

# Ohjelmistotuotanto

Syksy 2025

Luento 4

4.11.2025

## 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>
- ▶ *Olen yllättynyt siitä, kuinka epätieteelliseltä kurssi tuntuu vaikka kyseessä on yliopiston kurssi*
  - ▶ *Erityisesti kuvaus scrumista “extremely hard to master” tuntuu helpolta tavalta torjua kritiikki*

# Scrum

- ▶ **Kehittäjiensä mukaa Scrum on**
  - ▶ kevyt (lightweight)
  - ▶ helppo ymmärtää
  - ▶ mutta **äärimmäisen vaikea hallita** (extremely difficult to master)

# Scrum

- ▶ Kehittäjiensä mukaa Scrum on
  - ▶ kevyt (lightweight)
  - ▶ helppo ymmärtää
  - ▶ mutta **äärimmäisen vaikea hallita** (extremely difficult to master)
- ▶ Tutkimuksen tuomaan näkökulmaankin palataan

# 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
  - ▶ ti 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>

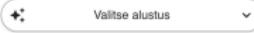
 CURRECHAT

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":

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
```

- ▶ 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

## Nopea kertaus eiliseltä

- ▶ User story
  - ▶ description
  - ▶ conversations
  - ▶ acceptance criteria

## Nopea kertaus eiliseltä

- ▶ User story
  - ▶ description
  - ▶ conversations
  - ▶ acceptance criteria
- ▶ Hyvä user story: INVEST

# Nopea kertaus eiliseltä

- ▶ User story
  - ▶ description
  - ▶ conversations
  - ▶ acceptance criteria
- ▶ Hyvä user story: INVEST
- ▶ Estimointi
  - ▶ Miksi? Kuka? Miten?

# Nopea kertaus eiliseltä

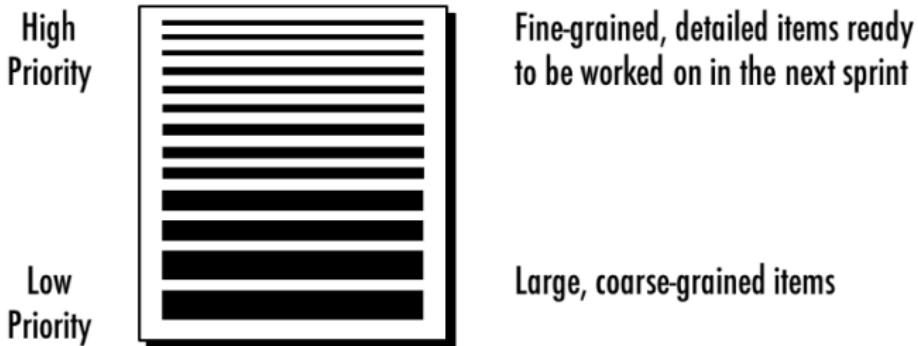
- ▶ User story
  - ▶ description
  - ▶ conversations
  - ▶ acceptance criteria
- ▶ Hyvä user story: INVEST
- ▶ Estimointi
  - ▶ Miksi? Kuka? Miten?
- ▶ Product Backlog
  - ▶ Kuka vastuussa?
  - ▶ Miten saadaan projektin alussa muodostettua?

## Hyvä product backlog on DEEP

- ▶ Mike Cohn lanseerasi lyhenteen DEEP kuvaamaan hyvän backlogin ominaisuuksia
  - ▶ Detailed appropriately
  - ▶ Estimated
  - ▶ Emergent
  - ▶ Prioritized

- ▶ Estimated, Prioritized

- ▶ Estimated, Prioritized
- ▶ *Detailed appropriately eli sopivan yksityiskohtainen*
  - ▶ ylempänä tarkkoja
  - ▶ alempaan suurpiirteisempiä



- ▶ *Emergent* kuvailee backlogin muuttuvaa luonnetta:
  - ▶ uusia storyja tulee
  - ▶ vanhoja poistetaan, uudelleenpriorisoidaan ja uudelleenestimoidaan, muokataan ja pilkotaan

- ▶ *Emergent* kuvaaa backlogin muuttuvaa luonnetta:
  - ▶ uusia storyja tulee
  - ▶ vanhoja poistetaan, uudelleenpriorisoidaan ja uudelleenestimoidaan, muokataan ja pilkotaan
- ▶ Muuttuvan luonteen takia backlogia tulee hoitaa projektin edetessä (engl. backlog refinement/grooming)
  - ▶ Pääasiallinen vastuu on product ownerilla
  - ▶ Backlogin hoitamiseen osallistuu koko kehitystiiimi
  - ▶ Scrum suosittlee että noin 10% sprintin työajasta käytetään backlog refinementiin

## “Ready” story ja epiikki

- ▶ Hyvä story on siis INVEST (independent, negotiable, valuable, estimable, small, testable)
- ▶ DEEP taas taas sanoo, että backlogin pitää olla *sopivan* yksityiskohtainen

## “Ready” story ja epiikki

- ▶ Hyvä story on siis INVEST (independent, negotiable, valuable, estimable, small, testable)
- ▶ DEEP taas taas sanoo, että backlogin pitää olla *sopivan* yksityiskohtainen
- ▶ INVEST päätekin vain backlogin korkeamman prioriteetin storyihin
  - ▶ Joskus sanotaan että story on **ready**, kun se on valmiina toteutettavaksi (hyvin tunnettu ja INVEST)

## “Ready” story ja epiikki

- ▶ Hyvä story on siis INVEST (independent, negotiable, valuable, estimable, small, testable)
- ▶ DEEP taas taas sanoo, että backlogin pitää olla *sopivan* yksityiskohtainen
- ▶ INVEST päätekin vain backlogin korkeamman prioriteetin storyihin
  - ▶ Joskus sanotaan että story on **ready**, kun se on valmiina toteutettavaksi (hyvin tunnettu ja INVEST)
- ▶ Alemman prioriteetin storyt voivat olla **epiikkejä** (epic)
  - ▶ scope ei tiedossa, ei mielekästä estimoida

## Velositeetti

- ▶ Estimoinnin yksi tarkoitus on mahdollistaa koko projektin viemän aikamäärän summittainen arvointi

## Velositeetti

- ▶ Estimoinnin yksi tarkoitus on mahdollistaa koko projektin viemän aikamäärän summittainen arvointi
- ▶ Estimoinnin yksikkönä on abstrakti käsite *story point*, miten sen avulla voidaan arvioida projektin kestoa?

## Velositeetti

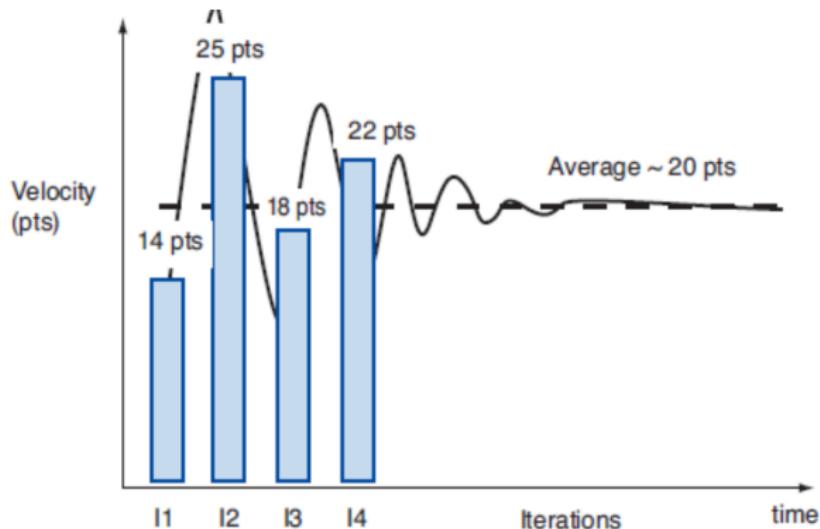
- ▶ Estimoinnin yksi tarkoitus on mahdollistaa koko projektin viemän aikamäärän summittainen arvointi
- ▶ Estimoinnin yksikkönä on abstrakti käsite *story point*, miten sen avulla voidaan arvioida projektin kestoa?
- ▶ Kehitystiimin *velositeetti* (engl velocity) tarjoaa osittaisen ratkaisun tähän
- ▶ Velositeetilla tarkoitetaan *tiimin keskimäärin yhdessä sprintissä toteuttamien story pointtien määrää*

# Velositeetti

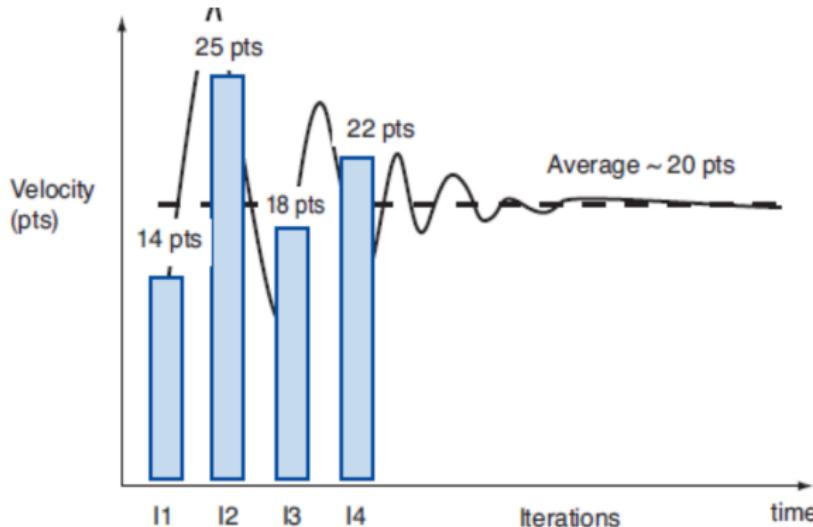
- ▶ Estimoinnin yksi tarkoitus on mahdollistaa koko projektin viemän aikamäärän summittainen arvointi
- ▶ Estimoinnin yksikkönä on abstrakti käsite *story point*, miten sen avulla voidaan arvioida projektin kestoa?
- ▶ Kehitystiimin *velositeetti* (engl velocity) tarjoaa osittaisen ratkaisun tähän
- ▶ Velositeetilla tarkoitetaan *tiimin keskimäärin yhdessä sprintissä toteuttamien story pointtien määrää*
- ▶ Jos velositeetti selvillä ja toteutettavaksi tarkoitettut storyt estimoitu, projektin keston arvio on helppo laskea  
$$(\text{estimaattien summa}) / \text{velositeetti} * \text{sprintin pituus}$$

- ▶ Projektin alkaessa velositeetti ei ole selvillä
  - ▶ ellei kyseessä ole jo yhdessä työskennellyt tiimi

- ▶ Projektin alkaessa velositeetti ei ole selvillä
  - ▶ ellei kyseessä ole jo yhdessä työskennellyt tiimi
- ▶ Velositeetti vaihtelee alussa melko paljon
  - ▶ Estimointi aluksi vaikeampaa varsinkin jos sovellusalue ja käytetyt teknologiat eivät ole täysin tuttuja



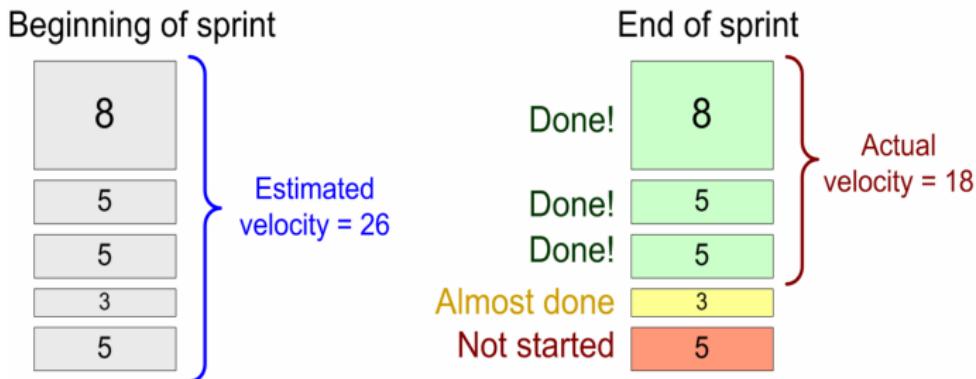
- ▶ Projektin alkaessa velositeetti ei ole selvillä
  - ▶ ellei kyseessä ole jo yhdessä työskennellyt tiimi
- ▶ Velositeetti vaihtelee alussa melko paljon
  - ▶ Estimointi aluksi vaikeampaa varsinkin jos sovellusalue ja käytetyt teknologiat eivät ole täysin tuttuja



- ▶ Velositeetti ja siihen perustuva projektin keston arvio tarkentuu pikkuhiljaa

- ▶ Ketterissä menetelmissä on oleellista kuvata mahdollisimman realistisesti projektin etenemistä

- ▶ Ketterissä menetelmissä on oleellista kuvata mahdollisimman realistisesti projektin etenemistä
- ▶ Velositeettiin lasketaan mukaan ainoastaan definition of doneen mukaisesti toteutetut storyt
  - ▶ “lähes valmiiksi” tehtyä työtä ei katsota ollenkaan tehdyksi

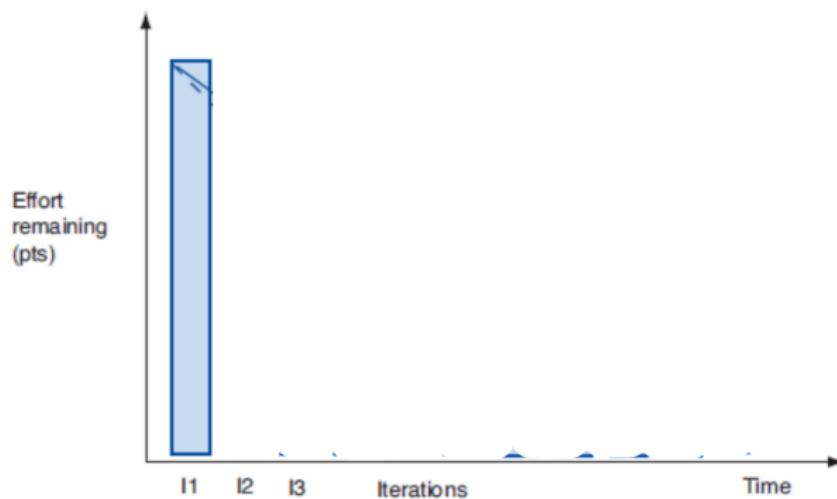


# Burndown

- ▶ Projektin etenemistä kuvataan joskus release burndown -kaavion avulla

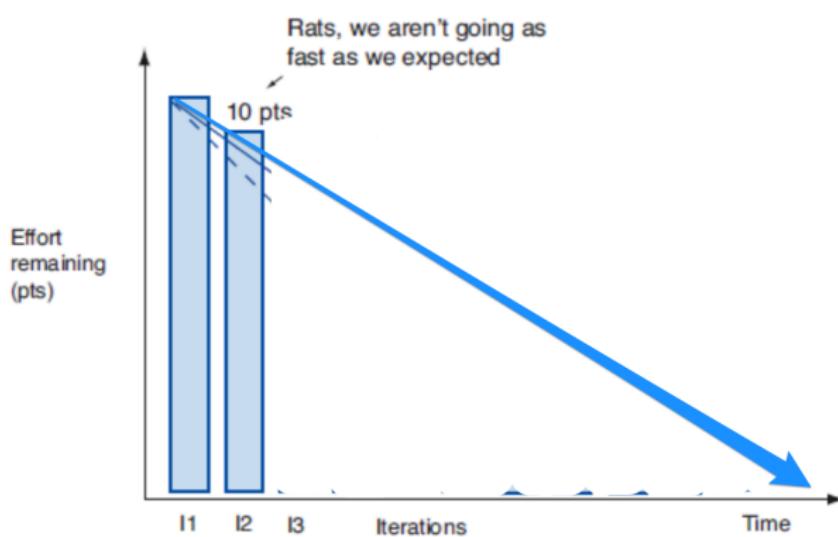
# Burndown

- ▶ Projektin etenemistä kuvataan joskus release burndown -kaavion avulla



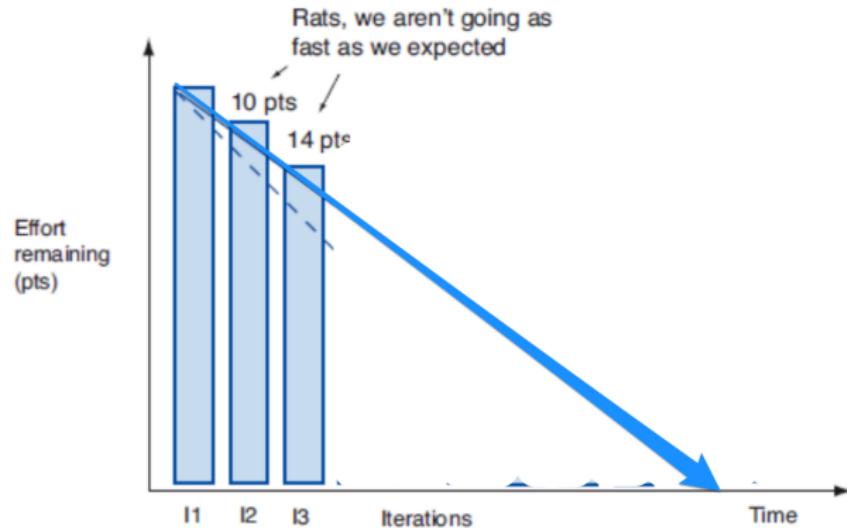
# Burndown

- ▶ Projektin etenemistä kuvataan yleensä release burndown -kaavion avulla



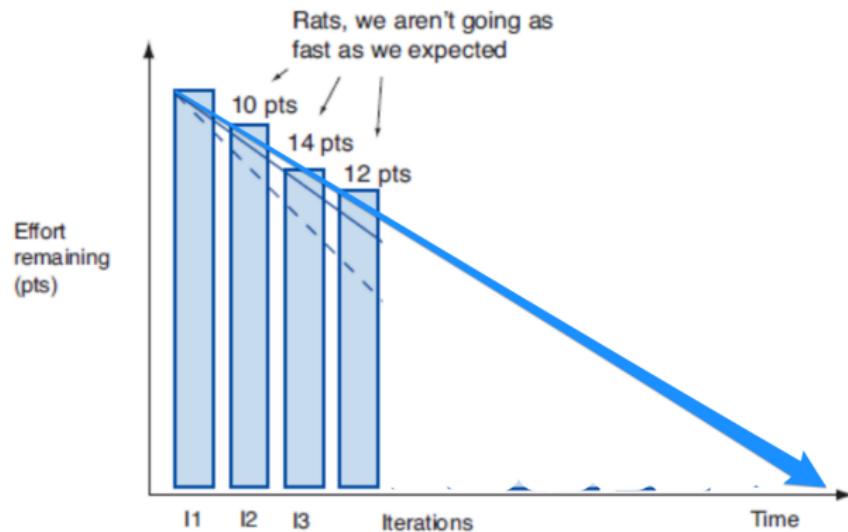
# Burndown

- ▶ Projektin etenemistä kuvataan yleensä release burndown -kaavion avulla



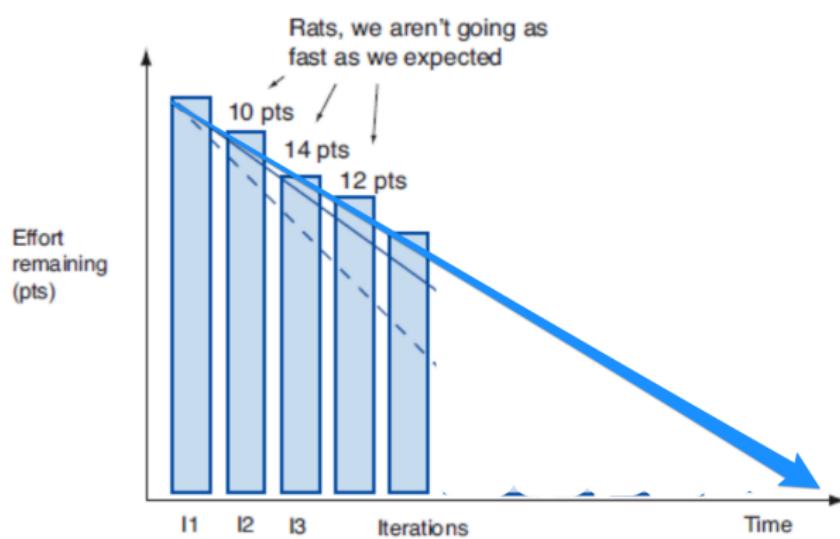
# Burndown

- ▶ Projektin etenemistä kuvataan yleensä release burndown -kaavion avulla



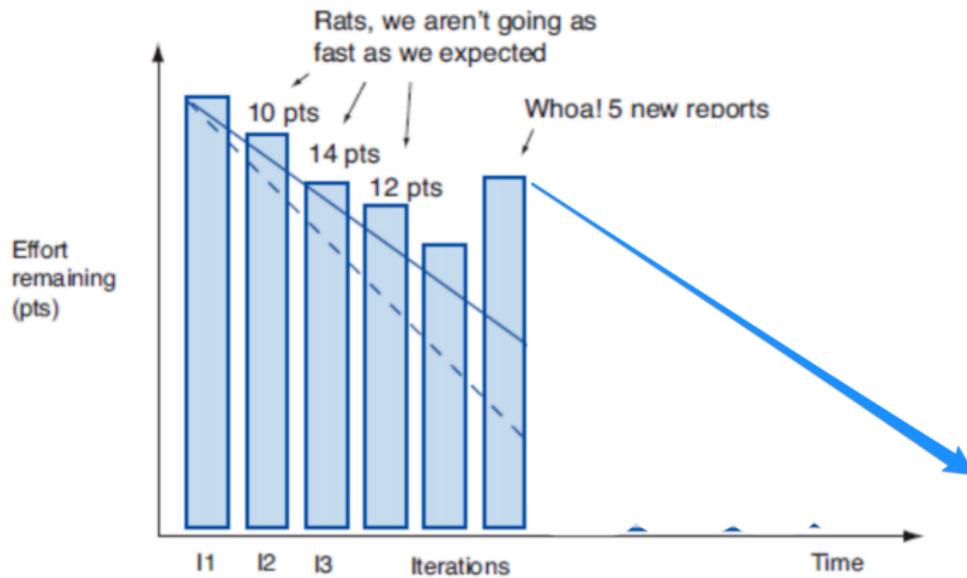
# Burndown

- ▶ Projektin etenemistä kuvataan yleensä release burndown -kaavion avulla



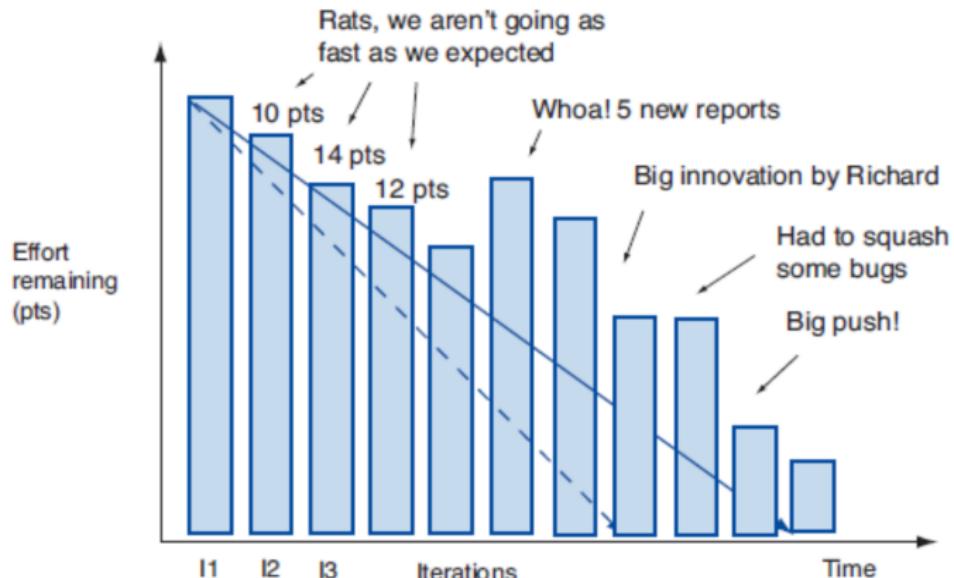
# Burndown

- ▶ Projektin etenemistä kuvataan yleensä release burndown -kaavion avulla



# Burndown

- ▶ Projektin etenemistä kuvataan yleensä release burndown -kaavion avulla



## Kannattaako estimointi? #NoEstimates

- ▶ Storyjen viemän työmääärän arvioimiseen kaksi motivaatiota
  - ▶ auttaa asiakasta priorisoinnissa
  - ▶ mahdollistaa koko projektin tai kokonaisuuden viemän ajan ja kustannuksen arvioinnin

# Kannattaako estimointi? #NoEstimates

- ▶ Storyjen viemän työmääärän arvioimiseen kaksi motivaatiota
  - ▶ auttaa asiakasta priorisoinnissa
  - ▶ mahdollistaa koko projektin tai kokonaisuuden viemän ajan ja kustannuksen arvioinnin
- ▶ Story point -pohjainen suhteellinen estimointi on saavuttanut vankan aseman
  - ▶ Scrum guide mainitsee että backlogin vaatimukset estimoituja
  - ▶ Samoin kuten monet parhaat käytänteet kuten DEEP

# Kannattaako estimointi? #NoEstimates

- ▶ Storyjen viemän työmääärän arvioimiseen kaksi motivaatiota
  - ▶ auttaa asiakasta priorisoinnissa
  - ▶ mahdollistaa koko projektin tai kokonaisuuden viemän ajan ja kustannuksen arvioinnin
- ▶ Story point -pohjainen suhteellinen estimointi on saavuttanut vankan aseman
  - ▶ Scrum guide mainitsee että backlogin vaatimukset estimoituja
  - ▶ Samoin kuten monet parhaat käytänteet kuten DEEP
- ▶ #NoEstimates-liike ruvennut kyseenalaistamaan story point -perustaista estimointitapaa
  - ▶ pitää siitä saavutettuja hyötyjä liian vähäisinä verrattuna käytettyyn aikaan ja vaivan

# Kannattaako estimointi? #NoEstimates

- ▶ Storyjen viemän työmääärän arvioimiseen kaksi motivaatiota
  - ▶ auttaa asiakasta priorisoinnissa
  - ▶ mahdollistaa koko projektin tai kokonaisuuden viemän ajan ja kustannuksen arvioinnin
- ▶ Story point -pohjainen suhteellinen estimointi on saavuttanut vankan aseman
  - ▶ Scrum guide mainitsee että backlogin vaatimukset estimoituja
  - ▶ Samoin kuten monet parhaat käytänteet kuten DEEP
- ▶ *#NoEstimates*-liike ruvennut kyseenalaistamaan story point -perustaista estimointitapaa
  - ▶ pitää siitä saavutettuja hyötyjä liian vähäisinä verrattuna käytettyyn aikaan ja vaivaan
- ▶ Yksinkertainen vaihtoehto: **arvioidaan velositeetti laskemalla kussakin sprintissä valmistuneiden storyjen lukumäärää**

# Kannattaako estimointi? #NoEstimates

- ▶ Storyjen viemän työmääärän arvioimiseen kaksi motivaatiota
  - ▶ auttaa asiakasta priorisoinnissa
  - ▶ mahdollistaa koko projektin tai kokonaisuuden viemän ajan ja kustannuksen arvioinnin
- ▶ Story point -pohjainen suhteellinen estimointi on saavuttanut vankan aseman
  - ▶ Scrum guide mainitsee että backlogin vaatimukset estimoituja
  - ▶ Samoin kuten monet parhaat käytänteet kuten DEEP
- ▶ *#NoEstimates*-liike ruvennut kyseenalaistamaan story point -perustaista estimointitapaa
  - ▶ pitää siitä saavutettuja hyötyjä liian vähäisinä verrattuna käytettyyn aikaan ja vaivan
- ▶ Yksinkertainen vaihtoehto: **arvioidaan velositeetti laskemalla kussakin sprintissä valmistuneiden storyjen lukumäärää**
- ▶ Toimii jos storyt riittävän tasakokoisia?

Tauko 10 min

Sprintti

## Sprintin suunnittelu

- ▶ Kertauksena alkuviikolta: Scrum määrittelee pidettäväksi ennen jokaista sprinttiä *suunnittelupalaverin*

## Sprintin suunnittelu

- ▶ Kertauksena alkuviikolta: Scrum määrittelee pidettäväksi ennen jokaista sprinttiä *suunnittelupalaverin*
- ▶ Palaverin ensimmäinen tavoite on selvittää *mitä* sprintin aikana tehdään
  - ▶ Lähtökohtana DEEP product backlog

# Sprintin suunnittelu

- ▶ Kertauksena alkuviikolta: Scrum määrittelee pidettäväksi ennen jokaista sprinttiä *suunnittelupalaverin*
- ▶ Palaverin ensimmäinen tavoite on selvittää *mitä* sprintin aikana tehdään
  - ▶ Lähtökohtana DEEP product backlog
- ▶ Product owner esittelee backlogin kärjessä olevat vaatimukset
  - ▶ Tiimin on tarkoitus olla riittävällä tasolla selville mitä vaatimuksilla tarkoitetaan

## Sprintin suunnittelu

- ▶ Kertauksena alkuviikolta: Scrum määrittelee pidettäväksi ennen jokaista sprinttiä *suunnittelupalaverin*
- ▶ Palaverin ensimmäinen tavoite on selvittää *mitä* sprintin aikana tehdään
  - ▶ Lähtökohtana DEEP product backlog
- ▶ Product owner esittelee backlogin kärjessä olevat vaatimukset
  - ▶ Tiimin on tarkoitus olla riittävällä tasolla selville mitä vaatimuksilla tarkoitetaan
- ▶ Tiimi valitsee niin monta storyä kuin se arvioi kykenevänsä sprintin aikana toteuttamaan definition of doneen laadulla

## Sprintin tavoite

- ▶ Suunnittelun yhteydessä määritellään *sprintin tavoite* (goal)
  - ▶ Lyhyt, yhden tai kahden lauseen kuvausta siitä, mitä tiimi on aikissa sprintin aikana tehdä

## Sprintin tavoite

- ▶ Suunnittelun yhteydessä määritellään *sprintin tavoite* (goal)
  - ▶ Lyhyt, yhden tai kahden lauseen kuvausta siitä, mitä tiimi on aikissa sprintin aikana tehdä
- ▶ K. Schwaber, ensimmäisen sprintin tavoite:
  - ▶ *demonstrate a key piece of user functionality on the selected technology*

## Sprintin tavoite

- ▶ Suunnittelun yhteydessä määritellään *sprintin tavoite* (goal)
  - ▶ Lyhyt, yhden tai kahden lauseen kuvausta siitä, mitä tiimi on aikeissa sprintin aikana tehdä
- ▶ K. Schwaber, ensimmäisen sprintin tavoite:
  - ▶ *demonstrate a key piece of user functionality on the selected technology*
- ▶ Verkkokaupan sprinttien tavoitteita voisivat olla:
  - ▶ Ostoskorin perustoiminnallisuus: tuotteiden lisäys ja poisto
  - ▶ Ostosten maksaminen ja toimitustavan valinta

# Sprintin tavoite

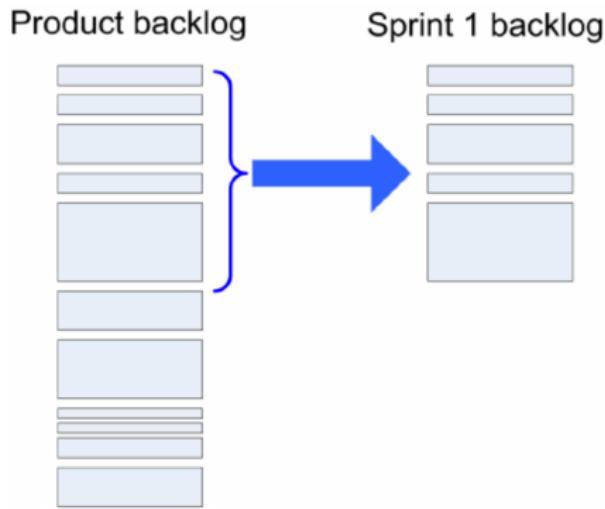
- ▶ Suunnittelun yhteydessä määritellään *sprintin tavoite* (goal)
  - ▶ Lyhyt, yhden tai kahden lauseen kuvausta siitä, mitä tiimi on aikeissa sprintin aikana tehdä
- ▶ K. Schwaber, ensimmäisen sprintin tavoite:
  - ▶ *demonstrate a key piece of user functionality on the selected technology*
- ▶ Verkkokaupan sprinttien tavoitteita voisivat olla:
  - ▶ Ostoskorin perustoiminnallisuus: tuotteiden lisäys ja poisto
  - ▶ Ostosten maksaminen ja toimitustavan valinta
- ▶ Lyhyt kuvaus parempi niille sidosryhmäläisille, joita ei kiinnosta seurata tapahtumia yksittäisten storyjen tarkkuudella

## Sprintiin valittavat storyt

- ▶ Sprintin tavoitteen asettamisen lisäksi tulee valita backlogista sprintin aikana toteutettavat storyt
  - ▶ Kehitystiimi päättää kuinka monta storya sprinttiin otetaan

## Sprintiin valittavat storyt

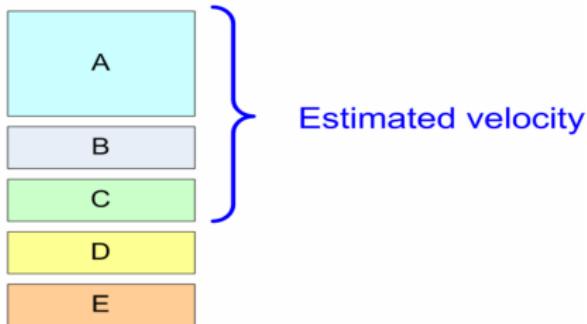
- ▶ Sprintin tavoitteen asettamisen lisäksi tulee valita backlogista sprintin aikana toteutettavat storyt
  - ▶ Kehitystiimi päättää kuinka monta storya sprinttiin otetaan
- ▶ Jos velositeetti on selvillä, on valinta periaatteessa helppo



- ▶ Jos velositettia ei tiedossa, käytetään harkintaa

- ▶ Product owner voi vaikuttaa sprinttiin mukaan otettaviin storyihin tekemällä uudelleenpriorisointia

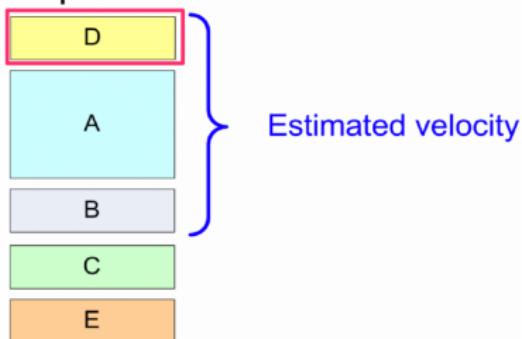
### Product backlog



- ▶ Entä jos myös D halutaan sprinttiin?

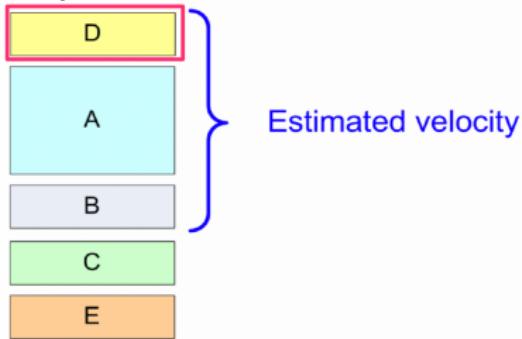
► Uudelleenpriorisoidaan

Option 1



- ▶ Uudelleenpriorisoidaan

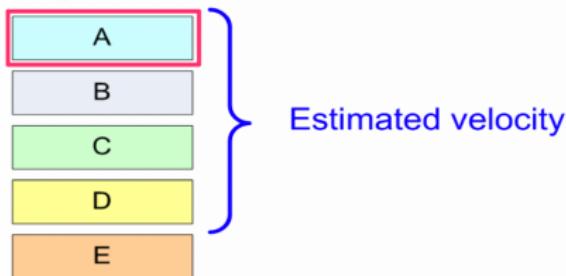
Option 1



- ▶ Entä jos myös C halutaan mukaan?

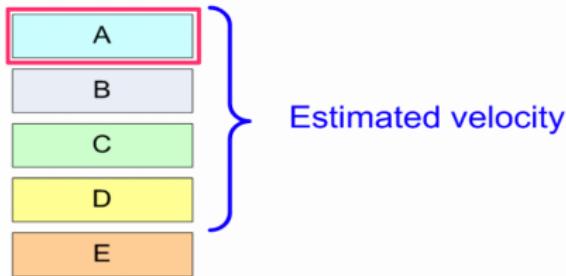
► Pienennetään A:n kuvaamaa toiminnallisuutta

Option 2



- ▶ Pienennetään A:n kuvaamaa toiminnallisuutta

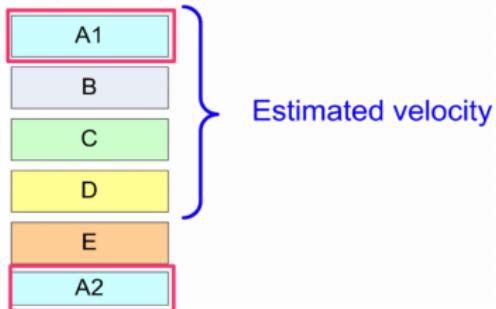
### Option 2



- ▶ Entä jos A ei saa pienentyä

- ▶ Jaetaan A kahteen osaan

Option 3



- ▶ Tärkeämpi osa toiminnallisuutta eli A1 mahtuu mukaan sprinttiin, vähemmän tärkeät osat eli A2 jää myöhempiin sprintteihin

## User storyjen jakaminen useampaan osaan

- ▶ Haastava aihe, palataan siihen tänään jos aikaa jää
- ▶ Kurssinmateriaalissa jonkin verran ohjeistusta asiaan
- ▶ Pääperiaate: jakamisessa syntyvien storyjen edelleen noudatettava INVEST-kriteerejä

## Miten sprintin tavoitteeseen päästään?

- ▶ Sprintin suunnittelun yhteydessä sprinttiin valituille user storyille tehdään karkean tason suunnittelu

## Miten sprintin tavoitteeseen päästään?

- ▶ Sprintin suunnittelun yhteydessä sprinttiin valituille user storyille tehdään karkean tason suunnittelu
- ▶ Mietitään mitä *teknisen tason tehtäviä* (task) on toteutettava, jotta user story saadaan valmiaksi

## Miten sprintin tavoitteeseen päästään?

- ▶ Sprintin suunnittelun yhteydessä sprinttiin valituille user storyille tehdään karkean tason suunnittelu
- ▶ Mietitään mitä *teknisen tason tehtäviä* (task) on toteutettava, jotta user story saadaan valmiiksi
- ▶ Suunnitellaan komponentteja ja rajapintoja karkealla tasolla

## Miten sprintin tavoitteeseen päästään?

- ▶ Sprintin suunnittelun yhteydessä sprinttiin valituille user storyille tehdään karkean tason suunnittelu
- ▶ Mietitään mitä *teknisen tason tehtäviä* (task) on toteutettava, jotta user story saadaan valmiiksi
- ▶ Suunnitellaan komponentteja ja rajapintoja karkealla tasolla
- ▶ Huomioidaan uusien storyjen aiheuttamat muutokset olemassa olevaan osaan sovelluksesta

## Storyn jako taskeihin, esimerkki

- ▶ Esimerkiksi *tuotteen lisääminen ostoskoriin*, voitaisiin jakaa seuraaviin teknisiin taskeihin:
  - ▶ sessio, joka muistaa asiakkaan tila
  - ▶ oliot ja tietorakenteet ostoskorin ja ostoksen esittämiseen
  - ▶ laajennus tietokanttaskeemaan
  - ▶ html-näkymää päivitettävä tarvittavilla painikkeilla
  - ▶ kontrolleri painikkeiden käsittelyyn
  - ▶ yksikkötestit kontrollerille ja ostoskorin logiikalle
  - ▶ hyväksymätestien automatisointi

## Storyn jako taskeihin, esimerkki

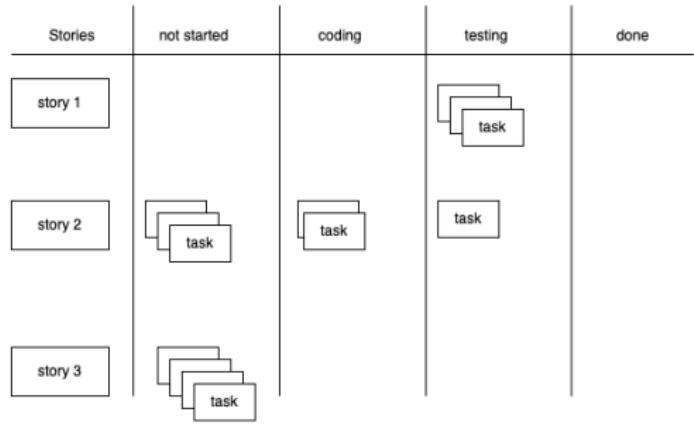
- ▶ Esimerkiksi *tuotteen lisääminen ostoskoriin*, voitaisiin jakaa seuraaviin teknisiin taskeihin:
  - ▶ sessio, joka muistaa asiakkaan tila
  - ▶ oliot ja tietorakenteet ostoskorin ja ostoksen esittämiseen
  - ▶ laajennus tietokantaskeemaan
  - ▶ html-näkymää päivitettyvä tarvittavilla painikkeilla
  - ▶ kontrolleri painikkeiden käsittelyyn
  - ▶ yksikkötestit kontrollerille ja ostoskorin logiikalle
  - ▶ hyväksymätestien automatisointi
- ▶ Kaikkia storyyn liittyviä taskeja ei sprintin suunnittelun aikana löydetä
  - ▶ Uusia taskeja generoidaan tarvittaessa sprintin edetessä

## Sprint backlog

- ▶ *Sprint backlog* koostuu sprinttiin valituista storyista ja niihin liittyvistä tehtävistä eli taskeista

# Sprint backlog

- ▶ *Sprint backlog* koostuu sprinttiin valituista storyista ja niihin liittyvistä tehtävistä eli taskeista
- ▶ Sprint backlog usein organisoitu taskboardiksi



- ▶ Taskit niiden valmistumisastetta kuvaavassa sarakkeessa

## Sprint backlogin työmäääräarviot

- ▶ Sprintissä arvioidaan päivittäin kunkin taskin *jäljellä olevaksi arvioitua työmääräää*
  - ▶ Usein tapana tehdä arviot tunteina

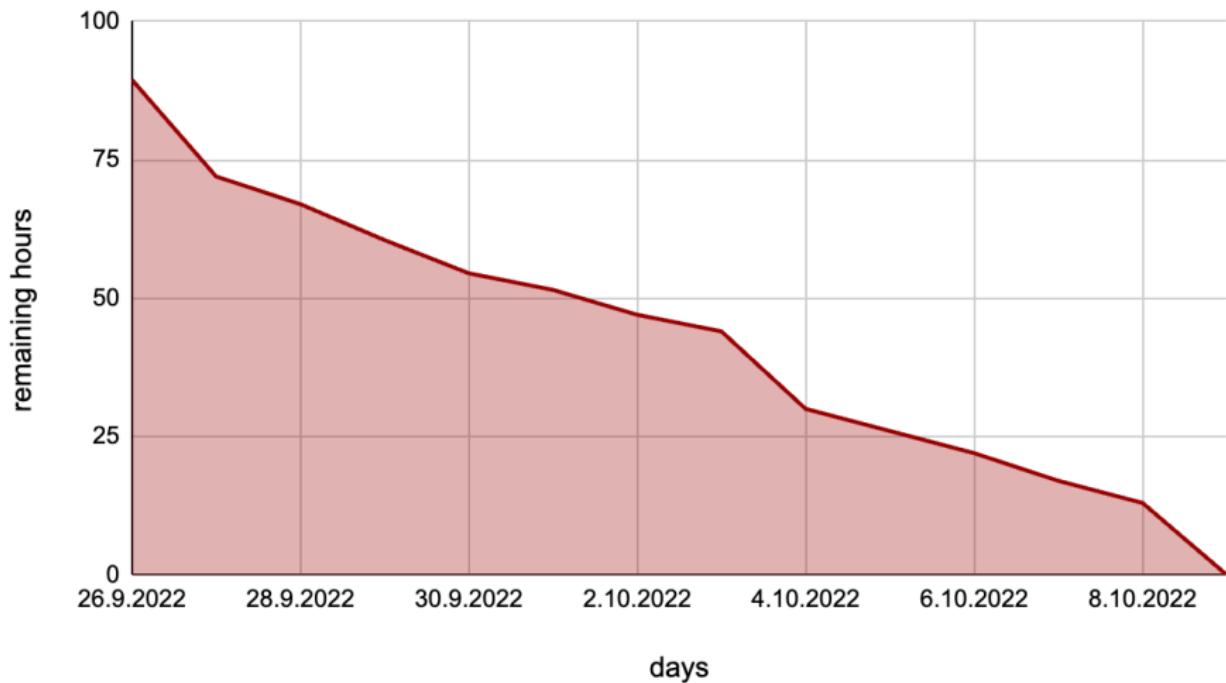
# Sprint backlogin työmäääräarviot

- ▶ Sprintissä arvioidaan päivittäin kunkin taskin *jäljellä olevaksi arvioitua työmääräää*
  - ▶ Usein tapana tehdä arviot tunteina

	Tasks	Initial estimate (h)	mon	tue	wed	thu	fri	sat	sun
		9.10.2022	10.10.2022	11.10.2022	12.10.2022	13.10.2022	14.10.2022	15.10.2022	16.10.2022
As a user I can view inventory reports on a map, so I can see all searched places	(front) Add the map to the home page (back) Endpoint for getting the areas (front) Get the areas from the backend (front) Show the areas on the map (front) When area is clicked show popup with small summary of the report	1 1 1 5 2	0 0,5 1 5 2	0 0 0 0,5 2	0 0 0 0,5 2	0 0 0 0,5 1	0 0 0 0 0,5	0 0 0 0 0,5	0 0 0 0 0,5
As a user I can view details of submitted water area inventory report	(front) Create page/component for inventory report (back) Create endpoint for getting report details by id	2 2	2 2	2 1	2 0	2 0	2 0	2 0	2 0
As a user I can view list of submitted water area inventory reports	(front) create report list component (back) create endpoint for getting list of reports (front) add list filtering (front) find a way to get city from location (front) get (all) reports from the backend	3 3 2 5 1	3 1 2 5 1	3 1 2 5 0,5	2,5 0 2 4 0,5	1 0 2 1 0,5	1 0 2 1 0,5	1 0 2 1 0,5	1 0 2 1 0,5
As a logged in user I can submit a report without giving contact details again	(front) Name, email and phone are prefilled on the report form (back) User reference is attached to the report document (front) Logged in user is sent to the form	3 4 2	3 4 2	3 3 0	2 0 0	1 0 0	1 0 2	1 0 2	1 0 0

# Sprintin burndown etenemisen seurantaan

## Burndown



# Kannattaako taskeille tehdä työmäääräarviot?

- ▶ A *Scrum book* 2019 ei suosittele taskien tasolla tehtävää työmäääräarviointia
  - ▶ Kehottaa seuraamaan sprinttien aikana ainoastaan sitä kuinka monen story pointin verran storyja saatu valmiiksi

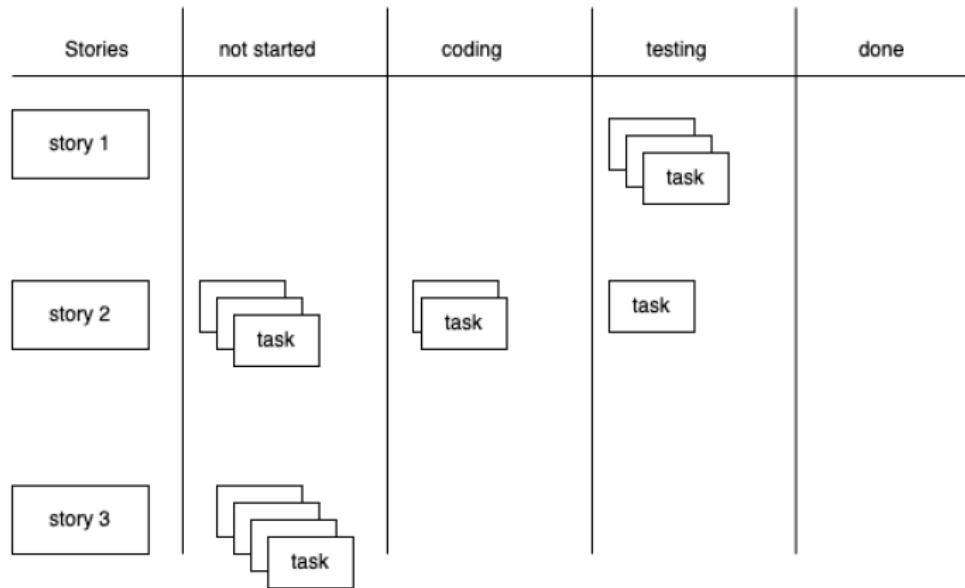
# Kannattaako taskeille tehdä työmäääräarviot?

- ▶ A *Scrum book 2019* ei suosittele taskien tasolla tehtävää työmäääräarviointia
  - ▶ Kehottaa seuraamaan sprinttien aikana ainoastaan sitä kuinka monen story pointin verran storyja saatu valmiiksi
- ▶ On mahdollista, että tiimi saa sprintissä valmiiksi lähes kaikki taskit, saamatta valmiiksi yhtäkään storya
  - ▶ Burn down voi näyttää pitkään melko hyvältä, mutta asiakkaan saama arvo on lopulta nolla

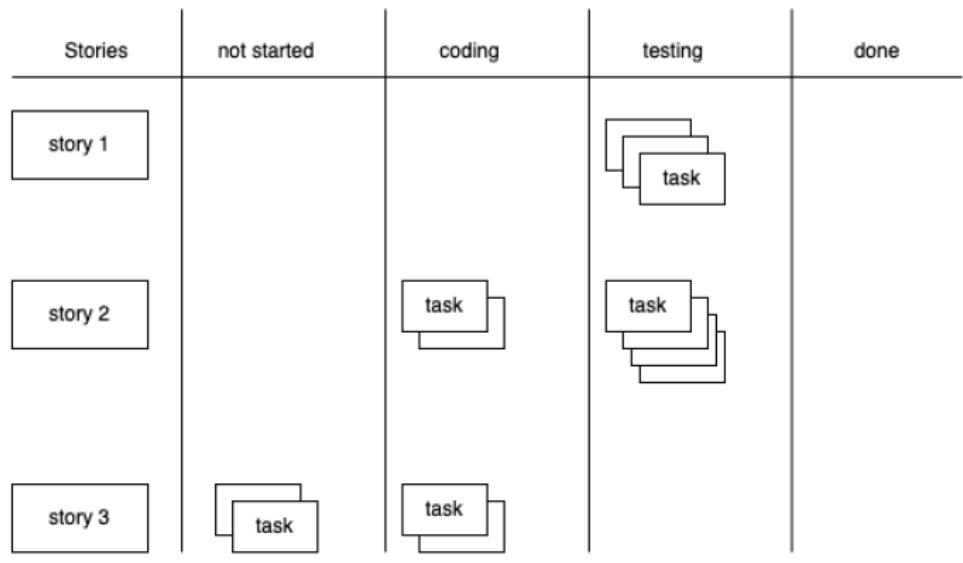
# Kannattaako taskeille tehdä työmäääräarviot?

- ▶ A *Scrum book 2019* ei suosittele taskien tasolla tehtävää työmäääräarviointia
  - ▶ Kehottaa seuraamaan sprinttien aikana ainoastaan sitä kuinka monen story pointin verran storyja saatu valmiiksi
- ▶ On mahdollista, että tiimi saa sprintissä valmiiksi lähes kaikki taskit, saamatta valmiiksi yhtäkään storya
  - ▶ Burn down voi näyttää pitkään melko hyvältä, mutta asiakkaan saama arvo on lopulta nolla
- ▶ Yksinkertainen tapa sprintin etenemisen seurantaan
  - ▶ laske, tai katsoa taskboardilta, mikä on jo valmiiden ja vielä valmistumattomien sprinttiin kuuluvien taskien lukumäärä

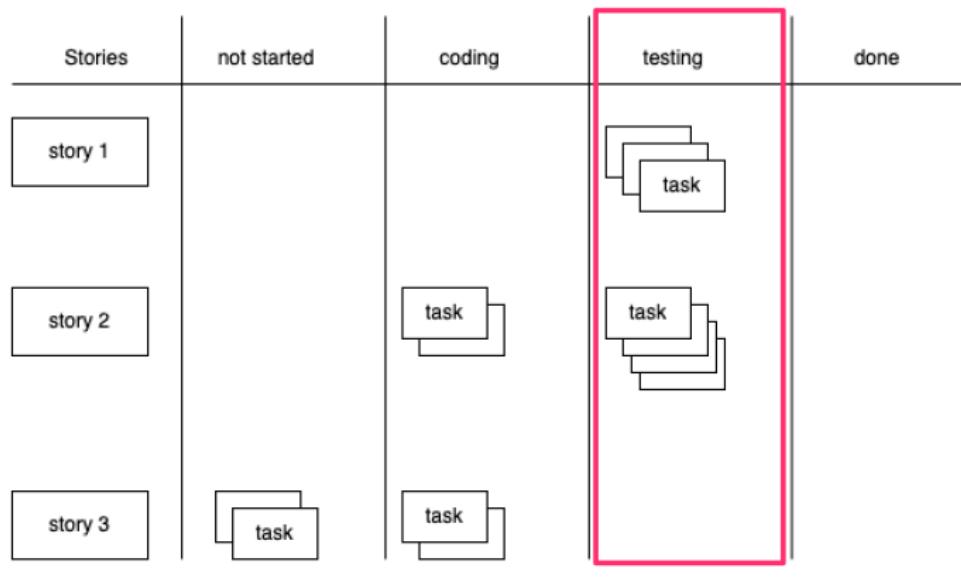
# Joskus Sprinteissä ...



# Joskus Sprinteissa käy näin



# Puolivalmis työ kasautuu ja asiat eivät valmistu

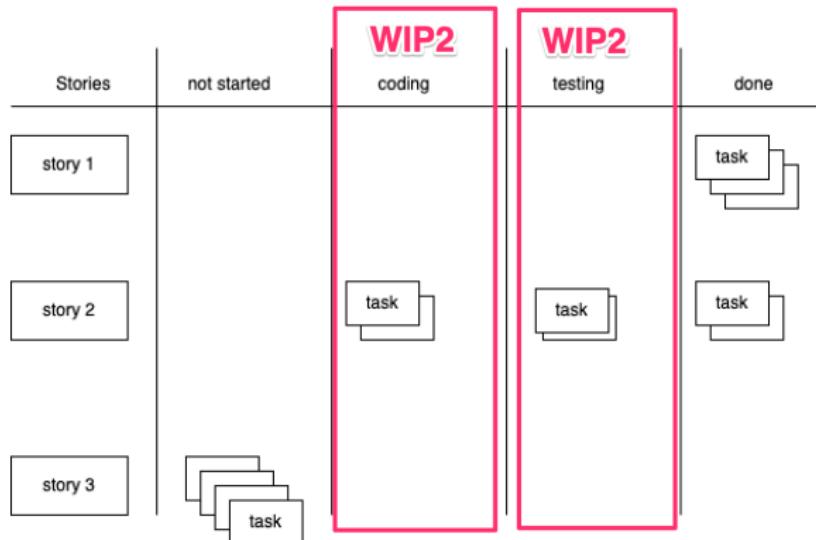


## WIP-rajoitteet

- ▶ Yhtä aikaa työn alla olevien taskien suuri määrä voi koitua ongelmaksi
  - ▶ Riski sille, että sprintin päätyttyä paljon osittain valmiita storyja kasvaa

# WIP-rajoitteet

- ▶ Yhtä aikaa työn alla olevien taskien suuri määrä voi koitua ongelmaksi
  - ▶ Riski sille, että sprintin päätyttyä paljon osittain valmiita storyja kasvaa
- ▶ Ratkaisu: *work in progress eli WIP* -rajoitteet



# Kanban ja Lean

- ▶ WIP-rajoitusten idea on peräisin *Kanban*-menetelmästä, joka on eräs keskeisimmistä *Lean*-ajattelun työkaluista
  - ▶ Lean-ajattelu on peräisin jo kymmeniä vuosia vanhasta Toyota Production Systemistä

# Kanban ja Lean

- ▶ WIP-rajoitusten idea on peräisin *Kanban*-menetelmästä, joka on eräs keskeisimmistä *Lean*-ajattelun työkaluista
  - ▶ Lean-ajattelu on peräisin jo kymmeniä vuosia vanhasta Toyota Production Systemistä
- ▶ Lean-ajattelun taustalla on idea *hukan* eli asiakkaalle arvoa tuottamattomien asioiden eliminoimisessa

# Kanban ja Lean

- ▶ WIP-rajoitusten idea on peräisin *Kanban*-menetelmästä, joka on eräs keskeisimmistä *Lean*-ajattelun työkaluista
  - ▶ Lean-ajattelu on peräisin jo kymmeniä vuosia vanhasta Toyota Production Systemistä
- ▶ Lean-ajattelun taustalla on idea *hukan* eli asiakkaalle arvoa tuottamattomien asioiden eliminoimisessa
- ▶ Toiminnallisuudet tuovat arvoa vasta käytössä, sitä ennen ne sitovat turhaan kustannuksia ja tuovat riskejä

# Kanban ja Lean

- ▶ WIP-rajoitusten idea on peräisin *Kanban*-menetelmästä, joka on eräs keskeisimmistä *Lean*-ajattelun työkaluista
  - ▶ Lean-ajattelu on peräisin jo kymmeniä vuosia vanhasta Toyota Production Systemistä
- ▶ Lean-ajattelun taustalla on idea *hukan* eli asiakkaalle arvoa tuottamattomien asioiden eliminoimisessa
- ▶ Toiminnallisuudet tuovat arvoa vasta käytössä, sitä ennen ne sitovat turhaan kustannuksia ja tuovat riskejä
- ▶ Hukkaa muun muassa: **osittain tehty työ, välivarastointi ja turha odottaminen**

## WIP-rajoitteiden soveltaminen

- ▶ WIP-rajoitteita voidaan soveltaa Scrumin yhteydessä monella tavalla

## WIP-rajoitteiden soveltaminen

- ▶ WIP-rajoitteita voidaan soveltaa Scrumin yhteydessä monella tavalla
- ▶ Aika tavallista on rajoittaa eri työvaiheessa, esim. toteutuksen olevien taskien määrää

## WIP-rajoitteiden soveltaminen

- ▶ WIP-rajoitteita voidaan soveltaa Scrumin yhteydessä monella tavalla
- ▶ Aika tavallista on rajoittaa eri työvaiheessa, esim. toteutuksen olevien taskien määrää
- ▶ tai yksittäisellä sovelluskehittäjän kerrallaan työn alla olevien töiden määrää

## WIP-rajoitteiden soveltaminen

- ▶ WIP-rajoitteita voidaan soveltaa Scrumin yhteydessä monella tavalla
- ▶ Aika tavallista on rajoittaa eri työvaiheessa, esim. toteutuksen olevien taskien määrää
- ▶ tai yksittäisellä sovelluskehittäjän kerrallaan työn alla olevien töiden määrää
- ▶ Järkevästi lienee rajoittaa sprintin aikana yhtäaikaa työn alla olevien storyjen määrää mahdollisimman pieneksi

## WIP-rajoitteiden soveltaminen

- ▶ WIP-rajoitteita voidaan soveltaa Scrumin yhteydessä monella tavalla
- ▶ Aika tavallista on rajoittaa eri työvaiheessa, esim. toteutuksen olevien taskien määrää
- ▶ tai yksittäisellä sovelluskehittäjän kerrallaan työn alla olevien töiden määrää
- ▶ Järkevästi lienee rajoittaa sprintin aikana yhtäaikaa työn alla olevien storyjen määrää mahdollisimman pieneksi
- ▶ WIP-rajoitteita säädetään usein retrospektiivien yhteydessä jos kehitystyössä havaitaan ongelmia



## Storyjen jakaminen

- ▶ Haastava aihe aloittelijalle ja joskus myös kokeneille ohjelmistokehittäjille
- ▶ Pääperiaate: jakamisessa syntyvien storyjen edelleen noudatettava INVEST-kriteerejä
- ▶ Richard Lawrencen ohjeita

## Pattern 1: business rule variations

*As a user, I can search for flights with flexible dates.*

## Pattern 1: business rule variations

*As a user, I can search for flights with flexible dates.*

kannattaa jakaa siten että jokainen näistä ehdosta eritellään omaksi storykseen

- ▶ ... as “*between dates x and y*”
- ▶ ... as “*a weekend in December*”
- ▶ ... as “*± n days of dates x and y*”

## Pattern 2: simple/complex

*As a user, I can search for flights between two destinations*

## Pattern 2: simple/complex

*As a user, I can search for flights between two destinations*

voidaan jakaa seuraavasti

- ▶ ... *when only direct flights used*
- ▶ ... *specifying a max number of stops*
- ▶ ... *including nearby airports*

## Pattern 3: major effort

*As a user, I can pay for my flight with VISA, MasterCard, Diners Club, or American Express.*

## Pattern 3: major effort

*As a user, I can pay for my flight with VISA, MasterCard, Diners Club, or American Express.*

voitaisiin jakaa kahtia, missä ensimmäisessä storyssa vasta hoidettaisiin yksi luottokorttityyppi, ja seuraava story yleistäisi toiminnan kaikkiin kortteihin:

- ▶ ... *I can pay with VISA*
- ▶ ... *I can pay with all four credit card types (VISA, MC, DC, AMEX) (given one card type already implemented)*

## Pattern 4: data entry methods

*As a user, I can search for flights between two destinations*

## Pattern 4: data entry methods

*As a user, I can search for flights between two destinations*

jakaantuukin helposti kahteen esim. seuraavasti

- ▶ ... *using simple date input*
- ▶ ... *with a fancy calendar UI*

## Pattern 5: Defer Performance

*As a user, I can search for flights between two destinations*

## Pattern 5: Defer Performance

*As a user, I can search for flights between two destinations*

jakaantuu kahtia seuraavasti:

- ▶ ... slow—just get it done, show a “searching” animation
- ▶ ... in under 5 seconds

## Pattern 6: Operations

*As a user, I can manage my account*

## Pattern 6: Operations

*As a user, I can manage my account*

jakaantuu moneen osaan

- ▶ ... *I can sign up for an account*
- ▶ ... *I can edit my account settings*
- ▶ ... *I can cancel my account*

## Pattern 7: Break Out a Spike

Jos tiimi ei ole toteuttanut koskaan luottokorttimaksuun liittyvää toiminnallisuutta, user storysta

*As a user, I can pay by credit card*

## Pattern 7: Break Out a Spike

Jos tiimi ei ole toteuttanut koskaan luottokorttimaksuun liittyvää toiminnallisuutta, user storysta

*As a user, I can pay by credit card*

kannattaa eriyttää aikarajattu eksperimentti joka suoritetaan aiemmassa sprintissä.

Tämän jälkeen toivon mukaan varsinaisen toiminnallisuuden toteuttava story osataan estimoida paremmin:

- ▶ *Investigate credit card processing*
- ▶ *Implement credit card processing*