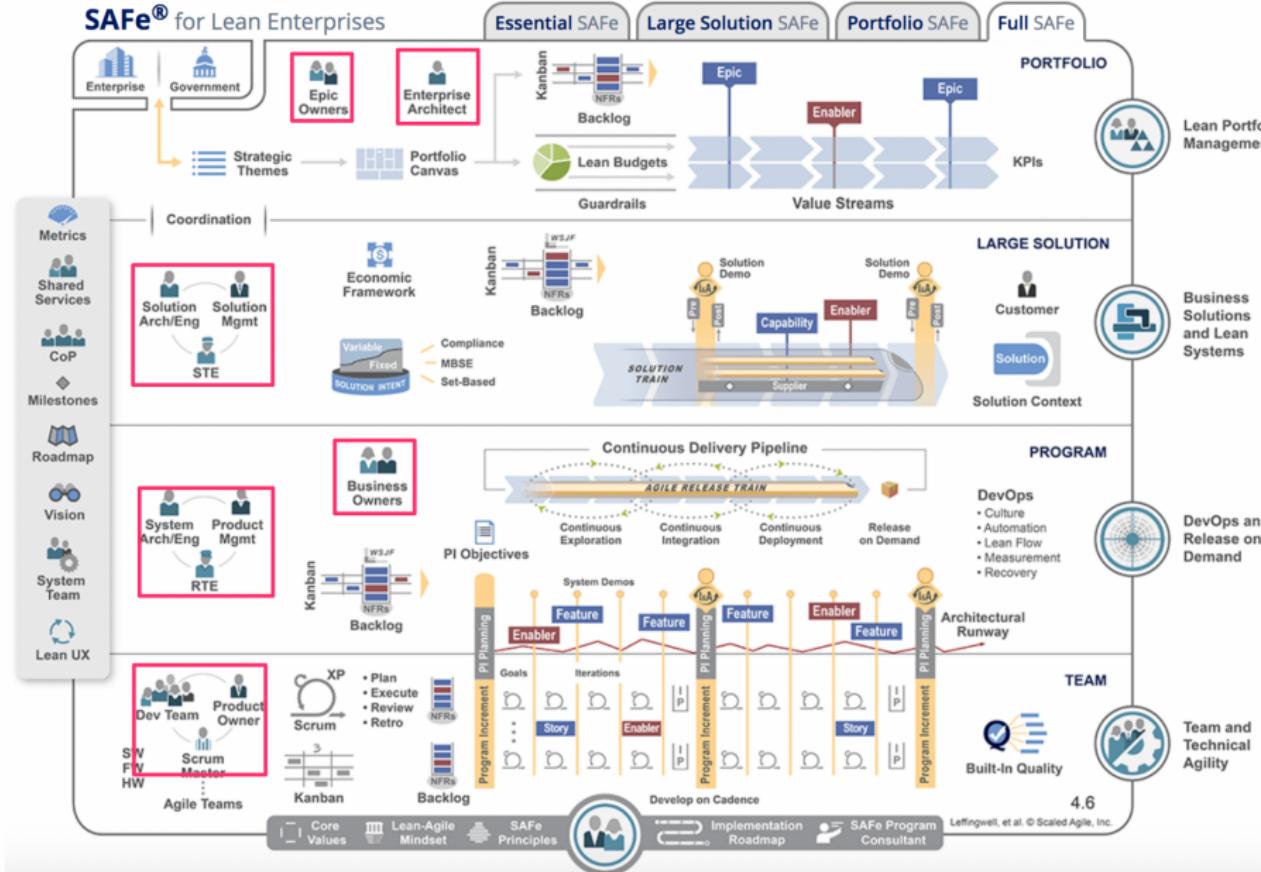


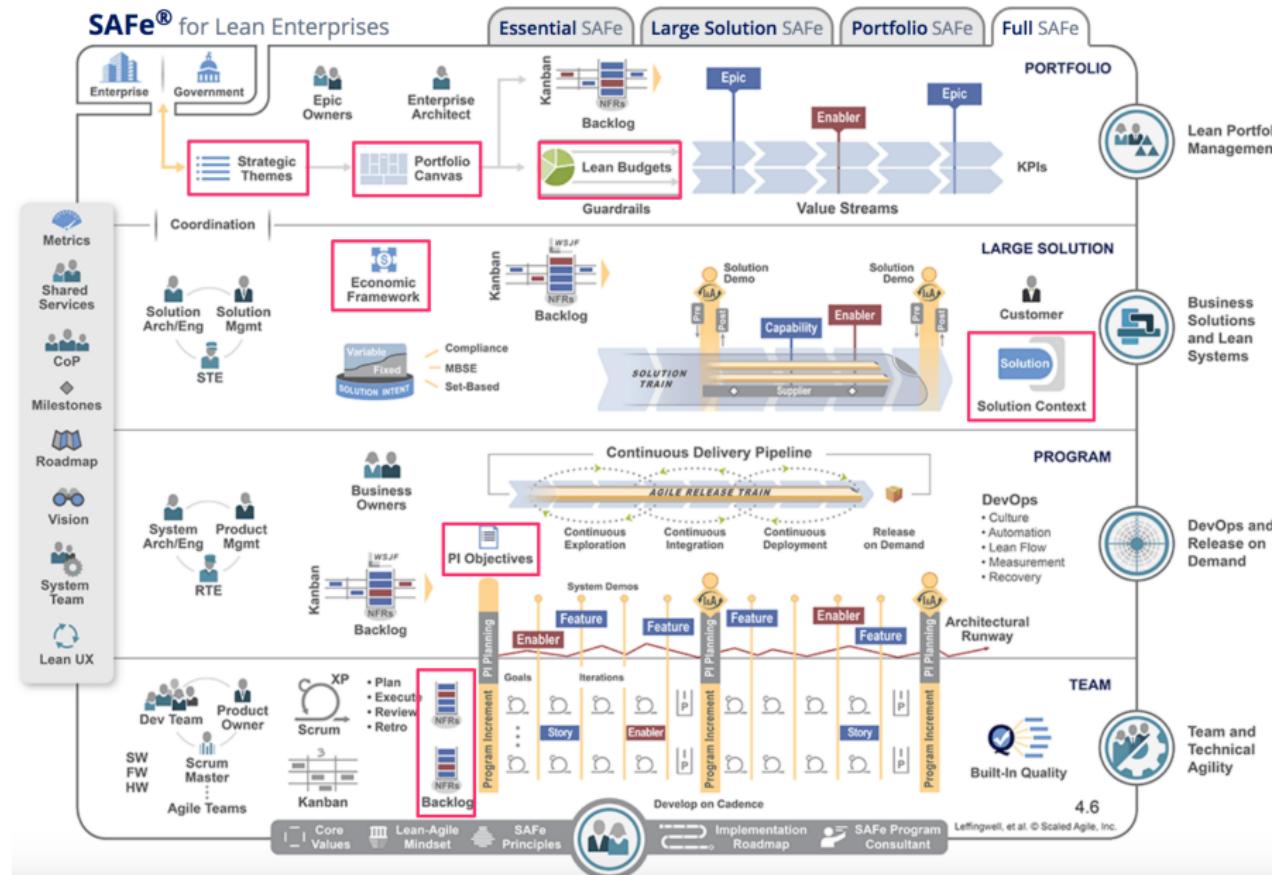
Luento 10

28.11.2023

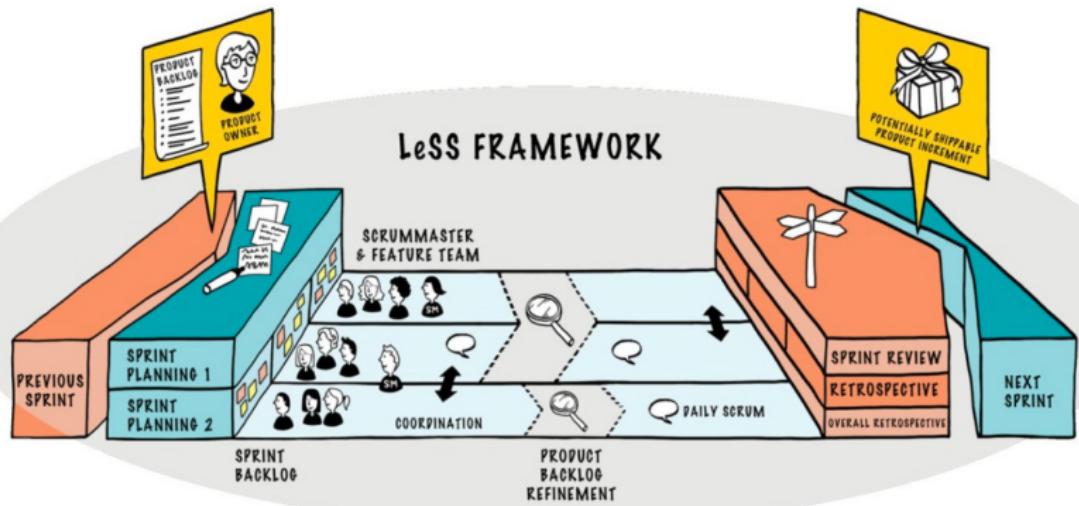
# SAFe: paljon rooleja



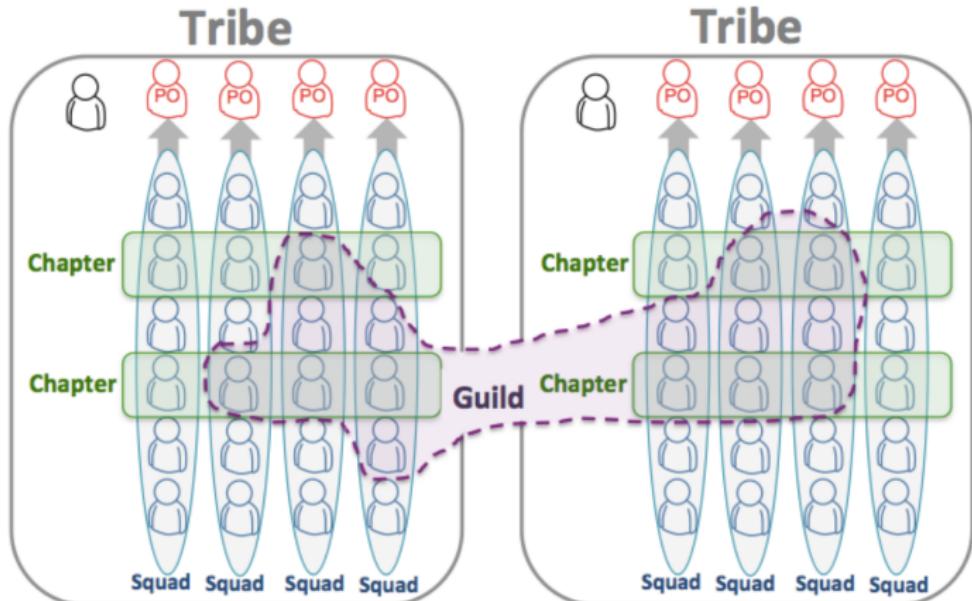
# SAFe: paljon artefakteja



# LeSS



# Spotifyn malli



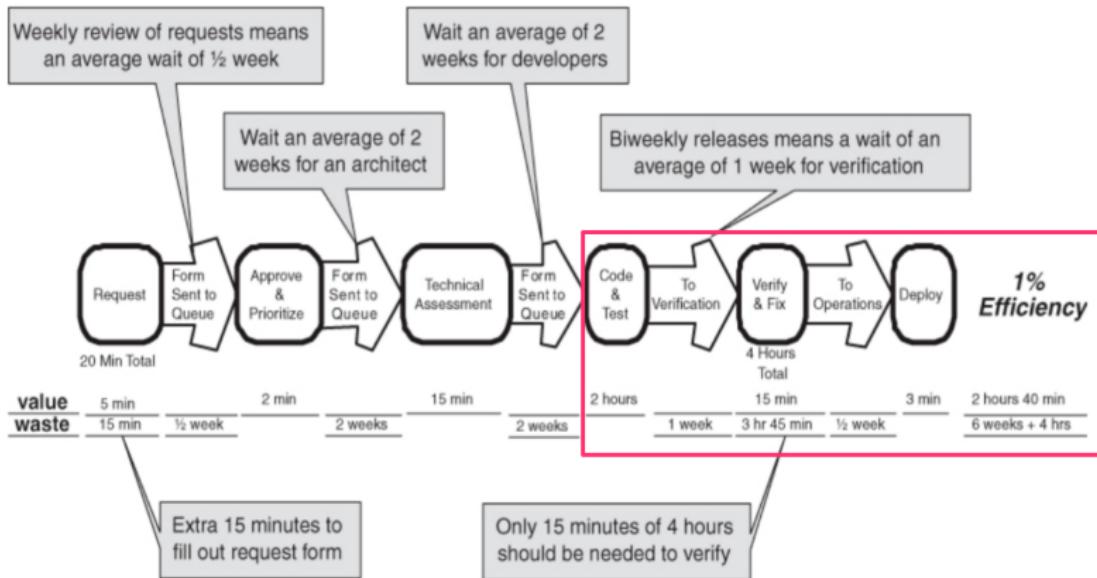
Luento 9

29.11.2023

# Lean thinking house

|  |  |   |
|--|--|---|
| Sustainable shortest lead time, best quality and value (to people and society), most customer delight, lowest cost, high morale, safety  |  |   |
| <p><b>Respect for People</b></p> <ul style="list-style-type: none"><li>- don't trouble your 'customer'</li><li>- "develop people, then build products"</li><li>- no wasteful work</li><li>- teams &amp; individuals evolve their own practices and improvements</li><li>- build partners with stable relationships, trust, and coaching in lean thinking</li><li>- develop teams</li></ul> | <p><b>Product Development</b></p> <ul style="list-style-type: none"><li>- long-term great engineers</li><li>- mentoring from manager-engineer-teacher</li><li>- cadence</li><li>- cross-functional</li><li>- team room + visual mgmt</li><li>- entrepreneurial chief engineer/product mgr</li><li>- set-based concurrent dev</li><li>- create more knowledge</li></ul> <p><b>14 Principles</b></p> <ul style="list-style-type: none"><li>long-term, flow, pull, less variability &amp; overburden, Stop &amp; Fix, master norms, simple visual mgmt, good tech, leader-teachers from within, develop exceptional people, help partners be lean, Go See, consensus, reflection &amp; kaizen</li></ul> | <p><b>Continuous Improvement</b></p> <ul style="list-style-type: none"><li>- Go See</li><li>- kaizen</li><li>- spread knowledge</li><li>- small, relentless</li><li>- retrospectives</li><li>- 5 Whys</li><li>- eyes for waste<ul style="list-style-type: none"><li>* variability, overburden, NVA ... (handoff, WIP, info scatter, delay, multi-tasking, defects, wishful thinking..)</li></ul></li><li>- perfection challenge</li><li>- work toward flow (lower batch size, Q size, cycle time)</li></ul> |
| Management applies and teaches lean thinking, and bases decisions on this long-term philosophy   |  |   |

# Perimmäisen syyn analyysi: five whys





Luento 8

21.11.2023

## Laadukkaan koodin tuntomerkejä

## Laadukkaan koodin tuntomerkejä

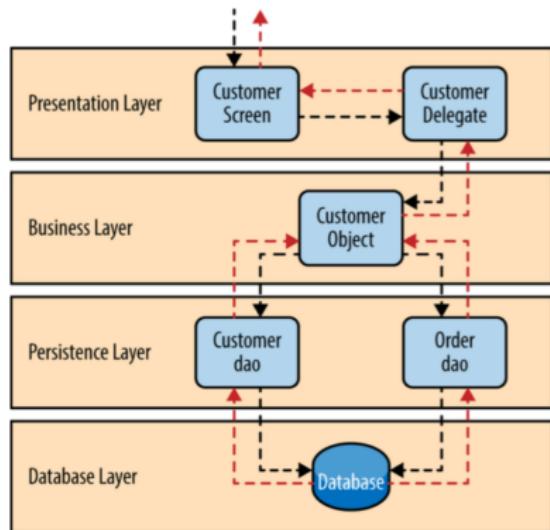
- ▶ Laadukkaalla koodilla joukko yhteneviä ominaisuuksia, tai *laatuattribuutteja*, esim. seuraavat:
  - ▶ kapselointi
  - ▶ korkea koheesion aste
  - ▶ riippuvuuksien vähäisyys
  - ▶ toisteettomuus
  - ▶ testattavuus
  - ▶ selkeys

Luento 7

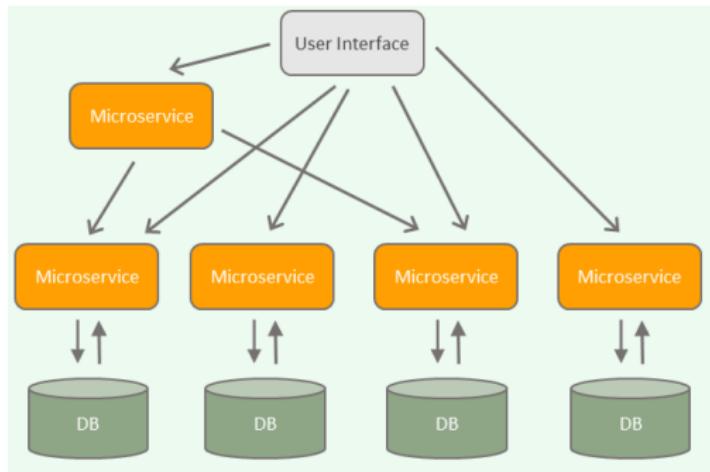
20.11.2023

# Kerrosarkkitehtuuri

- ▶ Kerros on kokoelma toisiinsa liittyviä olioita, jotka muodostavat toiminnallisuuden suhteen loogisen kokonaisuuden



# Mikropalveluarkkitehtuuri

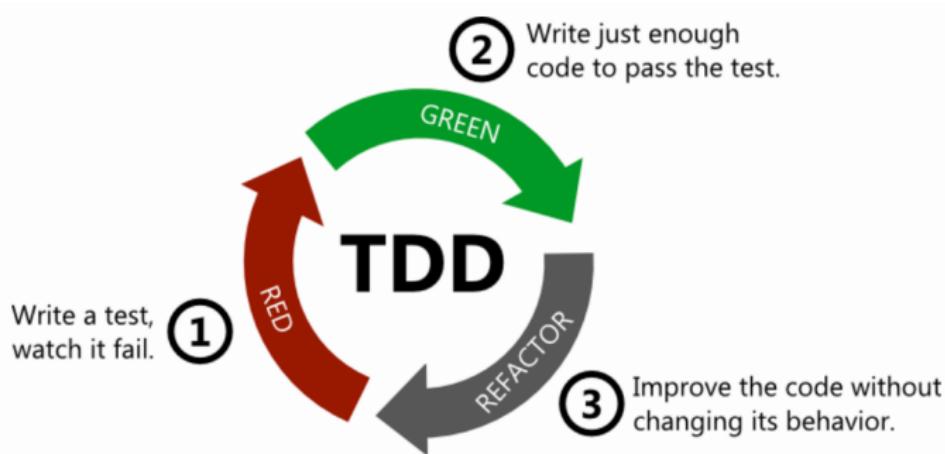


- ▶ sovellus koostataan useista (jopa sadoista) pienistä verkossa toimivista autonomisista palveluista

Luento 6

14.11.2023

# Test driven development (TDD)



1. Kirjoitetaan sen verran testiä että testi ei mene läpi
2. Kirjoitetaan koodia sen verran, että testi menee läpi
3. Jos huomataan koodin rakenteen menneen huonoksi refaktoroidaan koodin rakenne paremmaksi
4. Jatketaan askeleesta 1

# Testit asiakkan kielellä

```
*** Test Cases ***
Register With Valid Username And Password
| Input Credentials  kalle  kalle123
| Output Should Contain  New user registered

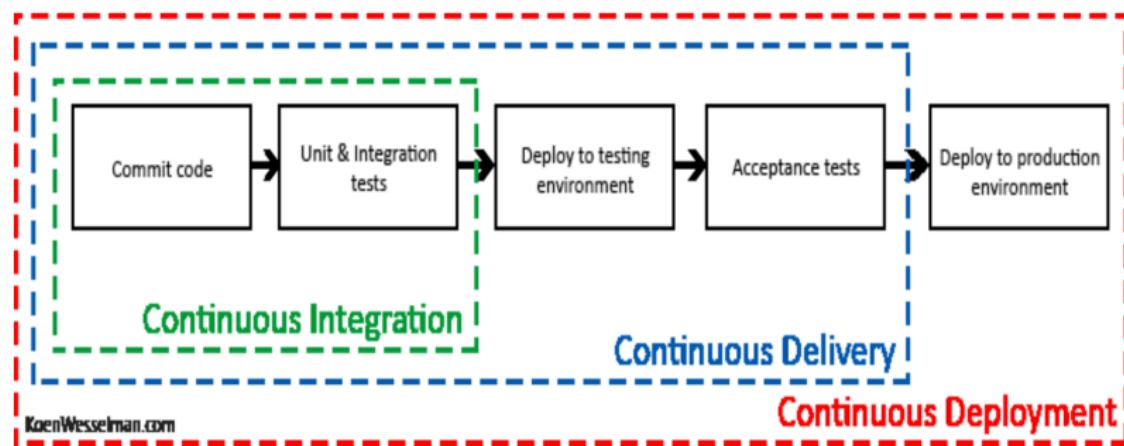
Register With Already Taken Username And Valid Password
| Input Credentials  kalle  kalle123
| Input Register Command
| Input Credentials  kalle  foobarfoo1
| Output Should Contain  User with username kalle already exists

Register With Too Short Username And Valid Password
| Input Credentials  k  kalle123
| Output Should Contain  Username too short

Register With Valid Username And Too Short Password
| Input Credentials  kalle  k
| Output Should Contain  Password too short
```

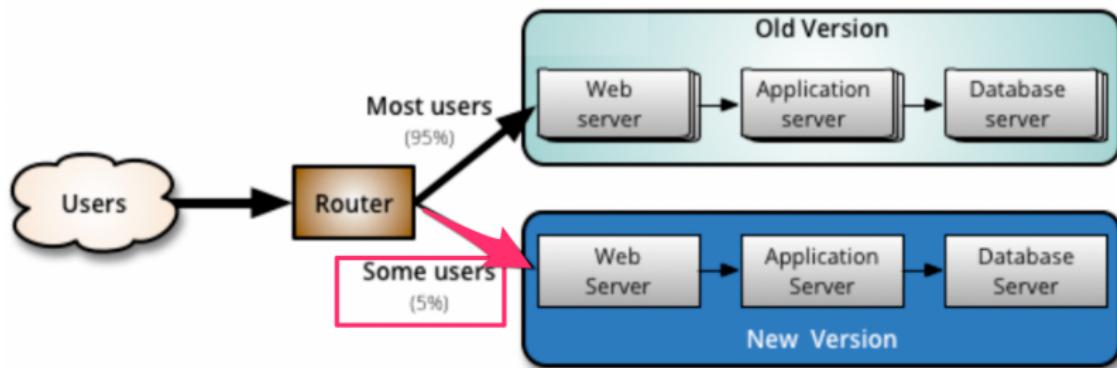
# Deployment pipeline

- Vaiheet, joiden suorittaminen edellytetään, että commitattu koodi saadaan siirrettyä staging/tuotantojärjestöön



# Canary release

- ▶ Kaksi rinnakkaista tuotanto-ympäristöä, joista uudet ominaisuudet viedään toiseen



Luento 5

13.11.2022

# Testauksen tasot

- ▶ *Yksikkötestaus* (unit testing)
  - ▶ Yksittäisten luokkien, metodien ja moduulien testaus erillään muusta kokonaisuudesta
- ▶ *Integraatiotestaus* (integration testing)
  - ▶ Yksittäin testattujen komponenttien liittäminen yhteen eli integrointi ja kokonaisuuden testaus
- ▶ *Järjestelmätestaus* (system testing)
  - ▶ Toimiiko ohjelmisto vaatimuksiin kirjatulla tavalla?
  - ▶ Tutkii järjestelmää kokonaisuudessaan: *end to end -testaus*
  - ▶ Jakautuu useisiin alalajeihin
- ▶ *Käyttäjän hyväksymistestaus* (user acceptance testing)
  - ▶ Loppukäyttäjän tuotteelle suorittama testaus

# Testisyötteiden valinta: palautussovellus

- Mitä testitapauksia kannattaisi valita palautussovelluksen testaamiseen?

## Create a submission for part2

Mark all exercises you have done (check the box if the exercise is done)

1  2  3  4  5  6  7  8  9  10  11  12

Mark all

Clear all

Used hours (reading the material and completing exercises)

GitHub repository

[https://github.com/mluukkai/put\\_your\\_repository\\_name\\_here](https://github.com/mluukkai/put_your_repository_name_here)

Comments

Pressing send will submit this whole part. Any exercises you have not marked done above for this part can not be marked done later. If you by accident submit the wrong number of exercises contact the course teacher or Discord admins.

Send

Cancel

## Ohtuvarasto: tyhjä, puolitäysi, täysi

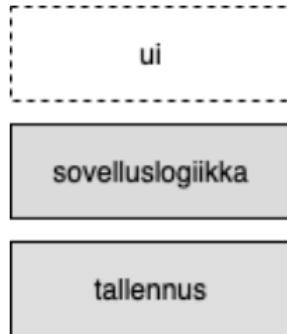
```
class Varasto:
    def __init__(self, tilavuus, alkusalto = 0):
        self.tilavuus = tilavuus
        self.saldo = alkusalto

    def ota_varastosta(self, maara):
        if maara < 0:
            return 0.0

        if maara > self.saldo:
            kaikki_mita_voidaan = self.saldo
            self.saldo = 0.0
            return kaikki_mita_voidaan

        self.saldo = self.saldo - maara
        return maara
```

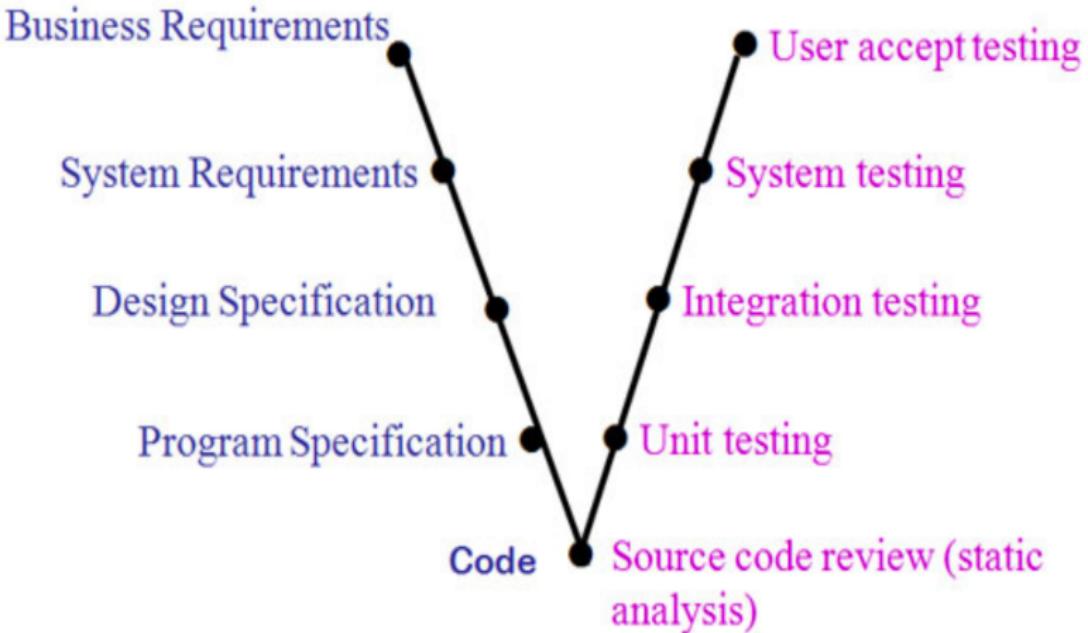
► Rakenteeseen perustuva integraatio



► Ominaisuksiin perustuva integraatio

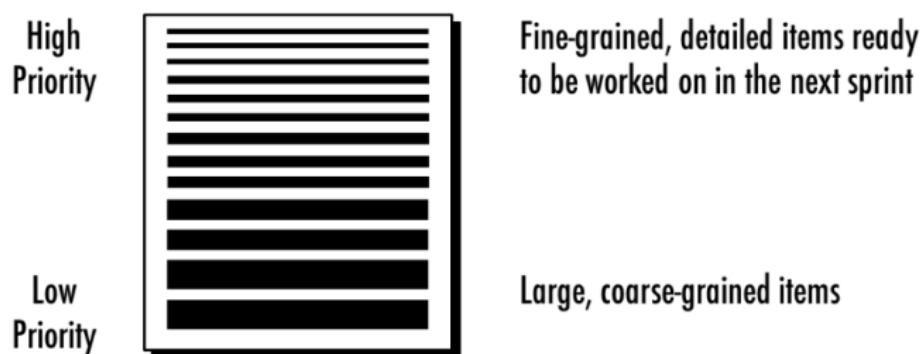


# “V-malli”

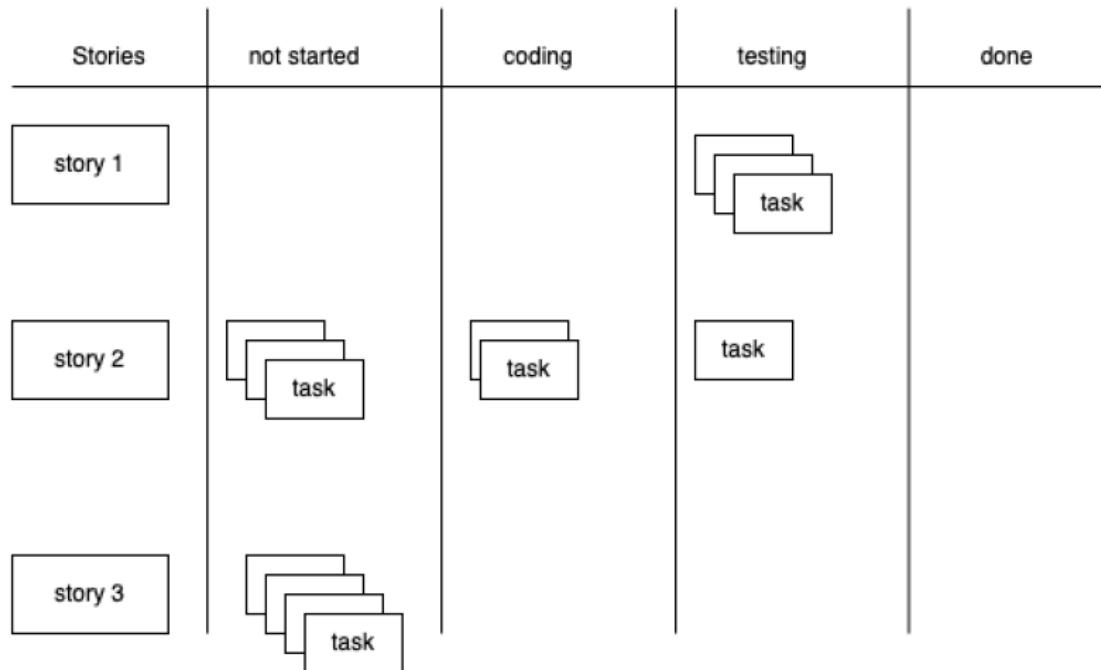


# Hyvä product backlog on DEEP

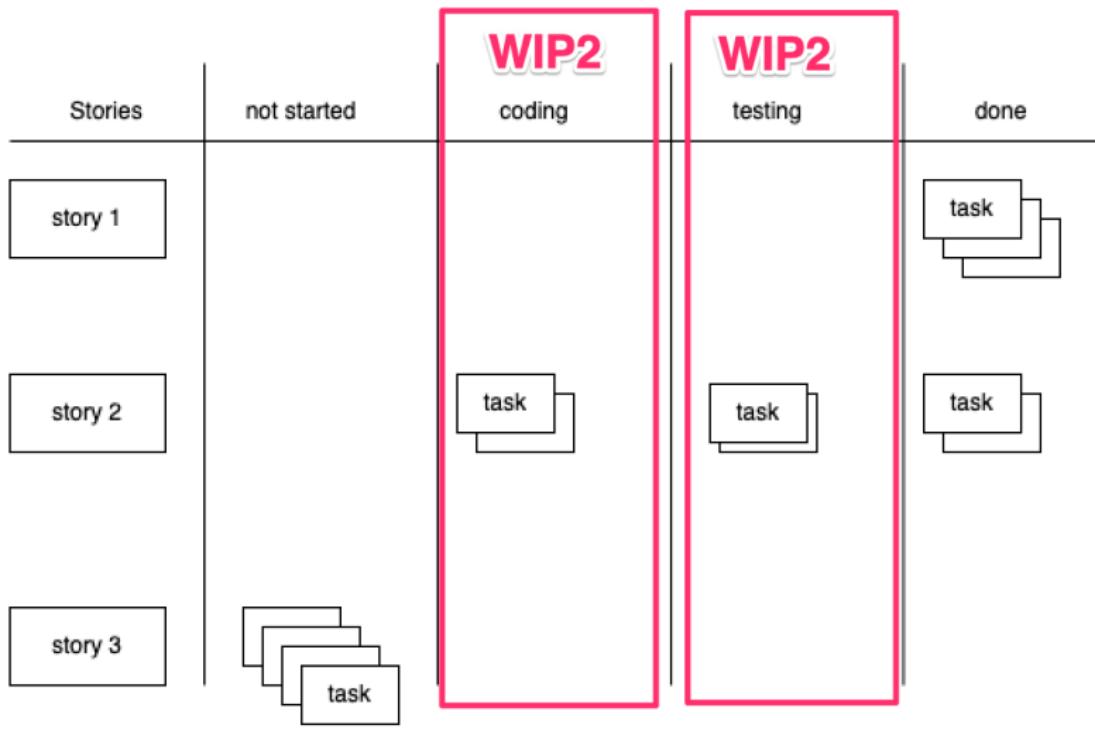
- ▶ Detailed appropriately
- ▶ Estimated
- ▶ Emergent
- ▶ Prioritized



# Sprint backlog

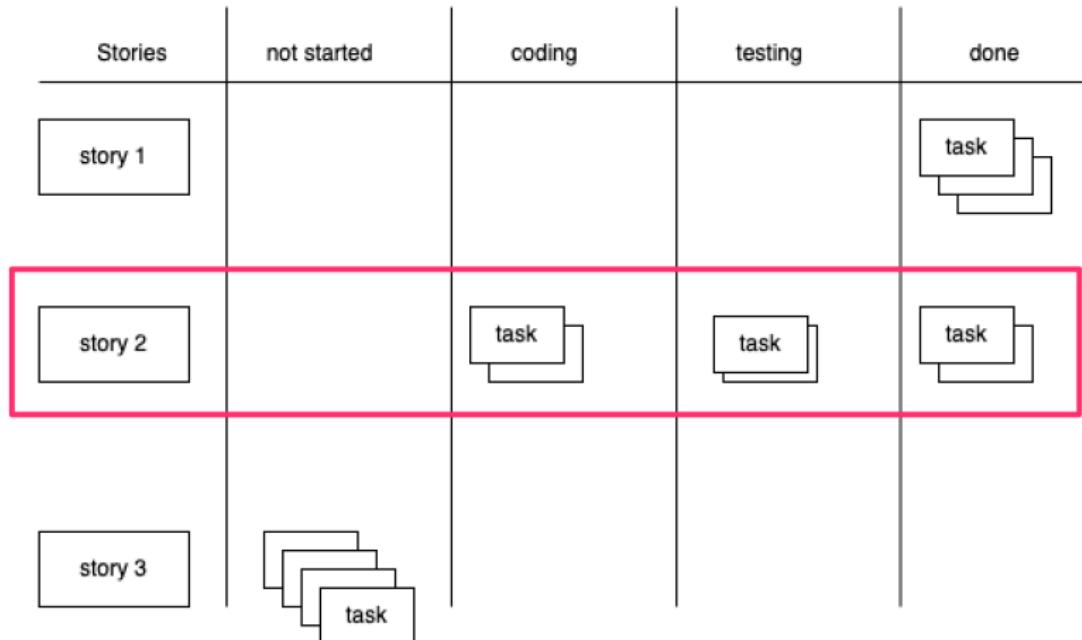


# WIP-rajoitteet



# WIP-rajoitteet

**WIP vain yksi story työn alla**



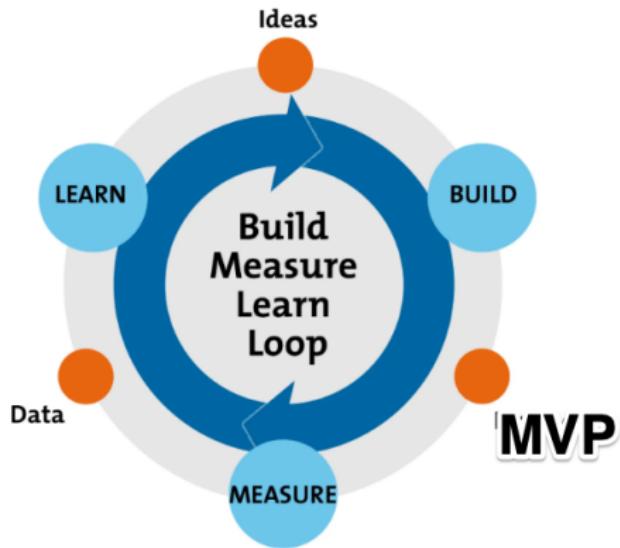
Luento 3

6.11.2023

# Ohjelmiston elinkaari (software lifecycle)

- ▶ **Vaatimusten analysointi ja määrittely**
- ▶ Suunnittelu
- ▶ Toteutus
- ▶ Testaus
- ▶ Ohjelmiston ylläpito ja evoluutio

# Vaatimusmäärittely 2010-luvulla: Lean startup



## User story

- ▶ Mike Cohn:
  - ▶ *A user story describes **functionality that will be valuable** to either user or purchaser of software.*
- ▶ User stories are composed of three aspects:
  1. **A written description**
  2. **Conversations**
  3. **Tests**

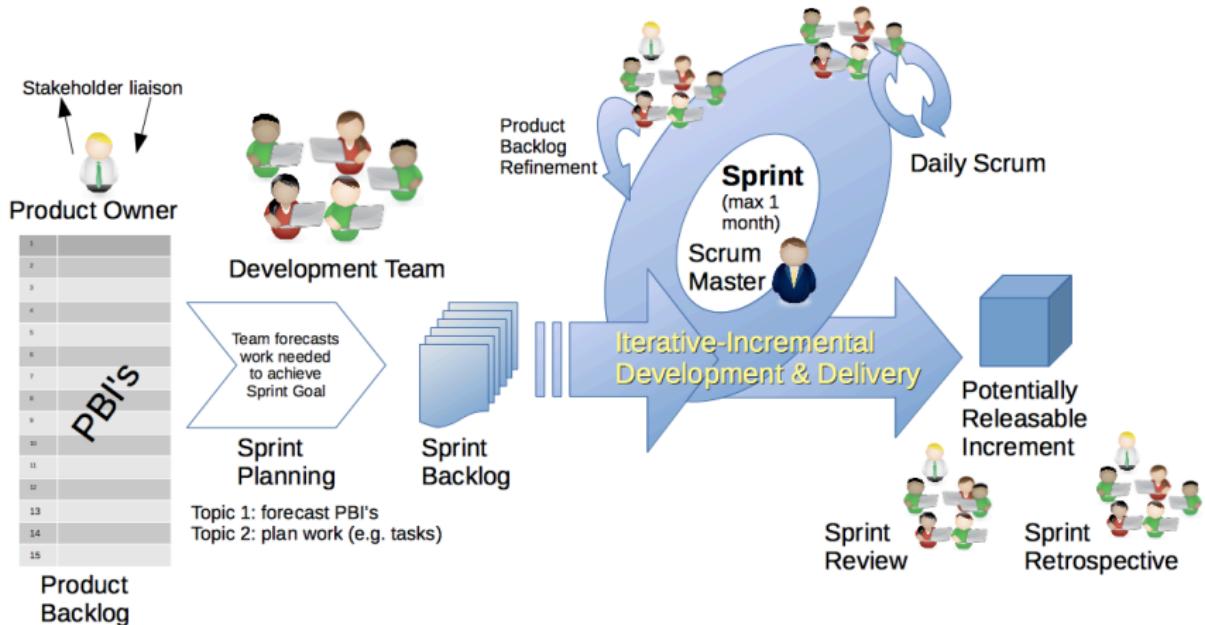
## Hyvän storyn kriteerit

- ▶ Bill Wake *INVEST in good User Stories*, kuusi toivottavaa ominaisuutta
  - ▶ Independent
  - ▶ Negotiable
  - ▶ Valuable to user or customer
  - ▶ Estimable
  - ▶ Small
  - ▶ Testable

Luento 2

31.10.2022

# Scrum kuvana



Luento 1

30.10.2022

# Ohjelmiston elinkaari (software lifecycle)

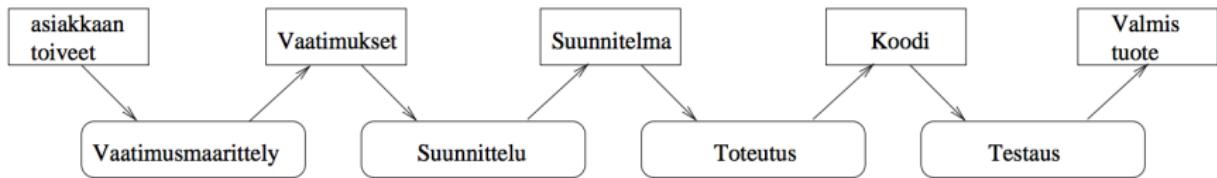
Riippumatta tyylistä ja tavasta, jolla ohjelmisto tehdään, käy ohjelmisto läpi seuraavat *vaiheet*

- ▶ Vaatimusten analysointi ja määrittely
- ▶ Suunnittelu
- ▶ Toteutus
- ▶ Testaus
- ▶ Ohjelmiston ylläpito ja evoluutio

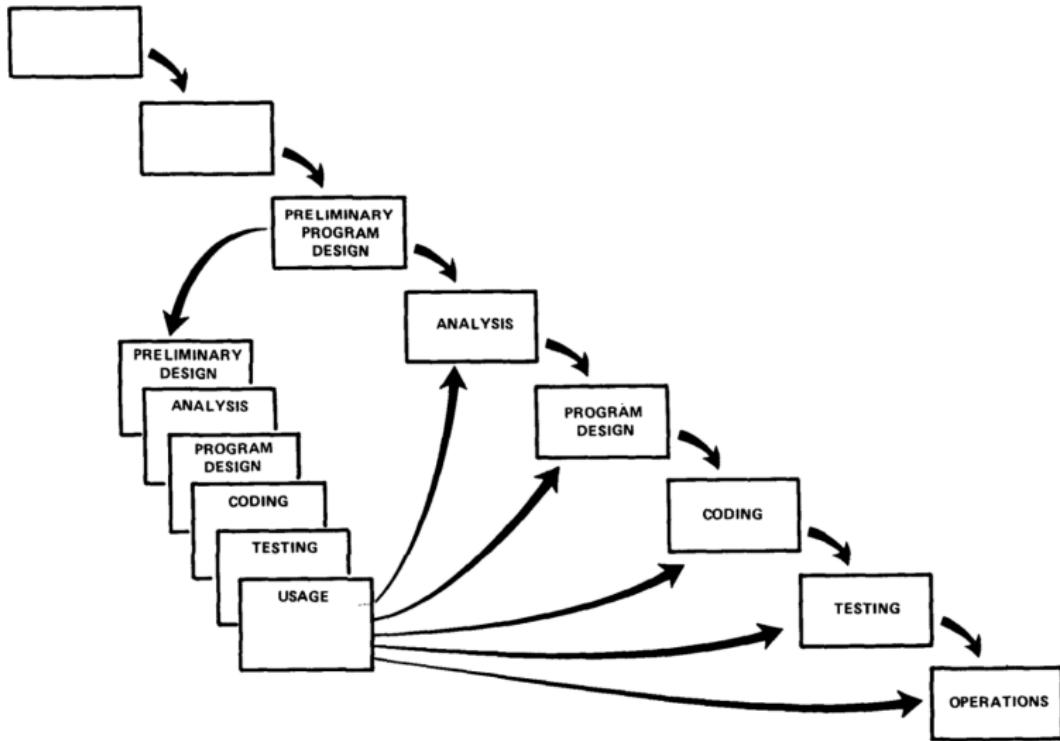
Vaiheista muodostuu ohjelmiston “elinkaari”

# Vesiputousmalli

Winston W. Royce: Management of the development of Large Software, 1970



# Roycen kahden iteraation malli



# Iteratiivinen ohjelmistokehitys

