

# Ohjelmistotuotanto

Syksy 2025

Luento 3

3.11.2025

## Kurssipalaute

- ▶ Kurssipalaute
  - ▶ Kurssilla lopussa kerättävän palautteen lisäksi ns. jatkuva palaute <https://norppa.helsinki.fi>

## Kurssipalaute

- ▶ Kurssipalaute
  - ▶ Kurssilla lopussa kerättävän palautteen lisäksi ns. jatkuva palaute <https://norppa.helsinki.fi>
- ▶ laskareita tehdessä oli haastavaa pysyä kärryllä siitä, mikä osa olikaan varsinaista tehtävää

# Paja

- ▶ BK107
  - ▶ ma 14.30-16.30
  - ▶ ti 12-16
  - ▶ to 12-16
  - ▶ pe 12-14

# Miniprojektit

- ▶ Käynnistyttyvät 10.11 alkavalla viikolla
- ▶ **Ilmoittautumisen deadline la 8.11. klo 23.59**
- ▶ Aloitustilaisuudet
  - ▶ ma 14-16
  - ▶ to 14-16
  - ▶ ke 10-12
  - ▶ ke 12-14
  - ▶ to 14-16
  - ▶ to 16-18
- ▶ Seuraavat viikot: sprinttien katselmuks ja suunnittelu samassa aikaikkunassa
- ▶ Loppudemot
  - ▶ ke 10.12. klo 10-12 B123
  - ▶ to 11.12. klo 10-12 A111

# CurreChat

- ▶ Käytettävissä HY:n GPT-chat
  - ▶ <https://curre.helsinki.fi/chat>

# CurreChat

- ▶ Käytettävissä HY:n GPT-chat
- ▶ <https://curre.helsinki.fi/chat>

 CURRECHAT

TKT20006  
**Ohjelmistotuotanto**

27.10.–10.01.2026  
kurssin asetukset

 Tyhjennä keskustelu

 Kielimalli: gpt-4o

 Valitse alustus

 Tallenna sähköpostina

 Keskustelun asetukset

 Käyttöohje

Ota yhteyttä tukeen:  
[opetuksen teknologia@helsinki.fi](mailto:opetuksen teknologia@helsinki.fi)

Palvelinversio v3.19.13   
Selainversio v3.19.13  
Uppime: noin 8 tuntia

lääpi, miten tämä metodi: [KURSSICHATIT](#) [OPETTAMANI KURSSIT](#) [KURSSITILASTOT](#) [YLLÄPITO](#) [SISÄLLÖT](#)

**Metodin tarkoitus:**

ota\_varastosta -metodi käsitteli varaston saldoa. Se vähentää pyydetyn määärän varastosta, mutta huomioi, ettei varastosta voida ottaa enempää kuin sen saldo. Lisäksi negatiivisia määriä ei käsitellä.

**Parametrit:**

- self: Viittaa luokkaan, johon metodi kuuluu. Tämän avulla metodi voi käyttää luokan attribuutteja, kuten selfsaldo.
- maara: Määärä, joka halutaan ottaa varastosta.

**Toiminta:**

1. Negatiivisen määärän käsitteily:

```
python
if maara < 0:
    return 0.0
```

Jos pyydetty määrä (maara) on negatiivinen, metodi palauttaa arvon 0.0. Negatiivisia määriä ei käsitellä, eikä varaston saldo muutu.

2. Jos pyydetty määrä ylittää varaston saldon:

```
python
if maara > selfsaldo:
    kaikki_mita_voidaan = selfsaldo
```

# CurreChat

- ▶ Käytettävissä HY:n GPT-chat
  - ▶ <https://curre.helsinki.fi/chat>

The screenshot shows the CURRECHAT application interface. On the left is a sidebar with the following sections and content:

- TKT20006**
- Ohjelmistotuotanto**
- 27.10.–10.01.2026
- [kurssin asetukset](#)
- [Tyhjennä keskustelu](#)
- [Kielimalli: gpt-4o](#)
- [Valitse alustus](#)
- [Tallenna sähköpostina](#)
- [Keskustelun asetukset](#)
- [Käyttöohje](#)

Below the sidebar, there is a message from the user "Ota\_varastosta" asking about a method. The main panel displays the following information:

**Metodin tarkoitus:**  
ota\_varastosta -metodi käsitteli varaston saldoa. Se vähentää pyydetyn määärän varastosta, mutta huomioi, ettei varastosta voida ottaa enempää kuin sen saldo. Lisäksi negatiivisia määriä ei käsitellä.

**Parametrit:**

- self: Viittaa luokkaan, johon metodi kuuluu. Tämän avulla metodi voi käyttää luokan attribuutteja, kuten selfsaldo.
- maara: Määärä, joka halutaan ottaa varastosta.

**Toiminta:**

- Negatiivisen määärän käsitteily:**

```
python
if maara < 0:
    return 0.0
```
- Jos pyydetty määrä (maara) on negatiivinen, metodi palauttaa arvon 0.0. Negatiivisia määriä ei käsitellä, eikä varaston saldo muutu.**
- Jos pyydetty määrä ylittää varaston saldon:**

```
python
if maara > selfsaldo:
    kaikki_mita_voidaan = selfsaldo
```

- ▶ Keskusteluja ei vuodeta mihinkään

# CurreChat: Kurssimateriaali-indeksi

TKT2006

## Ohjelmistotuotanto

27.10.–10.01.2026

[Kurssin asetukset](#)

Tyhjennä keskustelu

Kielimalli: gpt-4o

**Kurssimateriaali**

Tallenna sähköpostina

Keskustelun asetukset

Käyttöohje

Ota yhteyttä tukeen:

[opetusteknologia@helsinki.fi](mailto:opetusteknologia@helsinki.fi)

milloin kurssikoe pidetään

Kurssimateriaali (gpt-4o)

Kurssikoe pidetään tiistaina 16.12. klo 13–16  
luokkahuoneissa A111 ja CHE A110. Muistathan ilmoittautua  
kokeeseen!

Lähteet hakusanalla "kurssikoe ajankohta": etusivu.txt,...

## Hakutulokset

Hakusanat: **kurssikoe ajankohta**

1 Kurssilla käsitellään  
ohjelmistotuotantoprojektiin  
hallinnan, työvaiheiden ja  
työvälineiden perusteita. Erityinen  
painotus ketterissä...

2 Kokeessa suurin paino tulee  
olemaan teoriassa ja sen  
soveltamisessa käytäntöön.  
Laskareiden teknisimpia asioita,  
kuten versionhallintaa ei kokees...

Lue lisää

# CurreChat: Kurssimateriaali-indeksi

TKT2006

## Ohjelmistotuotanto

27.10.–10.01.2026

[Kurssin asetukset](#)

∅ Tyhjennä keskustelu

✉ Kieliruumi: gpt-4o

**+ Kurssimateriaali**

✉ Tallenna sähköpostina

☰ Keskustelun asetukset

💡 Käyttöohje

Ota yhteyttä tukeen:

[opetusteknologia@helsinki.fi](mailto:opetusteknologia@helsinki.fi)

milloin kurssikoe pidetään

↳ Kurssimateriaali (gpt-4o)

Kurssikoe pidetään tiistaina 16.12. klo 13–16 luokkahuoneissa A111 ja CHE A110. Muistathan ilmoittautua kokeeseen!

📄 Lähteet hakusanalla 'kurssikoe ajankohta': etusivu.txt,...

## Hakutulokset

Hakusanat: **kurssikoe ajankohta**

1 Kurssilla käsitellään ohjelmistotuotantoprojektiin halinnan, työvaiheiden ja työvälineiden perusteita. Erityinen painotus ketterissä...

2 Kokeessa suurin paino tulee olemaan teoriassa ja sen soveltamisessa käytäntöön. Laskareiden teknisimpia asioita, kuten versionhallintaa ei kokees...

☰ Lue lisää

- ▶ Ei (ehkä) hallusinoi
- ▶ Osin herkkä kysymysten sanamuodon suhteen

# Ohjelmiston elinkaari (software lifecycle)

# Ohjelmiston elinkaari (software lifecycle)

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

# Vaatimusmäärittely engl requirements engineering

- ▶ Keskeisin ongelma ohjelmistotuotantoprosessissa on määritellä asiakkaan vaatimukset rakennettavalle ohjelmistolle

# Vaatimusmäärittely engl requirements engineering

- ▶ Keskeisin ongelma ohjelmistotuotantoprosessissa on määritellä asiakkaan vaatimukset rakennettavalle ohjelmistolle
- ▶ Jakaantuvat kahteen luokkaan
  - ▶ Toiminnalliset vaatimukset
    - ▶ ohjelman toiminnot
  - ▶ Ei-toiminnalliset vaatimukset
    - ▶ koko ohjelmistoa koskevat "laatuvaatimukset" ja
    - ▶ toimintaympäristön asettamat rajoitteet

# Vaatimusmäärittely engl requirements engineering

- ▶ Keskeisin ongelma ohjelmistotuotantoprosessissa on määritellä asiakkaan vaatimukset rakennettavalle ohjelmistolle
- ▶ Jakaantuvat kahteen luokkaan
  - ▶ Toiminnalliset vaatimukset
    - ▶ ohjelman toiminnot
  - ▶ Ei-toiminnalliset vaatimukset
    - ▶ koko ohjelmistoa koskevat "laatuvaatimukset" ja
    - ▶ toimintaympäristön asettamat rajoitteet
- ▶ Vaatimusmäärittelyn tulee ainakin alkaa ennen ohjelmiston suunnittelua ja toteuttamista
  - ▶ vesiputouksessa vaatimukset määritellään heti alussa
  - ▶ iteratiivisessa ja ketterässä kehityksestä projektin kuluessa

## Vaatimusmäärittelyn vaiheet

- ▶ Vaatimusmäärittelyn luonne vaihtelee paljon riippuen
  - ▶ kehitettävästä ohjelmistosta
  - ▶ kehittäjäorganisaatiosta
  - ▶ ohjelmistokehitykseen käytettävästä prosessimallista

## Vaatimusmäärittelyn vaiheet

- ▶ Vaatimusmäärittelyn luonne vaihtelee paljon riippuen
  - ▶ kehitettävästä ohjelmistosta
  - ▶ kehittäjäorganisaatiosta
  - ▶ ohjelmistokehitykseen käytettävästä prosessimallista
- ▶ Asiakkaan tai asiakkaan edustajan on oltava prosessissa aktiivisesti mukana

## Vaatimusmäärittelyn vaiheet

- ▶ Vaatimusmäärittelyn luonne vaihtelee paljon riippuen
  - ▶ kehitettävästä ohjelmistosta
  - ▶ kehittäjäorganisaatiosta
  - ▶ ohjelmistokehitykseen käytettävästä prosessimallista
- ▶ Asiakkaan tai asiakkaan edustajan on oltava prosessissa aktiivisesti mukana
- ▶ Jaotellaan yleensä muutamaan työvaiheeseen
  - ▶ kartoitus (engl. elicitation)
  - ▶ analyysi
  - ▶ validointi
  - ▶ dokumentointi
  - ▶ hallinnointi

## Vaatimusmäärittelyn vaiheet

- ▶ Vaatimusmäärittelyn luonne vaihtelee paljon riippuen
  - ▶ kehitettävästä ohjelmistosta
  - ▶ kehittäjäorganisaatiosta
  - ▶ ohjelmistokehitykseen käytettävästä prosessimallista
- ▶ Asiakkaan tai asiakkaan edustajan on oltava prosessissa aktiivisesti mukana
- ▶ Jaotellaan yleensä muutamaan työvaiheeseen
  - ▶ kartoitus (engl. elicitation)
  - ▶ analyysi
  - ▶ validointi
  - ▶ dokumentointi
  - ▶ hallinnointi
- ▶ Työvaiheet limittivät ja vaatimusmäärittely etenee spiraalimaisesti tarkentuen

## Vaatimusten kartoituksen menetelmiä

- ▶ Selvitetään järjestelmän sidosryhmät (stakeholders) eli tahot, jotka tekemisissä järjestelmän kanssa

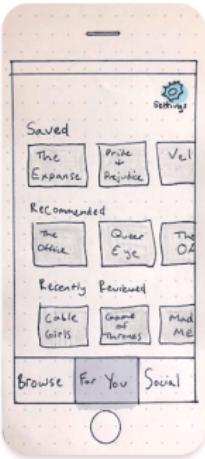
## Vaatimusten kartoituksen menetelmiä

- ▶ Selvitetään järjestelmän sidosryhmät (stakeholders) eli tahot, jotka tekemisissä järjestelmän kanssa
- ▶ Käytetään kaikki mahdolliset keinot:
  - ▶ Haastatellaan sidosryhmien edustajia
  - ▶ Pidetään brainstormaussessioita asiakkaan ja kehittäjien kesken

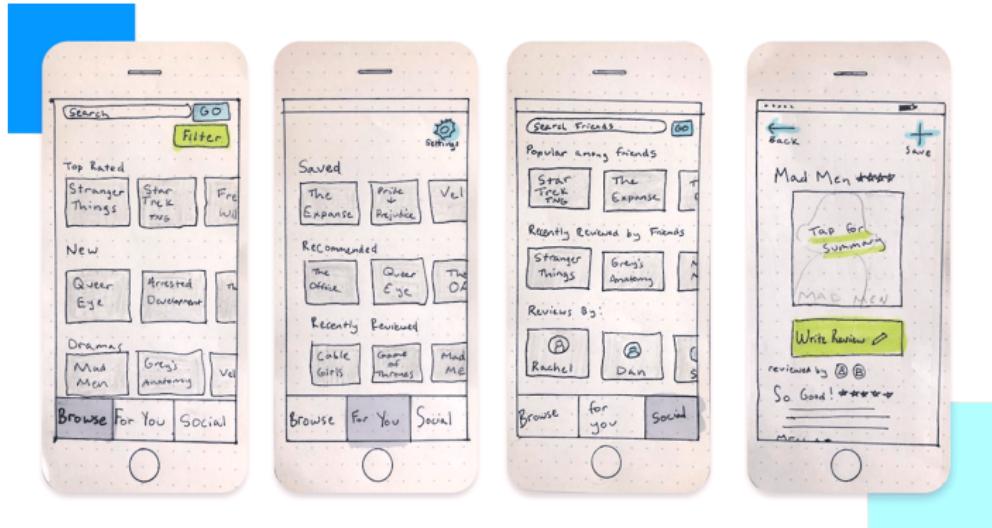
## Vaatimusten kartoituksen menetelmiä

- ▶ Selvitetään järjestelmän sidosryhmät (stakeholders) eli tahot, jotka tekemisissä järjestelmän kanssa
- ▶ Käytetään kaikki mahdolliset keinot:
  - ▶ Haastatellaan sidosryhmien edustajia
  - ▶ Pidetään brainstormaussessioita asiakkaan ja kehittäjien kesken
- ▶ Kehittäjätiimi voi strukturoida vaatimusten kartoitusta
  - ▶ Mietitään *kuviteltuja käyttäjiä* ja keksitään käyttäjille tyypillisiä *käytöskenaarioita*
  - ▶ Tehdään paperiprototyypejä ja käyttöliittymäluonoksia

# Vaatimusten kartoituksen menetelmiä

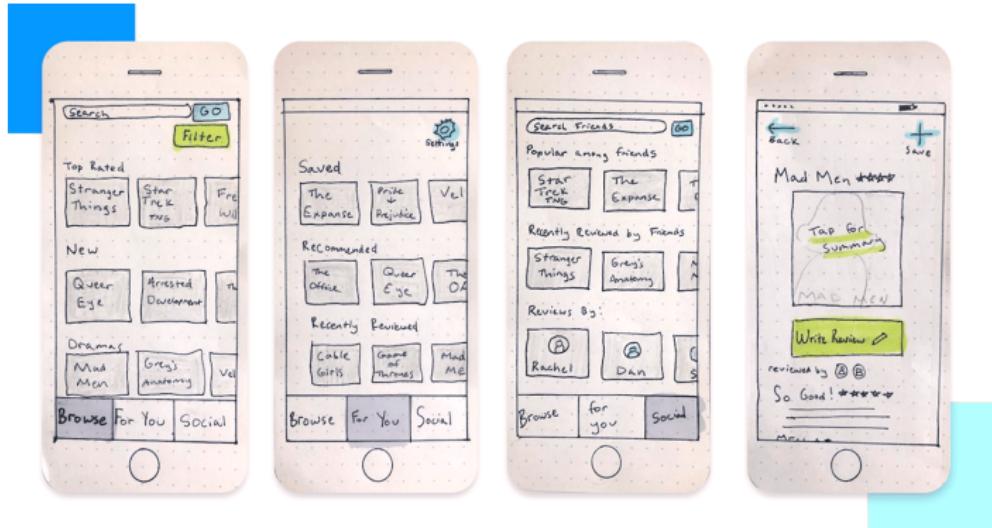


# Vaatimusten kartoituksen menetelmiä



- ▶ Skenarioita ja prototyypejä läpikäymällä asiakas näkemys tarkentuu

# Vaatimusten kartoituksen menetelmiä



- ▶ Skenarioita ja prototyypejä läpikäymällä asiakas näkemys tarkentuu
- ▶ Jos ollaan korvaamassa vanhaa järjestelmää, voidaan havainnoida loppukäyttäjän työskentelyä (etnografia)

- ▶ Kartoitettuja vaatimuksia täytyy **analysoida**, eli ovatko ne
  - ▶ riittävän kattavat
  - ▶ keskenään ristiriidattomia
  - ▶ testattavissa
  - ▶ toteutuminen on mahdollista ja taloudellisesti järkevää

- ▶ Kartoitettuja vaatimuksia täytyy **analysoida**, eli ovatko ne
  - ▶ riittävän kattavat
  - ▶ keskenään ristiriidattomia
  - ▶ testattavissa
  - ▶ toteutuminen on mahdollista ja taloudellisesti järkevää
- ▶ Vaatimukset on myös pakko **dokumentoida** muodossa tai toisessa
  - ▶ Ohjelmistokehittäjiä varten: mitä tehdään
  - ▶ Testaajia varten: toimiiko järjestelmä kuten vaatimukset määrittelevät

- ▶ Kartoitettuja vaatimuksia täytyy **analysoida**, eli ovatko ne
  - ▶ riittävän kattavat
  - ▶ keskenään ristiriidattomia
  - ▶ testattavissa
  - ▶ toteutuminen on mahdollista ja taloudellisesti järkevää
- ▶ Vaatimukset on myös pakko **dokumentoida** muodossa tai toisessa
  - ▶ Ohjelmistokehittäjiä varten: mitä tehdään
  - ▶ Testaajia varten: toimiiko järjestelmä kuten vaatimukset määrittelevät
- ▶ Joskus vaatimusdokumentti toimii oleellisena osana asiakkaan ja kehittäjien välisessä sopimuksessa

- ▶ Kartoitettuja vaatimuksia täytyy **analysoida**, eli ovatko ne
  - ▶ riittävän kattavat
  - ▶ keskenään ristiriidattomia
  - ▶ testattavissa
  - ▶ toteutuminen on mahdollista ja taloudellisesti järkevää
- ▶ Vaatimukset on myös pakko **dokumentoida** muodossa tai toisessa
  - ▶ Ohjelmistokehittäjiä varten: mitä tehdään
  - ▶ Testaajia varten: toimiiko järjestelmä kuten vaatimukset määrittelevät
- ▶ Joskus vaatimusdokumentti toimii oleellisena osana asiakkaan ja kehittäjien välisessä sopimuksessa
- ▶ Ja **validoida**:
  - ▶ Onko asiakas sitä mieltä että kirjatut vaatimukset kuvaavat sellaisen järjestelmät mitä asiakas tarvitsee

## Toiminnalliset vaatimukset

- ▶ Vaatimukset jakaantuvat toiminnallisiin ja ei-toiminnallisiin vaatimuksiin

## Toiminnalliset vaatimukset

- ▶ Vaatimukset jakaantuvat toiminnallisiin ja ei-toiminnallisiin vaatimuksiin
- ▶ *Toiminnalliset vaatimukset* (functional requirements) kuvaavat mitä toimintoja järjestelmällä on
- ▶ Esim:
  - ▶ Asiakas voi lisätä tuotteen ostoskoriiin
  - ▶ Onnistuneen luottokorttimaksun yhteydessä asiakkaalle vahvistetaan ostotapahtuman onnistuminen sähköpostitse

## Toiminnalliset vaatimukset

- ▶ Vaatimukset jakaantuvat toiminnallisiin ja ei-toiminnallisiin vaatimuksiin
- ▶ *Toiminnalliset vaatimukset* (functional requirements) kuvaavat mitä toimintoja järjestelmällä on
- ▶ Esim:
  - ▶ Asiakas voi lisätä tuotteen ostoskoriiin
  - ▶ Onnistuneen luottokorttimaksun yhteydessä asiakkaalle vahvistetaan ostotapahtuman onnistuminen sähköpostitse
- ▶ Toiminnallisten vaatimusten dokumentointi voi tapahtua esim.
  - ▶ feature-listoina
  - ▶ UML-käyttötapaauksina (joita käsiteltiin kurssilla Ohjelmistotekniikka ennen vuotta 2018)
  - ▶ Ketterissä menetelmissä usein **user storyinä**

## Ei-toiminnalliset vaatimukset

- ▶ Ei-toiminnalliset vaatimukset jakautuvat kahteen luokkaan

## Ei-toiminnalliset vaatimukset

- ▶ Ei-toiminnalliset vaatimukset jakautuvat kahteen luokkaan
- ▶ *Laatuvaatimukset* (quality attributes), ovat koko järjestelmän toiminnallisuutta rajoittavia/ohjaavia tekijöitä, esim.
  - ▶ Käytettävyys
  - ▶ Saavutettavuus
  - ▶ Tietoturva
  - ▶ Suorituskyky
  - ▶ Skaalautuvuus
  - ▶ Testattavuus
  - ▶ Laajennettavuus

## Ei-toiminnalliset vaatimukset

- ▶ Ei-toiminnalliset vaatimukset jakautuvat kahteen luokkaan
- ▶ *Laatuvaatimukset* (quality attributes), ovat koko järjestelmän toiminnallisuutta rajoittavia/ohjaavia tekijöitä, esim.
  - ▶ Käytettävyys
  - ▶ Saavutettavuus
  - ▶ Tietoturva
  - ▶ Suorituskyky
  - ▶ Skaalautuvuus
  - ▶ Testattavuus
  - ▶ Laajennettavuus
- ▶ *Toimintaympäristön rajoitteita* (constraints) ovat esim:
  - ▶ Toteusteknologia (tulee toteuttaa NodeJS:llä ja Reactilla)
  - ▶ Integroituminen muihin järjestelmiin (kirjautuminen HY-tunnuksilla, data SISU:sta)
  - ▶ Mukautuminen lakeihin ja standardeihin (ei riko GDPR:ää)

## Ei-toiminnalliset vaatimukset

- ▶ Ei-toiminnalliset vaatimukset jakautuvat kahteen luokkaan
- ▶ *Laatuvaatimukset* (quality attributes), ovat koko järjestelmän toiminnallisuutta rajoittavia/ohjaavia tekijöitä, esim.
  - ▶ Käytettävyys
  - ▶ Saavutettavuus
  - ▶ Tietoturva
  - ▶ Suorituskyky
  - ▶ Skaalautuvuus
  - ▶ Testattavuus
  - ▶ Laajennettavuus
- ▶ *Toimintaympäristön rajoitteita* (constraints) ovat esim:
  - ▶ Toteusteknologia (tulee toteuttaa NodeJS:llä ja Reactilla)
  - ▶ Integroituminen muihin järjestelmiin (kirjautuminen HY-tunnuksilla, data SISU:sta)
  - ▶ Mukautuminen lakeihin ja standardeihin (ei riko GDPR:ää)
- ▶ Ei-toiminnalliset vaatimukset vaikuttavat yleensä ohjelman arkkitehtuurin suunnittelun

## Vaatimusmäärittely vesiputouksen aikakaudella

## Vaatimusmäärittely vesiputouksen aikakaudella

- ▶ Vesiputousmallissa vaatimusmäärittely erillinen ohjelmistoprosessin vaihe
  - ▶ tehdään kokonaan ennen suunnittelun aloittamista

## Vaatimusmäärittely vesiputouksen aikakaudella

- ▶ Vesiputoosmallissa vaatimusmäärittely erillinen ohjelmistoprosessin vaihe
  - ▶ tehdään kokonaan ennen suunnittelun aloittamista
- ▶ Jos määrittelyssä tehdään virhe, joka huomataan vasta testauksessa, muutoksen tekeminen kallista

## Vaatimusmäärittely vesiputouksen aikakaudella

- ▶ Vesiputoosmallissa vaatimusmäärittely erillinen ohjelmistoprosessin vaihe
  - ▶ tehdään kokonaan ennen suunnittelun aloittamista
- ▶ Jos määrittelyssä tehdään virhe, joka huomataan vasta testauksessa, muutoksen tekeminen kallista
- ▶ Tästä loogisena johtopäätöksenä oli tehdä vaatimusmäärittelystä erittäin järeä ja huolella tehty työvaihe

## Vaatimusmäärittely vesiputouksen aikakaudella: vaikeaa

- ▶ Ideaali jonka mukaan vaatimusmäärittely voidaan irrottaa erilliseksi vaiheeksi on osoittautunut utopiaksi

## Vaatimusmäärittely vesiputouksen aikakaudella: vaikeaa

- ▶ Ideaali jonka mukaan vaatimusmäärittely voidaan irrottaa erilliseksi vaiheeksi on osoittautunut utopiaksi
- ▶ Vaatimusten muuttumien on väistämätöntä
  - ▶ asiakas ei osaa ilmaista tarpeita, toimintaympäristö muuttuu, vaatimusdokumenttia tulkitaan väärin...

## Vaatimusmäärittely vesiputouksen aikakaudella: vaikeaa

- ▶ Ideaali jonka mukaan vaatimusmäärittely voidaan irrottaa erilliseksi vaiheeksi on osoittautunut utopiaksi
- ▶ Vaatimusten muuttumien on väistämätöntä
  - ▶ asiakas ei osaa ilmaista tarpeita, toimintaympäristö muuttuu, vaatimusdokumenttia tulkitaan väärin...
- ▶ Vaatimusmäärittelyä ei ole mahdollista/järkevää irrottaa suunnittelusta ja toteutuksesta
  - ▶ Suunnittelu auttaa ymmärtämään ongelma-alueutta syvälliisemmin ja generoi muutoksia vaatimuksiin
  - ▶ Ohjelmia tehdään maksimoiden valmiiden ja muualta, esim. open sourcena saatavien komponenttien käyttö

## Vaatimusmäärittely vesiputouksen aikakaudella: vaikeaa

- ▶ Ideaali jonka mukaan vaatimusmäärittely voidaan irrottaa erilliseksi vaiheeksi on osoittautunut utopiaksi
- ▶ Vaatimusten muuttumien on väistämätöntä
  - ▶ asiakas ei osaa ilmaista tarpeita, toimintaympäristö muuttuu, vaatimusdokumenttia tulkitaan väärin...
- ▶ Vaatimusmäärittelyä ei ole mahdollista/järkevää irrottaa suunnittelusta ja toteutuksesta
  - ▶ Suunnittelu auttaa ymmärtämään ongelma-alueutta syvällisemmin ja generoi muutoksia vaatimuksiin
  - ▶ Ohjelmia tehdään maksimoiden valmiiden ja muualta, esim. open sourcena saatavien komponenttien käyttö
- ▶ **Jos toteutus otetaan huomioon, on helpompi arvioda vaatimusten toteuttamisen hintaa**

# Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

## Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

- ▶ Iteratiivisen ja ketterän ohjelmistotuotannon tapa on integroida kaikki ohjelmistotuotannon vaiheet yhteen

## Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

- ▶ Iteratiivisen ja ketterän ohjelmistotuotannon tapa on integroida kaikki ohjelmistotuotannon vaiheet yhteen
- ▶ Projektin alussa määritellään vaatimuksia tarkemmassa tasolla ainakin yhden iteration tarpeiden verran

## Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

- ▶ Iteratiivisen ja ketterän ohjelmistotuotannon tapa on integroida kaikki ohjelmistotuotannon vaiheet yhteen
- ▶ Projektin alussa määritellään vaatimuksia tarkemmassa tasolla ainakin yhden iteraation tarpeiden verran
- ▶ Ohjelmistokehittäjät arvioivat vaatimusten toteuttamisen hintaa
- ▶ Asiakas priorisoi: iteraatioon valitaan toteutettavaksi ne vaatimukset, jotka maksimoivat liiketoiminnallisen arvon

## Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

- ▶ Jokaisen iteraation aikana tehdään määrittelyä, suunnittelua, ohjelointia ja testausta

## Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

- ▶ Jokaisen iteraation aikana tehdään määrittelyä, suunnittelua, ohjelointia ja testausta
- ▶ Jokainen iteraatio tuottaa valmiin osan järjestelmää

## Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

- ▶ Jokaisen iteraation aikana tehdään määrittelyä, suunnittelua, ohjelointia ja testausta
- ▶ Jokainen iteraatio tuottaa valmiin osan järjestelmää
- ▶ Edellisen iteraation tuotos toimii syötteenä seuraavan iteraation vaatimusten määrittelyyn

## Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

- ▶ Jokaisen iteraation aikana tehdään määrittelyä, suunnittelua, ohjelointia ja testausta
- ▶ Jokainen iteraatio tuottaa valmiin osan järjestelmää
- ▶ Edellisen iteraation tuotos toimii syötteenä seuraavan iteraation vaatimusten määrittelyyn
- ▶ **Ohjelmisto on mahdollista saada tuotantoon jo ennen kaikkien vaatimusten valmistumista**

## Vaatimusmäärittely iteratiivisessa ja ketterässä ohjelmistokehityksessä

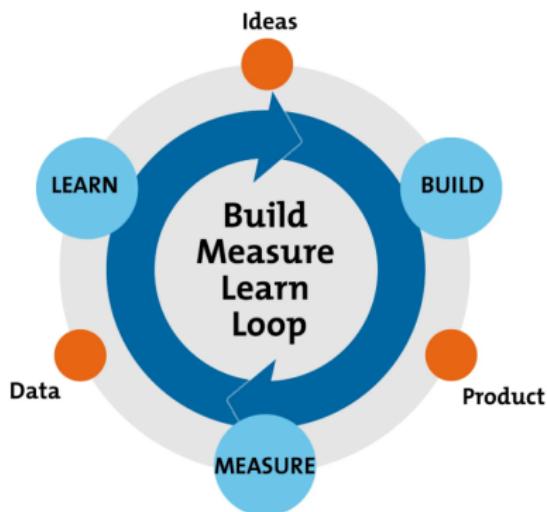
- ▶ Jokaisen iteraation aikana tehdään määrittelyä, suunnittelua, ohjelointia ja testausta
- ▶ Jokainen iteraatio tuottaa valmiin osan järjestelmää
- ▶ Edellisen iteraation tuotos toimii syötteenä seuraavan iteraation vaatimusten määrittelyyn
- ▶ **Ohjelmisto on mahdollista saada tuotantoon jo ennen kaikkien vaatimusten valmistumista**
- ▶ Kattavana teemana tuottaa asiakkaalle maksimaalisesti arvoa

## Uuden ajan vaatimusmäärittely: Lean startup

- ▶ Eric Ries (2011): *Lean startup*
  - ▶ kuvaa systemaattisen tavan kartoittaa vaatimuksia erityisen epävarmoissa konteksteissa

# Uuden ajan vaatimusmäärittely: Lean startup

- ▶ Eric Ries (2011): *Lean startup*
  - ▶ kuvaaa systemaattisen tavan kartoittaa vaatimuksia erityisen epävarmoissa konteksteissa
- ▶ Malli perustuu kolmiosaisen build-measure-learn-syklin toistamiseen



## Lean startup

- ▶ Esim. internetpalveluja tai mobiilisovelluksia rakennettaessa käyttäjien tarpeista ei minkäänlaista varmuutta

## Lean startup

- ▶ Esim. internetpalveluja tai mobiilisovelluksia rakennettaessa käyttäjien tarpeista ei minkäänlaista varmuutta
- ▶ Otetaan lähtökohdaksi jokin idea siitä, mitä käyttäjät haluavat

## Lean startup

- ▶ Esim. internetpalveluja tai mobiilisovelluksia rakennettaessa käyttäjien tarpeista ei minkäänlaista varmuutta
- ▶ Otetaan lähtökohdaksi jokin idea siitä, mitä käyttäjät haluavat
- ▶ Rakennetaan nopeasti **minimal viable product (MVP)** joka toteuttaa ominaisuuden

# Lean startup

- ▶ Esim. internetpalveluja tai mobiilisovelluksia rakennettaessa käyttäjien tarpeista ei minkäänlaista varmuutta
- ▶ Otetaan lähtökohdaksi jokin idea siitä, mitä käyttäjät haluavat
- ▶ Rakennetaan nopeasti **minimal viable product (MVP)** joka toteuttaa ominaisuuden
- ▶ MVP laitetaan tuotantoon ja **mitataan miten asiakkaat käyttäytyvät** uuden toiminnallisuuden suhteeseen

# Lean startup

- ▶ Jos MVP jonkin toiminnallisuuden uusi versio, käytetään **A/B-testausta**
  - ▶ uusi ominaisuus julkaistaan osalle käyttäjistä, loput jatkavat vanhan ominaisuuden käyttöä

# Lean startup

- ▶ Jos MVP jonkin toiminnallisuuden uusi versio, käytetään **A/B-testausta**
  - ▶ uusi ominaisuus julkaistaan osalle käyttäjistä, loput jatkavat vanhan ominaisuuden käyttöä
- ▶ Mitattua käyttäytymistä:
  - ▶ olivatko toteutetut toiminnallisuuden käyttäjien mieleen

# Lean startup

- ▶ Jos MVP jonkin toiminnallisuuden uusi versio, käytetään **A/B-testausta**
  - ▶ uusi ominaisuus julkaistaan osalle käyttäjistä, loput jatkavat vanhan ominaisuuden käyttöä
- ▶ Mitattua käyttäytymistä:
  - ▶ olivatko toteutetut toiminnallisuuden käyttäjien mieleen
- ▶ Jos idea on hyvä, toteutetaan sen toiminnallisuus robustilla tavalla
  - ▶ jos ei, palataan järjestelmän edelliseen versioon

# Lean startup

- ▶ Jos MVP jonkin toiminnallisuuden uusi versio, käytetään **A/B-testausta**
  - ▶ uusi ominaisuus julkaistaan osalle käyttäjistä, loput jatkavat vanhan ominaisuuden käyttöä
- ▶ Mitattua käyttäytymistä:
  - ▶ olivatko toteutetut toiminnallisuuden käyttäjien mieleen
- ▶ Jos idea on hyvä, toteutetaan sen toiminnallisuus robustilla tavalla
  - ▶ jos ei, palataan järjestelmän edelliseen versioon
- ▶ **Menetelmällä on siis tarkoitus oppia systemaattisesti ja mahdollisimman nopeasti mitä asiakkaat haluavat**

TAUKO 10 minuuttia

# Vaatimusmäärittely ja projektisuunnittelu ketterässä prosessimallissa

# User story

## User story

- ▶ Ketterän vaatimusmäärittelyn tärkein työväline on user story
- ▶ Mike Cohn:
  - ▶ *A user story describes **functionality that will be valuable** to either user or purchaser of software.*

# User story

- ▶ Ketterän vaatimusmäärittelyn tärkein työväline on 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** of the story, used for planning and reminder
  2. **Conversations** about the story to serve to flesh the details of the story
  3. **Tests** that convey and document details and that will be used to determine that the story is complete

## User story

- ▶ User storyt kuvaavat loppukäyttäjän kannalta arvokkaita toiminnallisuuksia

## User story

- ▶ User storyt kuvaavat loppukäyttäjän kannalta arvokkaita toiminnallisuuksia
- ▶ User story on karkean tason tekstuaalinen kuvaus
- ▶ ja lupaus/muistutus siitä, että toiminnallisuuden vaatimukset on selvitettävä asiakkaan kanssa

## User story

- ▶ User storyt kuvaavat loppukäyttäjän kannalta arvokkaita toiminnallisuksia
- ▶ User story on karkean tason tekstuaalinen kuvaus
- ▶ ja lupaus/muistutus siitä, että toiminnallisuuden vaatimukset on selvitettävä asiakkaan kanssa
- ▶ Seuraavat voisivat olla verkkokaupan user storyjen tekstuaalisia kuvaauksia:
  - ▶ Asiakas voi lisätä tuotteen ostoskoriin
  - ▶ Asiakas voi poistaa ostoskorissa olevan tuotteen
  - ▶ Asiakas voi maksaa luottokortilla ostoskorissa olevat tuotteet

## User story

- ▶ Kun user story päätetään toteuttaa, on sen tarkat vaatimukset pakko selvittää

## User story

- ▶ Kun user story päätetään toteuttaa, on sen tarkat vaatimukset pakko selvittää
- ▶ Story on lupaus kommunikoinnista asiakkaan kanssa *conversations about the story to serve to flesh the details of the story*

## User story

- ▶ Kun user story päätetään toteuttaa, on sen tarkat vaatimukset pakko selvittää
- ▶ Story on lupaus kommunikoinnista asiakkaan kanssa *conversations about the story to serve to flesh the details of the story*
- ▶ Määritelmän kolmas alikohta sanoo että storyyn kuuluu *Tests that convey and document details and that will be used to determine that the story is complete*

## User story

- ▶ Kun user story päättetään toteuttaa, on sen tarkat vaatimukset pakko selvittää
- ▶ Story on lupaus kommunikoinnista asiakkaan kanssa *conversations about the story to serve to flesh the details of the story*
- ▶ Määritelmän kolmas alikohta sanoo että storyyn kuuluu *Tests that convey and document details and that will be used to determine that the story is complete*
- ▶ Storyyn kuuluvia testejä kutsutaan **hyväksymätesteiksi** tai **hyväksymäkriteereiksi**

## User story

- ▶ Kun user story päättäään toteuttaa, on sen tarkat vaatimukset pakko selvittää
- ▶ Story on lupaus kommunikoinnista asiakkaan kanssa *conversations about the story to serve to flesh the details of the story*
- ▶ Määritelmän kolmas alikohta sanoo että storyyn kuuluu *Tests that convey and document details and that will be used to determine that the story is complete*
- ▶ Storyyn kuuluvia testejä kutsutaan **hyväksymätesteiksi** tai **hyväksymäkriteereiksi**
- ▶ Yleensä joukko konkreettisia testiskenaarioita joiden toimittava, jotta storyn voidaan todeta olevan valmis

## User story

- ▶ Kun user story päättäään toteuttaa, on sen tarkat vaatimukset pakko selvittää
- ▶ Story on lupaus kommunikoinnista asiakkaan kanssa *conversations about the story to serve to flesh the details of the story*
- ▶ Määritelmän kolmas alikohta sanoo että storyyn kuuluu *Tests that convey and document details and that will be used to determine that the story is complete*
- ▶ Storyyn kuuluvia testejä kutsutaan **hyväksymätesteiksi** tai **hyväksymäkriteereiksi**
- ▶ Yleensä joukko konkreettisia testiskenaarioita joiden toimittava, jotta storyn voidaan todeta olevan valmis
- ▶ Luonne vaihtelee projekteittain
  - ▶ Tekstinä dokumentoituja skenaarioita
  - ▶ Parhaassa tapauksessa automaattisesti suoritettavia testejä

## Esimerkki user storystä

# Esimerkki user storystä

Front of Card	Back of Card
<p>1B</p> <p>As a student I want to purchase a parking pass so that I can drive to school</p> <p>Priority: <del>1000</del> Should Estimate: 4</p>	<p>Confirmations:</p> <p><del>The student must pay the correct amount</del></p> <p>One pass for one month is issued at a time</p> <p>The student will not receive a pass if the payment isn't sufficient</p> <p>The person buying the pass must be a currently enrolled student.</p> <p>The student may only buy one pass per month.</p>

## Hyvän storyn kriteerejä

## Hyvän storyn kriteerejä

- ▶ User storyn tulee kuvata sovelluksen käyttäjälle arvokkaita toimintoja
  - ▶ Käytettävä asiakkaan kielty, ei teknistä jargonia

## Hyvän storyn kriteerejä

- ▶ User storyn tulee kuvata sovelluksen käyttäjälle arvokkaita toimintoja
  - ▶ Käytettävä asiakkaan kielty, ei teknistä jargonia
- ▶ User story tulisi kuvata “end to end”-toiminnallisuutta
  - ▶ kattaen käyttöliittymän, bisneslogiikan, ja tietokannan
  - ▶ esimerkki huonosta storystä *lisää jokaisesta asiakkaasta rivi tietokantatauluun customers*

## Hyvän storyn kriteerejä

- ▶ User storyn tulee kuvata sovelluksen käyttäjälle arvokkaita toimintoja
  - ▶ Käytettävä asiakkaan kielty, ei teknistä jargonia
- ▶ User story tulisi kuvata ”end to end”-toiminnallisuutta
  - ▶ kattaen käyttöliittymän, bisneslogiikan, ja tietokannan
  - ▶ esimerkki huonosta storystä *lisää jokaisesta asiakkaasta rivi tietokantatauluun customers*
- ▶ Edellinen sivu erään muodon mukaisessa muodossa
  - ▶ As a *type of user*, I want *functionality* so that *business value*
  - ▶ As a *student* I want *to purchase a parking pass* so that *I can drive to school*

## Hyvän storyn kriteerit

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

## Hyvän storyn kriteerit

- ▶ Bill Wake *INVEST in good User Stories*, kuusi toivottavaa ominaisuutta
  - ▶ Independent
  - ▶ Negotiable
  - ▶ Valuable to user or customer
  - ▶ Estimable
  - ▶ Small
  - ▶ Testable
- ▶ **Independent:** storyjen pitäisi olla toteutusjärjestykseltään mahdollisimman riippumattomia
  - ▶ antaa asiakkaalle enemmän vapaauksia

## Hyvän storyn kriteerit

- ▶ Bill Wake *INVEST in good User Stories*, kuusi toivottavaa ominaisuutta
  - ▶ Independent
  - ▶ Negotiable
  - ▶ Valuable to user or customer
  - ▶ Estimable
  - ▶ Small
  - ▶ Testable
- ▶ **Independent:** storyjen pitäisi olla toteutusjärjestykseltään mahdollisimman riippumattomia
  - ▶ antaa asiakkaalle enemmän vapaauksia
- ▶ **Negotiable:** storyn luonne “muistilappuna” ja keskusteluna
- ▶ **Valuable**

## Hyvän storyn kriteerit

- ▶ **Estimatable:** storyn toteuttamisen vaatima työmäärä pitää olla arvioitavissa kohtuullisella tasolla

## Hyvän storyn kriteerit

- ▶ **Estimatable:** storyn toteuttamisen vaatima työmäärä pitää olla arvioitavissa kohtuullisella tasolla
- ▶ **Small** storyt on oltava riittävän pieniä, yhden sprintin aikana toteutettavissa olevia

## Hyvän storyn kriteerit

- ▶ **Estimatable:** storyn toteuttamisen vaatima työmäärä pitää olla arvioitavissa kohtuullisella tasolla
- ▶ **Small** storyt on oltava riittävän pieniä, yhden sprintin aikana toteutettavissa olevia
- ▶ **Testability:** storylle pitää pystyä laatimaan kriteerit, joiden avulla voi todeta onko story toteutettu hyväksyttävästi

## Hyvän storyn kriteerit

- ▶ **Estimatable:** storyn toteuttamisen vaatima työmäärä pitää olla arvioitavissa kohtuullisella tasolla
- ▶ **Small** storyt on oltava riittävän pieniä, yhden sprintin aikana toteutettavissa olevia
- ▶ **Testability:** storylle pitää pystyä laatimaan kriteerit, joiden avulla voi todeta onko story toteutettu hyväksyttävästi
- ▶ Ei-toiminnalliset vaatimukset (esim. suorituskyky, käytettävyys) aiheuttavat usein haasteita testattavuudelle
  - ▶ Esim. story *verkkokaupan tulee toimia tarpeeksi nopeasti kovassakin kuormituksessa*
  - ▶ voidaan muotoilla testattavaksi seuraavasti: *käyttäjän vasteaika saa olla korkeintaan 0.5 sekuntia 99% tapauksissa jos yhtäaikaisia käyttäjiä sivulla on maksimissaan 1000*

Ketterää vaatimusten hallintaa...

## Alustava backlog

- ▶ Projektin alussa etsitään ja määritteellään user storyja ja muodostaa näistä alustava product backlog

## Alustava backlog

- ▶ Projektin alussa etsitään ja määritteellään user storyja ja muodostaa näistä alustava product backlog
- ▶ Käytettävissä ovat kaikki yleiset vaatimusten kartoitustekniikat:
  - ▶ haastattelut, brainstormaus, paperiprototypit, käyttöliittymälouonnokset...

## Alustava backlog

- ▶ Projektin alussa etsitään ja määritteellään user storyja ja muodostaa näistä alustava product backlog
- ▶ Käytettävissä ovat kaikki yleiset vaatimusten kartoitustekniikat:
  - ▶ haastattelut, brainstormaus, paperiprototyypit, käyttöliittymälouonnokset...
- ▶ Alustavan storyjen keräämisvaiheen ei ole tarkoituksenmukaista kestää kovin kauaa, maksimissaan muutaman päivän

## Alustava backlog

- ▶ Projektin alussa etsitään ja määritteellään user storyja ja muodostaa näistä alustava product backlog
- ▶ Käytettävissä ovat kaikki yleiset vaatimusten kartoitustekniikat:
  - ▶ haastattelut, brainstormaus, paperiprototyypit, käyttöliittymälouonnokset...
- ▶ Alustavan storyjen keräämisvaiheen ei ole tarkoituksenmukaista kestää kovin kauaa, maksimissaan muutaman päivän
- ▶ User story on muistilappu ja lupaus tarkennuksesta:
  - ▶ Turhiin detaljeihin ei puututa
  - ▶ Ei edes tavoitella täydellistä ja kattavaa lista vaatimuksista, asioita tarkennetaan myöhemmin

## Alustava backlog

- ▶ Projektin alussa etsitään ja määritteellään user storyja ja muodostaa näistä alustava product backlog
- ▶ Käytettävissä ovat kaikki yleiset vaatimusten kartoitustekniikat:
  - ▶ haastattelut, brainstormaus, paperiprototyypit, käyttöliittymälouonnokset...
- ▶ Alustavan storyjen keräämisvaiheen ei ole tarkoituksensa mukaista kestää kovin kauaa, maksimissaan muutaman päivän
- ▶ User story on muistilappu ja lupaus tarkennuksesta:
  - ▶ Turhiin detaljeihin ei puututa
  - ▶ Ei edes tavoitella täydellistä ja kattavaa lista vaatimuksista, asioita tarkennetaan myöhemmin
- ▶ Kun alustavat storyt identifioitu, ne priorisoidaan ja työmäärä arvioidaan karkealla tasolla

# Backlogin priorisointi

## Backlogin priorisointi

- ▶ Prioriteetti määräää järjestyksen, missä ohjelmistokehittäjät toteuttavat ohjelmiston ominaisuuksia
- ▶ **Priorisoinnin hoitaa product owner**

## Backlogin priorisointi

- ▶ Prioriteetti määräää järjestyksen, missä ohjelmistokehittäjät toteuttavat ohjelmiston ominaisuuksia
- ▶ **Priorisoinnin hoitaa product owner**
- ▶ Motivaationa on pyrkiä maksimoimaan asiakkaan kehitettävästä ohjelmistosta saama hyöty/arvo

## Backlogin priorisointi

- ▶ Prioriteetti määräää järjestyksen, missä ohjelmistokehittäjät toteuttavat ohjelmiston ominaisuuksia
- ▶ **Priorisoinnin hoitaa product owner**
- ▶ Motivaationa on pyrkiä maksimoimaan asiakkaan kehitettävästä ohjelmistosta saama hyöty/arvo
- ▶ Tärkeimmät asiat halutaan toteuttaa nopeasti
  - ▶ saadaan tuotteen alustava versio nopeasti julkaistua

## Backlogin priorisointi

- ▶ Prioriteetti määräää järjestyksen, missä ohjelmistokehittäjät toteuttavat ohjelmiston ominaisuuksia
- ▶ **Priorisoinnin hoitaa product owner**
- ▶ Motivaationa on pyrkiä maksimoimaan asiakkaan kehitettävästä ohjelmistosta saama hyöty/arvo
- ▶ Tärkeimmät asiat halutaan toteuttaa nopeasti
  - ▶ saadaan tuotteen alustava versio nopeasti julkaistua
- ▶ Arvon lisäksi priorisoinnissa kannattaa huomioida
  - ▶ Storyn toteuttamiseen kuluva työmääriä
  - ▶ Storyn kuvaamaan ominaisuuteen sisältyvä tekninen riski

# Estimointi

# Estimointi

- ▶ User storyjen estimointiin eli niiden viemän työmäärän arvioimiseen on kaksi motivaatiota
  - ▶ Auttaa asiakasta priorisoinnissa
  - ▶ Mahdollistaa koko projektin viemän ajan arvioinnin

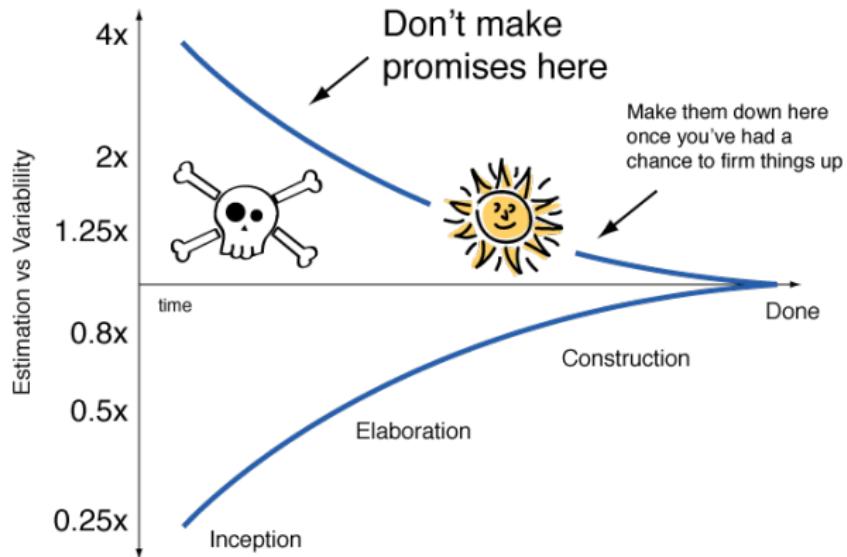
# Estimointi

- ▶ User storyjen estimointiin eli niiden viemän työmäärään arvioimiseen on kaksi motivaatiota
  - ▶ Auttaa asiakasta priorisoinnissa
  - ▶ Mahdollistaa koko projektin viemän ajan arvioinnin
- ▶ Työmäärään arvioimiseen on kehitetty vuosien varrella useita erilaisia menetelmiä
  - ▶ Kaikille yhteistä on se, että ne eivät toimi kunnolla
  - ▶ **tarkkoja työmääräarvioita on mahdoton antaa**

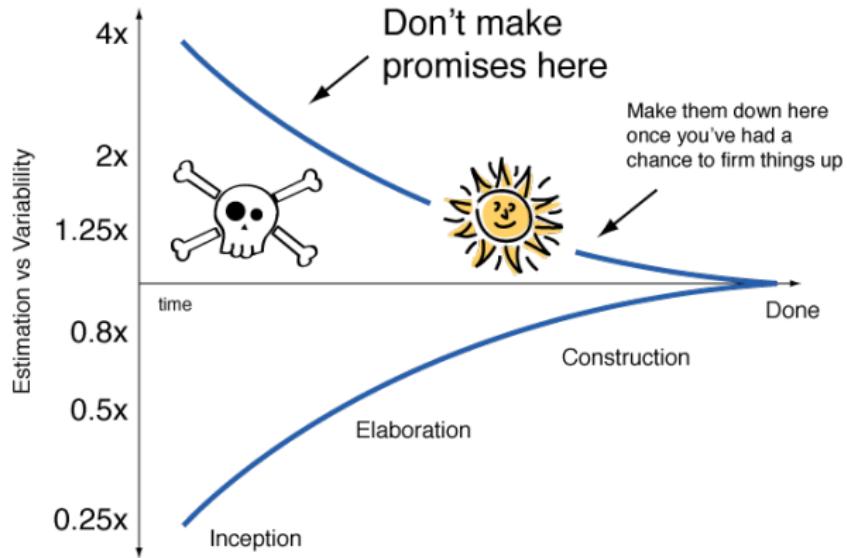
# Estimointi

- ▶ User storyjen estimointiin eli niiden viemän työmäärään arvioimiseen on kaksi motivaatiota
  - ▶ Auttaa asiakasta priorisoinnissa
  - ▶ Mahdollistaa koko projektin viemän ajan arvioinnin
- ▶ Työmäärään arvioimiseen on kehitetty vuosien varrella useita erilaisia menetelmiä
  - ▶ Kaikille yhteistä on se, että ne eivät toimi kunnolla
  - ▶ **tarkkoja työmääräarvioita on mahdoton antaa**
- ▶ Mitä kauempana tuotteen/ominaisuuden valmistuminen on, sitä epätarkempia työmääräarviot ovat

# cone of uncertainty



# cone of uncertainty



- ▶ Ketterän kehityksen *lähtökohta* on että estimointi on epävarmaa ja tarkentuu vasta projektin kuluessa
  - ▶ ei tehdä sitovia estimointiin perustuvia lupauksia

## Suhteelliseen kokoon perustuva estimointi

- ▶ Ominaisuuksien toteuttamiseen menevän tarkan ajan arvioiminen on vaikeaa

## Suhteelliseen kokoon perustuva estimointi

- ▶ Ominaisuuksien toteuttamiseen menevän tarkan ajan arvioiminen on vaikeaa
- ▶ Ohjelmistokehittäjät pystyvät jossain määrin arvioida eri ominaisuuksien vaatimaa työmääriä suhteessa toisiinsa

## Suhteelliseen kokoon perustuva estimointi

- ▶ Ominaisuuksien toteuttamiseen menevän tarkan ajan arvioiminen on vaikeaa
- ▶ Ohjelmistokehittäjät pystyvät jossain määrin arvioida eri ominaisuuksien vaatimaa työmääriä suhteessa toisiinsa
- ▶ Esim.
  - ▶ *Tuotteen lisääminen ostoskoriin toteuttaminen* vie yhtä kauan kuin *Tuotteen poistaminen ostoskorista*
  - ▶ *Ostoskorissa olevien tuotteiden maksaminen luottokortilla* taas vie noin kolme kertaa kauemmin kun edelliset

## Suhteelliseen kokoon perustuva estimointi

- ▶ Ominaisuuksien toteuttamiseen menevän tarkan arvioiminen on vaikeaa
- ▶ Ohjelmistokehittäjät pystyvät jossain määrin arvioida eri ominaisuuksien vaatimaa työmääriä suhteessa toisiinsa
- ▶ Esim.
  - ▶ *Tuotteen lisääminen ostoskoriin toteuttaminen* vie yhtä kauan kuin *Tuotteen poistaminen ostoskorista*
  - ▶ *Ostoskorissa olevien tuotteiden maksaminen luottokortilla* taas vie noin kolme kertaa kauemmin kun edelliset
- ▶ Ketterissä menetelmissä käytetäänkin yleisesti *suhteelliseen kokoon perustuvaa estimointia*
  - ▶ Yksikkönä arvioinnissa on yleensä **story point**
  - ▶ Ei yleensä vastaa mitään todellista tuntimääriä

## Kehittäjät iimi estimoii

- ▶ Estimointi tapahtuu **aina** ohjelmistokehitystiimin toimesta
  - ▶ Product owner tarkentaa estimoitaviin storyihin liittyviä vaatimuksia

## Kehittäjät iimi estimoii

- ▶ Estimointi tapahtuu **aina** ohjelmistokehitystiimin toimesta
  - ▶ Product owner tarkentaa estimoitaviin storyihin liittyviä vaatimuksia
- ▶ Estimointia auttaa user storyn pilkkomisen teknisiin työvaiheisiin

## Kehittäjät iimi estimoii

- ▶ Estimointi tapahtuu **aina** ohjelmistokehitystiimin toimesta
  - ▶ Product owner tarkentaa estimoitaviin storyihin liittyviä vaatimuksia
- ▶ Estimointia auttaa user storyn pilkkomisen teknisiin työvaiheisiin
- ▶ *Tuotteen lisääminen ostoskoriiin*, voisi sisältää toteutuksen kannalta seuraavat tekniset tehtävät:
  - ▶ tarvitaan sessio, joka muistaa asiakkaan
  - ▶ oliot/tietorakenteet ostoskorin ja ostoksen esittämiseen
  - ▶ html-näkymää päivitettävä tarvittavilla painikkeilla
  - ▶ Kontrolleri painikkeiden käsitteilyyn
  - ▶ yksikkötestit kontrollerille ja tietorakenteille
  - ▶ hyväksymätestien automatisointi
- ▶ Jos kyseessä on samantapainen toiminnallisuus kuin joku aiemmin toteutettu, ei pilkkomista välttämättä tarvita

## Estimointi definition of doneen tarkkuudella

- ▶ Estimoinnissa tulee arvioida storyn viemä aika *definition of doneen* tarkkuudella
- ▶ Tämä sisältää yleensä kaiken storyn toteuttamiseen liittyvän
  - ▶ määrittely, suunnittelu, toteutus, automatisoitujen tekstien tekeminen, testaus, integrointi ja dokumentointi

## Estimointi definition of doneen tarkkuudella

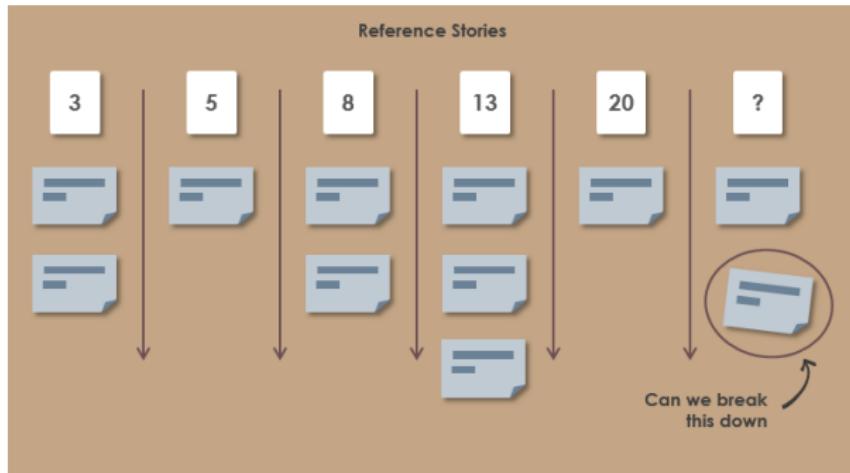
- ▶ Estimoinnissa tulee arvioida storyn viemä aika *definition of doneen* tarkkuudella
- ▶ Tämä sisältää yleensä kaiken storyn toteuttamiseen liittyvän
  - ▶ määrittely, suunnittelu, toteutus, automatisoitujen tekstien tekeminen, testaus, integrointi ja dokumentointi
- ▶ Estimointi on joka tapauksessa suhteellisen epätarkkaa, joten estimoinnin on tarkoitus tapahtua nopeasti
  - ▶ Storyn estimointiin kannattaa käyttää aikaa max 15 minuuttia

## Estimointi definition of doneen tarkkuudella

- ▶ Estimoinnissa tulee arvioida storyn viemä aika *definition of doneen* tarkkuudella
- ▶ Tämä sisältää yleensä kaiken storyn toteuttamiseen liittyvän
  - ▶ määrittely, suunnittelu, toteutus, automatisoitujen tekstien tekeminen, testaus, integrointi ja dokumentointi
- ▶ Estimointi on joka tapauksessa suhteellisen epätarkkaa, joten estimoinnin on tarkoitus tapahtua nopeasti
  - ▶ Storyn estimointiin kannattaa käyttää aikaa max 15 minuuttia
- ▶ Jos se ei riitä, storya ei tunneta niin hyvin että se kannattaisi estimoida
  - ▶ story kannattaanee pilkkoja

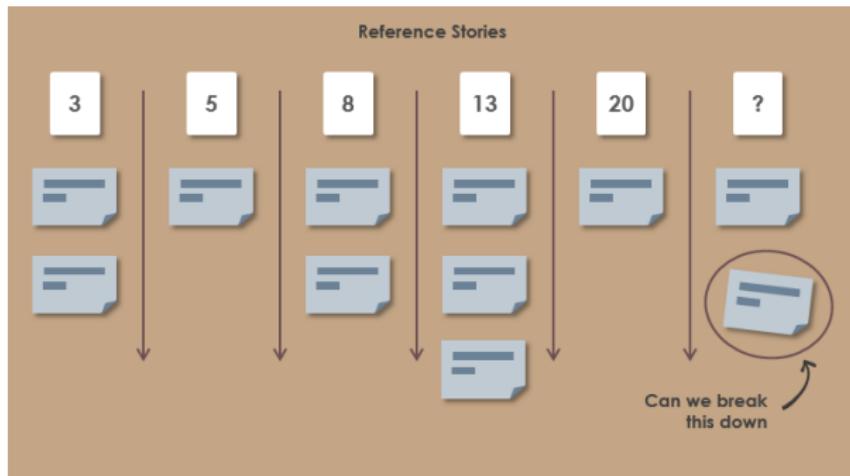
# Estimoinnin menetelmiä

- ▶ Kiinnitetään muutama erikokoinen story *referenssiksi* ja verrata muiden storyjen työmäärää näihin



# Estimoinnin menetelmiä

- ▶ Kiinnitetään muutama erikokoinen story *referenssiksi* ja verrata muiden storyjen työmäärää näihin



- ▶ Käytetään yläpäästä harvenevaa skaalaa esim. 1, 2, 3, 5, 10, 20, 40, 100
- ▶ Koska isojen storyjen estimointiin liittyy suuri epävarmuus, ei teeskennellä että skaala olisi yläpäästä tarkka

# Planning poker: osallistetaan koko tiimi

1. Customer reads story.



2. Team estimates.

This includes testing.



3. Team discusses.



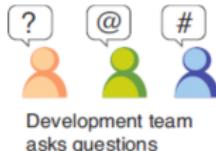
4. Team estimates again.

Repeat until consensus reached.



# Planning poker: osallistetaan koko tiimi

1. Customer reads story.



2. Team estimates.

This includes testing.



3. Team discusses.



4. Team estimates again.

Repeat until consensus reached.



- ▶ Kaikille yhtenäinen näkemys sisällöstä ja tieto levämään kaikille (transparency)