

## Ohjelmistotuotanto

Matti Luukkainen ja ohjaajat Kalle Ilves, Silva Perander, Topias Pyykönen, Jussi Laisi, Petrus Peltola, Kristian Krok

syksy 2020

Luento 4

2.11.2020

## 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?

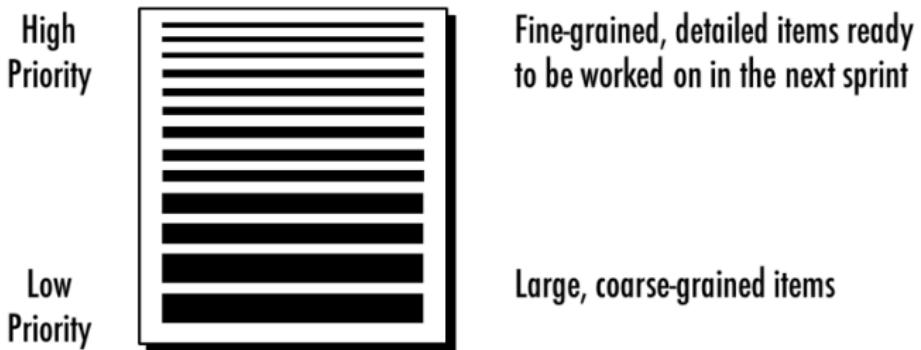
## nopea kertaus eiliseltä

- ▶ 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
- ▶ *Detailed appropriately eli sopivan detaljoitu*
  - ▶ ylempänä tarkkoja
  - ▶ alempana 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 grooming/refinement)
  - ▶ Pääasiallinen vastuu on product ownerilla
  - ▶ Backlogin hoitamiseen osallistuu koko kehitystiiimi
  - ▶ Scrum suosittlee että noin 10% sprintin työajasta käytetään backlog groomingiin

## “Ready” story ja epiikki

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

## “Ready” story ja epiikki

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

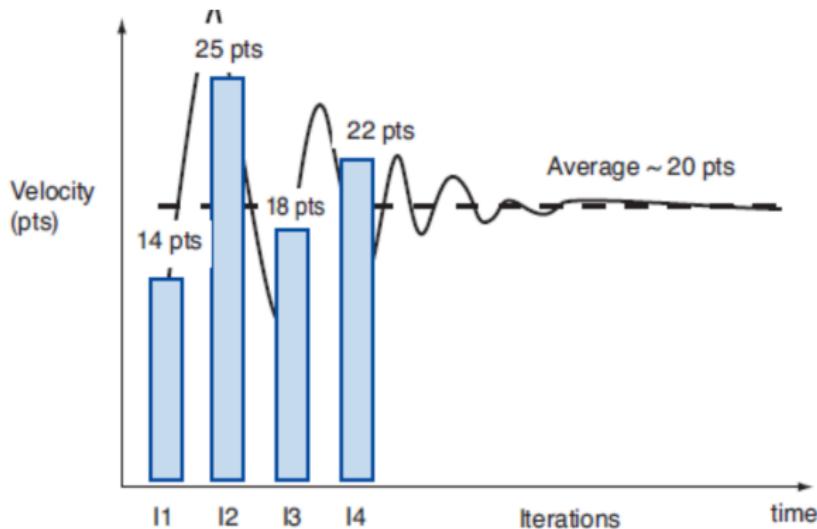
# Velositeetti

- ▶ 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

*(storyjen estimaattien summa) / velositeetti \* sprintin pituus*

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

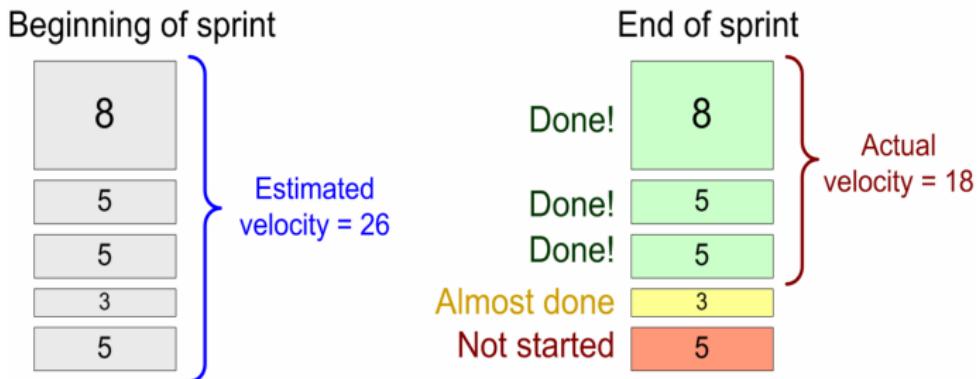
- ▶ Projektin alkaessa velositeetti ei ole selvillä, ellei kyseessä ole jo yhdessä työskennellyt tiimi
- ▶ Velositeetti vaihtelee alussa melko paljon
  - ▶ Estimointi aluksi vaikeampaa varsinkin jos sovellusalue ja käytetyt teknologiat eivät ole täysin tuttuja



- ▶ Velositeetti ja siihen perustuva projektin keston arvio alkaakin tarkentumaan pikkuhiljaa

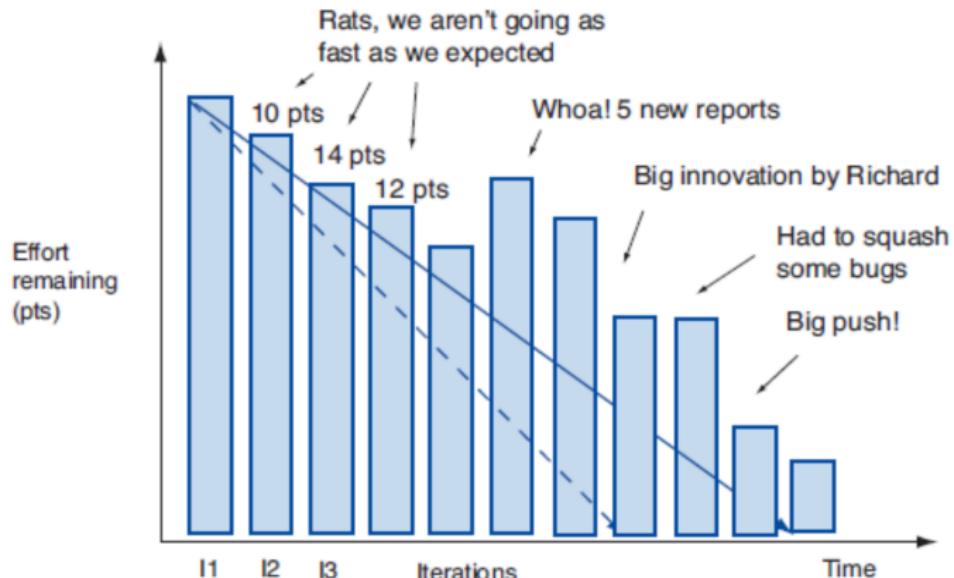
- ▶ Ketterissä menetelmissä on oleellista kuvata mahdollisimman realistisesti projektin etenemistä

- ▶ 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



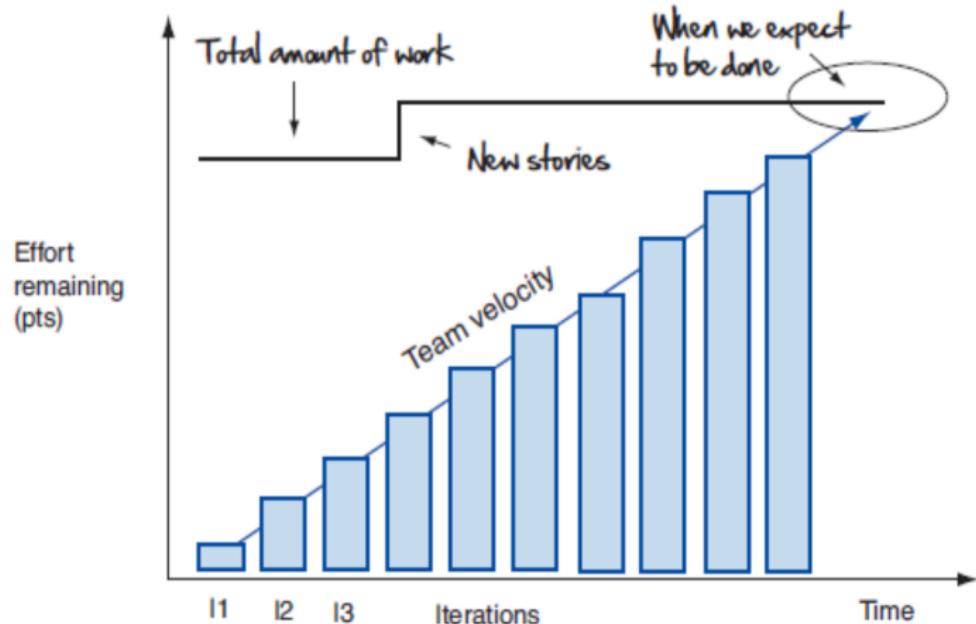
# Burndown

- ▶ Projektin etenemistä kuvataan yleensä release burndown -kaavion avulla



# Burnup

- ▶ Tuo selkeämmin esiin kesken projektin etenemisen tapahtuvan työmääärän kasvun



## 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 estimoitujia
  - ▶ 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 vaivaan

# 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*
- ▶ Monien kokemuksen mukaan toimii varsin hyvin, jos storyt riittävän tasakokoisia

## Sprintin suunnittelu

- ▶ Kertauksena viime viikolta: Scrum määrittelee pidettäväksi ennen jokaista sprinttiä suunnittelupalaverin

## Sprintin suunnittelu

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

## Sprintin suunnittelu

- ▶ Kertauksena viime viikolta: 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 viime viikolta: 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* (sprint goal)
- ▶ Lyhyt, yhden tai kahden lauseen kuvausta siitä, mitä tiimi on aikeissa sprintin aikana tehdä

## Sprintin tavoite

- ▶ Suunnittelun yhteydessä määritellään *sprintin tavoite* (sprint 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

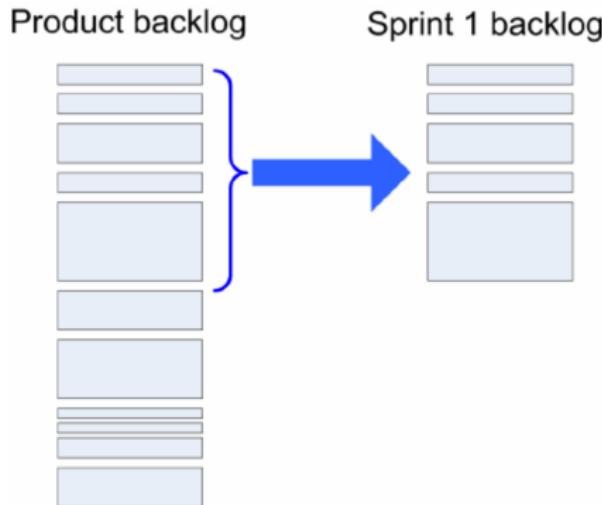
- ▶ Suunnittelun yhteydessä määritellään *sprintin tavoite* (sprint goal)
- ▶ Lyhyt, yhden tai kahden lauseen kuvausta siitä, mitä tiimi on aikeissa sprintin aikana tehdä
- ▶ K. Schwaber ensimmäisen sprintin tavoite: *demonstrate a key piece of user functionality on the selected technology*
- ▶ Verkkokaupan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

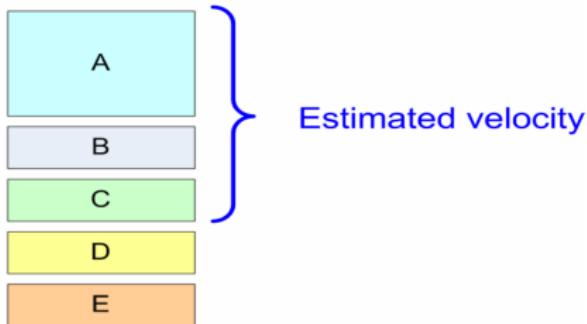
- ▶ Sprintin tavoitteen asettamisen lisäksi tulee valita backlogista sprintin aikana toteutettavat storyt
- ▶ Kehitystiimi päätää kuinka monta storya sprinttiin otetaan
- ▶ Jos velositeetti on selvillä, on valinta periaatteessa helppo



- ▶ Jos velositettia ei tiedossa, käytetään harkintaa

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

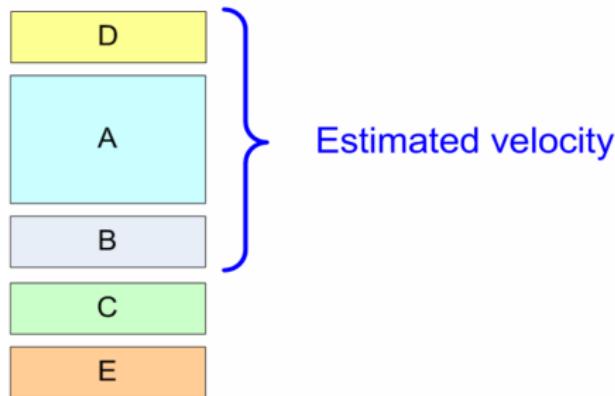
### Product backlog



- ▶ Entä jos myös D halutaan sprinttiin?

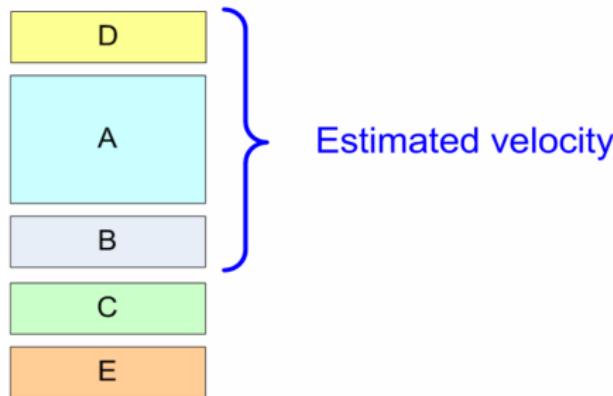
► Uudelleenpriorisoidaan

Option 1



- ▶ Uudelleenpriorisoidaan

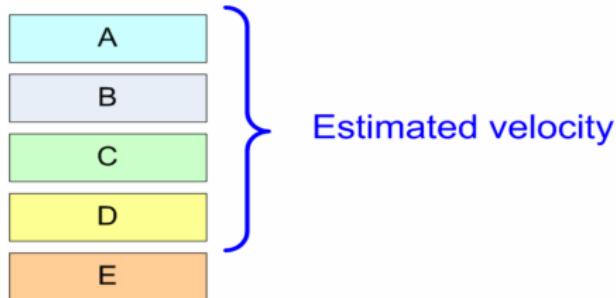
Option 1



- ▶ Entä jos myös C halutaan mukaan?

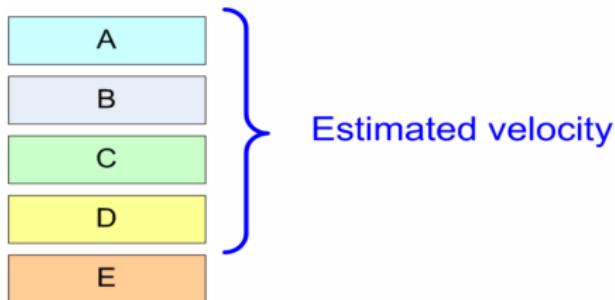
► Pienennetään A:n kuvaamaa toiminnallisuutta

### Option 2



- ▶ Pienennetään A:n kuvaamaa toiminnallisuutta

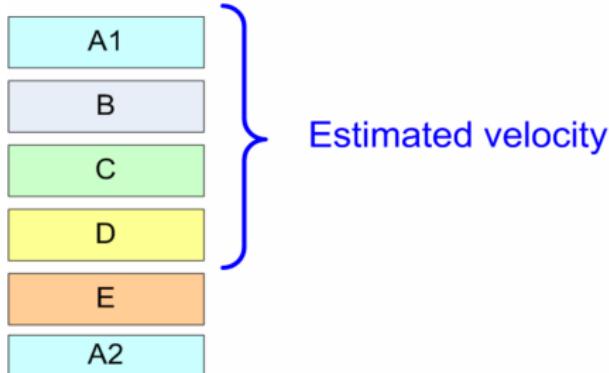
### Option 2



- ▶ 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 myöhempisiin sprintteihin

## User storyjen jakaminen useampaan osaan

- ▶ Haastava aihe, palataan siihen tänään jos aikaa jää
- ▶ Luentomateriaalissa 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 valmiiksi

## 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 storyn aiheuttamat muutokset olemassa olevaan osaan sovelluksesta

## Miten sprintin tavoitteeseen päästään?

- ▶ Sprintin suunnittelun yhteydessä sprinttiin valituille user storylle 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 storyn aiheuttamat muutokset olemassa olevaan osaan sovelluksesta
- ▶ Kaikkia storyyn liittyviä taskeja ei sprintin suunnittelun aikana löydetä
- ▶ Uusia taskeja generoidaan tarvittaessa sprintin edetessä

## Storyn jako taskeihin, esimerkki

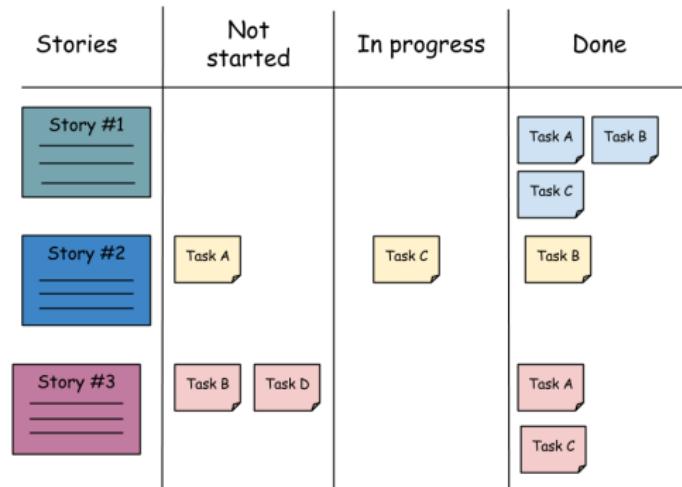
- ▶ Esimerkiksi *tuotteen lisääminen ostoskoriiin*, voitaisiin jakaa seuraaviin teknisiin taskeihin:
  - ▶ tarvitaan sessio, joka muistaa asiakkaan
  - ▶ sovelluslogiikkaolio ostoskorin ja ostoksen esittämiseen
  - ▶ laajennus tietokantaskeemaan
  - ▶ html-näkymää päivitetävä tarvittavilla painikkeilla
  - ▶ kontrolleri painikkeiden käsittelyyn
  - ▶ yksikkötestit kontrollerille ja domain-olioille
  - ▶ hyväksymätestien automatisointi

## Sprint backlog

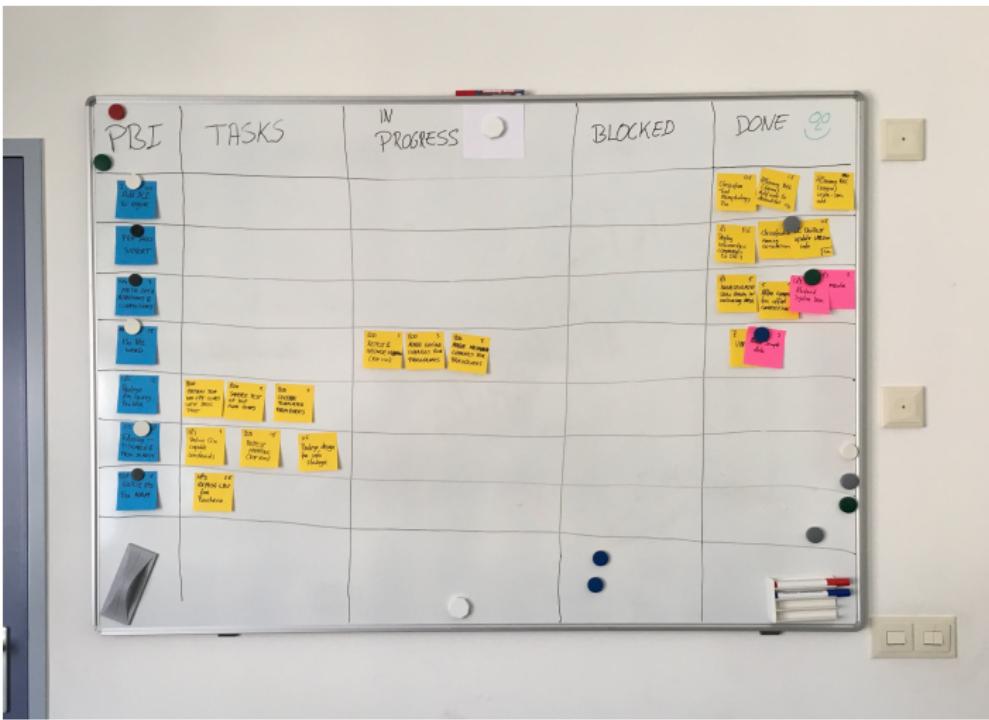
- ▶ *Sprint backlog* koostuu sprinttiin valituista storyista ja niihin liittyvistä tehtävistä eli taskeista

# Sprint backlog

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



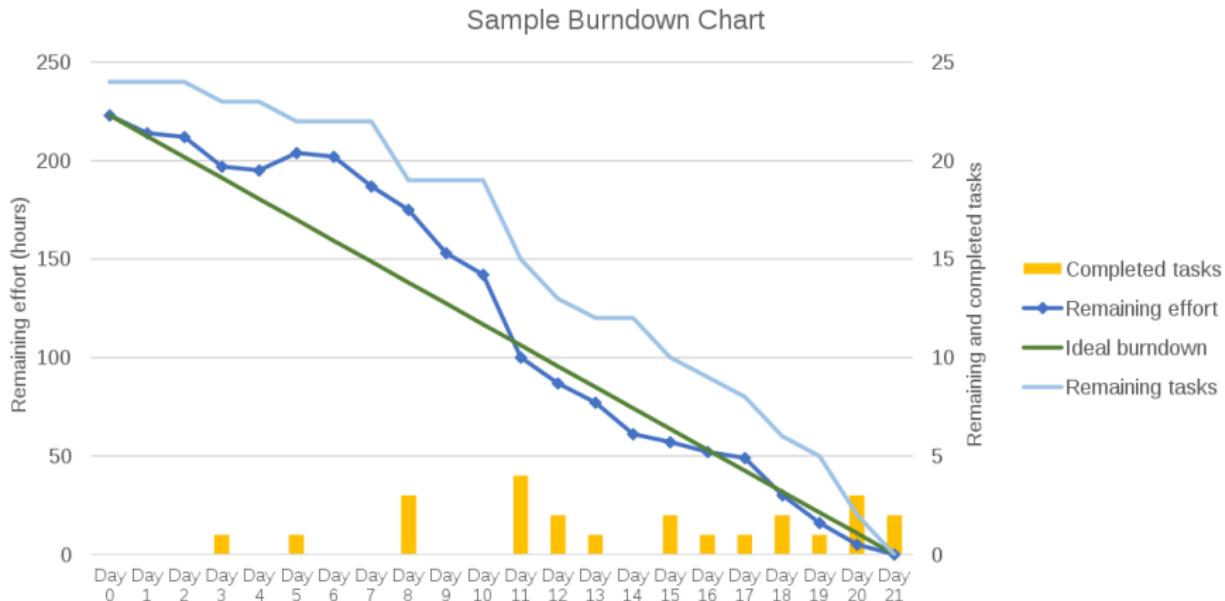
- ▶ Taskit niiden valmistumisastetta kuvaavassa sarakkeessa



► Taskeilla voi olla muitakin statuksia

# Taskien työmäääräarviot ja sprintin burndown

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



## 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 hyväältä, 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 hyväältä, 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

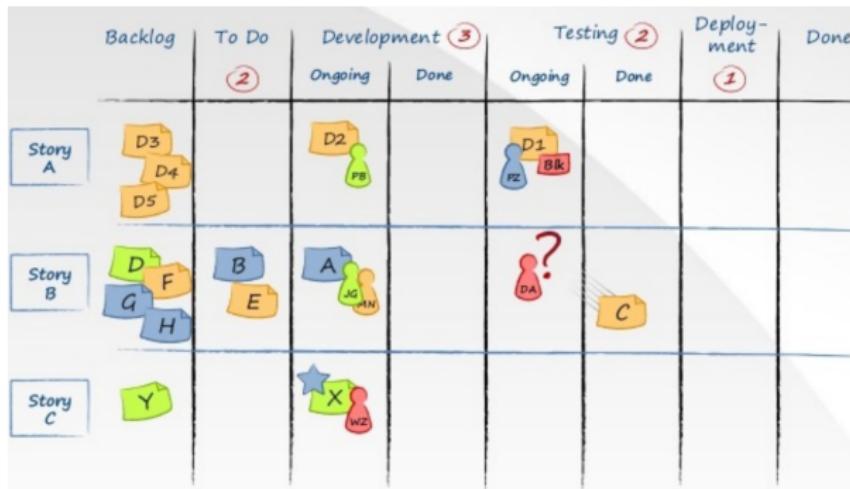
PBI	TASKS	IN PROGRESS	BLOCKED	DONE
1. PBI 2.0 Review				
2. Test and Support				
3. Define Acceptance Criteria				
4. Fix bugs				
5. Refactor code				
6. Update User Stories	BB: Define User Stories BB: Create User Stories BB: Review User Stories	BB: Create User Stories BB: Review User Stories		
7. Create Test Cases	BB: Create Test Cases BB: Review Test Cases BB: Fix bugs	BB: Create Test Cases BB: Review Test Cases BB: Fix bugs		
8. Fix bugs	BB: Fix bugs	BB: Fix bugs		
9. Refactor code	BB: Refactor code	BB: Refactor code		
10. Update User Stories	BB: Update User Stories	BB: Update User Stories		
11. PBI 2.0 Review				

## 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

## Kanban ja Lean

- ▶ WIP-rajoitusten idea on peräisin *Kanban*-menetelmästä, joka on eräs keskeisimmistä *Lean*-ajattelun työkaluista
- ▶ Lean-ajattelun taustalla on idea *hukan* eli asiakkaalle arvoa tuottamattomien asioiden eliminoimisessa
- ▶ 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-ajattelun taustalla on idea *hukan* eli asiakkaalle arvoa tuottamattomien asioiden eliminoimisessa
- ▶ Lean-ajattelu on peräisin jo kymmeniä vuosia vanhasta Toyota Production Systemistä
- ▶ Turhuksia: osittain tehty työ, välivarastointi ja turha odottaminen
  - ▶ Työn alla olevat storyt jotka eivät ole vielä DoD-tasolla valmiina
  - ▶ testaamista odottava toiminnallisuus
  - ▶ testatut mutta tuotantoon viemistä vielä odottavat toiminnallisuudet

# Kanban ja Lean

- ▶ WIP-rajoitusten idea on peräisin *Kanban*-menetelmästä, joka on eräs keskeisimmistä *Lean*-ajattelun työkaluista
- ▶ Lean-ajattelun taustalla on idea *hukan* eli asiakkaalle arvoa tuottamattomien asioiden eliminoimisessa
- ▶ Lean-ajattelu on peräisin jo kymmeniä vuosia vanhasta Toyota Production Systemistä
- ▶ Turhuksia: osittain tehty työ, välivarastointi ja turha odottaminen
  - ▶ Työn alla olevat storyt jotka eivät ole vielä DoD-tasolla valmiina
  - ▶ testaamista odottava toiminnallisuus
  - ▶ testatut mutta tuotantoon viemistä vielä odottavat toiminnallisuudet
- ▶ Toiminnallisuudet tuovat arvoa vasta käytössä, sitä ennen ne sitovat turhaan kustannuksia ja tuovat riskejä

## WIP-rajoitteiden soveltaminen

- ▶ WIP-rajoitteita voidaan soveltaa Scrumin yhteydessä monella tavalla

## WIP-rajoitteiden soveltaminen

- ▶ WIP-rajoitteita voidaan soveltaa Scrumin yhteydessä monella tavalla
- ▶ Järkevintä on rajoittaa sprintin aikana yhtäaikaa työn alla olevien storyjen määrää mahdollisimman pieneksi

## WIP-rajoitteiden soveltaminen

- ▶ WIP-rajoitteita voidaan soveltaa Scrumin yhteydessä monella tavalla
- ▶ Järkevintä on rajoittaa sprintin aikana yhtäaikaa työn alla olevien storyjen määrää mahdollisimman pieneksi
- ▶ On myös tavallista 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
- ▶ Järkevintä on rajoittaa sprintin aikana yhtäaikaa työn alla olevien storyjen määrää mahdollisimman pieneksi
- ▶ On myös tavallista rajoittaa eri työvaiheessa, esim. toteutuksen olevien taskien määrää
- ▶ tai yksittäisellä sovelluskehittäjän kerrallaan työn alla olevien töiden määrää
- ▶ 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: workflow steps

Tarkastellaan sovellusta jonka avulla voi julkaisuta artikkeleja yrityksen web-sivulle. Eräs sovelluksen user storyista on seuraava:

*As a content manager, I can publish a news story to the corporate website.*

Eräs tapa pilkkoa alkuperäinen story onkin jakaa se useampaan osaan eri työvaiheiden mukaan:

- ▶ ... *I can publish a news story directly to the corporate website*
- ▶ ... *I can publish a news story with editor review on a staging site*
- ▶ ... *I can publish a news story with legal review on a staging site*
- ▶ ... *I can view a news story on a staging site*
- ▶ ... *I can publish a news story from the staging site to production*

## Pattern 2: business rule variations

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

kannattaa jakaa siten että jokainen näistä ehdoista eritellään omaksi storykseen

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

## Pattern 3: simple/complex

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

voodaan jakaa seuraavasti

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

## Pattern 4: 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 5: 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 6: 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 7: 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 8: 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*