

Ohjelmistotuotanto

Syksy 2025

Luento 2

28.10.2025

Kurssipalaute

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

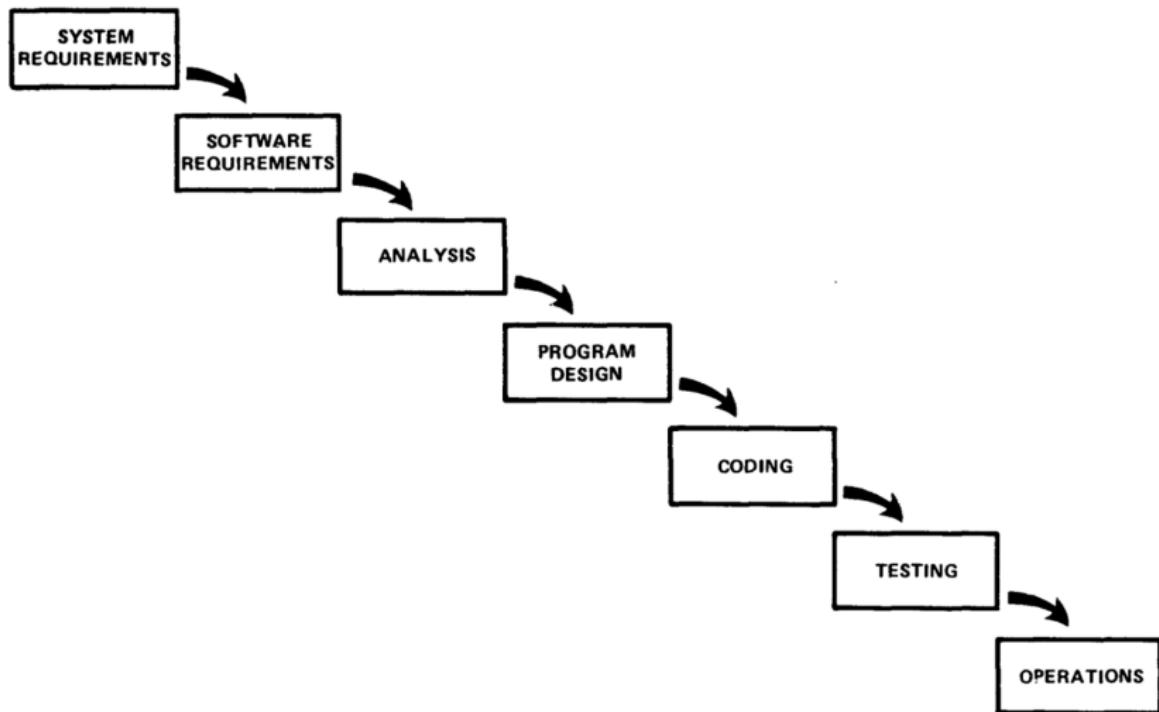
Paja

- ▶ 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 01.12. klo 10-12 B123
 - ▶ to 12.12. klo 10-12 A111

Vesiputousmalli



Vesiputousmallin suurimmat ongelmat

Vesiputousmallin suurimmat ongelmat

- ▶ Vaatimuksset mahdotonta määritellä tyhjentävästi heti alussa
 - ▶ asiakas ei ymmärrä vielä alussa mitä haluaa
 - ▶ bisnesympäristö muuttuu projektin kuluessa

Vesiputousmallin suurimmat ongelmat

- ▶ Vaatimuksset mahdotonta määritellä tyhjentävästi heti alussa
 - ▶ asiakas ei ymmärrä vielä alussa mitä haluaa
 - ▶ bisnesympäristö muuttuu projektin kuluessa
- ▶ Suunnittelu sillä tasolla, että ohjelmointi on suoraviivainen “rakennusvaihe” on mahdotonta
 - ▶ ohjelmointi on osa suunnitteluprosessia, ohjelmakoodi on tuotteen lopullinen suunnitelma

Vesiputousmallin suurimmat ongelmat

- ▶ Vaatimuksset mahdotonta määritellä tyhjentävästi heti alussa
 - ▶ asiakas ei ymmärrä vielä alussa mitä haluaa
 - ▶ bisnesympäristö muuttuu projektin kuluessa
- ▶ Suunnittelu sillä tasolla, että ohjelointi on suoraviivainen “rakennusvaihe” on mahdotonta
 - ▶ ohjelointi on osa suunnitteluprosessia, ohjelmakoodi on tuotteen lopullinen suunnitelma
- ▶ Suunnittelu taas on teknisesti haastavaa, riskejä sisältävää toimintaa

Vesiputousmallin suurimmat ongelmat

- ▶ Vaatimuksset mahdotonta määritellä tyhjentävästi heti alussa
 - ▶ asiakas ei ymmärrä vielä alussa mitä haluaa
 - ▶ bisnesympäristö muuttuu projektin kuluessa
- ▶ Suunnittelu sillä tasolla, että ohjelmointi on suoraviivainen "rakennusvaihe" on mahdotonta
 - ▶ ohjelmointi on osa suunnitteluprosessia, ohjelmakoodi on tuotteen lopullinen suunnitelma
- ▶ Suunnittelu taas on teknisesti haastavaa, riskejä sisältävää toimintaa
- ▶ Lopussa tehtävä testaus paljastaa ongelmat liian myöhään
 - ▶ ongelmien korjaaminen voi edellyttää kalliita muutoksia

Ohjelmiston tuottaminen ei ole kontrolloitu prosessi

- ▶ 90-luvun iteratiiviset prosessimallit korjaavat monia edellisen kalvon epäkohdista
- ▶ Olivat edelleen tarkkoihin etukäteissuunnitelmiin perustuvia
 - ▶ Tarkka projektisuunnitelma ja sen noudattaminen
 - ▶ Selkeä roolijako: projektipäälliköt, analyyttikot, arkkitehdit, ohjelmoijat, testaajat

Ohjelmiston tuottaminen ei ole kontrolloitu prosessi

- ▶ 90-luvun iteratiiviset prosessimallit korjaavat monia edellisen kalvon epäkohdista
- ▶ Olivat edelleen tarkkoihin etukäteissuunnitelmiin perustuvia
 - ▶ Tarkka projektisuunnitelma ja sen noudattaminen
 - ▶ Selkeä roolijako: projektipäälliköt, analyyttikot, arkkitehdit, ohjelmoijat, testaajat
- ▶ eli ne olettivat että ohjelmistotuotanto on jossain määrin *kontrolloitavissa oleva prosessi*

Ketterien menetelmien perusolettamuksia

Ketterien menetelmien perusolettamuksia

- ▶ Useimmat ohjelmistoprojektit ovat laadultaan uniikkeja
 - ▶ **Vaativiset erilaiset** kuin millään jo tehdyllä ohjelmistolla
 - ▶ **Uusi tekijätiimi**, varustettu omanlaisilla kompetensseilla ja persoonallisuksilla
 - ▶ **Toteutusteknologiat kehittyvät** tehdään todennäköisesti tavalla, joka ei ole kaikille tuttu

Ketterien menetelmien perusolettamuksia

- ▶ Useimmat ohjelmistoprojektit ovat laadultaan uniikkeja
 - ▶ **Vaativiset erilaiset** kuin millään jo tehdyllä ohjelmistolla
 - ▶ **Uusi tekijätiimi**, varustettu omanlaisilla kompetensseilla ja persoonallisuksilla
 - ▶ **Toteutusteknologiat kehittyvät** tehdään todennäköisesti tavalla, joka ei ole kaikille tuttu
- ▶ Järkevää lähteä oletuksesta että kyseessä ei ole *kontrolloitu prosessi*, joka voidaan tarkkaan etukäteen aikatauluttaa ja suunnitella

Ketterien menetelmien perusolettamuksia

- ▶ Useimmat ohjelmistoprojektit ovat laadultaan uniikkeja
 - ▶ **Vaativukset erilaiset** kuin millään jo tehdyllä ohjelmistolla
 - ▶ **Uusi tekijätiimi**, varustettu omanlaisilla kompetensseilla ja persoonallisuuksilla
 - ▶ **Toteutusteknologiat kehittyvät** tehdään todennäköisesti tavalla, joka ei ole kaikille tuttu
- ▶ Järkevä lähteä oletuksesta että kyseessä ei ole *kontrolloitu prosessi*, joka voidaan tarkkaan etukäteen aikatauluttaa ja suunnitella
- ▶ Parempi ajatella *tuotekehitysprojektina*, jonka kontrollointiin sopii paremmin *empiriinen prosessi* jonka periaatteina
 - ▶ *transparency* läpinäkyvyys
 - ▶ *inspection* tarkkailu
 - ▶ *adaption* mukauttaminen

Ketterien menetelmien perusolettamuksia

- ▶ Tekijät yksilöitää: toimivat paremmin kun heihin luotetaan ja annetaan tiimille vapaus organisoida itse toimintansa

Ketterien menetelmien perusolettamuksia

- ▶ Tekijät yksilöitää: toimivat paremmin kun heihin luotetaan ja annetaan tiimille vapaus organisoida itse toimintansa
- ▶ Oletuksena että perinteinen command-and-control ja jako eri vastuualueisiin ei tuota optimaalista tulosta

Ketterien menetelmien perusolettamuksia

- ▶ Tekijät yksilöitää: toimivat paremmin kun heihin luotetaan ja annetaan tiimille vapaus organisoida itse toimintansa
- ▶ Oletuksena että perinteinen command-and-control ja jako eri vastuualueisiin ei tuota optimaalista tulosta
- ▶ “The whole team”-periaate: tiimi kollektiivina vastuussa aikaansaannoksesta

Ketterien menetelmien perusolettamuksia

- ▶ Tekijät yksilöitää: toimivat paremmin kun heihin luotetaan ja annetaan tiimille vapaus organisoida itse toimintansa
- ▶ Oletuksena että perinteinen command-and-control ja jako eri vastuualueisiin ei tuota optimaalista tulosta
- ▶ “The whole team”-periaate: tiimi kollektiivina vastuussa aikaansaannoksesta

Eilisellä luennolla käsitelty ketterän *manifesti* heijastelee näitä olettamuksia

Ketterien menetelmien perusolettamuksia

- ▶ Tekijät yksilöitää: toimivat paremmin kun heihin luotetaan ja annetaan tiimille vapaus organisoida itse toimintansa
- ▶ Oletuksena että perinteinen command-and-control ja jako eri vastuualueisiin ei tuota optimaalista tulosta
- ▶ “The whole team”-periaate: tiimi kollektiivina vastuussa aikaansaannoksesta

Eilisellä luennolla käsitelty ketterän *manifesti* heijastelee näitä olettamuksia

Ovatko nämä valideja olettamuksia?

Scrum

- ▶ Tutustumme kurssilla Scrumiin, joka on jo vuosia ollut selvästi suosituin ketterä menetelmä/prosessimalli

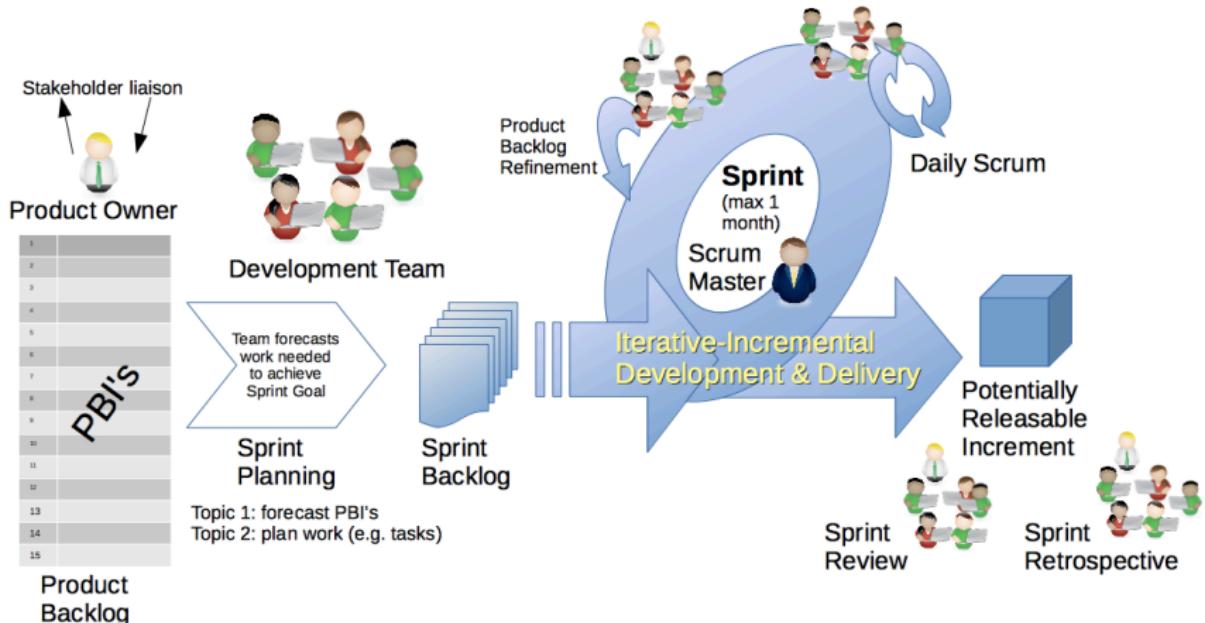
Scrum

- ▶ Tutustumme kurssilla Scrumiin, joka on jo vuosia ollut selvästi suosituin ketterä menetelmä/prosessimalli
- ▶ Kehittäjiensä mukaa Scrum on
 - ▶ menetelmäkehys
 - ▶ monimutkaisten ongelmien ratkaisuun
 - ▶ tuottavalla, luovalla ja maksimaalisen arvoa tuottavalla tavalla

Scrum

- ▶ Tutustumme kurssilla Scrumiin, joka on jo vuosia ollut selvästi suosituin ketterä menetelmä/prosessimalli
- ▶ Kehittäjiensä mukaa Scrum on
 - ▶ menetelmäkehys
 - ▶ monimutkaisten ongelmien ratkaisuun
 - ▶ tuottavalla, luovalla ja maksimaalisen arvoa tuottavalla tavalla
- ▶ Scrum on:
 - ▶ kevyt (lightweight)
 - ▶ helppo ymmärtää
 - ▶ mutta **äärimmäisen vaikea hallita** (extremely difficult to master)

Scrum kuvana



Scrum: roles, artifacts and events

Terminologiaa

- ▶ Scrum määrittelee 3 erilaista **roolia**:
 - ▶ Kehittäjä
 - ▶ Scrum master
 - ▶ Product owner

Scrum: roles, artifacts and events

Terminologiaa

- ▶ Scrum määrittelee 3 erilaista **roolia**:
 - ▶ Kehittäjä
 - ▶ Scrum master
 - ▶ Product owner
- ▶ Scrumiin kuuluvat **artefaktit** eli "konkreettiset asiat" ovat
 - ▶ Product backlog eli projektin kehitysjono
 - ▶ Sprint backlog eli sprintin tehtävälista
 - ▶ Työn alla olevan ohjelmiston uudet versiot (product increment)

Scrum: roles, artifacts and events

Terminologiaa

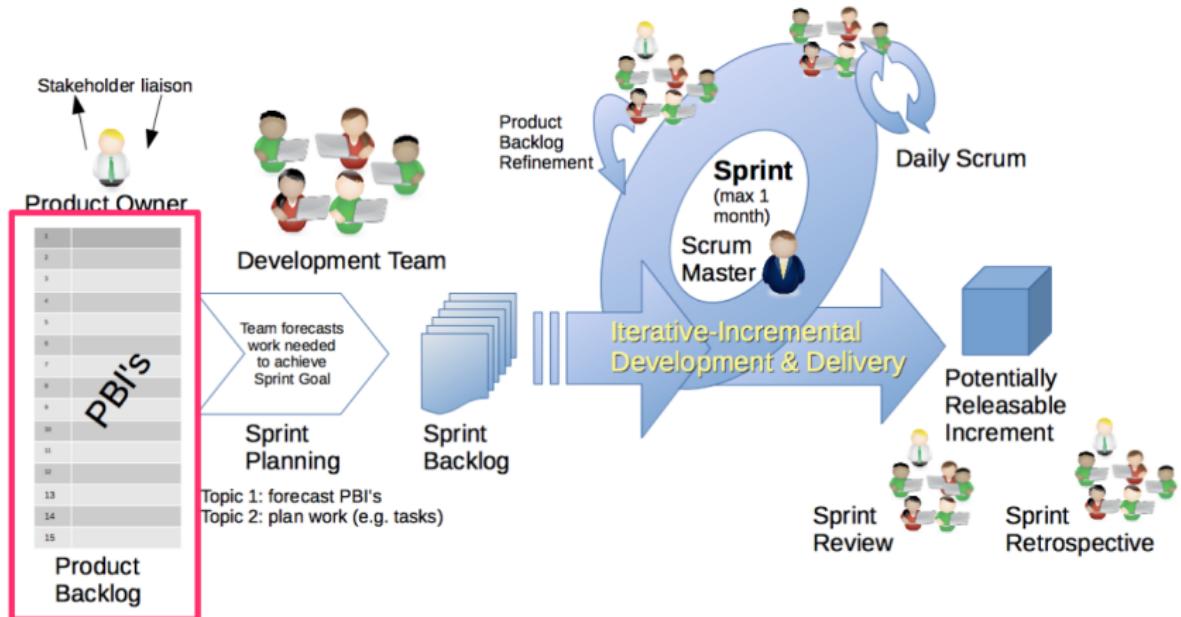
- ▶ Scrum määrittelee 3 erilaista **roolia**:
 - ▶ Kehittäjä
 - ▶ Scrum master
 - ▶ Product owner
- ▶ Scrumiin kuuluvat **artefaktit** eli "konkreettiset asiat" ovat
 - ▶ Product backlog eli projektin kehitysjono
 - ▶ Sprint backlog eli sprintin tehtävälista
 - ▶ Työn alla olevan ohjelmiston uudet versiot (product increment)
- ▶ Scrumissa tekeminen rytmittyy **sprintteihin** eli 1-4 viikon mittaisiin iteraatioihin

Scrum: roles, artifacts and events

Terminologiaa

- ▶ Scrum määrittelee 3 erilaista **roolia**:
 - ▶ Kehittäjä
 - ▶ Scrum master
 - ▶ Product owner
- ▶ Scrumiin kuuluvat **artefaktit** eli "konkreettiset asiat" ovat
 - ▶ Product backlog eli projektin kehitysjono
 - ▶ Sprint backlog eli sprintin tehtävälista
 - ▶ Työn alla olevan ohjelmiston uudet versiot (product increment)
- ▶ Scrumissa tekeminen rytmittyy **sprintteihin** eli 1-4 viikon mittaisiin iteraatioihin
- ▶ Sprintteihin kuuluu muutamia **standardipalavereja** (events):
 - ▶ Sprintin suunnittelupalaveri
 - ▶ Daily scrum -palaverit
 - ▶ Sprintin katselointi
 - ▶ Retrospektiivi

Product backlog



Product backlog

- ▶ Priorisoitu lista asiakkaan tuotteelle asettamista *vaatimuksista*
 - ▶ asiakkaan tasolla olevia *arvoa tuottavia toiminnallisuksia*, kirjattuna asiakkaan ymmärtämällä kielellä

Product backlog

- ▶ Priorisoitu lista asiakkaan tuotteelle asettamista *vaatimuksista*
 - ▶ asiakkaan tasolla olevia *arvoa tuottavia* toiminnallisuksia, kirjattuna asiakkaan ymmärtämällä kielellä
- ▶ Listan kärkipään vaatimukset valitaan toteutettavaksi seuraaviin sprintteihin
 - ▶ kirjattu tarkemmin kuin backlogin häntäpään vaatimukset

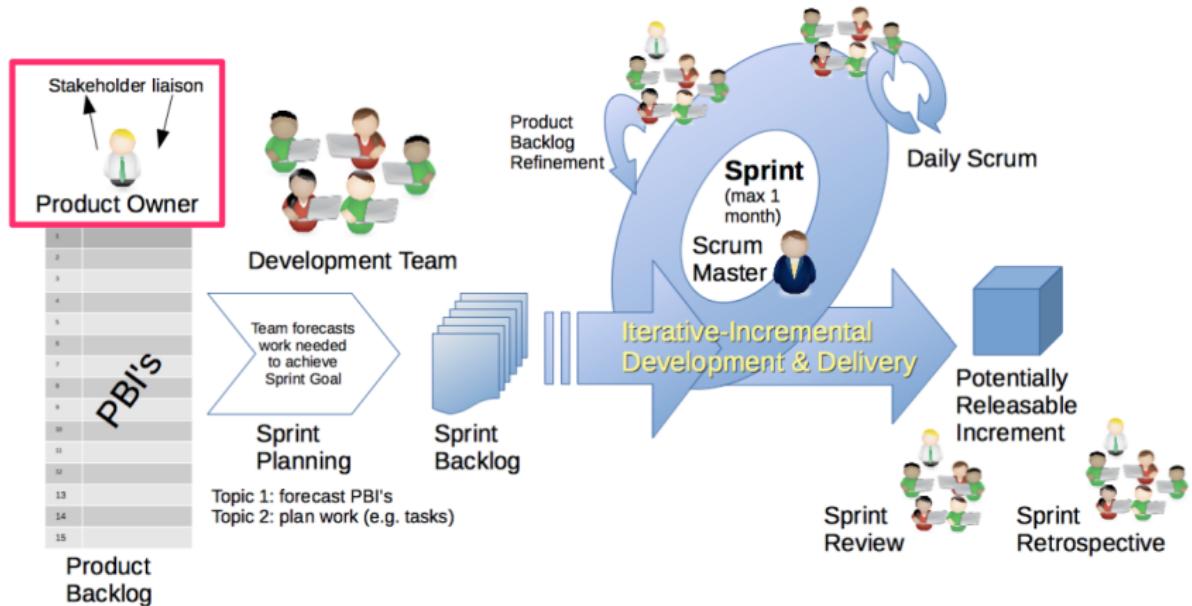
Product backlog

- ▶ Priorisoitu lista asiakkaan tuotteelle asettamista *vaatimuksista*
 - ▶ asiakkaan tasolla olevia *arvoa tuottavia* toiminnallisuksia, kirjattuna asiakkaan ymmärtämällä kielellä
- ▶ Listan kärkipään vaatimukset valitaan toteutettavaksi seuraaviin sprintteihin
 - ▶ kirjattu tarkemmin kuin backlogin häntäpään vaatimukset
- ▶ Vaatimukset ovat usein *estimoituja* eli toteutuksen vaatima työmäärä on arvioitu
 - ▶ Työmääräarviot tekee kehittäjätiimi

Product backlog

- ▶ Priorisoitu lista asiakkaan tuotteelle asettamista *vaatimuksista*
 - ▶ asiakkaan tasolla olevia *arvoa tuottavia* toiminnallisuksia, kirjattuna asiakkaan ymmärtämällä kielellä
- ▶ Listan kärkipään vaatimukset valitaan toteutettavaksi seuraaviin sprintteihin
 - ▶ kirjattu tarkemmin kuin backlogin häntäpään vaatimukset
- ▶ Vaatimukset ovat usein *estimoituja* eli toteutuksen vaatima työmäärä on arvioitu
 - ▶ Työmääräarviot tekee kehittäjätiimi
- ▶ Scrum ei määrittele missä muodossa backlog ja siinä olevat vaatimukset esitetään
 - ▶ nykyään käytetään usein *user story* -formaattia

Product owner



Product owner

- ▶ Baclogista vastaa *product owner* eli tuotteen omistaja
 - ▶ päättää mitä baclogille otetaan
 - ▶ priorisoi backlogin

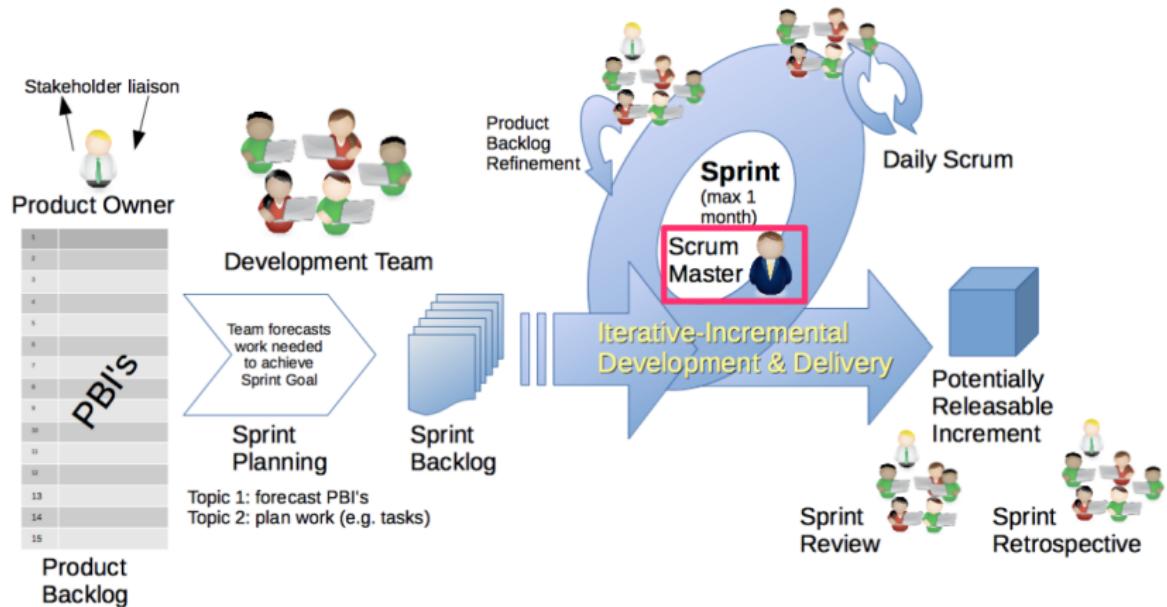
Product owner

- ▶ Baclogista vastaa *product owner* eli tuotteen omistaja
 - ▶ päättää mitä baclogille otetaan
 - ▶ priorisoi backlogin
- ▶ Product owner on yksittäinen henkilö
 - ▶ Priorisointiin voi toki olla vaikuttamassa useampikin henkilö
 - ▶ Product owner tekee lopulliset päätökset prioriteettien suhteen

Product owner

- ▶ Baclogista vastaa *product owner* eli tuotteen omistaja
 - ▶ päättää mitä baclogille otetaan
 - ▶ priorisoi backlogin
- ▶ Product owner on yksittäinen henkilö
 - ▶ Priorisointiin voi toki olla vaikuttamassa useampikin henkilö
 - ▶ Product owner tekee lopulliset päätökset prioriteettien suhteen
- ▶ Product owner on vastuussa backlogista
 - ▶ Varmistaa että kehittäjät iimi ymmärtää toteutettavaksi valitut vaatimukset
 - ▶ Priorisoi vaatimukset maksimoiden asiakkaan tuotteesta saaman hyödyn/arvon

Scrum master



Scrum master

- ▶ Tiimeillä on *scrum master*, eli henkilö joka huolehtii siitä että ohjelmistokehitys etenee sujuvasti

Scrum master

- ▶ Tiimeillä on *scrum master*, eli henkilö joka huolehtii siitä että ohjelmistokehitys etenee sujuvasti
- ▶ Ei perinteinen projektipäällikkö vaan *servant-leader*
 - ▶ järjestää Scrumiin liittyvät palaverit
 - ▶ huolehtii että Scrumia noudatetaan järkeväällä tavalla
 - ▶ opastaa hyvien käytänteiden noudattamisessa
 - ▶ rohkaisee ja auttaa tiimiä itseorganisoitumisessa

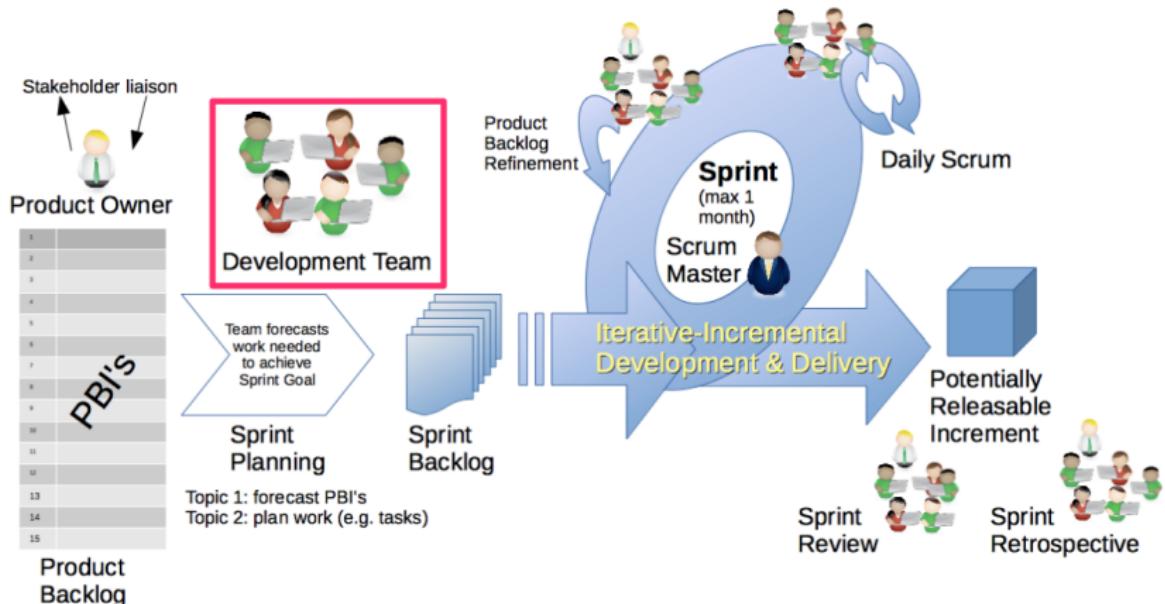
Scrum master

- ▶ Tiimeillä on *scrum master*, eli henkilö joka huolehtii siitä että ohjelmistokehitys etenee sujuvasti
- ▶ Ei perinteinen projektipäällikkö vaan *servant-leader*
 - ▶ järjestää Scrumiin liittyvät palaverit
 - ▶ huolehtii että Scrumia noudatetaan järkevällä tavalla
 - ▶ opastaa hyvien käytänteiden noudattamisessa
 - ▶ rohkaisee ja auttaa tiimiä itseorganisoitumisessa
- ▶ Pyrkii poistamaan kehitystyön **esteitä**
 - ▶ voi olla tiimistä riippumaton asia, jonka poistamiseksi scrum master joutuu neuvottelemaan yrityksen hallinnon kanssa
 - ▶ voi liittyä ryhmän työtapoihin, tällöin scrum master opastaa ryhmää toimimaan siten, että este poistuu

Scrum master

- ▶ Tiimeillä on *scrum master*, eli henkilö joka huolehtii siitä että ohjelmistokehitys etenee sujuvasti
- ▶ Ei perinteinen projektipäällikkö vaan *servant-leader*
 - ▶ järjestää Scrumiin liittyvät palaverit
 - ▶ huolehtii että Scrumia noudatetaan järkevällä tavalla
 - ▶ opastaa hyvien käytänteiden noudattamisessa
 - ▶ rohkaisee ja auttaa tiimiä itseorganisoitumisessa
- ▶ Pyrkii poistamaan kehitystyön **esteitä**
 - ▶ voi olla tiimistä riippumaton asia, jonka poistamiseksi scrum master joutuu neuvottelemaan yrityksen hallinnon kanssa
 - ▶ voi liittyä ryhmän työtapoihin, tällöin scrum master opastaa ryhmää toimimaan siten, että este poistuu
- ▶ Scrum master tekee kaikkensa, jotta tiimillä olisi optimaaliset olosuhteet kehittää tuotetta

Kehittäjätöimi



Kehittäjätiimi

- ▶ Kehittäjätiimi koostuu noin 3-9:stä henkilöstä, kaikista käytetään nimikettä developer
 - ▶ vaikka kaikilla nimike developer, voivat jotkut tiimin jäsenistä olla erikoistuneet omaan osa-alueeseensa

Kehittäjätiimi

- ▶ Kehittäjätiimi koostuu noin 3-9:stä henkilöstä, kaikista käytetään nimikettä developer
 - ▶ vaikka kaikilla nimike developer, voivat jotkut tiimin jäsenistä olla erikoistuneet omaan osa-alueeseensa
- ▶ koko tiimi kantaa aina yhteisen vastuun kehitystyöstä

Kehittäjätiimi

- ▶ Kehittäjätiimi koostuu noin 3-9:stä henkilöstä, kaikista käytetään nimikettä developer
 - ▶ vaikka kaikilla nimike developer, voivat jotkut tiimin jäsenistä olla erikoistuneet omaan osa-alueeseensa
- ▶ koko tiimi kantaa aina yhteisen vastuun kehitystyöstä
- ▶ Oletuksena on että tiimin jäsenet työskentelevät tiimissä 100%:lla työajalla

Kehittäjätiimi

- ▶ Kehittäjätiimi koostuu noin 3-9:stä henkilöstä, kaikista käytetään nimikettä developer
 - ▶ vaikka kaikilla nimike developer, voivat jotkut tiimin jäsenistä olla erikoistuneet omaan osa-alueeseensa
- ▶ koko tiimi kantaa aina yhteisen vastuun kehitystyöstä
- ▶ Oletuksena on että tiimin jäsenet työskentelevät tiimissä 100%:lla työajalla
- ▶ Tiimin tulee oletusarvoisesti työskennellä samassa paikassa, mieluiten yhteisessä tiimille varatuissa avokonttorissa
 - ▶ COVID ja sen jälkeinen hybridityöskentely aiheuttaneet haasteita...

Kehittäjätiiimi

- ▶ Tiiimi on *cross-functional*, eli sen tulisi sisältää kaikki tarvittava osaaminen järjestelmän suunnitteluun, toteuttamiseen ja testaamiseen

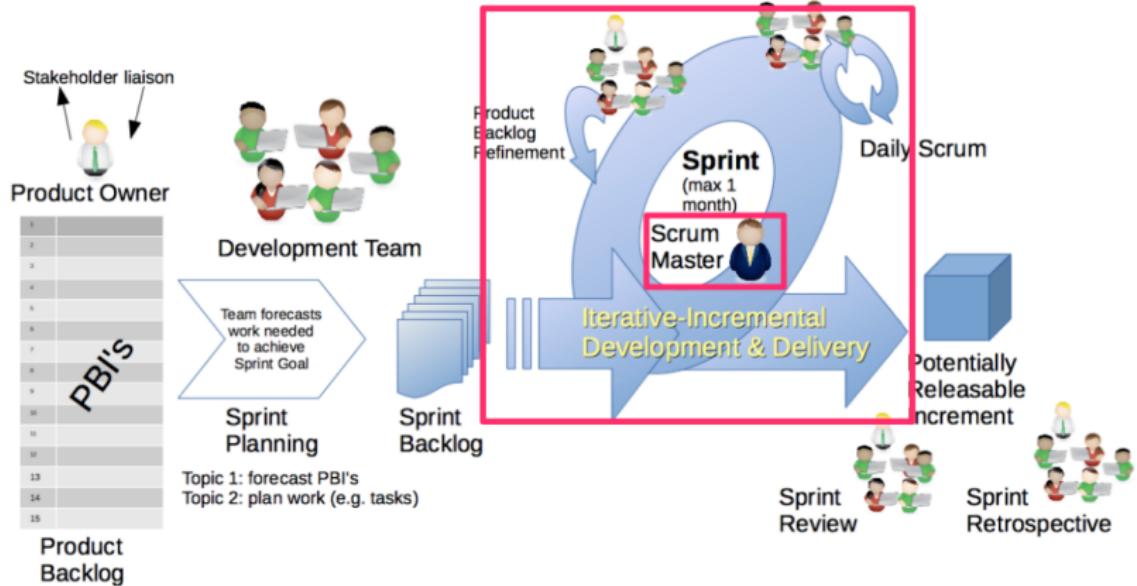
Kehittäjätiimi

- ▶ Tiimi on *cross-functional*, eli sen tulisi sisältää kaikki tarvittava osaaminen järjestelmän suunnittelun, toteuttamiseen ja testaamiseen
- ▶ Kehitystiimiä ei johdeta ulkopuolelta
 - ▶ päättää mihin tavoitteisiin se kussakin sprintissä sitoutuu, eli miten paljon vaatimuksia backlogilta valitaan sprintiin
 - ▶ päättää myös (tiettyjen reunaehojen puitteissa) itse miten se sprintin tavoiteen toteuttaa

Kehittäjätiimi

- ▶ Tiimi on *cross-functional*, eli sen tulisi sisältää kaikki tarvittava osaaminen järjestelmän suunnittelun, toteuttamiseen ja testaamiseen
- ▶ Kehitystiimiä ei johdeta ulkopuolelta
 - ▶ päättää mihin tavoitteisiin se kussakin sprintissä sitoutuu, eli miten paljon vaatimuksia backlogilta valitaan sprintiin
 - ▶ päättää myös (tiettyjen reunaehojen puitteissa) itse miten se sprintin tavoiteen toteuttaa
- ▶ Tiimi on siis *itseorganisoituva* (self organizing)

Sprintti



Sprintti

- ▶ Scrumissa kehitystyö siis jakautuu 1-4 viikon mittaisiin iteraatioihin eli sprintteihin
- ▶ Sprintti on *time-boxed*, sitä ei missään olosuhteissa pidennetä

Sprintti

- ▶ Scrumissa kehitystyö siis jakautuu 1-4 viikon mittaisiin iteraatioihin eli sprintteihin
- ▶ Sprintti on *time-boxed*, sitä ei missään olosuhteissa pidennetä
- ▶ Jokaisen sprintin alussa tiimi valitsee projektin backlogista sprintin aikana toteutettavat vaatimukset
 - ▶ Backlog on priorisoitu ja vaatimukset valitaan aina priorisoidun listan kärjestä
 - ▶ Product ownerin asettama prioriteettijärjestys määräää missä missä *järjestyksessä* asioita toteutetaan

Sprintti

- ▶ Scrumissa kehitystyö siis jakautuu 1-4 viikon mittaisiin iteraatioihin eli sprintteihin
- ▶ Sprintti on *time-boxed*, sitä ei missään olosuhteissa pidennetä
- ▶ Jokaisen sprintin alussa tiimi valitsee projektin backlogista sprintin aikana toteutettavat vaatimukset
 - ▶ Backlog on priorisoitu ja vaatimukset valitaan aina priorisoidun listan kärjestä
 - ▶ Product ownerin asettama prioriteettijärjestys määräää missä missä *järjestyksessä* asioita toteutetaan
- ▶ Tiimi valitsee sprinttiin ainoastaan sen verran toteutettavaa minkä valmistumiseen se uskoo kykenevänsä sitoutumaan

Sprintti

- ▶ Scrumissa kehitystyö siis jakautuu 1-4 viikon mittaisiin iteraatioihin eli sprintteihin
- ▶ Sprintti on *time-boxed*, sitä ei missään olosuhteissa pidennetä
- ▶ Jokaisen sprintin alussa tiimi valitsee projektin backlogista sprintin aikana toteutettavat vaatimukset
 - ▶ Backlog on priorisoitu ja vaatimukset valitaan aina priorisoidun listan kärjestä
 - ▶ Product ownerin asettama prioriteettijärjestys määräää missä missä *järjestyksessä* asioita toteutetaan
- ▶ Tiimi valitsee sprinttiin ainoastaan sen verran toteutettavaa minkä valmistumiseen se uskoo kykenevänsä sitoutumaan
- ▶ Sprintin aikana scrum-tiimi toteuttaa *itseorganisoidusti* sprinttiin valitut ohjelmiston ominaisuudet
- ▶ Sprintin aikana tiimille ei esitetä uusia vaatimuksia

Sprintti

- ▶ Scrumissa kehitystyö siis jakautuu 1-4 viikon mittaisiin iteraatioihin eli sprintteihin
- ▶ Sprintti on *time-boxed*, sitä ei missään olosuhteissa pidennetä
- ▶ Jokaisen sprintin alussa tiimi valitsee projektin backlogista sprintin aikana toteutettavat vaatimukset
 - ▶ Backlog on priorisoitu ja vaatimukset valitaan aina priorisoidun listan kärjestä
 - ▶ Product ownerin asettama prioriteettijärjestys määräää missä missä *järjestyksessä* asioita toteutetaan
- ▶ Tiimi valitsee sprinttiin ainoastaan sen verran toteutettavaa minkä valmistumiseen se uskoo kykenevänsä sitoutumaan
- ▶ Sprintin aikana scrum-tiimi toteuttaa *itseorganisoidusti* sprinttiin valitut ohjelmiston ominaisuudet
- ▶ Sprintin aikana tiimille ei esitetä uusia vaatimuksia
- ▶ Sprintin lopuksi tuotteesta on oltava olemassa *toimiva versio* (potentially shippable product increment)

TAUKO

Definition of done

- ▶ Jokaisessa sprintissä lopputuloksena **toimiva, valmiaksi tehty osa ohjelmistoa**

Definition of done

- ▶ Jokaisessa sprintissä lopputuloksena **toimiva, valmiaksi tehty osa ohjelmistoa**
- ▶ Scrumissa määritellään projektitasolla *definition of done*: mitä tarkoittaa, että jokin vaatimus on toteutettu valmiiksi

Definition of done

- ▶ Jokaisessa sprintissä lopputulosena **toimiva, valmiaksi tehty osa ohjelmistoa**
- ▶ Scrumissa määritellään projektitasolla *definition of done*: mitä tarkoittaa, että jokin vaatimus on toteutettu valmiiksi
- ▶ Määritellään yleensä tarkoittamaan sitä, että vaatimus on
 - ▶ *analysoitu, suunniteltu, ohjelmoitu, testattu, testaus automatisoitu, dokumentoitu, integroitu muuhun ohjelmistoon ja viety tuotantoymäristöön*

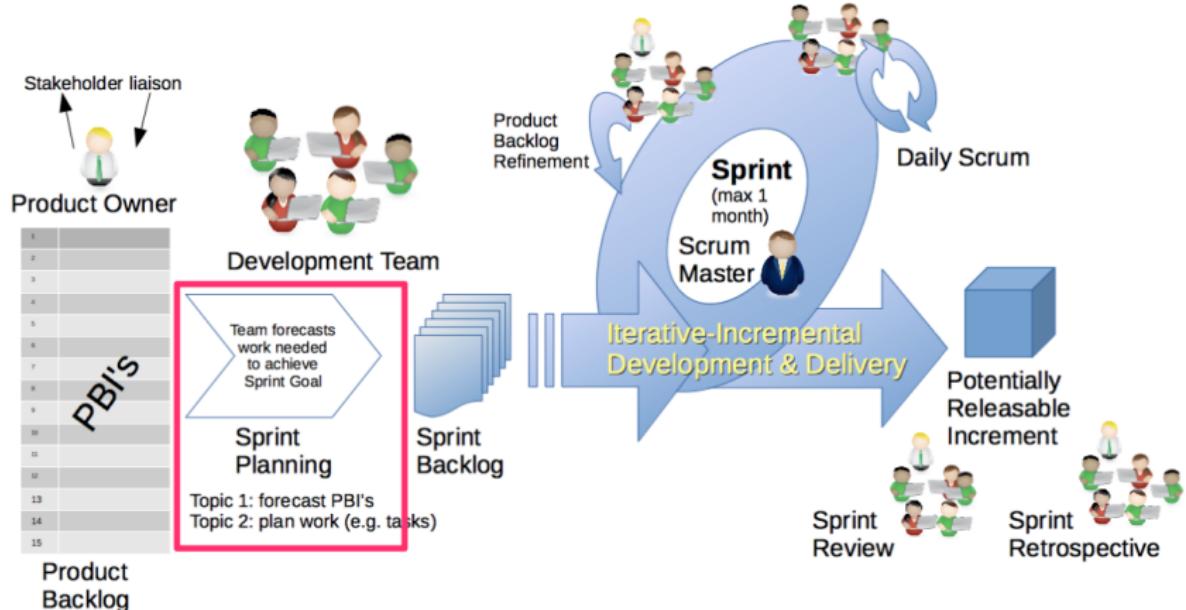
Definition of done

- ▶ Jokaisessa sprintissä lopputuloksena **toimiva, valmiaksi tehty osa ohjelmistoa**
- ▶ Scrumissa määritellään projektitasolla *definition of done*: mitä tarkoittaa, että jokin vaatimus on toteutettu valmiiksi
- ▶ Määritellään yleensä tarkoittamaan sitä, että vaatimus on
 - ▶ *analysoitu, suunniteltu, ohjelmoitu, testattu, testaus automatisoitu, dokumentoitu, integroitu muuhun ohjelmistoon ja viety tuotantoymäristöön*
- ▶ Jos Sprintissä on toteutettu joitain vaatimuksia puutteellisesti DoD:in kannalta, niitä ei tule raportoida valmiina

Definition of done

- ▶ Jokaisessa sprintissä lopputulosena **toimiva, valmiaksi tehty osa ohjelmistoa**
- ▶ Scrumissa määritellään projektitasolla *definition of done*: mitä tarkoittaa, että jokin vaatimus on toteutettu valmiiksi
- ▶ Määritellään yleensä tarkoittamaan sitä, että vaatimus on
 - ▶ *analysoitu, suunniteltu, ohjelmoitu, testattu, testaus automatisoitu, dokumentoitu, integroitu muuhun ohjelmistoon ja viety tuotantoymäristöön*
- ▶ Jos Sprintissä on toteutettu joitain vaatimuksia puutteellisesti DoD:in kannalta, niitä ei tule raportoida valmiina
- ▶ Jos sprintin aikana osoittautuu että tiimi ei ehdi toteuttamaan kaikkia vaatimuksia *laadusta ei tingitää*
 - ▶ osa vaatimuksista jätetään seuraavaan sprinttiin

Sprintin suunnittelu



Sprintin suunnittelu

- ▶ Ennen jokaista sprinttiä järjestetään sprintin *suunnittelukokous*
- ▶ Kokouksella kaksi tavoitetta, Scrumin sanoin *aiheetta*

Sprintin suunnittelu

- ▶ Ennen jokaista sprinttiä järjestetään sprintin *suunnittelukokous*
- ▶ Kokouksella kaksi tavoitetta, Scrumin sanoin *aiheetta*
- ▶ Ensimmäisen aihe on selvittää **mitä** sprintin aikana tehdään
 - ▶ Product owner esittelee product backlogin kärjessä olevat vaatimukset
 - ▶ Tiimin tulee olla selvillä siitä, mitä vaatimuksilla tarkoitetaan
 - ▶ Tiimi arvioi kuinka monta backlogin vaatimuksista se kykenee sprintin aikana toteuttamaan

Sprintin suunnittelu

- ▶ Suunnittelukokouksen toisena aiheena on selvittää **miten** sprintin tavoitteet saavutetaan

Sprintin suunnittelu

- ▶ Suunnittelukokouksen toisena aiheena on selvittää **miten** sprintin tavoitteet saavutetaan
- ▶ Tämä yleensä edellyttää että tiimi suunnittelee toteutettavaksi valitut vaatimukset tarvittavalla tasolla
 - ▶ Aikaansaannoksesta on usein lista teknisistä *tehtävistä* (task), jotka sprintin aikana on toteutettava

Sprintin suunnittelu

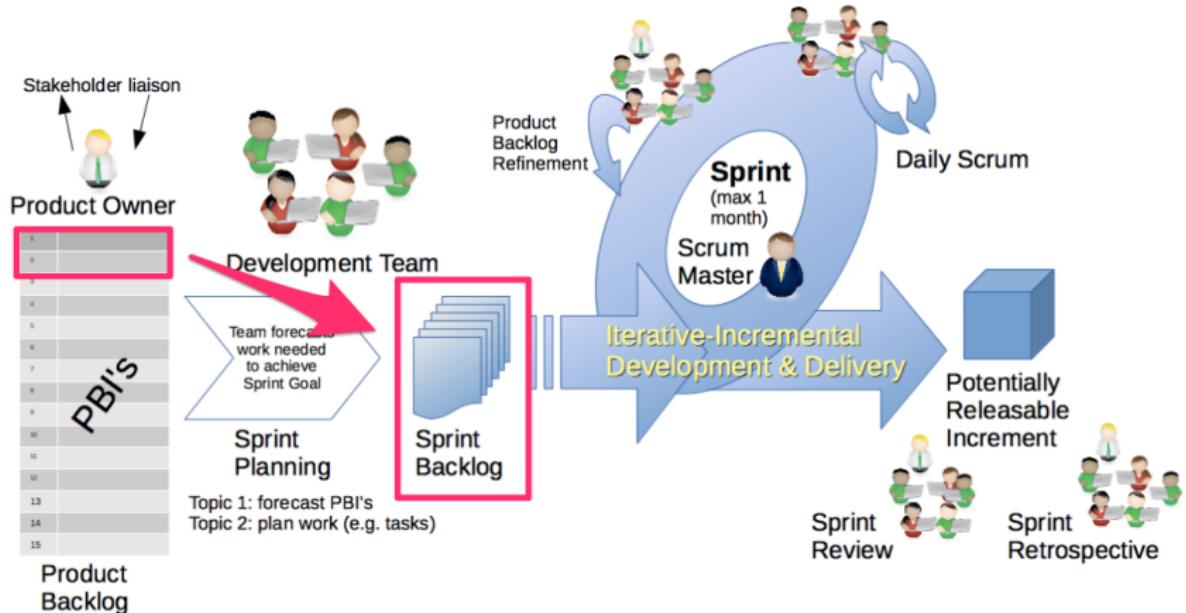
- ▶ Suunnittelukokouksen toisena aiheena on selvittää **miten** sprintin tavoitteet saavutetaan
- ▶ Tämä yleensä edellyttää että tiimi suunnittelee toteutettavaksi valitut vaatimukset tarvittavalla tasolla
 - ▶ Aikaansaannoksesta on usein lista teknisistä *tehtävistä* (task), jotka sprintin aikana on toteutettava
- ▶ Suunnittelun aikana identifioidut tehtävät kirjataan *sprintin backlogiin* eli sprintin tehtävälistaan

Sprintin suunnittelu

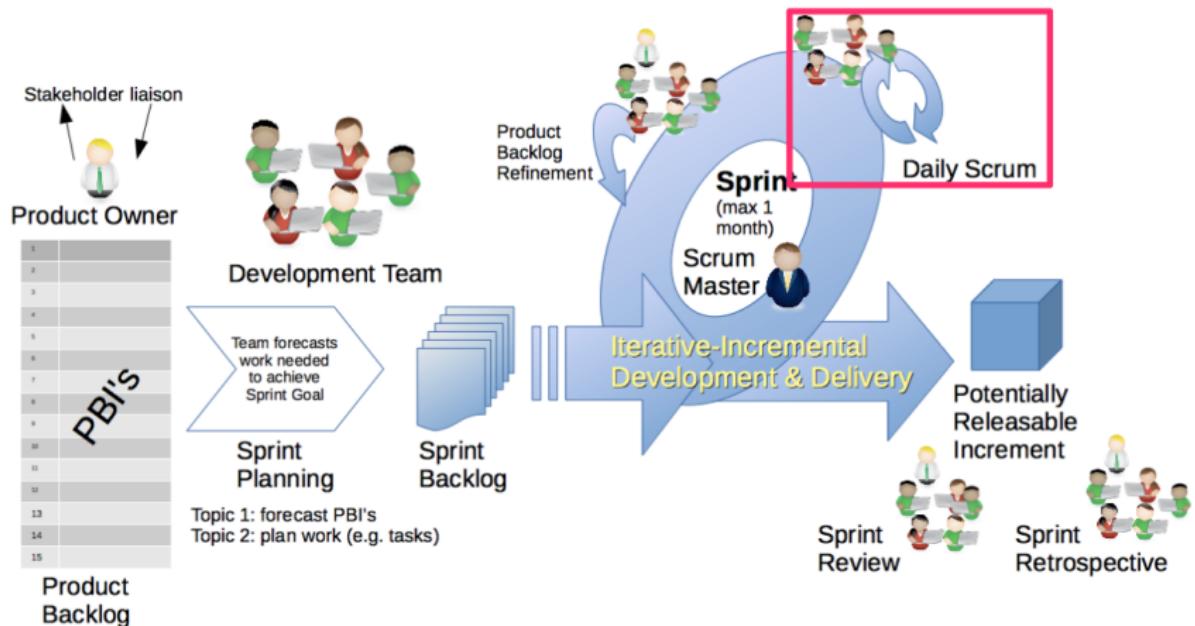
- ▶ Suunnittelukokouksen toisena aiheena on selvittää **miten** sprintin tavoitteet saavutetaan
- ▶ Tämä yleensä edellyttää että tiimi suunnittelee toteutettavaksi valitut vaatimukset tarvittavalla tasolla
 - ▶ Aikaansaannoksesta on usein lista teknisistä *tehtävistä* (task), jotka sprintin aikana on toteutettava
- ▶ Suunnittelun aikana identifioidut tehtävät kirjataan *sprintin backlogiin* eli sprintin tehtävälistaan

Palaamme sprintin suunnitteluun tarkemmin ja konkreettisten esimerkkien kanssa ensi viikolla

Sprintin suunnittelu: *product backlogilta sprint backlogille*



Daily scrum – päiväpalaveri



Daily scrum – päiväpalaveri

- ▶ Jokainen päivä sprintin aikana aloitetaan *daily scrumilla* eli korkeintaan 15 minuutin mittaisella palaverilla
 - ▶ Aina samaan aikaan, samassa paikassa, kaikkien kehittäjien oltava paikalla

Daily scrum – päiväpalaveri

- ▶ Jokainen päivä sprintin aikana aloitetaan *daily scrumilla* eli korkeintaan 15 minuutin mittaisella palaverilla
 - ▶ Aina samaan aikaan, samassa paikassa, kaikkien kehittäjien oltava paikalla
- ▶ Jokainen tiimin jäsen vastaa vuorollaan kolmeen kysymykseen
 - ▶ Mitä sain aikaan edellisen tapaamisen jälkeen?
 - ▶ Mitä aion saada aikaan ennen seuraavaa tapaamista?
 - ▶ Mitä esteitä etenemiselläni on?

Daily scrum – päiväpalaveri

- ▶ Jokainen päivä sprintin aikana aloitetaan *daily scrumilla* eli korkeintaan 15 minuutin mittaisella palaverilla
 - ▶ Aina samaan aikaan, samassa paikassa, kaikkien kehittäjien oltava paikalla
- ▶ Jokainen tiimin jäsen vastaa vuorollaan kolmeen kysymykseen
 - ▶ Mitä sain aikaan edellisen tapaamisen jälkeen?
 - ▶ Mitä aion saada aikaan ennen seuraavaa tapaamista?
 - ▶ Mitä esteitä etenemiselläni on?
- ▶ Kuka tahansa saa olla seuraamassa daily scrumia, mutta vain tiimin jäsenillä on puheoikeus

Daily scrum – päiväpalaveri

- ▶ Jokainen päivä sprintin aikana aloitetaan *daily scrumilla* eli korkeintaan 15 minuutin mittaisella palaverilla
 - ▶ Aina samaan aikaan, samassa paikassa, kaikkien kehittäjien oltava paikalla
- ▶ Jokainen tiimin jäsen vastaa vuorollaan kolmeen kysymykseen
 - ▶ Mitä sain aikaan edellisen tapaamisen jälkeen?
 - ▶ Mitä aion saada aikaan ennen seuraavaa tapaamista?
 - ▶ Mitä esteitä etenemiselläni on?
- ▶ Kuka tahansa saa olla seuraamassa daily scrumia, mutta vain tiimin jäsenillä on puheoikeus
- ▶ Palaverin on tarkoitustulo olla lyhyt, muu keskustelu ei sallittua
- ▶ Jos jollakin on ongelmia, scrum master keskustelee asianomaisen kanssa daily scrumin jälkeen

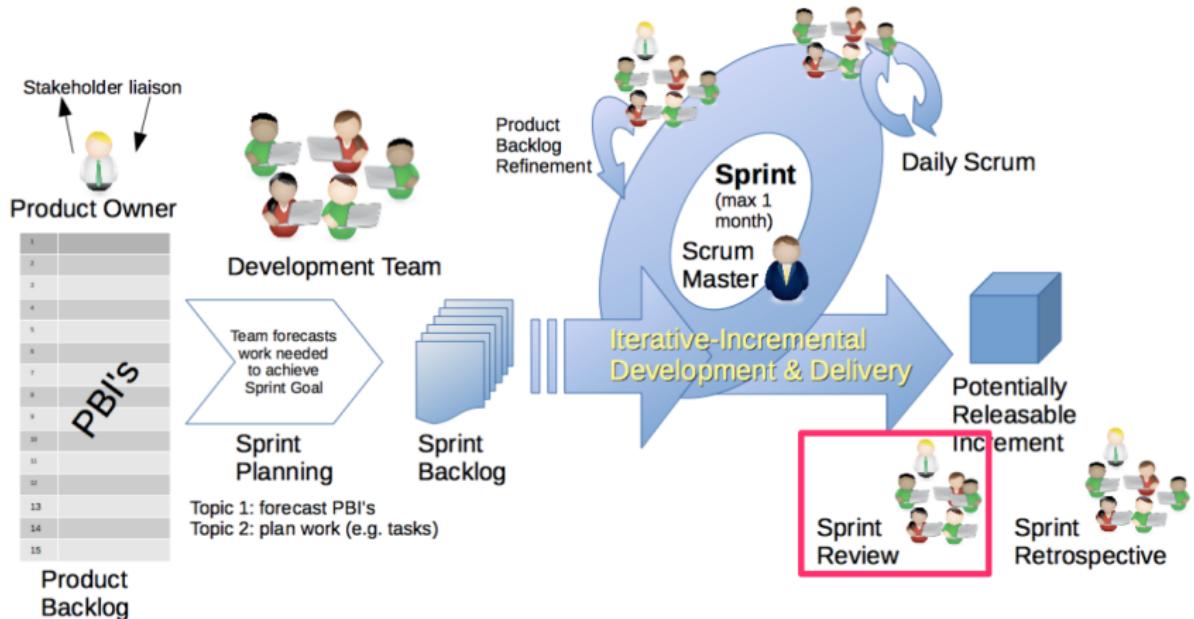
Daily scrum – päiväpalaveri

- ▶ Jokainen päivä sprintin aikana aloitetaan *daily scrumilla* eli korkeintaan 15 minuutin mittaisella palaverilla
 - ▶ Aina samaan aikaan, samassa paikassa, kaikkien kehittäjien oltava paikalla
- ▶ Jokainen tiimin jäsen vastaa vuorollaan kolmeen kysymykseen
 - ▶ Mitä sain aikaan edellisen tapaamisen jälkeen?
 - ▶ Mitä aion saada aikaan ennen seuraavaa tapaamista?
 - ▶ Mitä esteitä etenemiselläni on?
- ▶ Kuka tahansa saa olla seuraamassa daily scrumia, mutta vain tiimin jäsenillä on puheoikeus
- ▶ Palaverin on tarkoitus olla lyhyt, muu keskustelu ei sallittua
- ▶ Jos jollakin on ongelmia, scrum master keskustelee asianomaisen kanssa daily scrumin jälkeen
- ▶ Jos muuhun palaverointiin tarvetta, tulee palaverit järjestää daily scrumista erillään

Daily scrum – päiväpalaveri vuodesta 2020 eteenpäin

- ▶ The purpose of the Daily Scrum is to inspect progress toward the Sprint Goal and adapt the Sprint Backlog as necessary, adjusting the upcoming planned work.
- ▶ The Developers can select whatever structure and techniques they want, as long as their Daily Scrum focuses on progress toward the Sprint Goal and produces an actionable plan for the next day of work.

Sprintin katselmointi



Sprintin katselmointi

- ▶ Sprintin päättäeksi järjestetään *sprint review* eli katselmointi
- ▶ Katselmointiin voi osallistua kuka tahansa

Sprintin katselointi

- ▶ Sprintin päättäeksi järjestetään *sprint review* eli katselointi
- ▶ Katselointiin voi osallistua kuka tahansa
- ▶ Tiimi esittelee sprintin aikaansaannoksia
 - ▶ tarkastellaan/demotaan toteutettua toimivaa ohjelmistoa

Sprintin katselointi

- ▶ Sprintin päättäeksi järjestetään *sprint review* eli katselointi
- ▶ Katselointiin voi osallistua kuka tahansa
- ▶ Tiimi esittelee sprintin aikaansaannoksia
 - ▶ tarkastellaan/demotaan toteutettua toimivaa ohjelmistoa
- ▶ Scrum master huolehtii, että ainostaan definition of doneen mukaisesti toteutetut vaatimukset demotaan

Sprintin katselointi

- ▶ Sprintin päättäeksi järjestetään *sprint review* eli katselointi
- ▶ Katselointiin voi osallistua kuka tahansa
- ▶ Tiimi esittelee sprintin aikaansaannoksia
 - ▶ tarkastellaan/demotaan toteutettua toimivaa ohjelmistoa
- ▶ Scrum master huolehtii, että ainostaan definition of doneen mukaisesti toteutetut vaatimukset demotaan
- ▶ Product owner varmistaa, mitkä vaatimuksista toteutettiin hyväksyttävällä tavalla
- ▶ Ne vaatimukset joita ei hyväksytä toteutetuksi siirretään takaisin product backlogiin

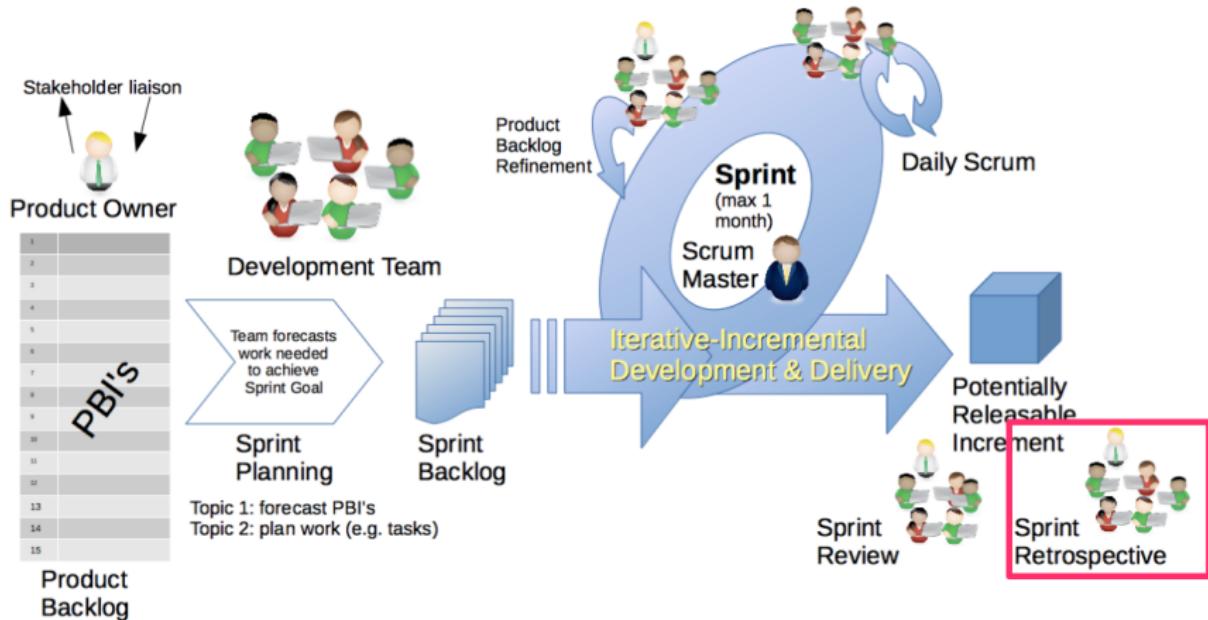
Sprintin katselointi

- ▶ Katselmoinnin aikana kuka tahansa saa antaa palautetta tuotteesta ja esim. ehdottaa uusia vaatimuksia lisättäväksi product backlogiin

Sprintin katselointi

- ▶ Katselmoinnin aikana kuka tahansa saa antaa palautetta tuotteesta ja esim. ehdottaa uusia vaatimuksia lisättäväksi product backlogiin
- ▶ Katselointi aiheuttaa usein myös tarpeen product backlogin uudelleenpriorisoimiseen

Retrospektiivi



Retrospektiivi

- ▶ *Retrospektiivi* on sprintin katselmoinnin ja seuraavan sprintin alun välissä pidettävä palaveri, jonka aikana tiimi tarkastelee omaa työskentelyprosessiaan

Retrospektiivi

- ▶ *Retrospektiivi* on sprintin katselmoinnin ja seuraavan sprintin alun välissä pidettävä palaveri, jonka aikana tiimi tarkastelee omaa työskentelyprosessiaan
- ▶ Identifioidaan mikä meni hyvin ja missä asioissa on parantamisen varaa

Retrospektiivi

- ▶ *Retrospektiivi* on sprintin katselmoinnin ja seuraavan sprintin alun välissä pidettävä palaveri, jonka aikana tiimi tarkastelee omaa työskentelyprosessiaan
- ▶ Identifioidaan mikä meni hyvin ja missä asioissa on parantamisen varaa
- ▶ Mietitään ratkaisuja ongelmakohtiin, joita pyritään korjaamaan seuraavan sprintin aikana

Retrospektiivi



Transparency - inspect - adapt

- ▶ Scrumin taustaperiaatteet ovat
 - ▶ *läpinäkyvyys* (transparency)
 - ▶ *tarkkailu* (inspection)
 - ▶ *mukauttaminen* (adaption)

Transparency - inspect - adapt

- ▶ Scrumin taustaperiaatteet ovat
 - ▶ *läpinäkyvyys* (transparency)
 - ▶ *tarkkailu* (inspection)
 - ▶ *mukauttaminen* (adaption)
- ▶ Asioiden läpinäkyvyys mahdollistaa niiden jatkuvan tarkkailun
- ▶ ja sen seurauksena toimintatapoja ja kehitettävää tuotetta on mahdollista mukauttaa

Transparency - inspect - adapt

- ▶ Scrumin taustaperiaatteet ovat
 - ▶ *läpinäkyvyys* (transparency)
 - ▶ *tarkkailu* (inspection)
 - ▶ *mukauttaminen* (adaption)
- ▶ Asioiden läpinäkyvyys mahdollistaa niiden jatkuvan tarkkailun
- ▶ ja sen seurauksena toimintatapoja ja kehitettävää tuotetta on mahdollista mukauttaa
- ▶ Läpinäkyvyys: backlogit, daily scrum, definition of done, sprintin katselointi, product increment...

Transparency - inspect - adapt

- ▶ Scrumin taustaperiaatteet ovat
 - ▶ *läpinäkyvyys* (transparency)
 - ▶ *tarkkailu* (inspection)
 - ▶ *mukauttaminen* (adaption)
- ▶ Asioiden läpinäkyvyys mahdollistaa niiden jatkuvan tarkkailun
- ▶ ja sen seurauksena toimintatapoja ja kehitettävää tuotetta on mahdollista mukauttaa
- ▶ Läpinäkyvyys: backlogit, daily scrum, definition of done, sprintin katselointi, product increment...
- ▶ Lyhyt kehityssykli mahdollistaa sekä tuotteen että toimintatapojen nopean inkrementaalisen parantamisen
 - ▶ backlogia uudelleenpriorisoidaan ja muokataan palautteen sekä opitun perusteella
 - ▶ retrospektiivi kannustaa tiimiä jatkuvasti parantamaan työprosessiaan

Scrumin arvot

- ▶ Scrum sisältää joukon *arvoja* joiden noudattamista se pitää oleellisena: *commitment, focus, courage, respect*

Scrumin arvot

- ▶ Scrum sisältää joukon *arvoja* joiden noudattamista se pitää oleellisena: *commitment, focus, courage, respect*
- ▶ tiimin tulee olla *sitoutunut* (commitment) yhteisen tavoitteen saavuttamiseksi

Scrumin arvot

- ▶ Scrum sisältää joukon *arvoja* joiden noudattamista se pitää oleellisena: *commitment, focus, courage, respect*
- ▶ tiimin tulee olla *sitoutunut* (commitment) yhteisen tavoitteen saavuttamiseksi
- ▶ ja *fokusoitua* (focus) oikeiden asioiden tekemiseen

Scrumin arvot

- ▶ Scrum sisältää joukon *arvoja* joiden noudattamista se pitää oleellisena: *commitment, focus, courage, respect*
- ▶ tiimin tulee olla *sitoutunut* (commitment) yhteisen tavoitteen saavuttamiseksi
- ▶ ja *fokusoitua* (focus) oikeiden asioiden tekemiseen
- ▶ tulee olla *rohkeutta* (courage) tehdä päätöksiä ja kohdata myös vaikeimpia asioita
 - ▶ tulee olla avoimia sekä onnistumisten että ongelmien suhteen

Scrumin arvot

- ▶ Scrum sisältää joukon *arvoja* joiden noudattamista se pitää oleellisena: *commitment, focus, courage, respect*
- ▶ tiimin tulee olla *sitoutunut* (commitment) yhteisen tavoitteen saavuttamiseksi
- ▶ ja *fokusoitua* (focus) oikeiden asioiden tekemiseen
- ▶ tulee olla *rohkeutta* (courage) tehdä päätöksiä ja kohdata myös vaikeimpia asioita
 - ▶ tulee olla avoimia sekä onnistumisten että ongelmien suhteen
- ▶ oleellista on *kunnioittaa* (respect) koko ajan kaikkia kehitystiimin jäseniä sekä ohjelmiston sidosryhmiä

Scrumin tehokas soveltaminen

- ▶ Jotta Scrum toimisi *tehokkaasti*, tarvitaan sen soveltamiseen sopiva asenne ja orientaatio, eli on noudatettava Scrumin arvoja

Scrumin tehokas soveltaminen

- ▶ Jotta Scrum toimisi *tehokkaasti*, tarvitaan sen soveltamiseen sopiva asenne ja orientaatio, eli on noudatettava Scrumin arvoja
- ▶ Scrumin tekemisen ei ole tarkoitus olla ainoastaan pelisääntöjen orjallista noudattamista

Scrumin tehokas soveltaminen

- ▶ Jotta Scrum toimisi *tehokkaasti*, tarvitaan sen soveltamiseen sopiva asenne ja orientaatio, eli on noudatettava Scrumin arvoja
- ▶ Scrumin tekemisen ei ole tarkoitus olla ainoastaan pelisäännöjen orjallista noudattamista
- ▶ Scrumin inspect-and-adapt (tarkkaile ja mukauta) -luonne ohjaa siihen, **tiimien on koko ajan mukautettava toimintaansa**

Scrumin tehokas soveltaminen

- ▶ Jotta Scrum toimisi *tehokkaasti*, tarvitaan sen soveltamiseen sopiva asenne ja orientaatio, eli on noudatettava Scrumin arvoja
- ▶ Scrumin tekemisen ei ole tarkoitus olla ainoastaan pelisäännöjen orjallista noudattamista
- ▶ Scrumin inspect-and-adapt (tarkkaile ja mukauta) -luonne ohjaa siihen, **tiimien on koko ajan mukautettava toimintaansa**
- ▶ *Tiimien optimaalisen toiminnan kannalta on joskus parempi toimia jopa joidenkin Scrumin ohjeiden vastaisesti*

Scrumin ongelmat

- ▶ Scrum on osoittautunut monin paikoin paremmaksi tavaksi ohjelmistojen tuottamiseen kuin vesiputoousmalli

Scrumin ongelmat

- ▶ Scrum on osoittautunut monin paikoin paremmaksi tavaksi ohjelmistojen tuottamiseen kuin vesiputoousmalli
- ▶ Yleinen ratkaisu ohjelmistotuotannon ongelmiin se ei ole
 - ▶ Scrumin käytön yleistyessä myös epäonnistuneiden Scrum-projektien määrä kasvaa

Scrumin ongelmat

- ▶ Scrum on osoittautunut monin paikoin paremmaksi tavaksi ohjelmistojen tuottamiseen kuin vesiputoousmalli
- ▶ Yleinen ratkaisu ohjelmistotuotannon ongelmien se ei ole
 - ▶ Scrumin käytön yleistyessä myös epäonnistuneiden Scrum-projektien määrä kasvaa
- ▶ Yksi ongelista on ns. **scrumbut**
 - ▶ We use Scrum, **but** having a Daily Scrum every day is too much overhead, so we only have one per week.
 - ▶ We use Scrum, **but** retrospectives are a waste of time, so we don't do them.
 - ▶ We use Scrum, **but** we can't build a piece of functionality in two weeks, so our Sprints are 3 months long

Scrumin ongelmat

- ▶ Scrum on osoittautunut monin paikoin paremmaksi tavaksi ohjelmistojen tuottamiseen kuin vesiputoousmalli
- ▶ Yleinen ratkaisu ohjelmistotuotannon ongelmien se ei ole
 - ▶ Scrumin käytön yleistyessä myös epäonnistuneiden Scrum-projektien määrä kasvaa
- ▶ Yksi ongelmista on ns. **scrumbut**
 - ▶ We use Scrum, **but** having a Daily Scrum every day is too much overhead, so we only have one per week.
 - ▶ We use Scrum, **but** retrospectives are a waste of time, so we don't do them.
 - ▶ We use Scrum, **but** we can't build a piece of functionality in two weeks, so our Sprints are 3 months long
- ▶ Transparency-inspect-adapt voi vaarantua

Scrumin ongelmia Robert Martinin listaamina

Scrumin ongelmia Robert Martinin listaamina

- ▶ No Technical Practices

Scrumin ongelmia Robert Martinin listaamina

- ▶ No Technical Practices
- ▶ Automated Testing

Scrumin ongelmia Robert Martinin listaamina

- ▶ No Technical Practices
- ▶ Automated Testing
- ▶ Certification in CSM

Scrumin ongelmia Robert Martinin listaamina

- ▶ No Technical Practices
- ▶ Automated Testing
- ▶ Certification in CSM
- ▶ Scrum Master sometimes turns into Project Manager

Scrumin ongelmia Robert Martinin listaamina

- ▶ No Technical Practices
- ▶ Automated Testing
- ▶ Certification in CSM
- ▶ Scrum Master sometimes turns into Project Manager
- ▶ Scrum over-emphasizes the role of the team as self-managing

Scrumin ongelmia Robert Martinin listaamina

- ▶ No Technical Practices
- ▶ Automated Testing
- ▶ Certification in CSM
- ▶ Scrum Master sometimes turns into Project Manager
- ▶ Scrum over-emphasizes the role of the team as self-managing
- ▶ Scrum and agile have little to say about how to scale

Scrumin ongelmia Robert Martinin listaamina

- ▶ No Technical Practices
- ▶ Automated Testing
- ▶ Certification in CSM
- ▶ Scrum Master sometimes turns into Project Manager
- ▶ Scrum over-emphasizes the role of the team as self-managing
- ▶ Scrum and agile have little to say about how to scale
- ▶ Insufficient guidance regarding the Product Backlog

Waterscrumfall

- ▶ Yleisesti raportoitu ongelma ketterään ohjelmistokehitykseen siirryttääessä on se, että muu organisaatio jää ennalleen

Waterscrumfall

- ▶ Yleisesti raportoitu ongelma ketterään ohjelmistokehitykseen siirryttääessä on se, että muu organisaatio jää ennalleen
- ▶ Waterscrumfall
 - ▶ ohjelmistokehitys tapahtuu Scrumia mukailien
 - ▶ budjetointi, vaatimusten hallinta sekä tuotantoonvienti etenevät edelleen vanhoja kontrolloituja prosesseja noudattaen

Waterscrumfall

- ▶ Yleisesti raportoitu ongelma ketterään ohjelmistokehitykseen siirryttääessä on se, että muu organisaatio jää ennalleen
- ▶ Waterscrumfall
 - ▶ ohjelmistokehitys tapahtuu Scrumia mukailien
 - ▶ budjetointi, vaatimusten hallinta sekä tuotantoonvienti etenevät edelleen vanhoja kontrolloituja prosesseja noudattaen
- ▶ Päätetään alustava Scrumiin tutustumisemme menetelmän kehittäjien sanoihin:

Scrum is easy to understand but extremely difficult to master