Introduction to Software Project Development

Ph. Collet, with slides from S Mosser

Lecture #1, 13.09.2017

pictures: sxc.hu & C.line

Software

[redacted]

Development


Easy!

Really?

[redacted]

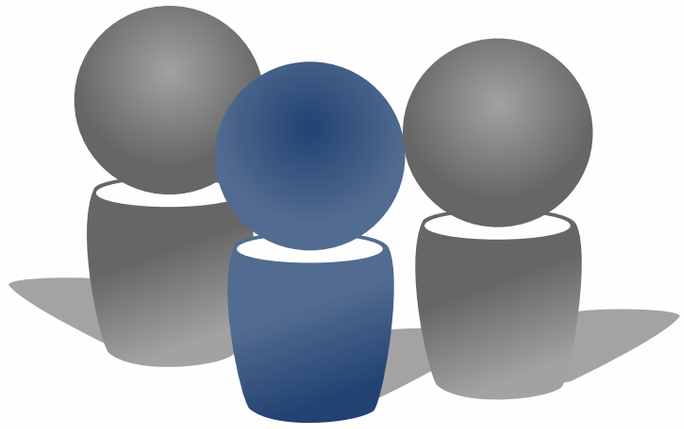
Project

[redacted]

Project?

customer

Needs

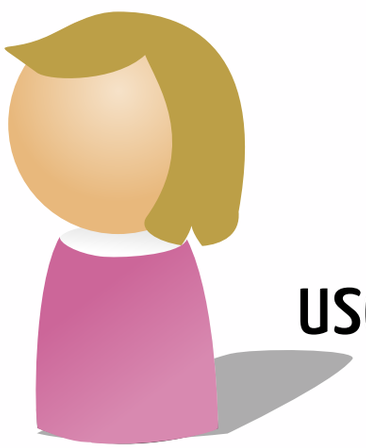


development team(s)

Project

project manager(s)

Product



user(s)

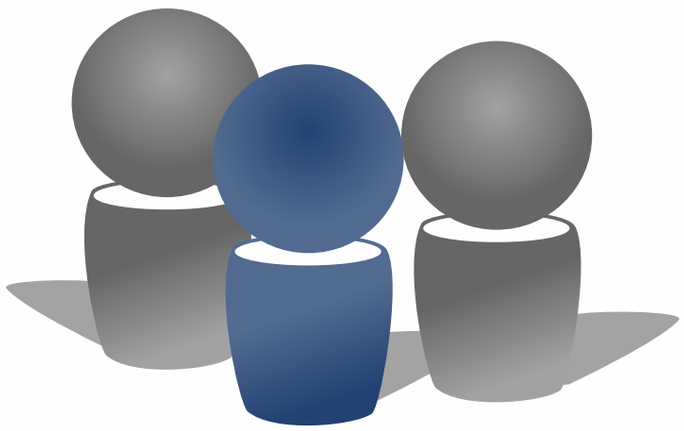
De la presquitude des choses.

As a Project Manager

This is what I understood.

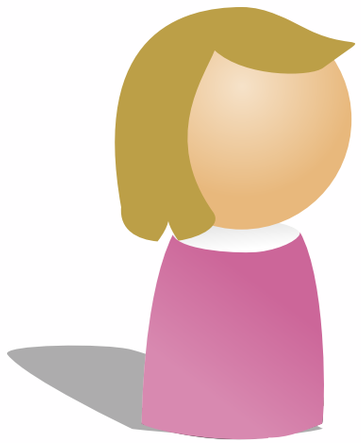
As Developers

This is what we'll deliver



As a final user

This is what I'll have to deal with.



As a Customer

This is what I actually needed

Activities

3rd year

Requirements

4th year

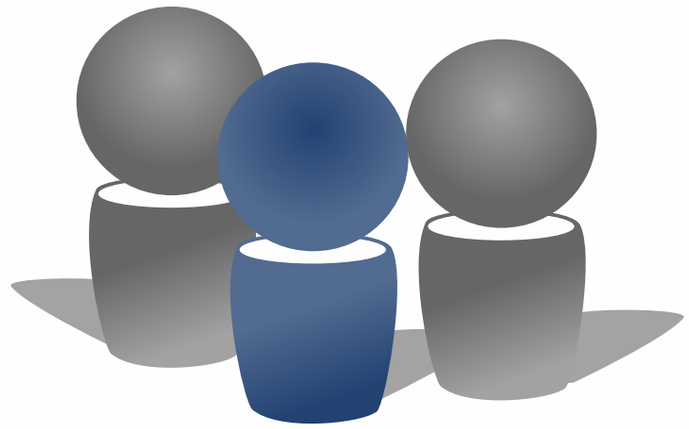
Resources

Budget

5th year

Cost

Course Focus



development
team(s)

Software Development Lifecycle(s)

How to organize software development activities?

Development Activities?

Requirements

Production

Design

Development

Specification

Validation

Maintenance

Tests

Lifecycle model?

Given **organization**

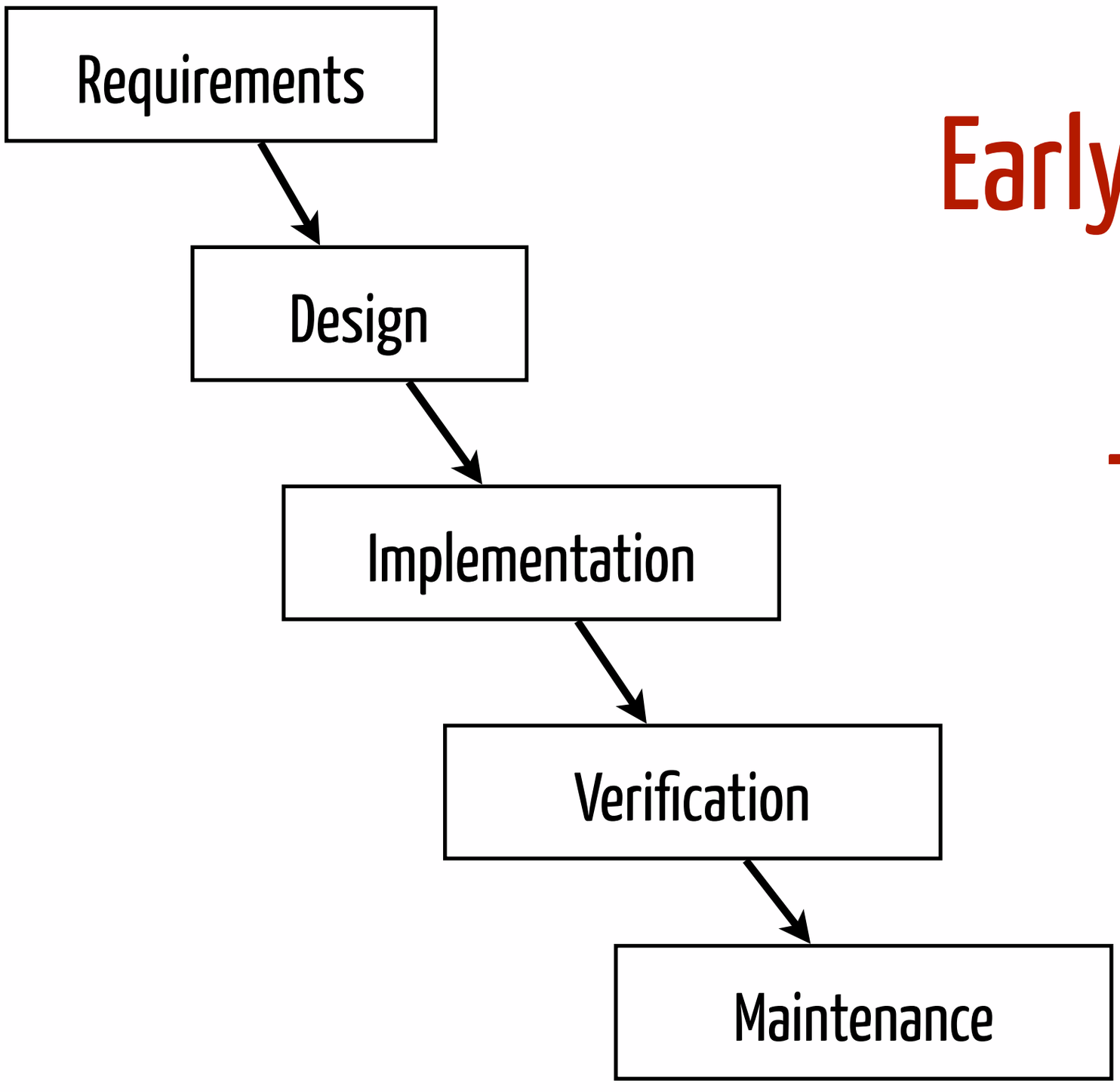
of such **activities**

Linear

or

Non-linear

Waterfall Model (Linear, ca. 1970)

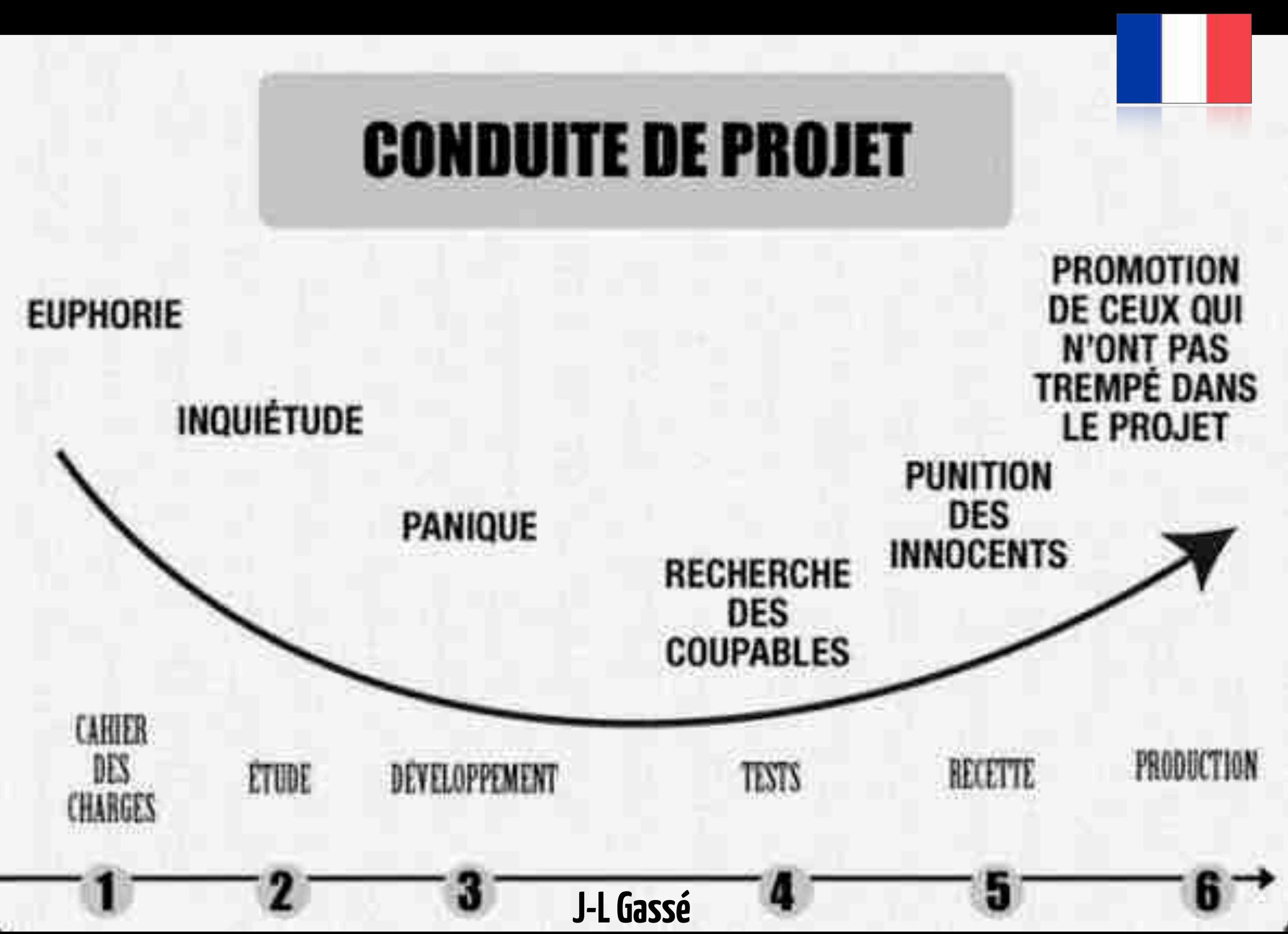


Early requirements

Tunnel effect

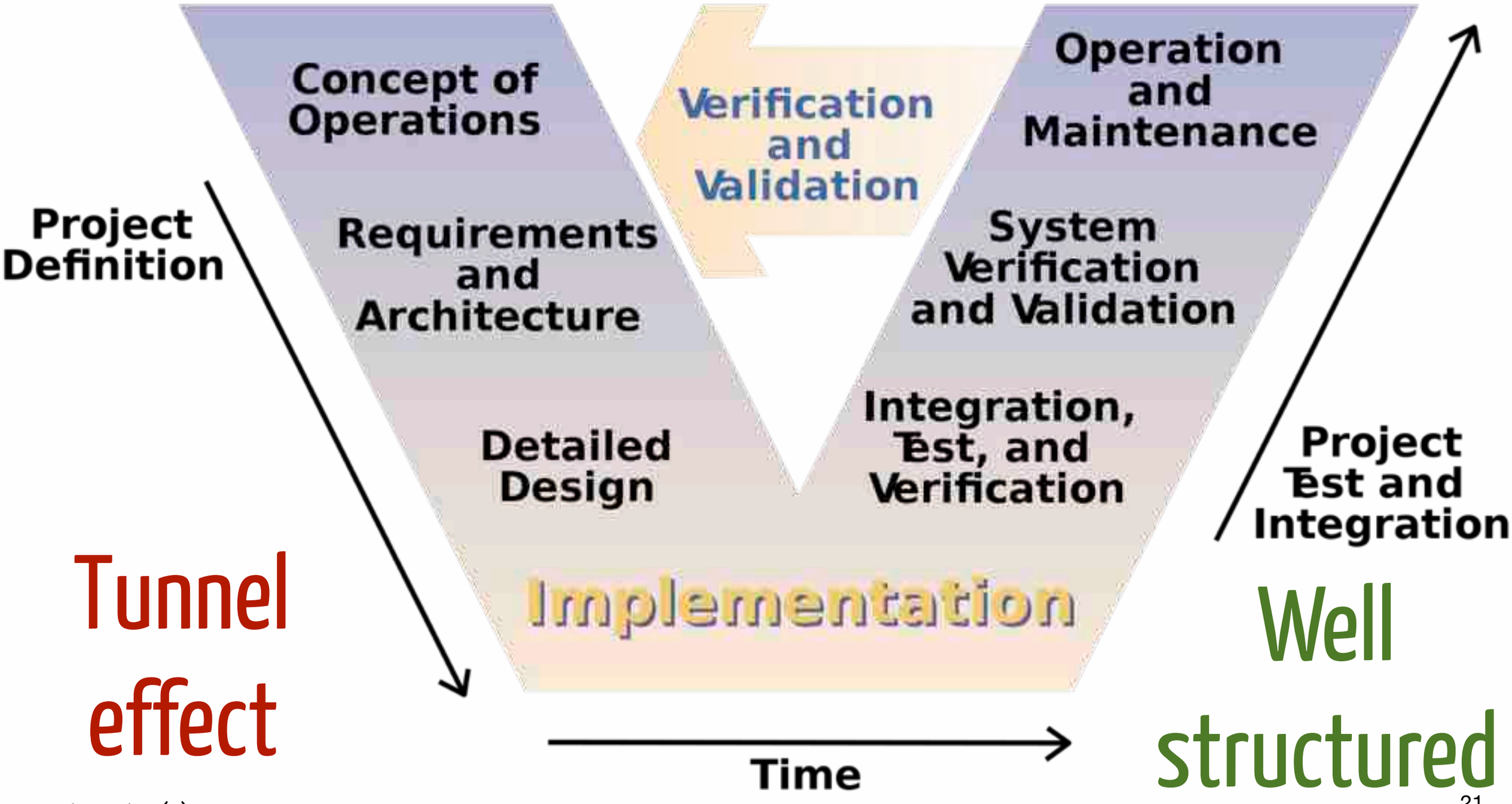
Late testing

CONDUITE DE PROJET



J-L Gassé

V-model (normative model)

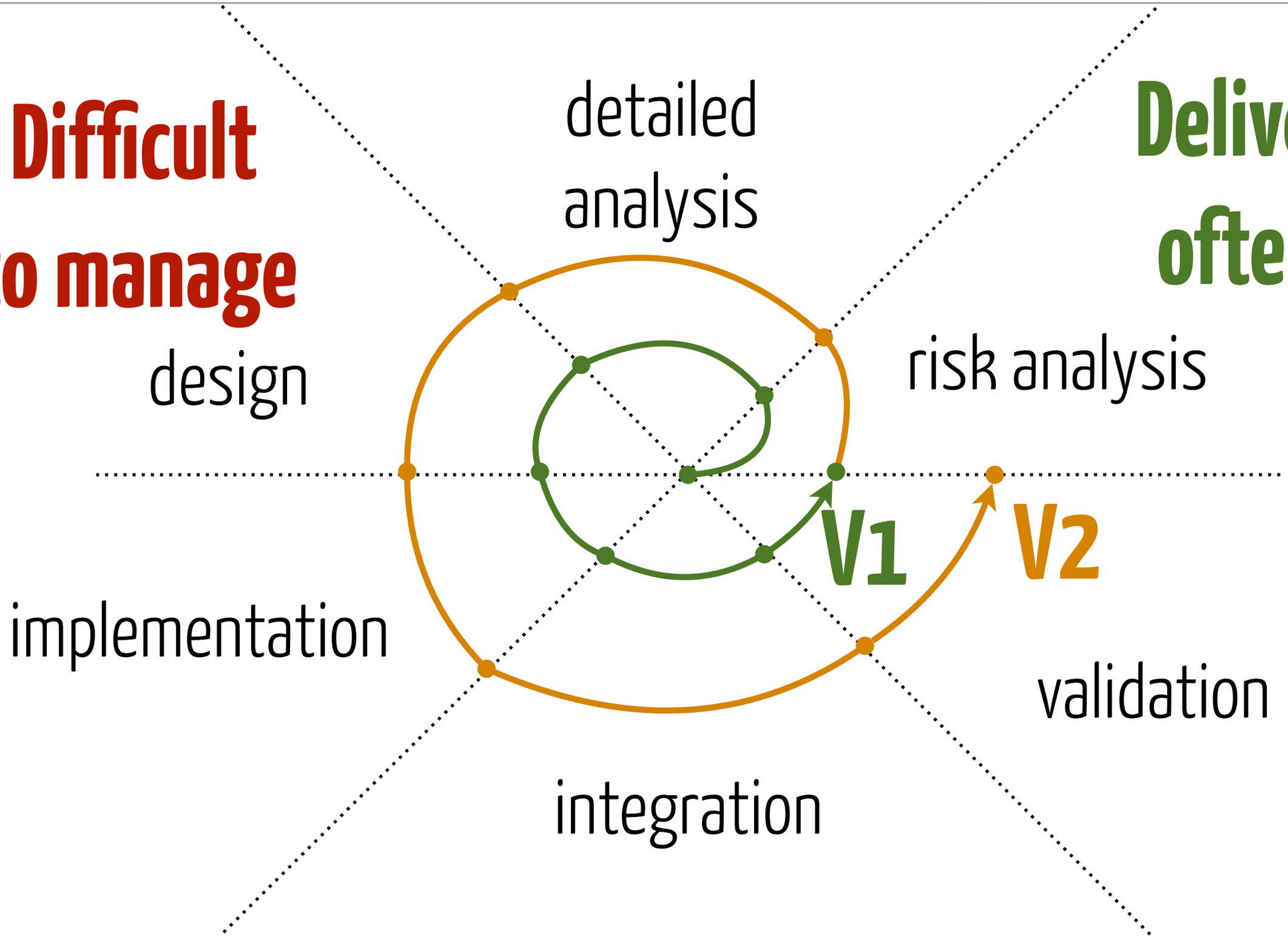


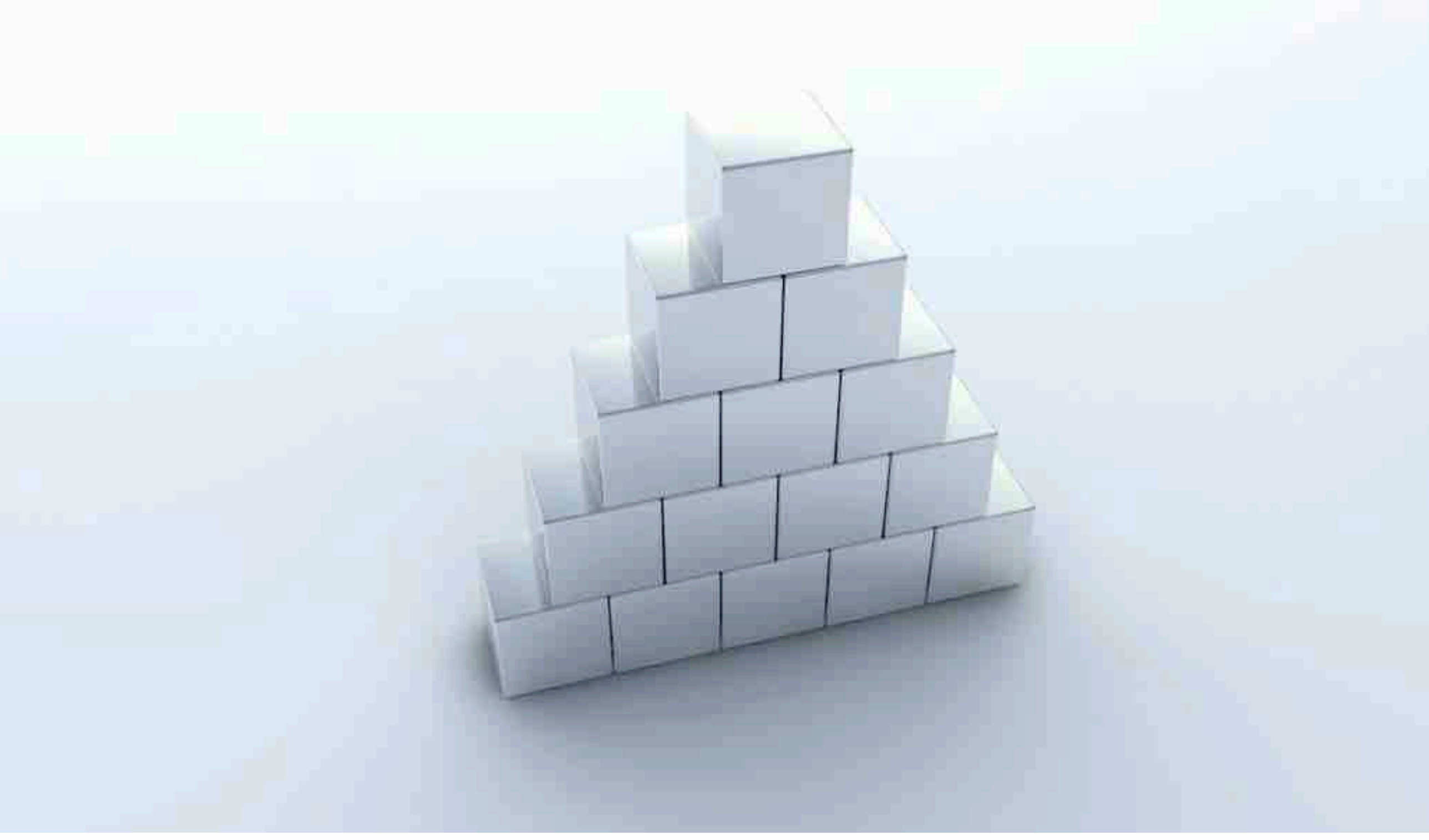
Tunnel effect

Well structured

wikipedia (c)

Spiral Model [Boehm86]





Focus on Task-based Development

«Just do it!»

Achievements

Tasks

Release

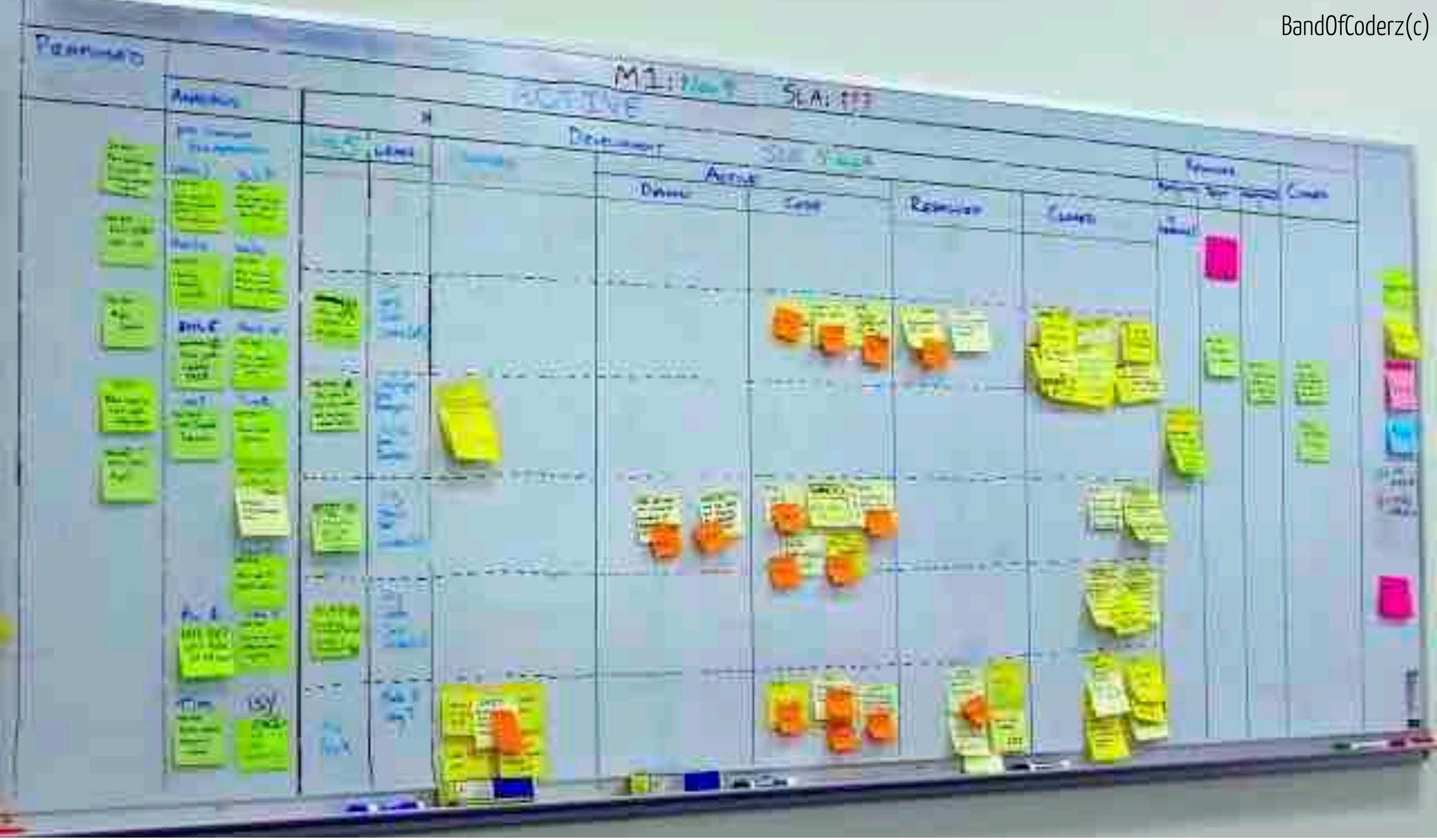
⇒ Build!

«Try not. Do or do not; there is no try.»

The empire strikes back (1980)

Maximize

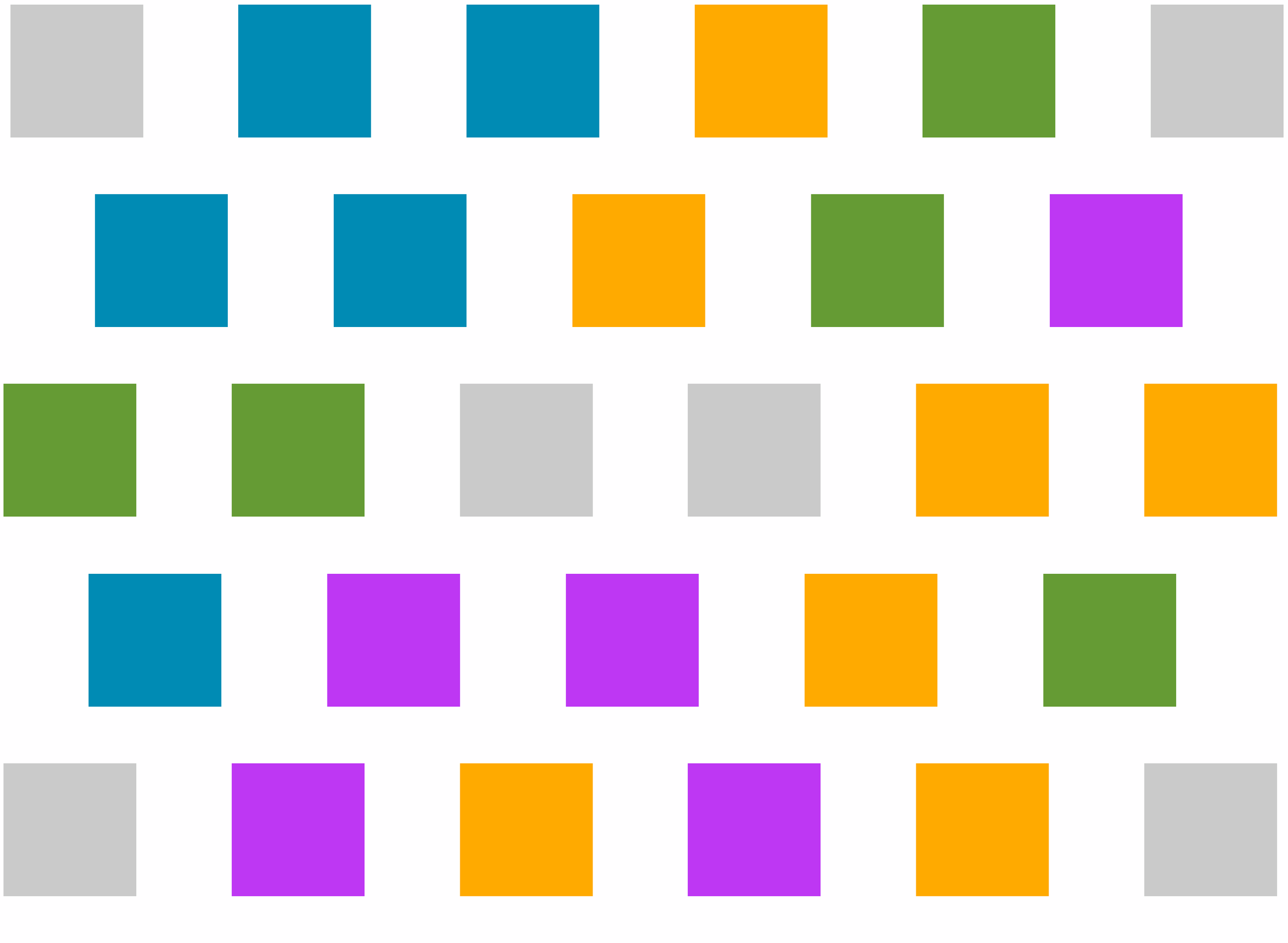
Minimalism!



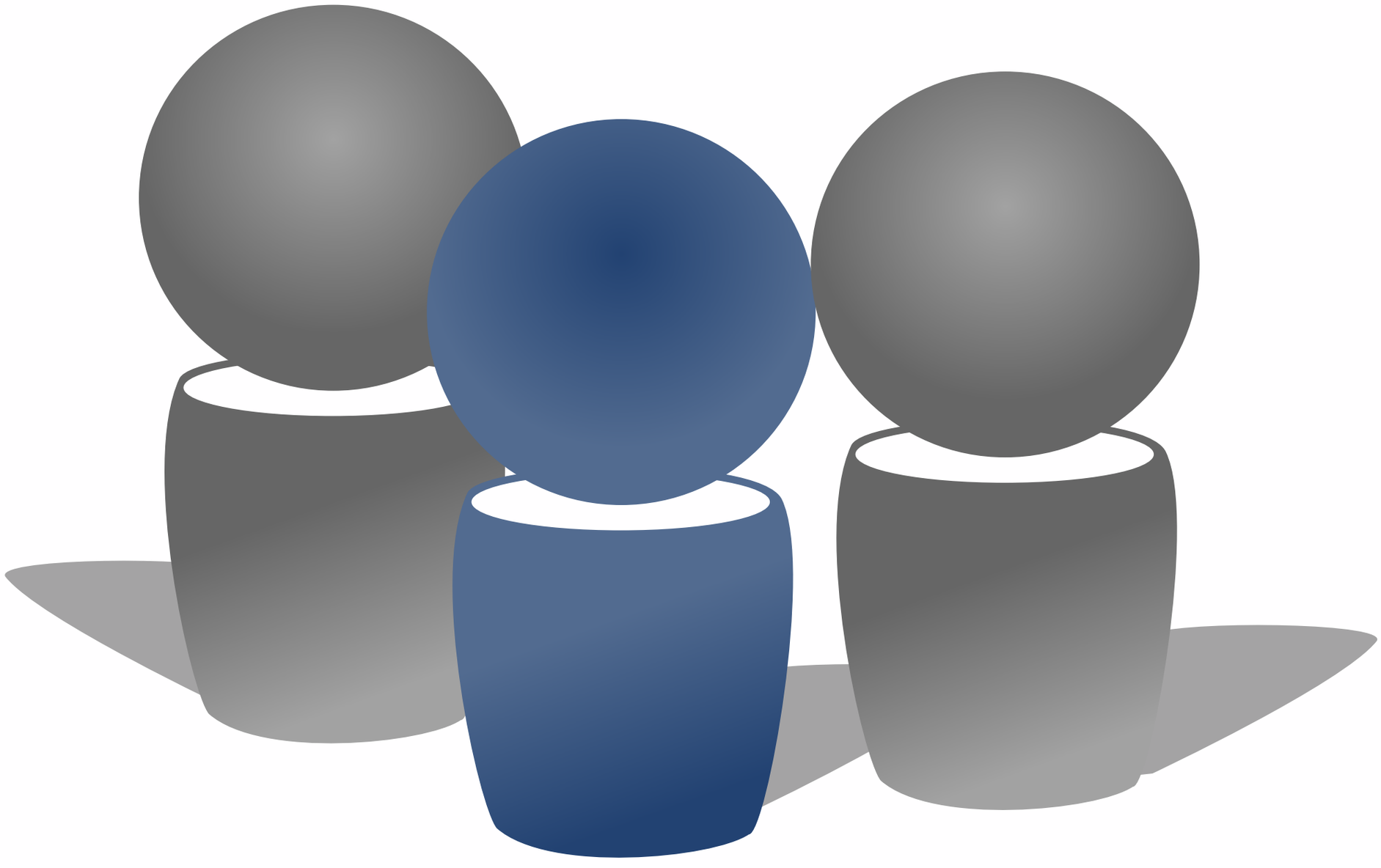
Organizing the task flow

"Haaave you met Kanban?"

http://blog.soat.fr/wp-content/uploads/2011/12/Kanban-booard.jpg



????



Problem

=

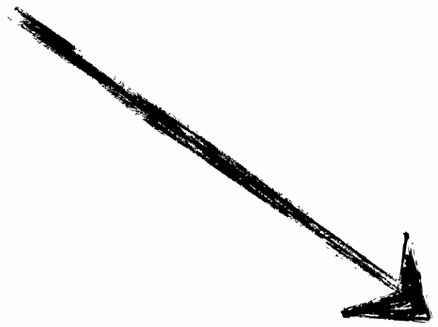
Workflow

Solution

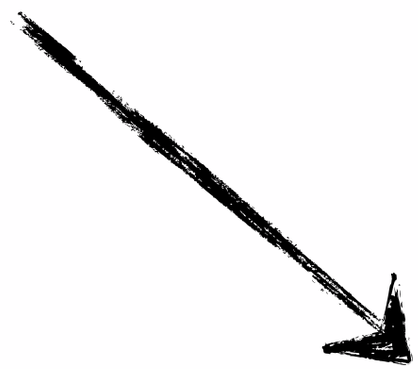
=

Draw it!

To Do



In Progress



Done

To Do

Chosen

Development

Delivery

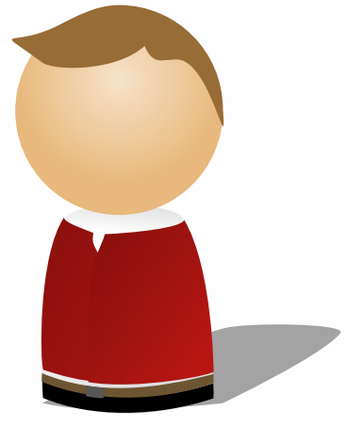
Tests

Done

To Do

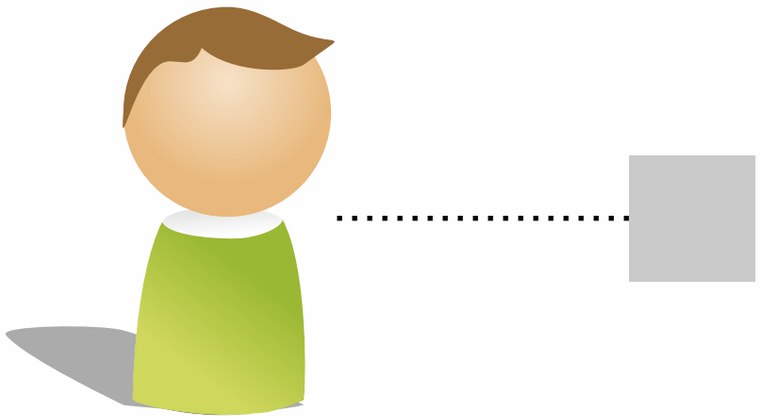
In Progress

Done



To Do

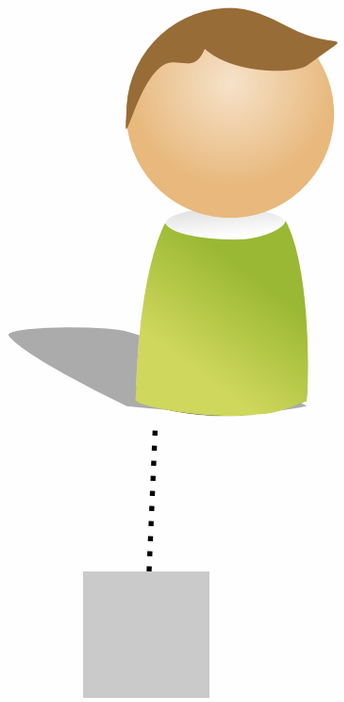
In Progress

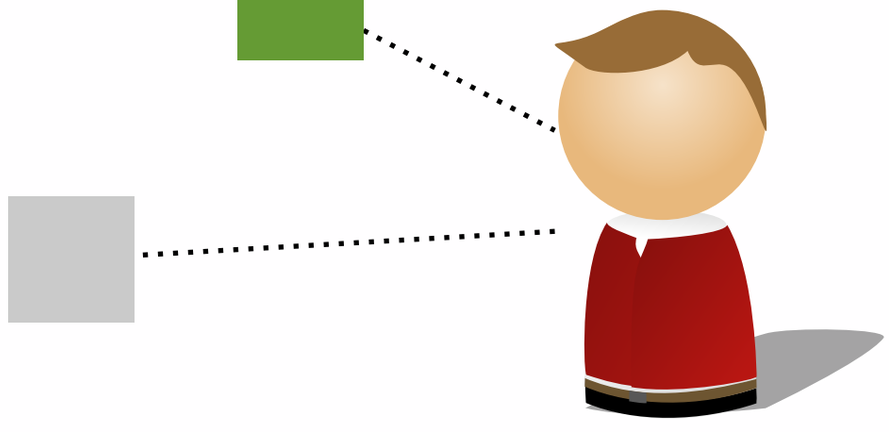


Done

To Do

In Progress



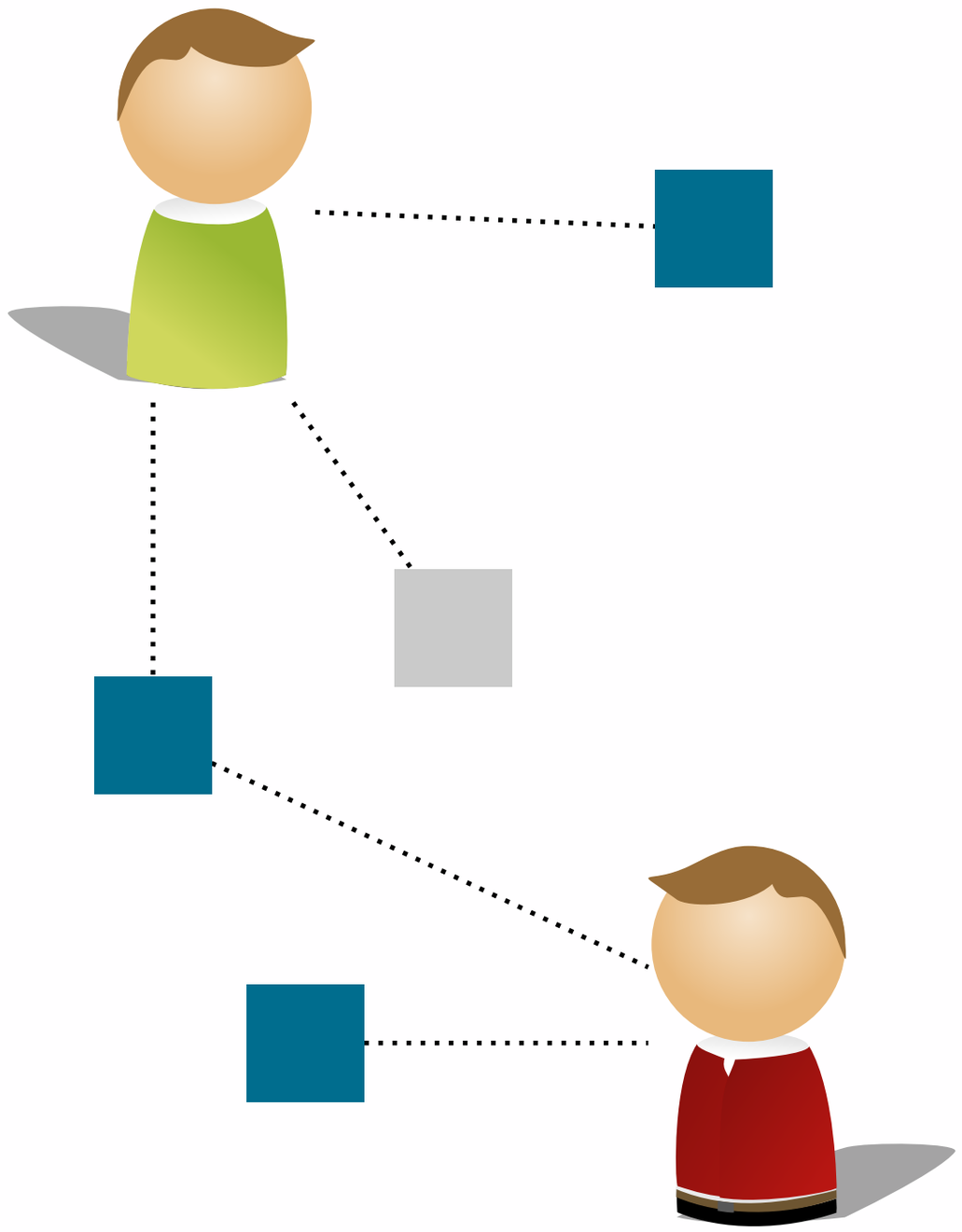


Done

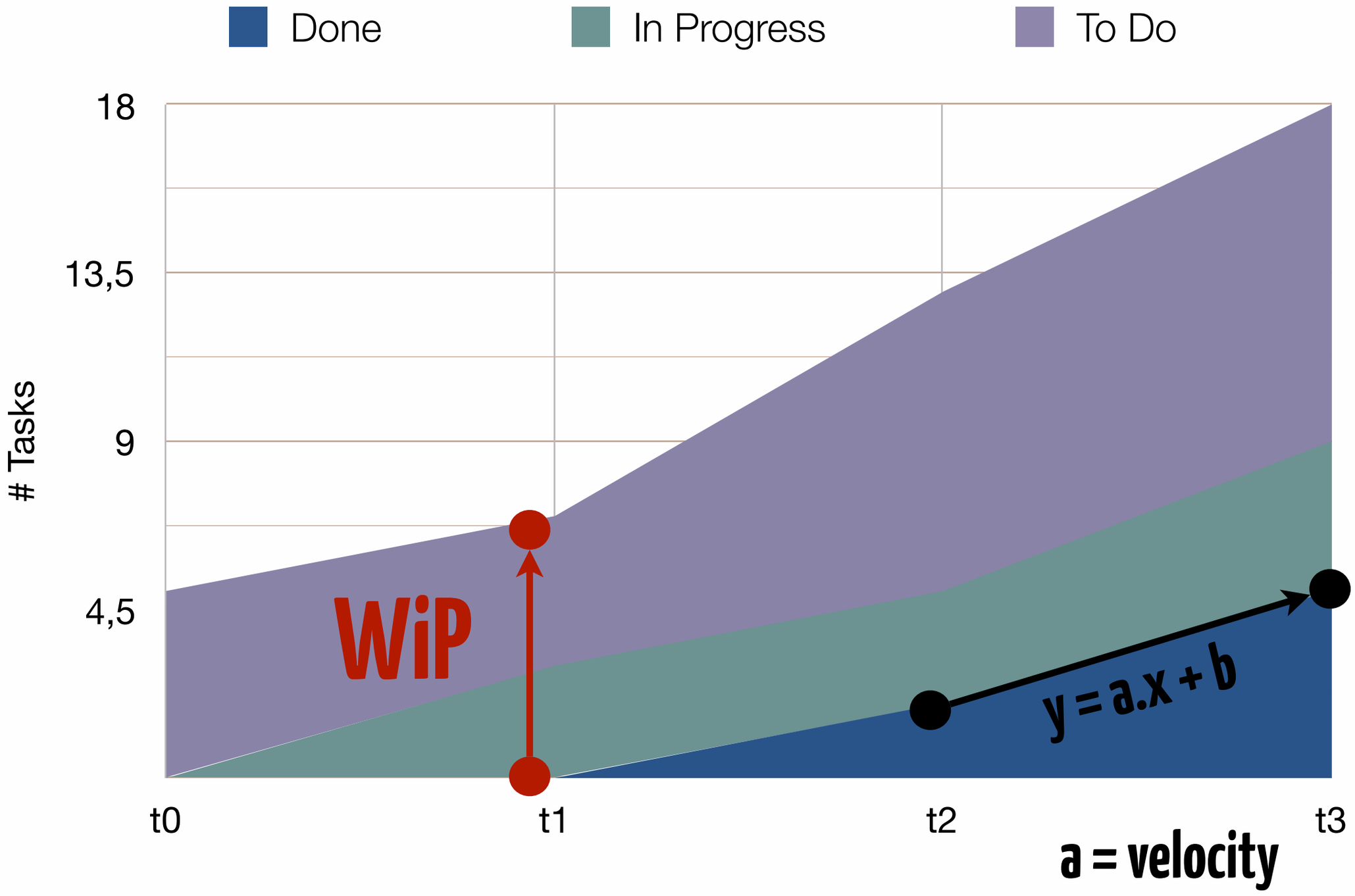
To Do

In Progress

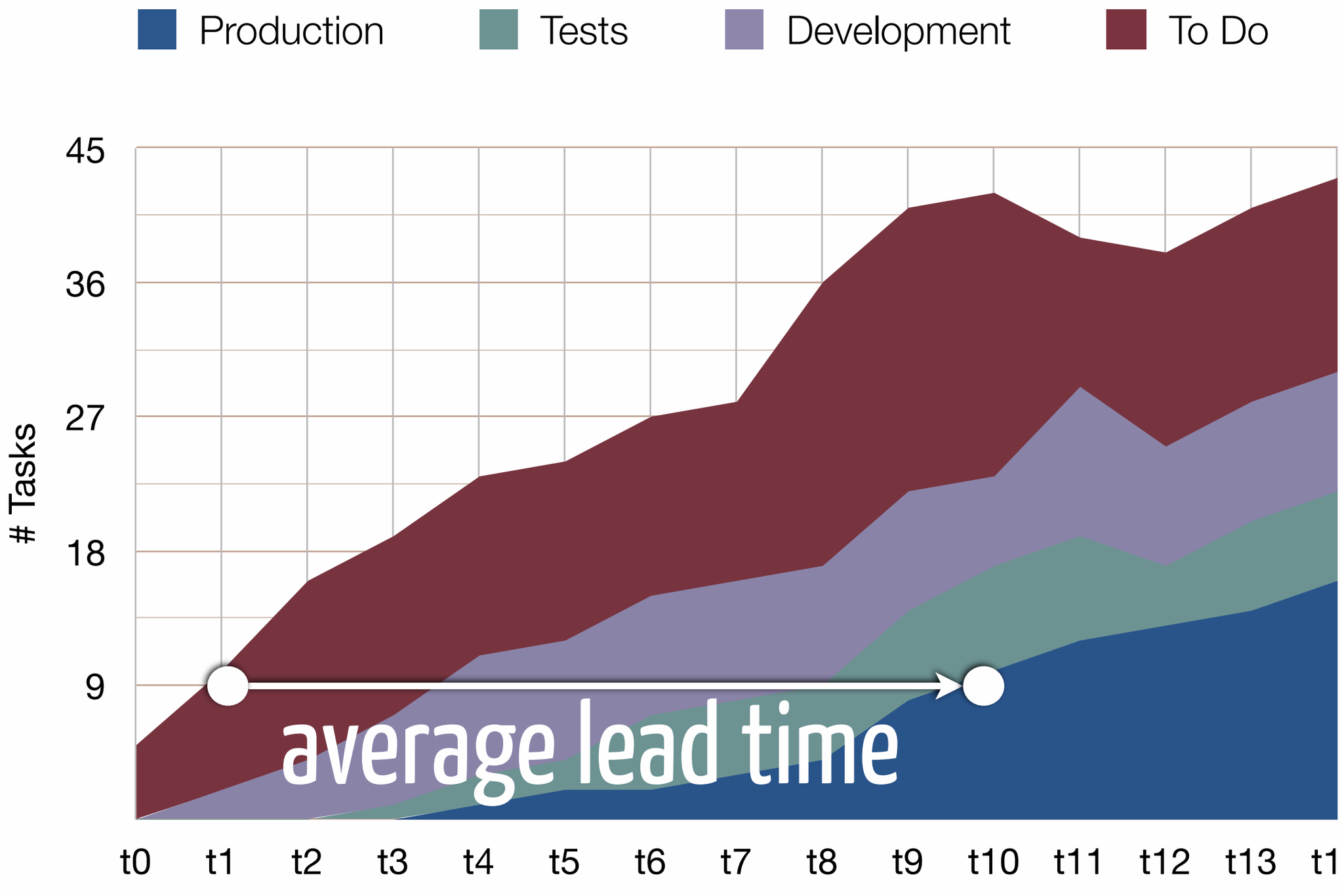
Done



Cumulative Flow Chart



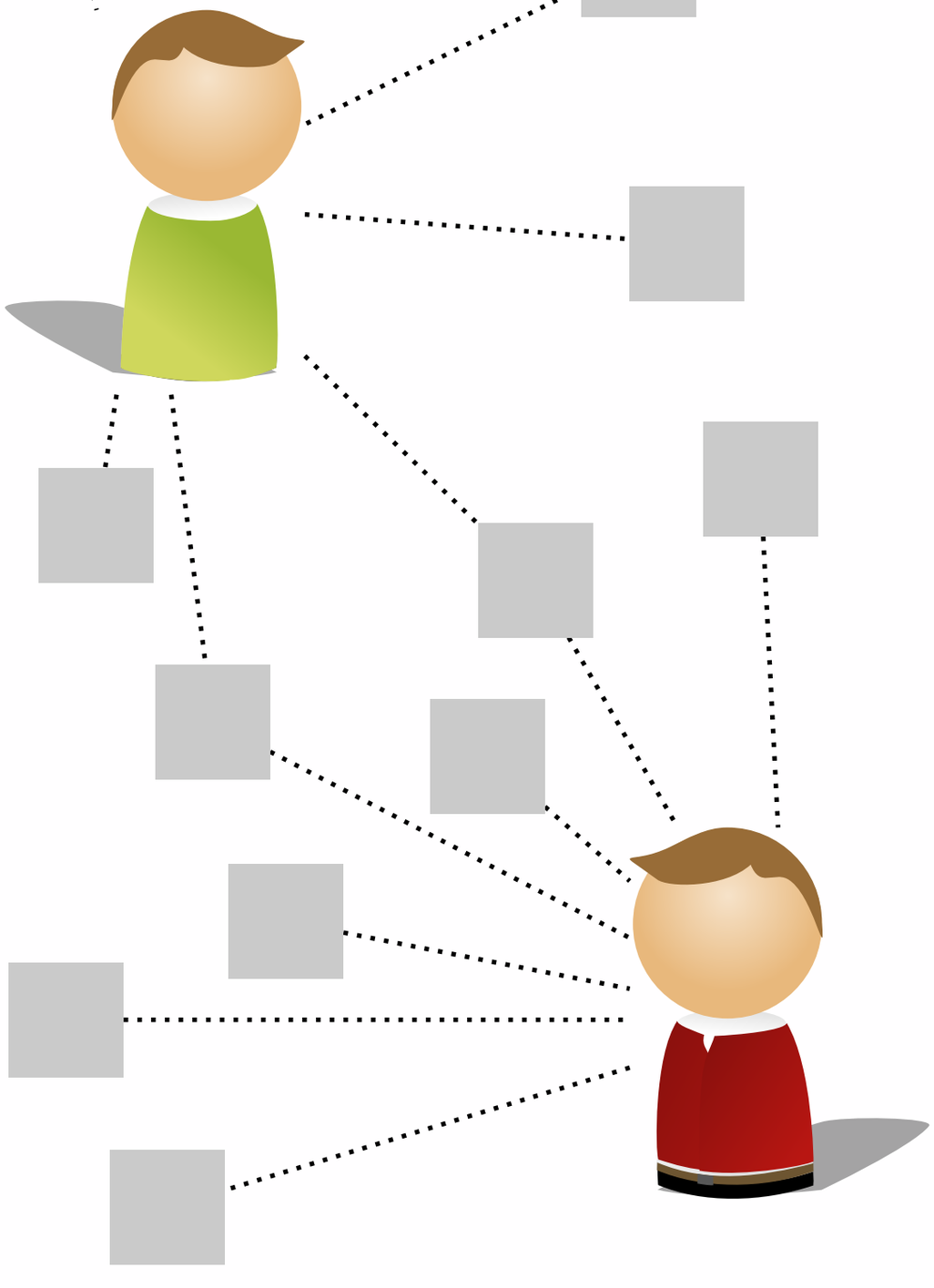
Exploiting a Cumulative Flow Chart



Does it really do the trick?

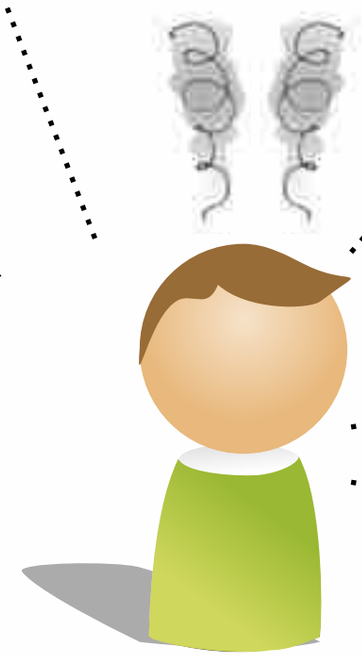
To Do

In Progress



Done

Overload



Problem

—

Work in Progress

Solution

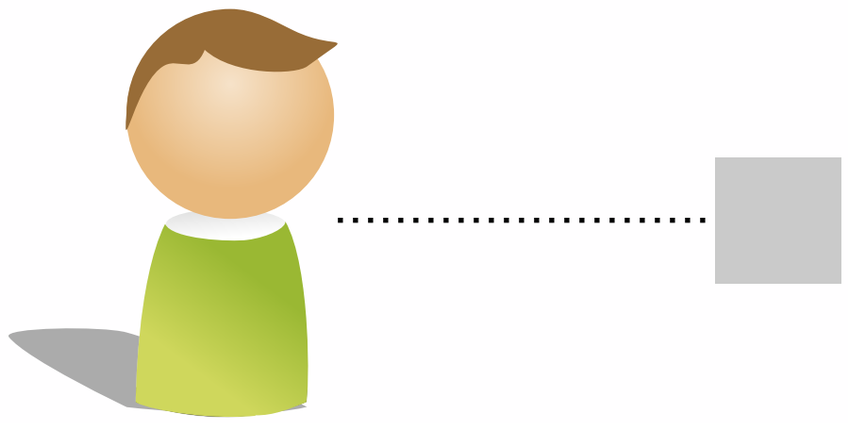
=

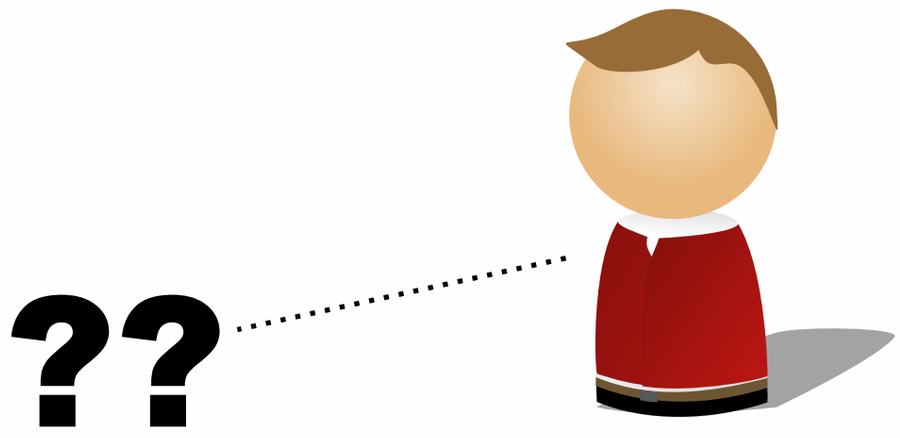
Limit the WiP!

To Do

In Progress (1)

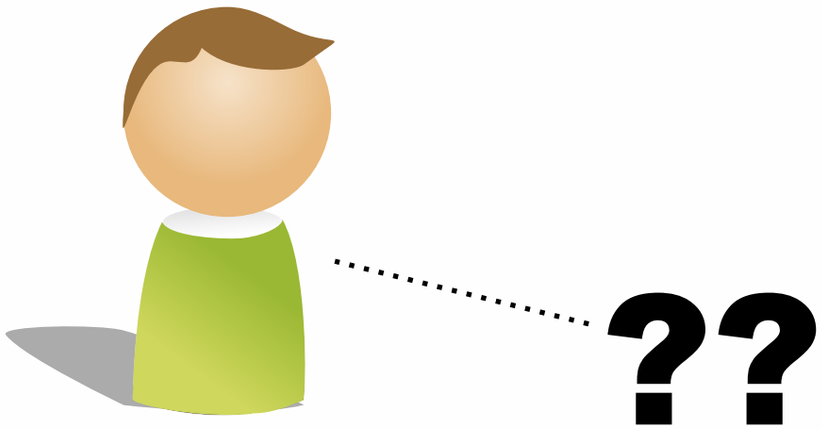
Done



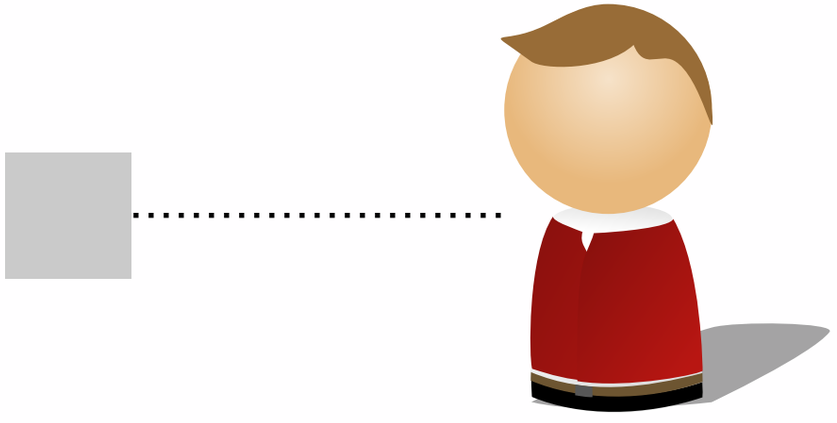


To Do

In Progress (1)



Done



Problem

=

Lead time

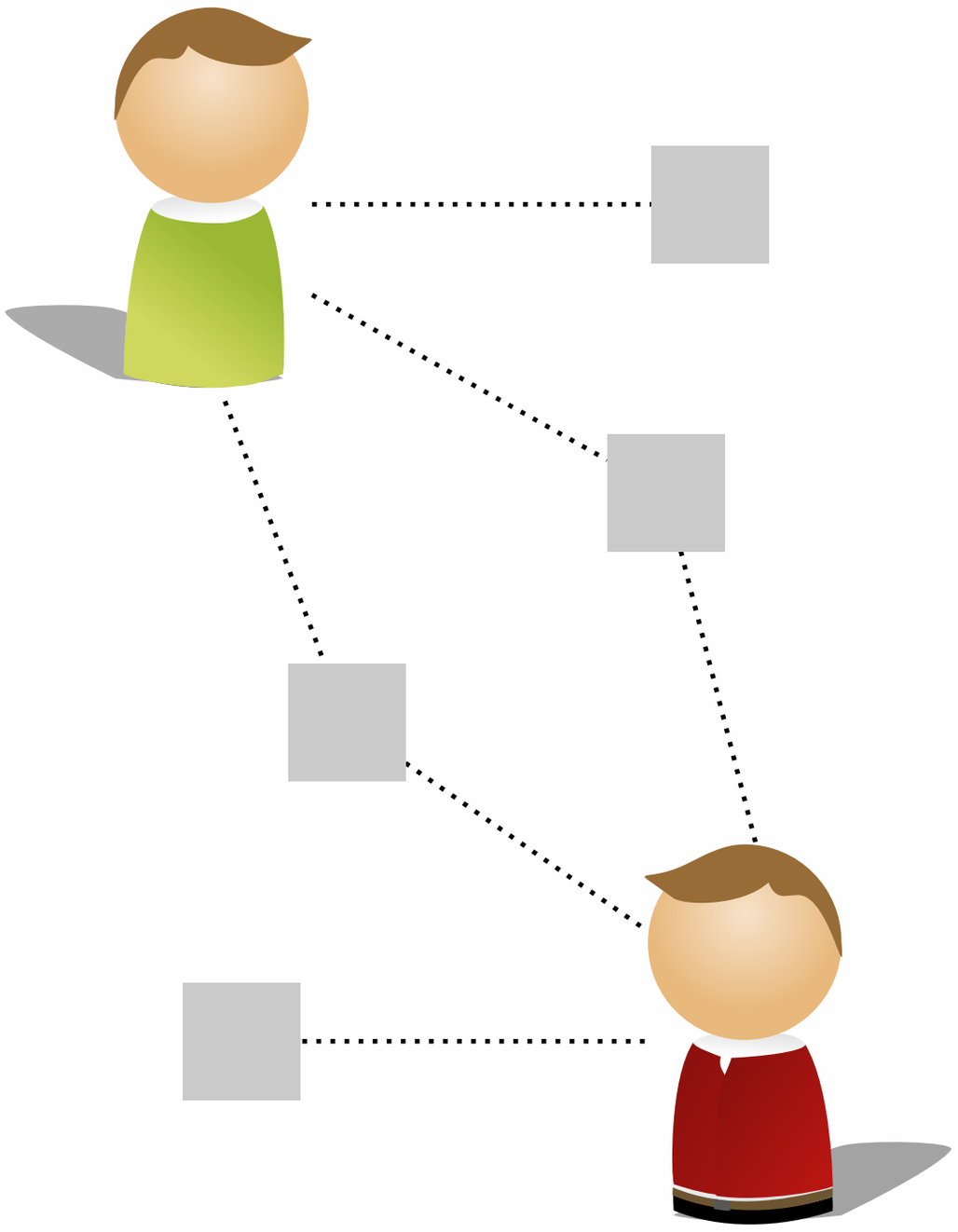
Solution

=

Adjust the limit!

To Do

In Progress (4)

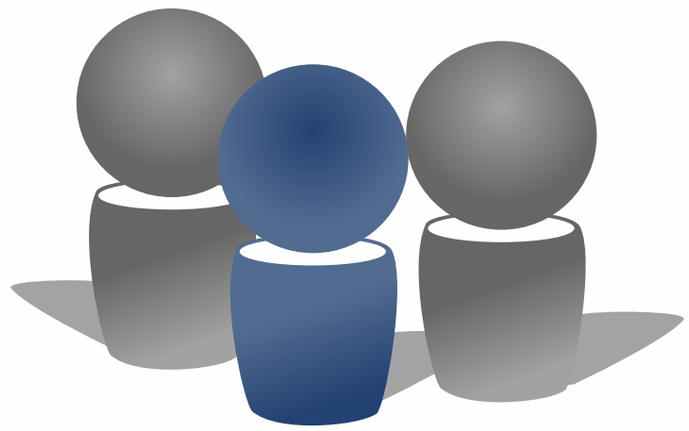


Done

There is no magic number!

Conclusions

Projects ⟹ Engineering discipline



development
team(s)

Lifecycle organizes project activities

Different models

linear vs non-linear

no silver bullet

Kanban in one single slide

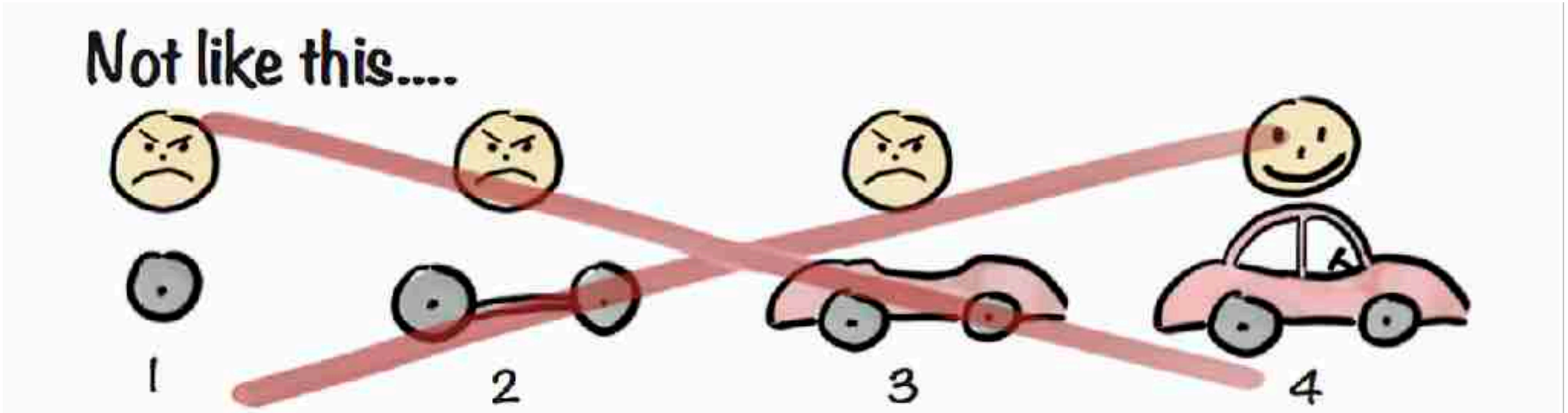
Visualize your workflow

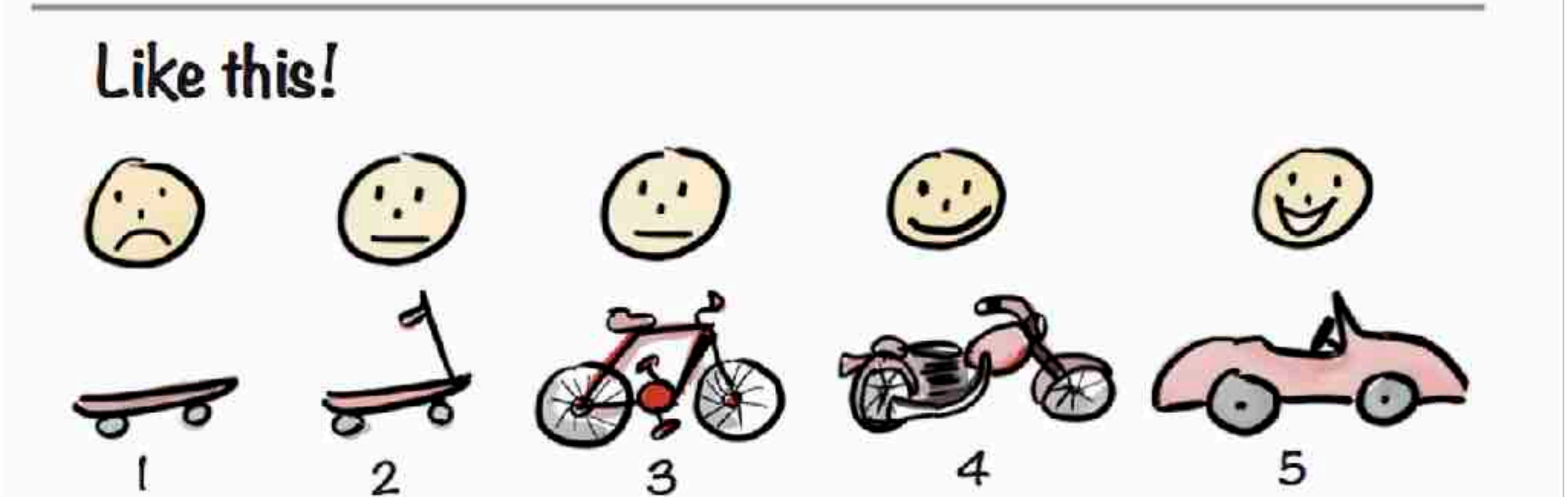
Limit the Work in Progress

Minimize lead time

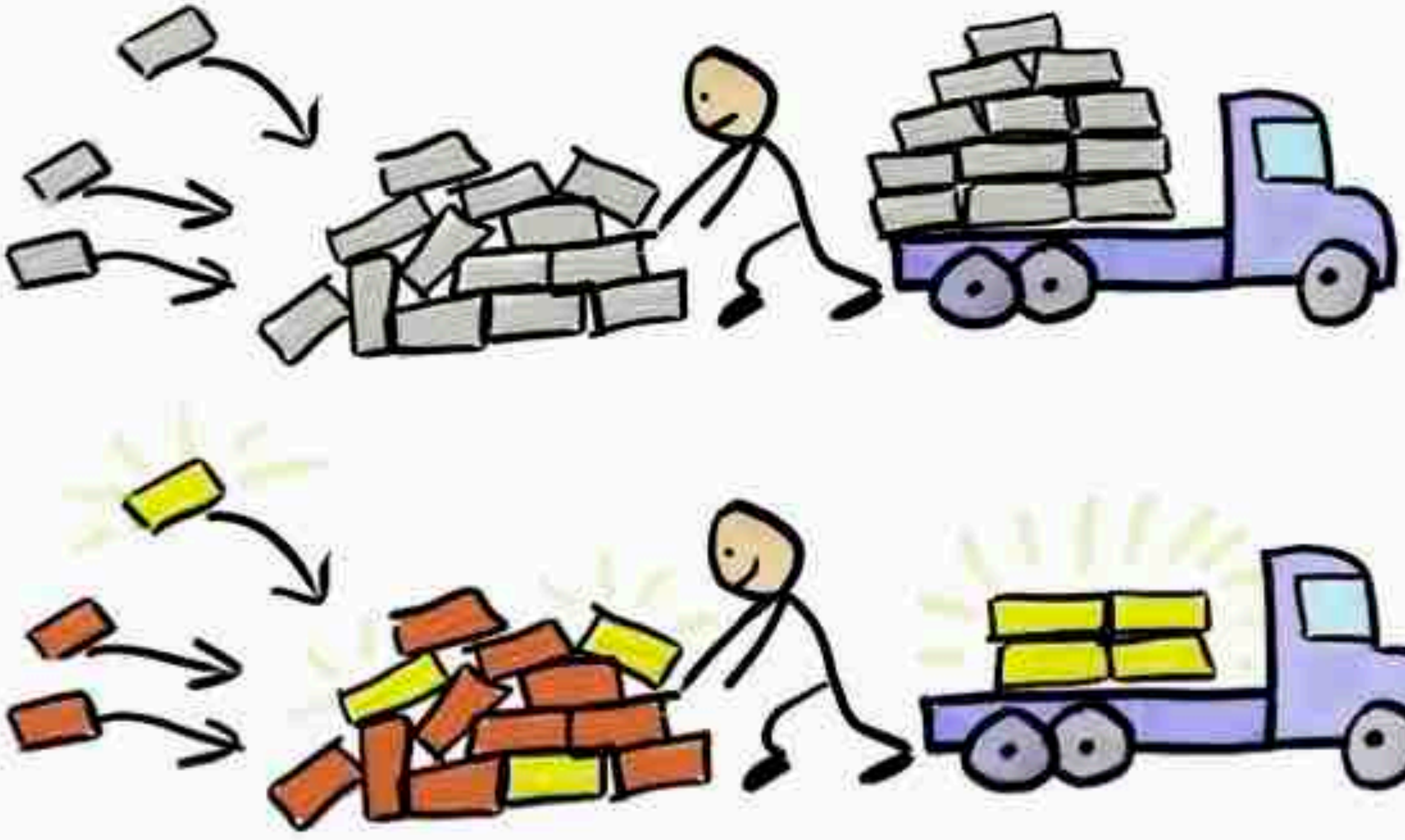
How to start?

Continuous Delivery



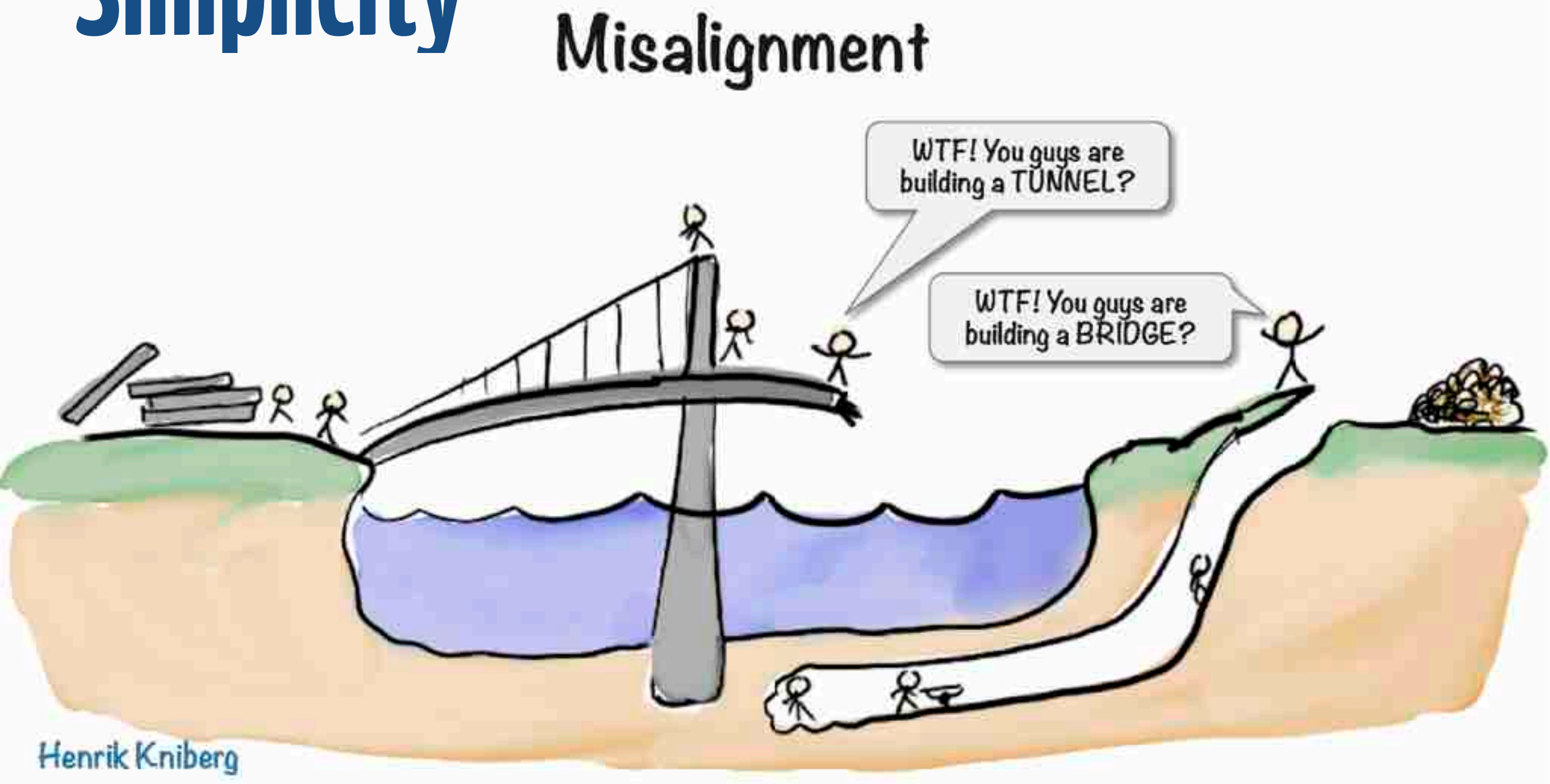


Maximize Value, not Output



Henrik Kniberg

Simplicity



Minimise useless work

Let’s try altogether on a projet

- Game engine with computer playing alone
  - No interaction with user
  - No graphics
- Several levels of computer players
- To be integrated later…

But

- Participer à la réalisation d'inventions pour gagner des points de victoire, mais également de faire progresser son équipe d'inventeurs.
- Chaque joueur dirige une équipe de 4 inventeurs, chacun possédant des compétences initiales propres.
- Les cartes inventions sont séparées en 3 époques

Basic Architecture

Slice

Robot Player

Game visualization

Game engine

Game representation