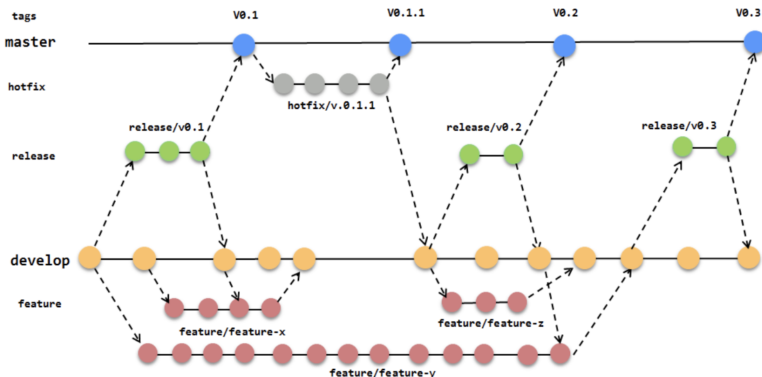


Luento 7

20.11.2023

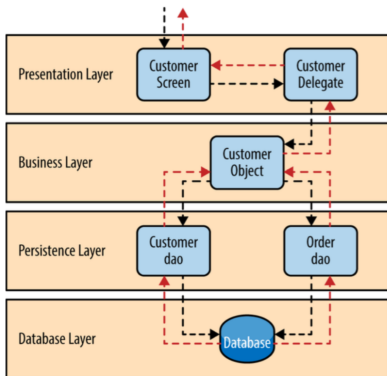
Branchayskäytänteet

- Toisin kuin aiemmissa versionhallintajärjestelmissä, Gitissä brancien teko on erittäin helppoa
 - Johtanut monimutkaisiin branchayskäytänteisiin

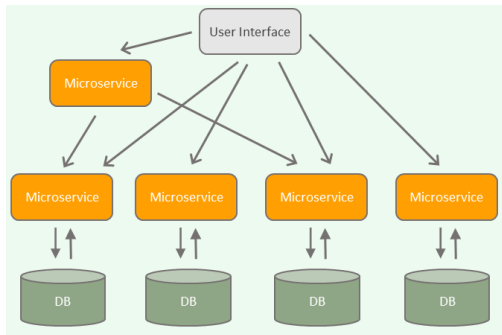


Kerrosarkkitehtuuri

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



Mikropalveluarkkitehtuuri

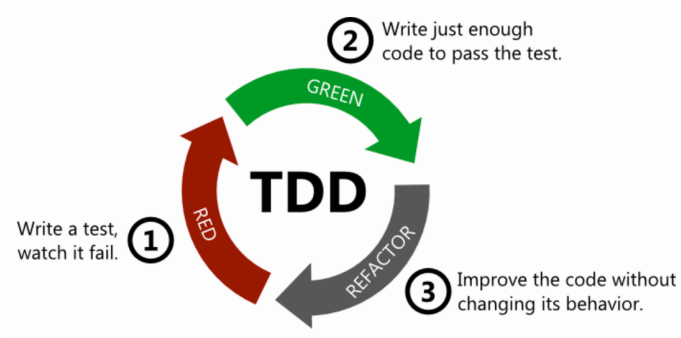


- sovellus koostetaan useista (jopa sadoista) pienistä verkossa toimivista autonomisista palveluista

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
```

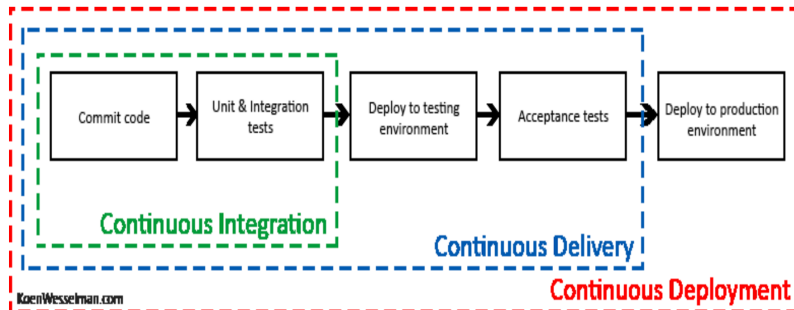
```
Register With Valid Username And Too Short Password
```

```
    Input Credentials  kalle  k
```

```
    Output Should Contain  Password too short
```

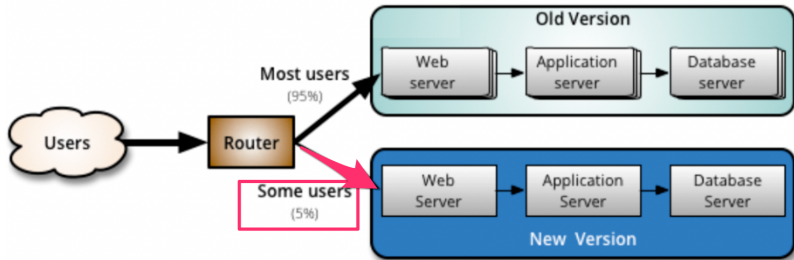
Deployment pipeline

- Vaiheet, joiden suorittaminen edellyttää, että commitattu koodi saadaan siirrettyä staging/tuotantoympäristöön



Canary release

- Kaksi rinnakkaista tuotantoympäristöä, 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

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

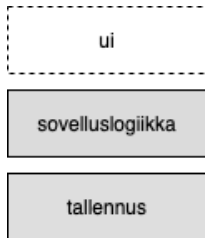
```
class Varasto
    def __init__(self, tilavuus, alku_saldo = 0):
        self.tilavuus = tilavuus
        self.saldo = alkusalto

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

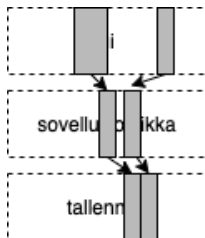
        if maara > self.saldo:
            kaikki_mita_voidaan = self.saldo
            self.saldo = 0.0
            return kaikki_mita_voidaan

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

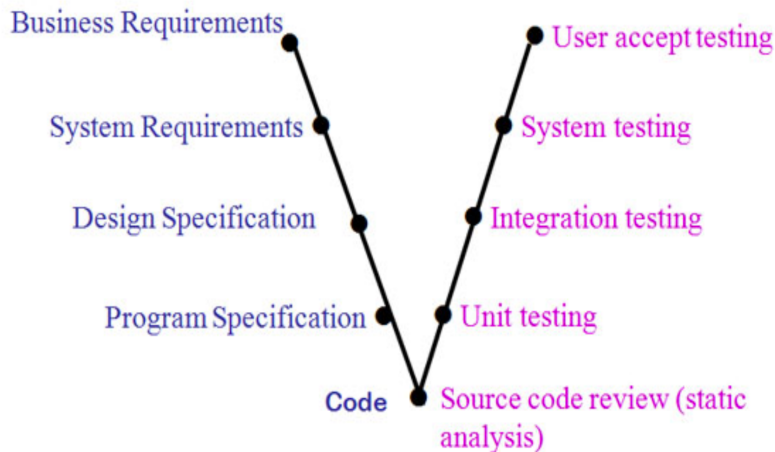
► Rakenteeseen perustuva integraatio



► Ominaisuuksiin perustuva integraatio



“V-malli”



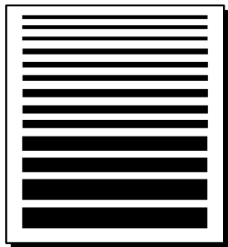
Luento 4

7.11.2022

Hyvä product backlog on DEEP

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

High
Priority

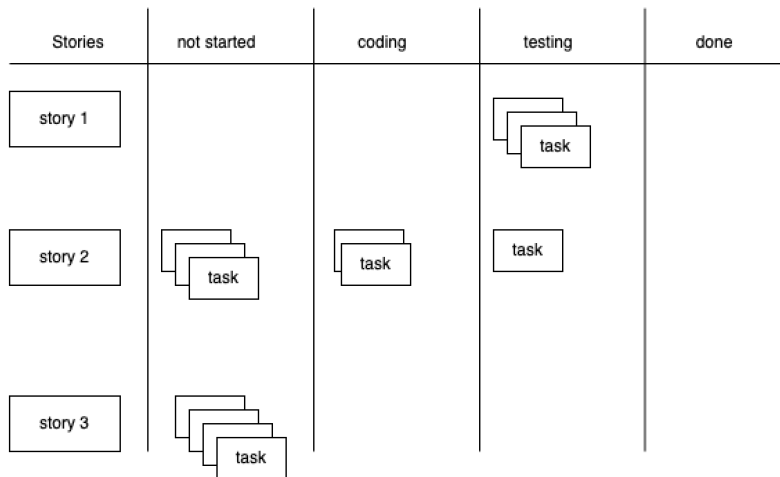


Low
Priority

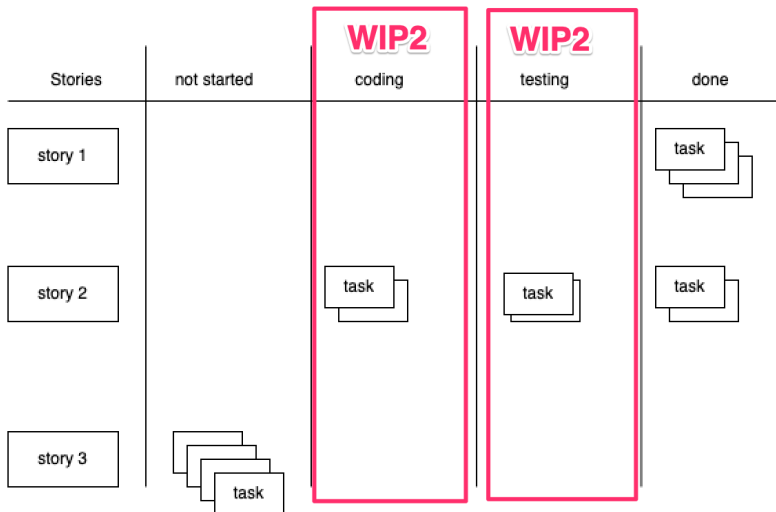
Fine-grained, detailed items ready
to be worked on in the next sprint

Large, coarse-grained items

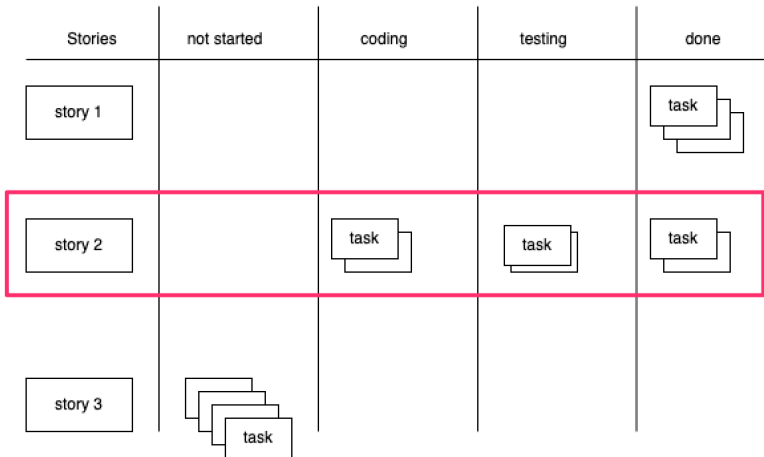
Sprint backlog



WIP-rajoitteet



WIP vain yksi story työn alla



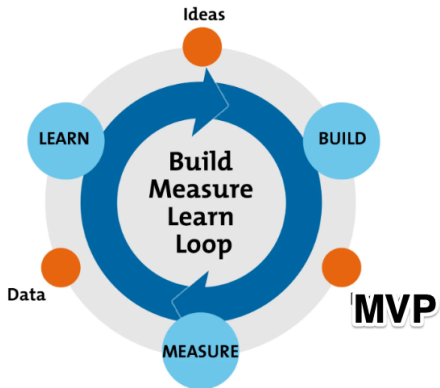
Luento 3

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**

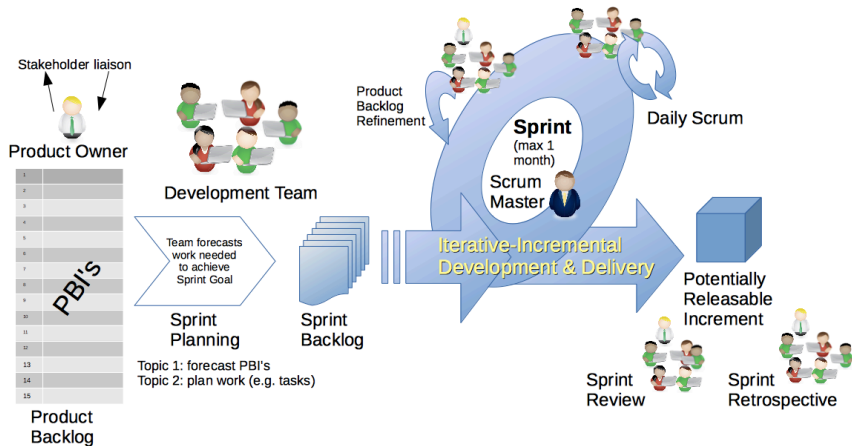
Hyvän storyn kriteerit

- ▶ 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

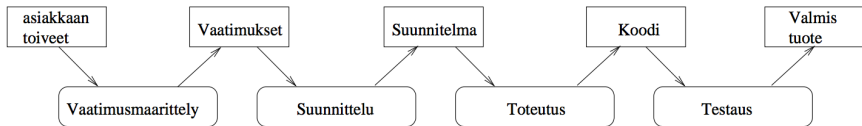
Ohjelmiston elinkaari (software lifecycle)

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

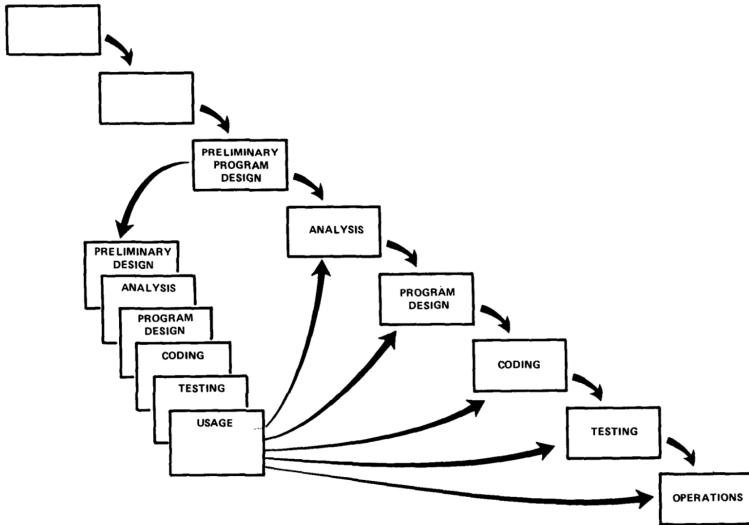
- ▶ Vaatimusten analysointi ja määrittely
- ▶ Suunnittelu
- ▶ Toteutus
- ▶ Testaus
- ▶ Ohjelmiston ylläpito ja evoluutio

Vaiheista muodostuu ohjelmiston “elinkaari”

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



Roycen kahden iteraation malli



Iteratiivinen ohjelmistokehitys

