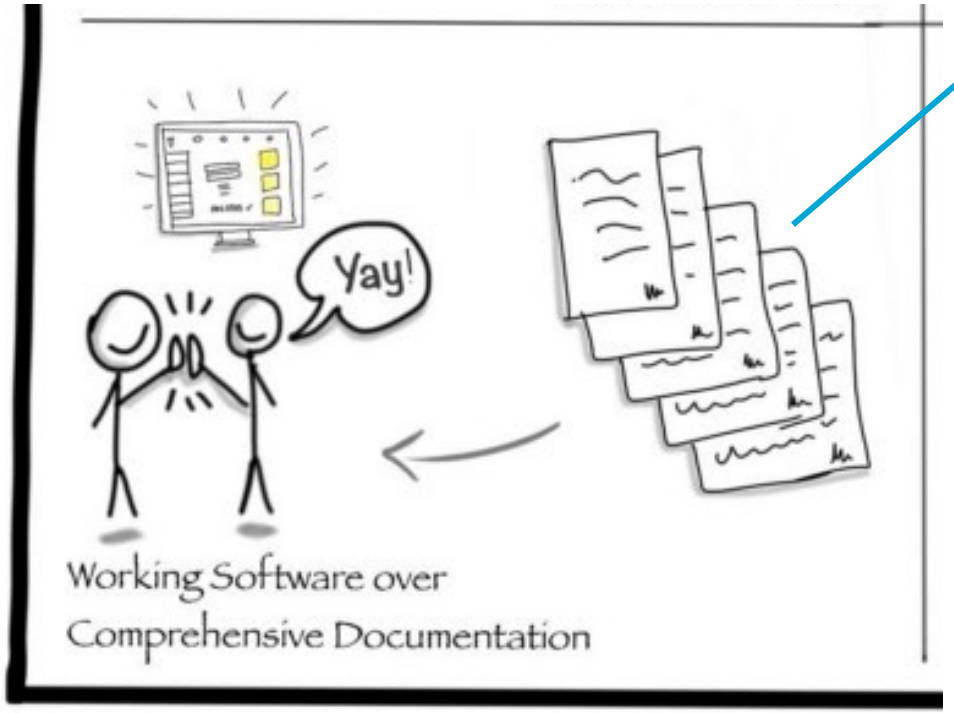
Team Lead Responsibility vs. Active Roles

30 min

- **Post-Meeting:**
  - What did we learn? → adapt plans
  - Discuss unclear things, surprises

TU Delft

Documentation



complete user documentation ahead of time ≠ keeping track of what the team does

Project Notes + Preparation for Final Presentation

Where to find everything

Decisions + Rationale

How are decisions documented in your field?

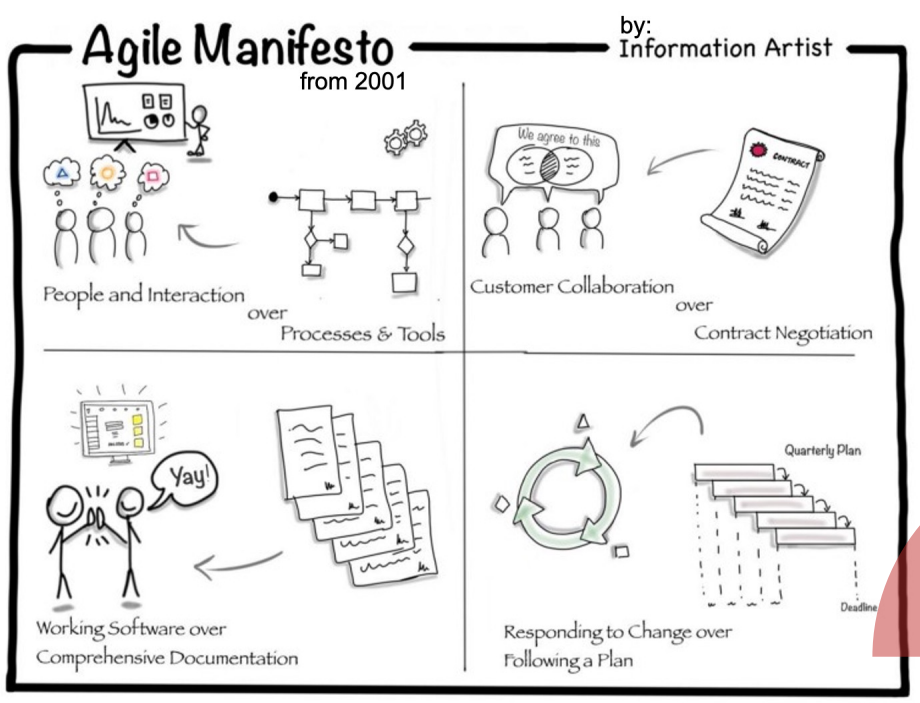
Industry Example: Architecture Decision Record

[short title of solved problem and solution]
Context and Problem Statement
...
Decision Drivers
•…
Considered Options
•…
Decision Outcome
Chosen option: "[option 1]", because ...

Positive Consequen
•…
Negative Consequences
•…
Pros and Cons of the Options
[option 1]
[description]
•Good, because ...
•Bad, because ...

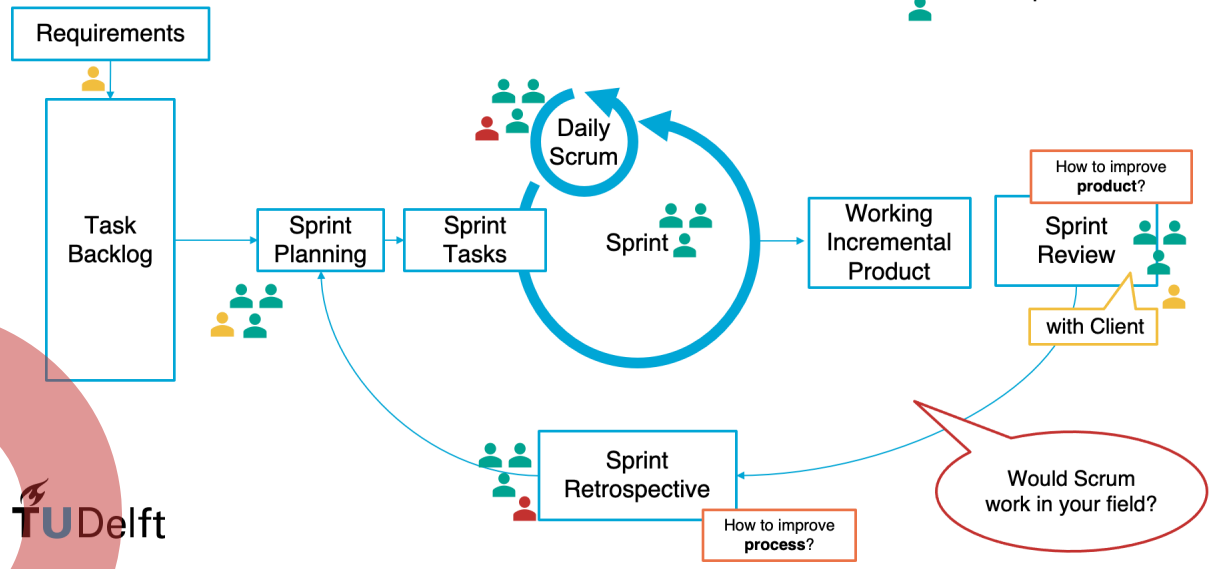
https://github.com/joelparkerhenderson/architecture-decision-record

TU Delft



Scrum: One way of Agile Development

Product Owner
Scrum Master
Development Team



TUDelft

How to have a (Client) Meeting

- **Pre-Meeting:** Prepare with the whole team beforehand (possibly with TA)
  - Agenda
  - Questions
  - Desired outcome

1-2 Days

- Have **the meeting**
  - Designated note-taker
  - Time keeper / Meeting leader

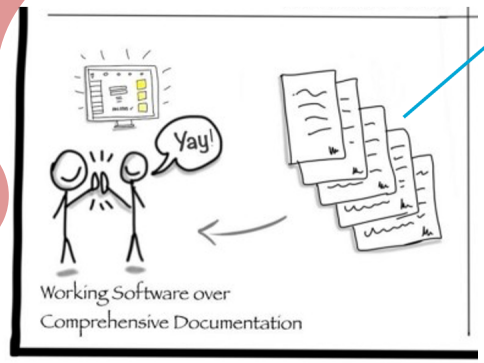
Team Lead Responsibility vs. Active Roles

30 min

- **Post-Meeting:**
  - What did we learn? → adapt plans
  - Discuss unclear things, surprises

TUDelft

Documentation



complete user documentation ahead of time ≠ keeping track of what the team does

Capstone Progress Document

Where to find everything

Decisions + Rationale

How are decisions documented in your field?

Industry Example: Architecture Decision Record

[short title of solved problem and solution]
Context and Problem Statement
...
Decision Drivers
•...
Considered Options
•...
Decision Outcome
Chosen option: "[option 1]", because ...

Positive Consequen
•...
Negative Consequences
•...
Pros and Cons of the Options
[option 1]
[description]
•Good, because ...
•Bad, because ...

https://github.com/joelparkerhenderson/architecture-decision-record

TUDelft

Let‘s get to work!

TU Delft

- **Review the initial requirements given by the client**
  - What do you (not) understand?
  - What extra information would you want from the client?
  - What would you look up online / in literature? Where?
  - How would you test that the requirement is fulfilled?
  - Can you see the steps needed to solve this requirement? Is it feasible for this project?
  - What could you offer in a negotiation?

- **Make a process plan**: How will you work together over this course?
  - When and where do you want to work?
  - How will you organize the project time? (Sprint length, meeting dates, deadlines)
  - Assign roles: project lead, project owner, scrum master, ...
  - Where and what will you document? Templates?
  - Make a big time plan: which week will you implement/do x?

- **(Start) preparing your proposed requirements**
  - Who is the stakeholder of the requirement?
  - Is it functional or non-functional?
  - Should the requirement be broken down?
  - Where on the MoSCoW scale do you want it?
  - Can you write the requirement better? (Unambiguous, Feasible, Verifiable, Traceable, Necessary; Consistent, Complete)

- Can you **start setting up your development environment**?
- **Bonus:** Apply what you learned about requirements to your Code of Conduct points

- What implementation steps are needed to fullfil the requirement?
- Sketch the validation approach for the requirement.

TU Delft

👆 Ideas on what you can apply right now 👆

Sources

- [Pre05] Pressman, Roger S. *Software engineering: a practitioner's approach*. Palgrave macmillan, 2005.
- [Som11] Sommerville, Ian. *Software Engineering, 9/E*. Pearson Education India, 2011.