

# Ohjelmistotuotanto

Matti Luukkainen ja ohjaajat Jami Kousa, Tero Tapio, Mauri Karlin

syksy 2019

Luento 10

26.11.2019

# Kurssin kaksi viimeistä viikkoa

- ▶ Miniprojektit jatkuvat
  - ▶ Loppudemot (jokainen ryhmä osallistuu toiseen demoista)
    - ▶ maanantai 9.12. klo 14-17
    - ▶ tiistai 10.12. klo 14-17

# Kurssin kaksi viimeistä viikkoa

- ▶ Miniprojektit jatkuvat
  - ▶ Loppudemot (jokainen ryhmä osallistuu toiseen demoista)
    - ▶ maanantai 9.12. klo 14-17
    - ▶ tiistai 10.12. klo 14-17
- ▶ Vierailuluennot
  - ▶ ma 2.12. klo 12 Juha Viljanen *Smartly*: testaus
  - ▶ ma 3.12. klo 12 Niko Laitinen *Nitor*: UX-suunnittelu
  - ▶ ma 9.12. klo 12 Hannu Kokko *Elisa*: Laajan mittakaavan ketterä sovelluskehitys
  - ▶ ma 10.12. klo 12 Marko Klemetti *Eficode*: DevOps

## Laajan mittakaavan ketterä

- ▶ Ketterät ohjelmistotuotantomenetelmät tarkoitettu alunperin pienien tiimien hallintaan
  - ▶ Scrum: 3-9 kehittäjää
  - ▶ Entä jos on kyseessä tuote, joka edellyttää suurempaa kehittäjämääärää?

## Laajan mittakaavan ketterä

- ▶ Ketterät ohjelmistotuotantomenetelmät tarkoitettu alunperin pienien tiimien hallintaan
  - ▶ Scrum: 3-9 kehittäjää
  - ▶ Entä jos on kyseessä tuote, joka edellyttää suurempaa kehittäjämääärää?
- ▶ Perusperiaatteena pitää tiimit pieninä, mutta kasvattaa tuotantokapasiteettia käyttämällä useampia tiimejä
  - ▶ Tämä edellyttää, että tiimien välistä työtä on koordinoitava

## Laajan mittakaavan ketterä

- ▶ Ketterät ohjelmistotuotantomenetelmät tarkoitettu alunperin pienien tiimien hallintaan
  - ▶ Scrum: 3-9 kehittäjää
  - ▶ Entä jos on kyseessä tuote, joka edellyttää suurempaa kehittäjämääärää?
- ▶ Perusperiaatteena pitää tiimit pieninä, mutta kasvattaa tuotantokapasiteettia käyttämällä useampia tiimejä
  - ▶ Tämä edellyttää, että tiimien välistä työtä on koordinoitava
- ▶ Jo kauan käytetty tapa Scrumin skaalaamiseen on *Scrum of Scrums*
  - ▶ koordinova tiimi, johon kuuluu jäseniä jokaisesta Scrum-tiimistä
  - ▶ esim. scrum master tai lead developer osallistuu

## Laajan mittakaavan ketterä

- ▶ Ketterät ohjelmistotuotantomenetelmät tarkoitettu alunperin pienien tiimien hallintaan
  - ▶ Scrum: 3-9 kehittäjää
  - ▶ Entä jos on kyseessä tuote, joka edellyttää suurempaa kehittäjämääärää?
- ▶ Perusperiaatteena pitää tiimit pieninä, mutta kasvattaa tuotantokapasiteettia käyttämällä useampia tiimejä
  - ▶ Tämä edellyttää, että tiimien välistä työtä on koordinoitava
- ▶ Jo kauan käytetty tapa Scrumin skaalaamiseen on *Scrum of Scrums*
  - ▶ koordinova tiimi, johon kuuluu jäseniä jokaisesta Scrum-tiimistä
  - ▶ esim. scrum master tai lead developer osallistuu
- ▶ Scrum of Scrums -tiimi voi tavata joka päivä tai jos se ei ole tarpeen niin esimerkiksi viikoittain

# Scrum of Scrums

- ▶ Scrum of Scrums -periaate on jo hyvin vanha
  - ▶ Jeff Sutherland kertoo käyttäneensä jo 1996

# Scrum of Scrums

- ▶ Scrum of Scrums -periaate on jo hyvin vanha
  - ▶ Jeff Sutherland kertoo käyttäneensä jo 1996
- ▶ Fimassa satoja sovelluskehittäjiä, kymmeniä Scrum-tiimejä ja useita eri tuotteita
  - ▶ Jokaisen tuotteen tiimejä kordinoi kerran viikossa kokoontuva Scrum of Scrums

# Scrum of Scrums

- ▶ Scrum of Scrums -periaate on jo hyvin vanha
  - ▶ Jeff Sutherland kertoo käyttäneensä jo 1996
- ▶ Fimassa satoja sovelluskehittäjiä, kymmeniä Scrum-tiimejä ja useita eri tuotteita
  - ▶ Jokaisen tuotteen tiimejä kordinoi kerran viikossa kokoontuva Scrum of Scrums
- ▶ Koko tuotejoukkoa hallinnoi kuukausittain kokoontuva "management Scrum"
  - ▶ koostui yrityksen johdosta, tuotepäälliköistä ja johtavista ohjelmistoarkkitehdeistä

# Scrum of Scrums

- ▶ Scrum of Scrums -periaate on jo hyvin vanha
  - ▶ Jeff Sutherland kertoo käytäneensä jo 1996
- ▶ Fimassa satoja sovelluskehittäjiä, kymmeniä Scrum-tiimejä ja useita eri tuotteita
  - ▶ Jokaisen tuotteen tiimejä kordinoi kerran viikossa kokoontuva Scrum of Scrums
- ▶ Koko tuotejoukkoa hallinnoi kuukausittain kokoontuva "management Scrum"
  - ▶ koostui yrityksen johdosta, tuotepäälliköistä ja johtavista ohjelmistoarkkitehdeistä
- ▶ Kuvaus ei ole kovin seikkaperäinen
  - ▶ miten esim. backlogien suhteiden tulisi toimia?

## Laajan mittakaavan ketterä

- ▶ Viimeisen kymmenen vuoden aikana ketterän skaalaamiseen esitellyt useita laajan mittakaavan ketteriä menetelmiä
  - ▶ Scaled Agile Framework eli SAFe
  - ▶ Large Scale Scrum eli LeSS
  - ▶ Disiplined Agile eli DA

## Laajan mittakaavan ketterä

- ▶ Viimeisen kymmenen vuoden aikana ketterän skaalaamiseen esitellyt useita laajan mittakaavan ketteriä menetelmiä
  - ▶ Scaled Agile Framework eli SAFe
  - ▶ Large Scale Scrum eli LeSS
  - ▶ Disiplined Agile eli DA
- ▶ Laajentavat ketteryyttä ottamalla mukaan lean-ajattelua
- ▶ Toisin kuin ketterä, lean on lähtökohtaisesti tarkoitettu toimimaan suuressa skaalassa
  - ▶ sisältääkin enemmän koko organisaation toimintaa ohjaavia periaatteita kuin perinteinen ketterä

# SAFe eli Scaled Agile Framework

- ▶ Pääasiallinen kehittäjä on David Leffingwell joka toimi Nokia Mobile Phonesissa (NMP) konstulttina 2000-luvulla
  - ▶ SAFe on syntynyt pitkälti Nokialla tehdyn työn pohjalta
  - ▶ NMP:lla olikin käytössä eräänlainen esiversio SAFe:sta
- ▶ SAFe:n virallinen ensimmäinen version julkaistiin 2011

# SAFe eli Scaled Agile Framework

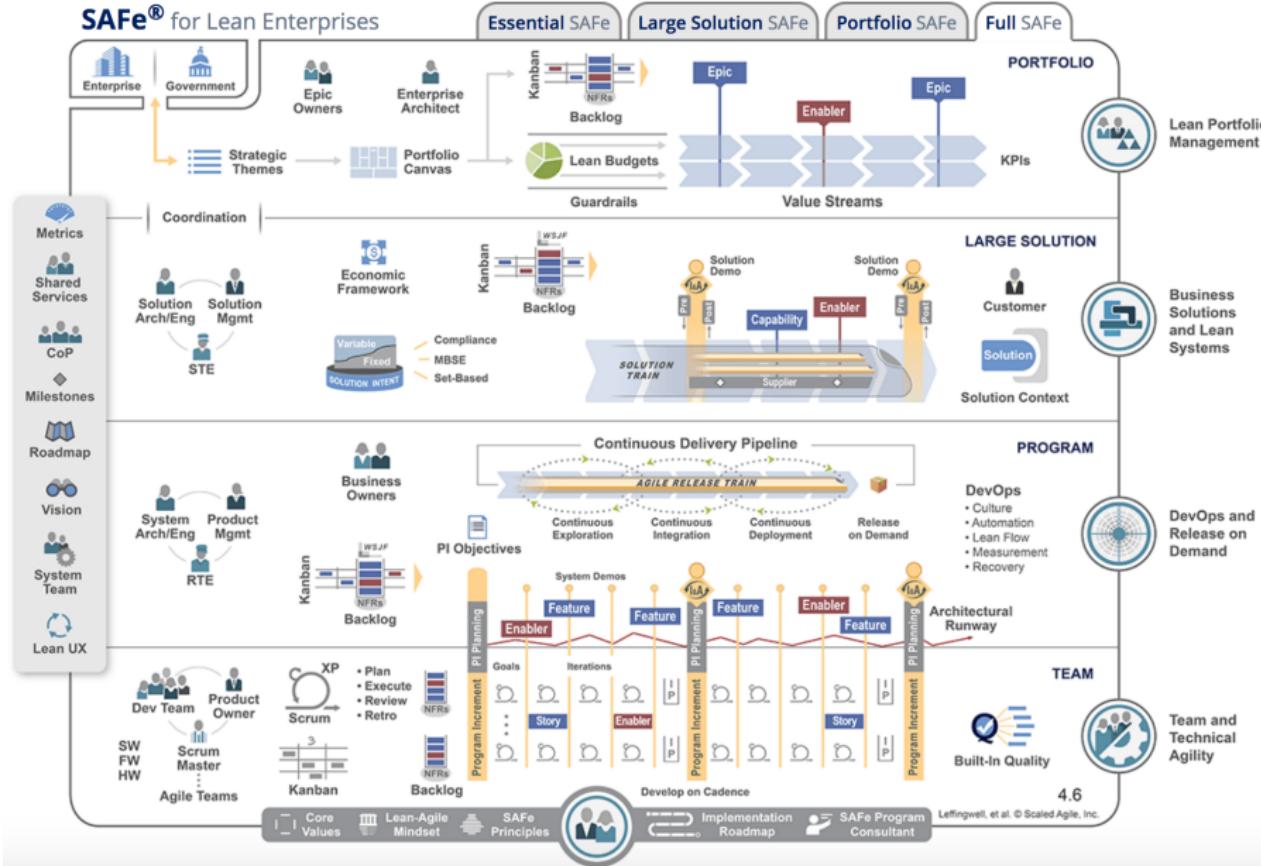
- ▶ Pääasiallinen kehittäjä on David Leffingwell joka toimi Nokia Mobile Phonesissa (NMP) konstulttina 2000-luvulla
  - ▶ SAFe on syntynyt pitkälti Nokialla tehdyн työn pohjalta
  - ▶ NMP:lla olikin käytössä eräänlainen esiversio SAFe:sta
- ▶ SAFe:n virallinen ensimmäinen version julkaistiin 2011
- ▶ yhdistää kaikki viimevuosien ketterän ja leanin parhaat käytänteet sekä joukon tuotteiden hallinnointiperiaatteita

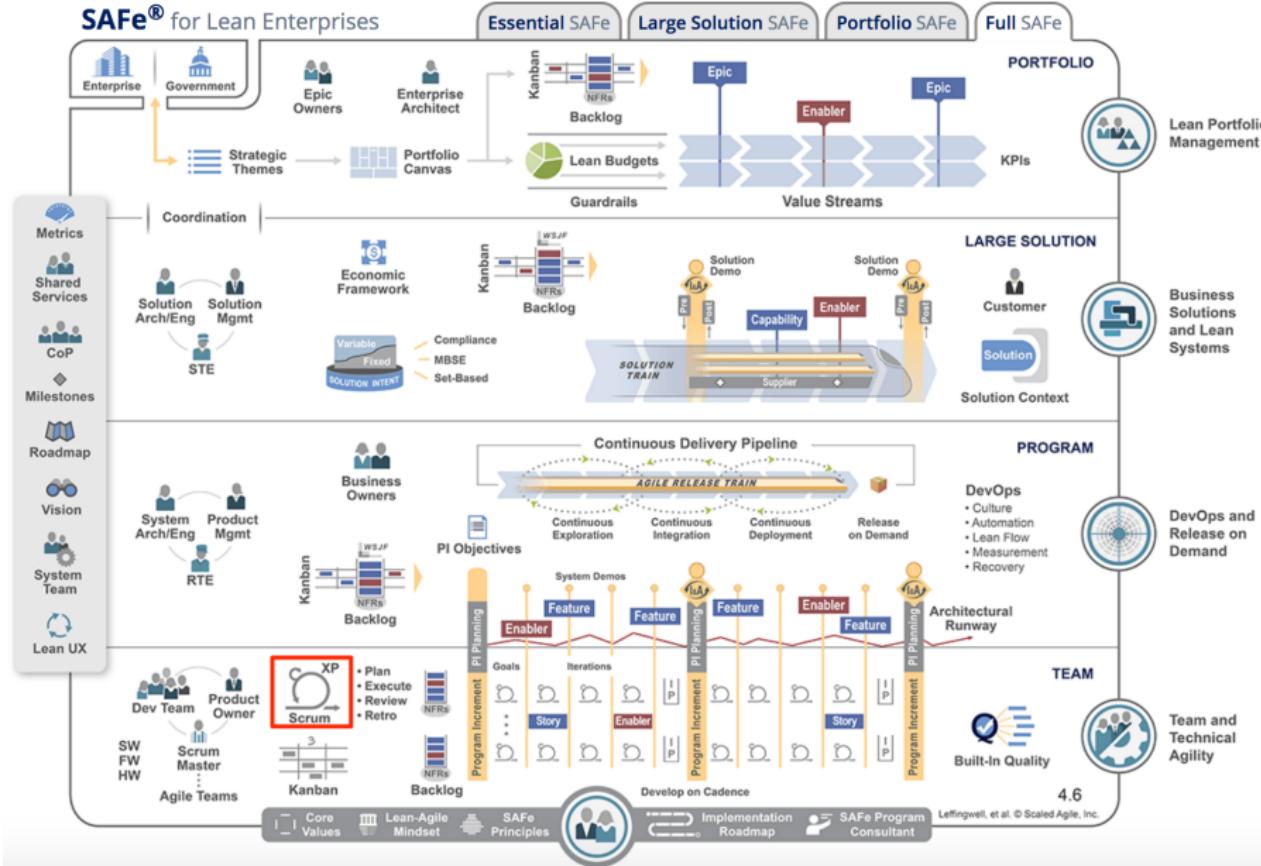
# SAFe eli Scaled Agile Framework

- ▶ Pääasiallinen kehittäjä on David Leffingwell joka toimi Nokia Mobile Phonesissa (NMP) konstulttina 2000-luvulla
  - ▶ SAFe on syntynyt pitkälti Nokialla tehdyн työn pohjalta
  - ▶ NMP:lla olikin käytössä eräänlainen esiversio SAFe:sta
- ▶ SAFe:n virallinen ensimmäinen version julkaistiin 2011
- ▶ yhdistää kaikki viimevuosien ketterän ja leanin parhaat käytänteet sekä joukon tuotteiden hallinnointiperiaatteita
- ▶ SAFe tarjoaa suuren määrään käytänteitä (principles), henkilö- ja tiimirooleja sekä käsitteitä
  - ▶ menetelmäkehys, yritykset räätälöivät itselleen sopivanlaisen prosessin käyttäen SAFe:n tarjoamia työkaluja

# SAFe eli Scaled Agile Framework

- ▶ Pääasiallinen kehittäjä on David Leffingwell joka toimi Nokia Mobile Phonesissa (NMP) konstulttina 2000-luvulla
  - ▶ SAFe on syntynyt pitkälti Nokialla tehdyн työn pohjalta
  - ▶ NMP:lla olikin käytössä eräänlainen esiversio SAFe:sta
- ▶ SAFe:n virallinen ensimmäinen version julkaistiin 2011
- ▶ yhdistää kaikki viimevuosien ketterän ja leanin parhaat käytänteet sekä joukon tuotteiden hallinnointiperiaatteita
- ▶ SAFe tarjoaa suuren määrään käytänteitä (principles), henkilö- ja tiimirooleja sekä käsitteitä
  - ▶ menetelmäkehys, yritykset räätälöivät itselleen sopivanlaisen prosessin käyttäen SAFe:n tarjoamia työkaluja
- ▶ Tarojaan 4 erikokoista valmiaksi räätälöityä konfiguraatiota
  - ▶ Essential SAFe: pienemmille yrityksille ja SAFen soveltamisen alkuvaiheeseen
  - ▶ Full SAFe: massiivisten, useita eri tuotteita hallitsevan yrityksen käyttöön





- ▶ Sovelluskehityksen ytimessä modifioitu Scrum

- ▶ Sovelluskehityksen ytimessä modifioitu Scrum
- ▶ Tiimien kordinointia hallitaan kokoamalla yhdestä tuotteesta vastaavien tiimien joukko käsitteen *release train* alle
  - ▶ tuottavat yhdessä isompia kokonaisuuksia useammasta sprintistä koostuvan *product increment* -ajanjakson aikana

- ▶ Sovelluskehityksen ytimessä modifioitu Scrum
- ▶ Tiimien kordinointia hallitaan kokoamalla yhdestä tuotteesta vastaavien tiimien joukko käsitteen *release train* alle
  - ▶ tuottavat yhdessä isompia kokonaisuuksia useammasta sprintistä koostuvan *product increment* -ajanjakson aikana
- ▶ Product incrementtejä ja niitä toteuttavia release traineja taas ohjaillaan yhä korkeammalta organisaatiosta erilaisten johtajien toimesta
- ▶ SAFe tarjoaa tähän paljon tukea käsitteistön ja roolien kautta

- ▶ Sovelluskehityksen ytimessä modifioitu Scrum
- ▶ Tiimien kordinointia hallitaan kokoamalla yhdestä tuotteesta vastaavien tiimien joukko käsitteen *release train* alle
  - ▶ tuottavat yhdessä isompia kokonaisuuksia useammasta sprintistä koostuvan *product increment* -ajanjakson aikana
- ▶ Product incrementtejä ja niitä toteuttavia release traineja taas ohjaillaan yhä korkeammalta organisaatiosta erilaisten johtajien toimesta
- ▶ SAFe tarjoaa tähän paljon tukea käsitteistön ja roolien kautta
- ▶ SAFe on dokumentoitu laajasti ja se antaa erittäin yksityiskohtaista ohjeistusta helpottamaan SAFen käyttöönottoa ja noudattamista

- ▶ Sovelluskehityksen ytimessä modifioitu Scrum
- ▶ Tiimien kordinointia hallitaan kokoamalla yhdestä tuotteesta vastaavien tiimien joukko käsitteen *release train* alle
  - ▶ tuottavat yhdessä isompia kokonaisuuksia useammasta sprintistä koostuvan *product increment* -ajanjakson aikana
- ▶ Product incrementtejä ja niitä toteuttavia release traineja taas ohjaillaan yhä korkeammalta organisaatiosta erilaisten johtajien toimesta
- ▶ SAFe tarjoaa tähän paljon tukea käsitteistön ja roolien kautta
- ▶ SAFe on dokumentoitu laajasti ja se antaa erittäin yksityiskohtaista ohjeistusta helpottamaan SAFen käyttöönottoa ja noudattamista
- ▶ SAFe vaikuttaa olevan firmojen johdon suosiossa
  - ▶ tarjoaakin firman managementille sopivasti tekemistä roolien ja käytänteiden muodossa

## SAFen kriitikkiä

- ▶ SAFe sisältää käytännössä kaikki mahdolliset ketterän ja lean-ohjelmistokehityksen parhaat käytänteet
  - ▶ Kaikki vieläpä selkeästi ja yksityiskohtaisesti dokumentoituna

## SAFen kriitikkiä

- ▶ SAFe sisältää käytännössä kaikki mahdolliset ketterän ja lean-ohjelmistokehityksen parhaat käytänteet
  - ▶ Kaikki vieläpä selkeästi ja yksityiskohtaisesti dokumentoituna
- ▶ SAFe on agile/lean-kehityksen supermarket, kaikki on helposti saatavissa valmiina pakattussa mudossa
  - ▶ Pick and mix, avaa paketti ja seuraa ohjetta...

## SAFen kritiikkiä

- ▶ SAFe sisältää käytännössä kaikki mahdolliset ketterän ja lean-ohjelmistokehityksen parhaat käytänteet
  - ▶ Kaikki vieläpä selkeästi ja yksityiskohtaisesti dokumentoituna
- ▶ SAFe on agile/lean-kehityksen supermarket, kaikki on helposti saatavissa valmiina pakattussa mudossa
  - ▶ Pick and mix, avaa paketti ja seuraa ohjetta...
- ▶ SAFe käytetään paljon ja se on erityisen suosittu Suomessa
- ▶ SAFe on saanut osakseen myös paljon kritiikkiä
  - ▶ Osa kritiikistä kohdistuu SAFen määrittelemän prosessin raskauteen
  - ▶ Osa taas SAFe:n top down -management luonteeseen

# SAFen kriitikkiä

- ▶ SAFe sisältää käytännössä kaikki mahdolliset ketterän ja lean-ohjelmistokehityksen parhaat käytänteet
  - ▶ Kaikki vieläpä selkeästi ja yksityiskohtaisesti dokumentoituna
- ▶ SAFe on agile/lean-kehityksen supermarket, kaikki on helposti saatavissa valmiina pakattussa mudossa
  - ▶ Pick and mix, avaa paketti ja seuraa ohjetta...
- ▶ SAFe käytetään paljon ja se on erityisen suosittu Suomessa
- ▶ SAFe on saanut osakseen myös paljon kriitikkiä
  - ▶ Osa kritiikistä kohdistuu SAFen määrittelemän prosessin raskauteen
  - ▶ Osa taas SAFe:n top down -management luonteseen
- ▶ Ken Schwaber on kyseenalaistanut onko SAFe ylipäättäään ketterä menetelmä
  - ▶ *Individuals and Interactions Over Processes and Tools*
  - ▶ SAFe taas prosessina vaikuttaa kovin raskaalta

## LeSS eli Large Scale Scrum

- ▶ LeSS:in taustalla on Craig Larman ja Bas Vodde jotka työskentelivät konsultteina 2000-luvun alussa mm. Nokia Siemens Networksilla

# LeSS eli Large Scale Scrum

- ▶ LeSS:in taustalla on Craig Larman ja Bas Vodde jotka työskentelivät konsultteina 2000-luvun alussa mm. Nokia Siemens Networksilla
- ▶ Toisin kuin SAFe, LeSS on erittäin yksinkertainen, hyvin vahvasti Scrumiin pohjautuva
  - ▶ Uusia rooleja, artifakteja ja palavereja ei ole

# LeSS eli Large Scale Scrum

- ▶ LeSS:in taustalla on Craig Larman ja Bas Vodde jotka työskentelivät konsultteina 2000-luvun alussa mm. Nokia Siemens Networksilla
- ▶ Toisin kuin SAFe, LeSS on erittäin yksinkertainen, hyvin vahvasti Scrumiin pohjautuva
  - ▶ Uusia rooleja, artifakteja ja palavereja ei ole
- ▶ LeSSistä on kaksi eri versiota
  - ▶ *LeSS* tilanteisiin, missä tuotetta tekee 2-8 scrum-tiimiä
  - ▶ *LeSS Huge* tilanteisiin, missä tiimejä tarvitaan suurempi määrä

# LeSS

- ▶ Sekä LeSS että LeSS Huge perustuvat seuraaville oletuksille
  - ▶ Kehitetään yhtä tuotetta, jolla on yksi product owner ja yksi product backlog
- ▶ Kaikilla tiimeillä on samaan aikaan etenevät sprintit
- ▶ Tiimit tekevät sprintin aikana yhdessä tuotteesta yhden uuden version *one shippable product increment*

- ▶ Sekä LeSS että LeSS Huge perustuvat seuraaville oletuksille
  - ▶ Kehitetään yhtä tuotetta, jolla on yksi product owner ja yksi product backlog
- ▶ Kaikilla tiimeillä on samaan aikaan etenevät sprintit
- ▶ Tiimit tekevät sprintin aikana yhdessä tuotteesta yhden uuden version *one shippable product increment*
- ▶ Jos yrityksellä on useita tuotteita, niitä kutakin varten on oma LeSS-toteutuksensa
- ▶ Oma erillinen product backlog ja Product owner kullekin LeSS-toteutukselle LeSS ei ota kantaa siihen miten firma hallinnoi tuoteperheitää

- ▶ LeSS isn't new and improved Scrum
  - ▶ It's not Scrum at the bottom for each team, and something different layered on top

- ▶ LeSS isn't new and improved Scrum
  - ▶ It's not Scrum at the bottom for each team, and something different layered on top
- ▶ About figuring out how to apply the Scrum in a large-scale, as simply as possible
  - ▶ Like truly agile frameworks, LeSS is “barely sufficient methodology”

- ▶ LeSS isn't new and improved Scrum
  - ▶ It's not Scrum at the bottom for each team, and something different layered on top
- ▶ About figuring out how to apply the Scrum in a large-scale, as simply as possible
  - ▶ Like truly agile frameworks, LeSS is “barely sufficient methodology”
- ▶ **applied to many teams** cross-functional, cross-component, full-stack feature teams that do all to create done items and a shippable product

- ▶ LeSS isn't new and improved Scrum
  - ▶ It's not Scrum at the bottom for each team, and something different layered on top
- ▶ About figuring out how to apply the Scrum in a large-scale, as simply as possible
  - ▶ Like truly agile frameworks, LeSS is “barely sufficient methodology”
- ▶ **applied to many teams** cross-functional, cross-component, full-stack feature teams that do all to create done items and a shippable product
- ▶ **working together** The teams have a common goal to deliver one common shippable product, and each team cares about this because they are a feature team responsible for the whole, not a part

- ▶ LeSS isn't new and improved Scrum
  - ▶ It's not Scrum at the bottom for each team, and something different layered on top
- ▶ About figuring out how to apply the Scrum in a large-scale, as simply as possible
  - ▶ Like truly agile frameworks, LeSS is “barely sufficient methodology”
- ▶ **applied to many teams** cross-functional, cross-component, full-stack feature teams that do all to create done items and a shippable product
- ▶ **working together** The teams have a common goal to deliver one common shippable product, and each team cares about this because they are a feature team responsible for the whole, not a part
- ▶ **one product** A broad complete end-to-end customer-centric solution that real customers use
  - ▶ It's not a component, platform, layer, or library

## More with less

- ▶ Periaatteet ovat lähes samat kuin SAFe:ssa, yksi periaatteesta tekee kuitenkin selvää eroa menetelmien välille

## More with less

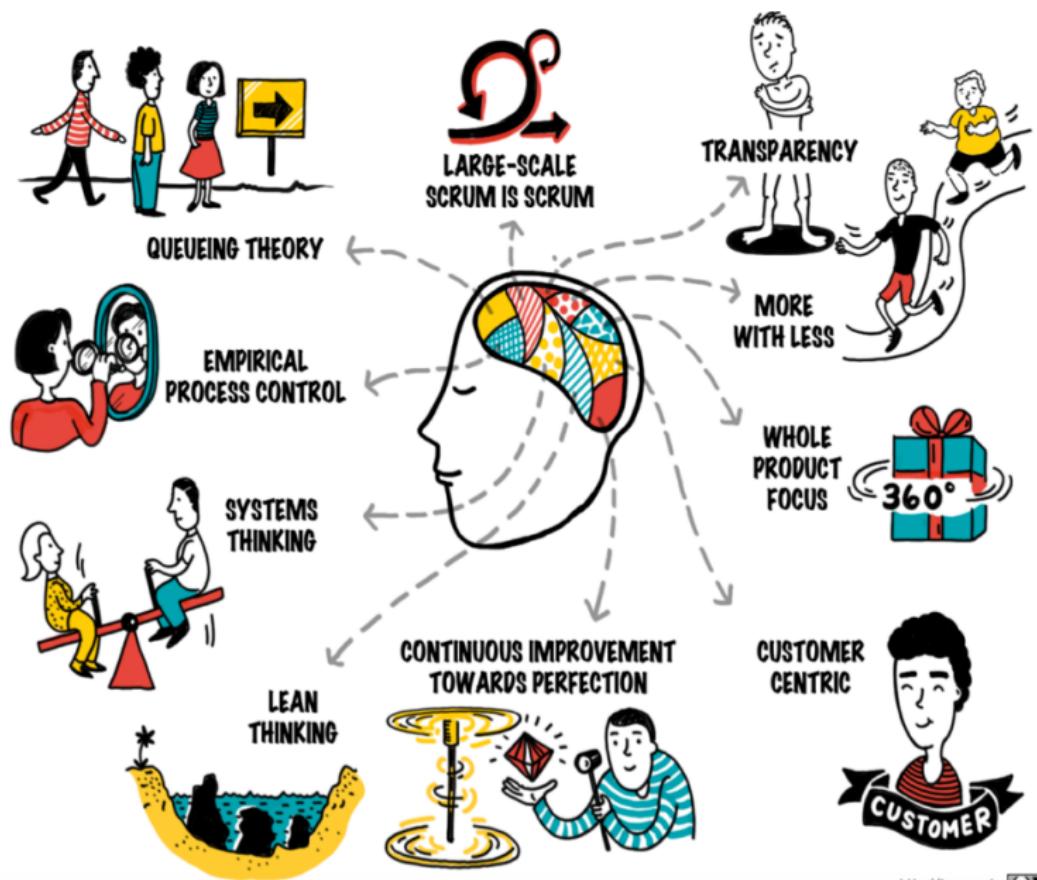
- ▶ Periaatteet ovat lähes samat kuin SAFe:ssa, yksi periaatteesta tekee kuitenkin selvää eroa menetelmien välille
- ▶ We don't want
  - ▶ more roles because more roles leads to less responsibility to Teams
  - ▶ more artifacts because more artifacts leads to a greater distance between Teams and customers
  - ▶ more process because that leads to less learning and team ownership of process

## More with less

- ▶ Periaatteet ovat lähes samat kuin SAFe:ssa, yksi periaatteesta tekee kuitenkin selvää eroa menetelmien välille
- ▶ We don't want
  - ▶ more roles because more roles leads to less responsibility to Teams
  - ▶ more artifacts because more artifacts leads to a greater distance between Teams and customers
  - ▶ more process because that leads to less learning and team ownership of process
- ▶ Instead We want
  - ▶ more **responsible Teams** by having less roles
  - ▶ more **customer-focused Teams** building useful products by having less artifacts
  - ▶ more **Team ownership of process** and more meaningful work by having less defined processes

## More with less

- ▶ Periaatteet ovat lähes samat kuin SAFe:ssa, yksi periaatteesta tekee kuitenkin selvää eroa menetelmien välille
- ▶ We don't want
  - ▶ more roles because more roles leads to less responsibility to Teams
  - ▶ more artifacts because more artifacts leads to a greater distance between Teams and customers
  - ▶ more process because that leads to less learning and team ownership of process
- ▶ Instead We want
  - ▶ more **responsible Teams** by having less roles
  - ▶ more **customer-focused Teams** building useful products by having less artifacts
  - ▶ more **Team ownership of process** and more meaningful work by having less defined processes
- ▶ We want more with less



# LeSS

- ▶ Katsotaan hieman tarkemmin LeSS:in pienempää konfiguraatiota

# LeSS

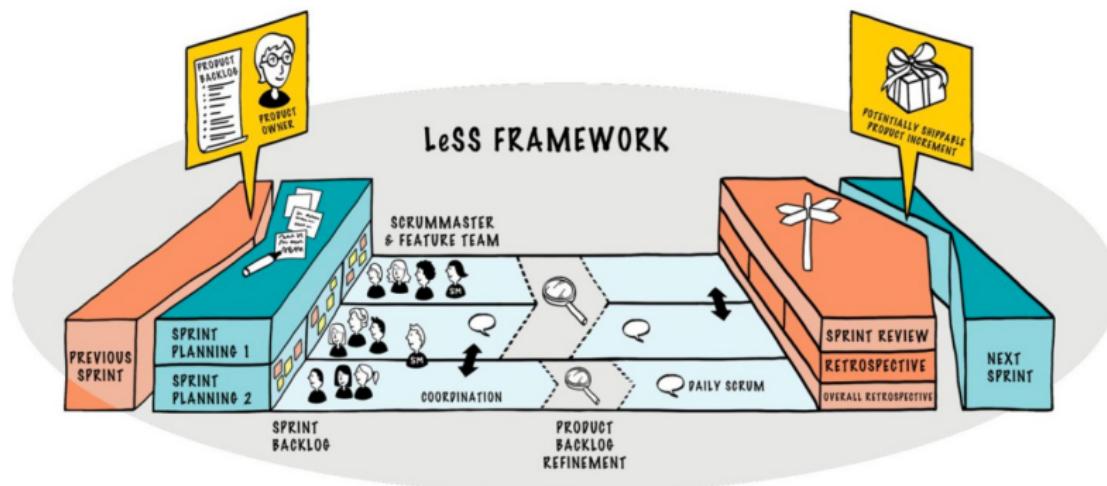
- ▶ Katsotaan hieman tarkemmin LeSS:in pienempää konfiguraatiota
- ▶ Roolit
  - ▶ yksi product owner
  - ▶ 2-8 tiimiä
  - ▶ Kyksi scrum master 1-3 tiimiä kohti

# LeSS

- ▶ Katsotaan hieman tarkemmin LeSS:in pienempää konfiguraatiota
- ▶ Roolit
  - ▶ yksi product owner
  - ▶ 2-8 tiimiä
  - ▶ Kyksi scrum master 1-3 tiimiä kohti
- ▶ cross functional featureiimit
  - ▶ true cross-functional and cross-component full-stack teams that work together in a shared code environment
  - ▶ each doing everything to create done items

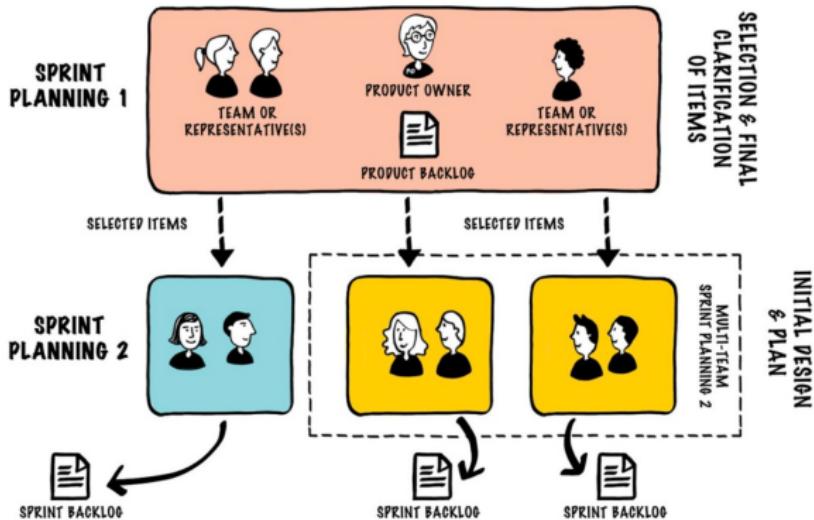
- ▶ Artefaktit
  - ▶ yksi product backlog
  - ▶ yhteinen kehitettävä tuote (potentially shippable product increment)
  - ▶ tiimikohtaiset sprinttibacklogit

- ▶ Artefaktit
  - ▶ yksi product backlog
  - ▶ yhteinen kehitettävä tuote (potentially shippable product increment)
  - ▶ tiimikohtaiset sprinttibacklogit
- ▶ Kaikille yhteinen sprintti

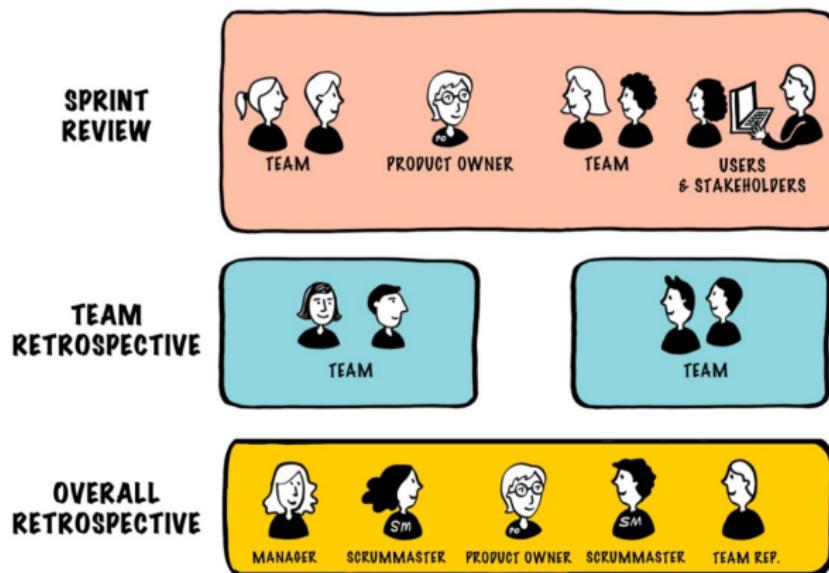


# LeSS: kaksiosainen sprintin suunnittelu

- ▶ Ensimmäisessä osassa product owner ja tiimien edustajat valitsevat backlogilta kaikille eri tiimille seuraavan sprintin storyt
- ▶ Toisessa osassa tiimit muodostavat omat sprint backlogit



- ▶ Kaikkien tiimien yhteinen aikaansaannos (one shippable product increment) katselmoidaan yhdessä
- ▶ Retrospektiivi on kaksitasoinen
  - ▶ tiimikohtainen
  - ▶ overall-retrospektiivi: edustus kaikista tiimeistä ja mahdollisesti yrityksen johdosta



## LeSS: Muu tiimien välinen koordinointi

- ▶ Kaikille tiimeille yhteisen sprintin suunnittelun, sprint reviewin ja overall-retrospektiivin lisäksi ei edellytä muita yhteisiä tapaamisia

## LeSS: Muu tiimien välinen koordinointi

- ▶ Kaikille tiimeille yhteisen sprintin suunnittelun, sprint reviewin ja overall-retrospektiivin lisäksi ei edellytä muita yhteisiä tapaamisia
- ▶ LeSS antaa joukon aiheeseen liittyviä ohjeita ja suosituksia  
Cross-team coordination is decided by the teams
  - ▶ Prefer decentralized and informal coordination over centralized coordination
  - ▶ Emphasize Just Talk and informal networks via communicate in code, cross-team meetings, component mentors, travelers, scouts, and open spaces
  - ▶ Just Talk korostaa suoran kommunikoinnin tärkeyttä
  - ▶ Communicate in code tarkoittaa ryhmien välistä työskentelyä helpottava ohjelointitapaa, mm. yhteisiä koodikäytänteitä ja jatkuvaan integraatiota

## LeSS: Muu tiimien välinen koordinointi

- ▶ Kaikille tiimeille yhteisen sprintin suunnittelun, sprint reviewin ja overall-retrospektiivin lisäksi ei edellytä muita yhteisiä tapaamisia
- ▶ LeSS antaa joukon aiheeseen liittyviä ohjeita ja suosituksia  
Cross-team coordination is decided by the teams
  - ▶ Prefer decentralized and informal coordination over centralized coordination
  - ▶ Emphasize Just Talk and informal networks via communicate in code, cross-team meetings, component mentors, travelers, scouts, and open spaces
  - ▶ Just Talk korostaa suoran kommunikoinnin tärkeyttä
  - ▶ Communicate in code tarkoittaa ryhmien välistä työskentelyä helpottava ohjelointitapaa, mm. yhteisiä koodikäytänteitä ja jatkuvaa integraatiota
- ▶ Scrum of Scrums -palaverit mainitaan, mutta suositellaan informaalimpia kommunikaation muotoja

## LeSS: Backlogin ylläpito

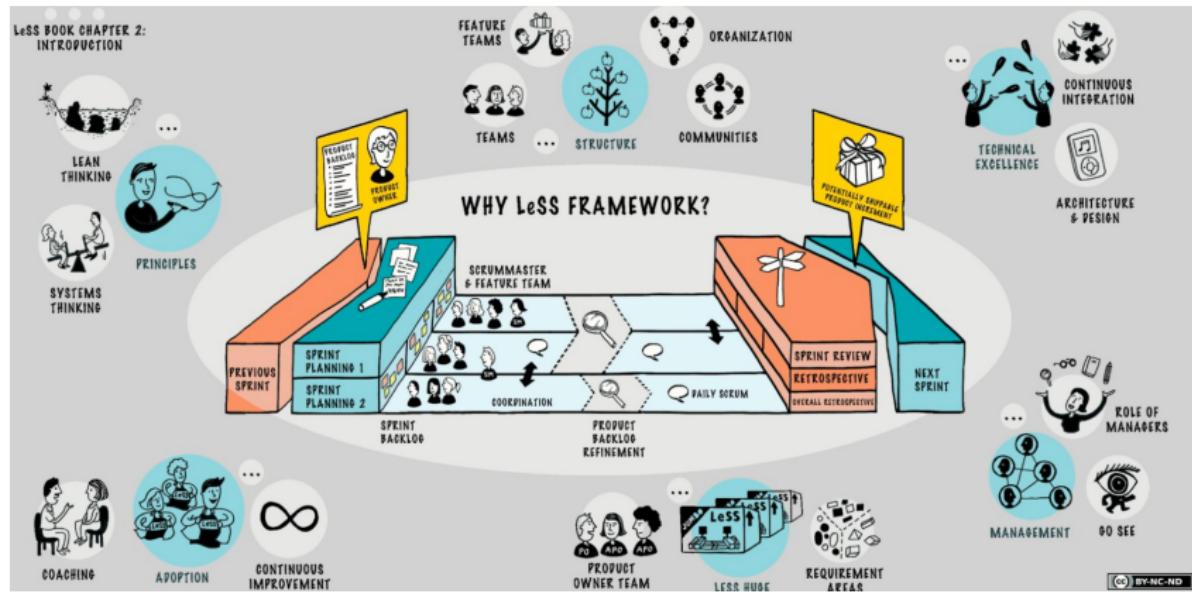
- ▶ LeSS kiinnittää eksplisiittisesti huomioita backlogin ylläpitämiseen
  - ▶ All prioritization goes through the Product Owner
  - ▶ The Product Owner shouldn't work alone on Backlog refinement
  - ▶ Supported by the multiple Teams working directly with customers/users
  - ▶ Clarification is as much as possible directly between the Teams and customer/users
  - ▶ Product Backlog Grooming is done per team for the items they are likely going to do in the future
  - ▶ Do multi-team and/or overall grooming to increase shared understanding and exploiting coordination opportunities

# LeSS: Backlogin ylläpito

- ▶ LeSS kiinnittää eksplisiittisesti huomioita backlogin ylläpitämiseen
  - ▶ All prioritization goes through the Product Owner
  - ▶ The Product Owner shouldn't work alone on Backlog refinement
  - ▶ Supported by the multiple Teams working directly with customers/users
  - ▶ Clarification is as much as possible directly between the Teams and customer/users
  - ▶ Product Backlog Grooming is done per team for the items they are likely going to do in the future
  - ▶ Do multi-team and/or overall grooming to increase shared understanding and exploiting coordination opportunities
- ▶ LeSS siis rohkaisee voimakkaasti sovelluskehittäjien ja asiakkaiden/loppukäyttäjien läheiseen kanssakäymiseen

# LeSS huge

- ▶ Yksi tuote, backlog ja vastuunalainen product owner
- ▶ Backlog jaetaan vaatimusalueisiin (requirement area)
  - ▶ jokaiselle alueelle siitä vastaava *area product owner*
  - ▶ muodostavat product owner -tiimi
  - ▶ backlogiin aluekohtaiset näkymät



## LeSS vs SAFe

- ▶ Sekä SAFe että LeSS ovat pitkälti syntyneet Suomessa ja Nokialla

## LeSS vs SAFe

- ▶ Sekä SAFe että LeSS ovat pitkälti syntyneet Suomessa ja Nokialla
- ▶ Nokian organisaatorakenteen takia Nokia Mobile Phonesin (NMP) ja Nokia Siemens Networksin (NSN) ohjelmistokehitystapa oli täysin erilainen
- ▶ SAFe (NMP) ja LeSS (NSN) ovat samoista taustaperiaatteistaan ja yhteisestä syntykonsernista huolimatta hyvin erilaisia menetelmiä

## LeSS vs SAFe

- ▶ Sekä SAFe että LeSS ovat pitkälti syntyneet Suomessa ja Nokialla
- ▶ Nokian organisaatorakenteen takia Nokia Mobile Phonesin (NMP) ja Nokia Siemens Networksin (NSN) ohjelmistokehitystapa oli täysin erilainen
- ▶ SAFe (NMP) ja LeSS (NSN) ovat samoista taustaperiaatteistaan ja yhtesisestä syntykonsernista huolimatta hyvin erilaisia menetelmiä
- ▶ SAFe suosittu yritysjohdon keskuudessa, saanut paljon kritiikkiä
- ▶ Ohjelmistokehittäjiltä en ole kuullut SAFesta juurikaan kiitosta

## LeSS vs SAFe

- ▶ Sekä SAFe että LeSS ovat pitkälti syntyneet Suomessa ja Nokialla
- ▶ Nokian organisaatorakenteen takia Nokia Mobile Phonesin (NMP) ja Nokia Siemens Networksin (NSN) ohjelmistokehitystapa oli täysin erilainen
- ▶ SAFe (NMP) ja LeSS (NSN) ovat samoista taustaperiaatteistaan ja yhtesisestä syntykonsernista huolimatta hyvin erilaisia menetelmiä
- ▶ SAFe suosittu yritysjohdon keskuudessa, saanut paljon kritiikkiä
- ▶ Ohjelmistokehittäjiltä en ole kuullut SAFesta juurikaan kiitosta
- ▶ SAFe:n kotia Nokia Mobile Phonesia ei enää ole olemassakaan, Nokia Networks taas on nykyinen Nokia ja soveltaa yhä LeSS-menetelmää

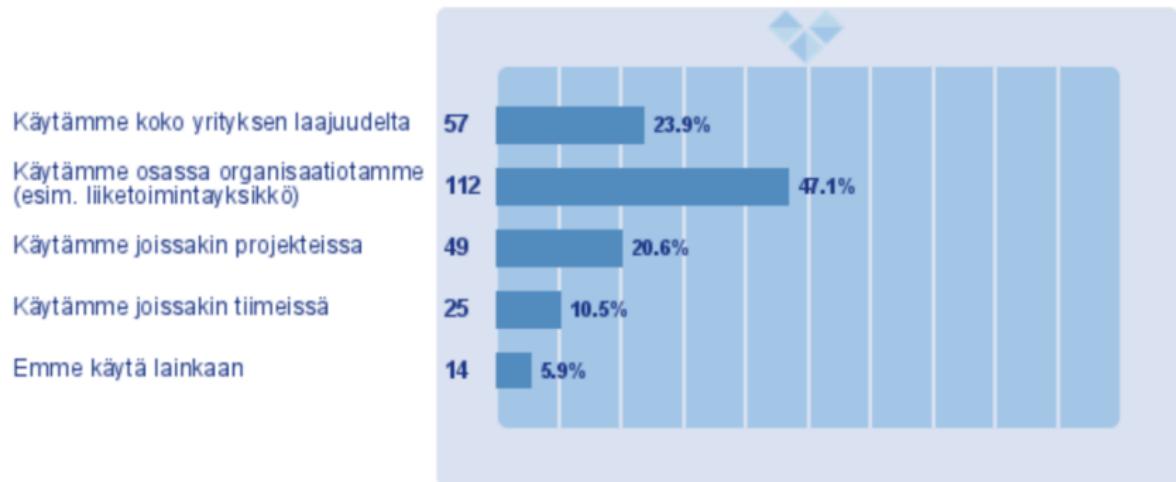
## Miten laajalti ketterää/leania käytetään

- ▶ Paljon kyselytutkimuksia, luvut vaihtelevat 46-86 % välillä
  - ▶ Project management institute (2018): 46 %
  - ▶ Stack overflow (yli 200000 vastaajaa, 2018): 85.9%

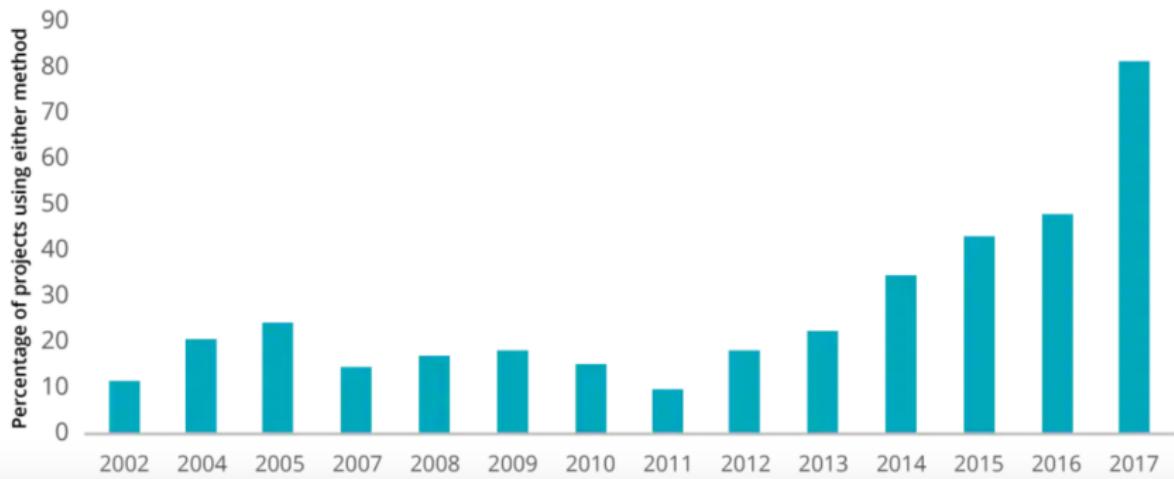
# Miten laajalti ketterää/leania käytetään

- ▶ Paljon kyselytutkimuksia, luvut vaihtelevat 46-86 % välillä
  - ▶ Project management institute (2018): 46 %
  - ▶ Stack overflow (yli 200000 vastaajaa, 2018): 85.9%
- ▶ Akateemisia tutkimuksia
  - ▶ Oulun yliopisto *Survey on Agile and Lean usage in Finnish software industry* 2012: 58% vastanneista 200 yrityksestä käytti agilea tai leania
  - ▶ Turun yliopisto ym *Adoption and Suitability of Software Development Methods and Practices*
    - ▶ Scrum 71.2%
    - ▶ Kanban 49.5%
    - ▶ Lean 39.7%
    - ▶ Vesiputous 35.3%

► Helsingin yliopiston ja Nitorin loppuvuodesta 2018 tekemän selvitys

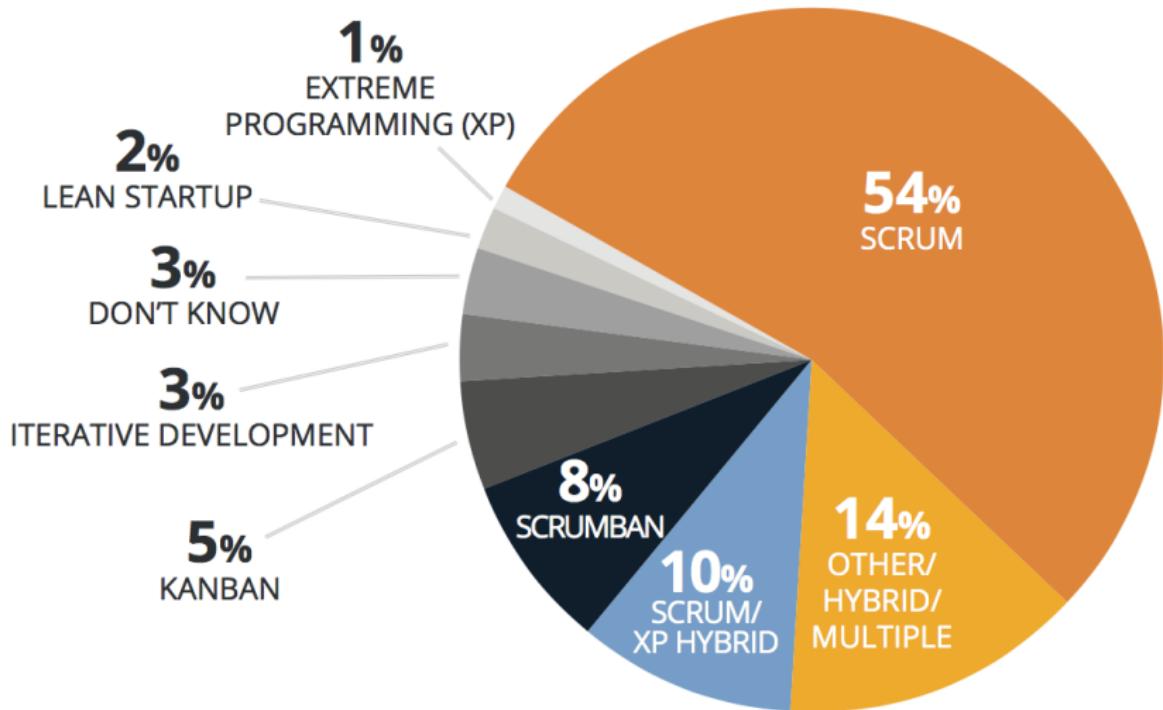


► Yhdysvaltojen hallituksen alaiset ohjelmistoprojektit

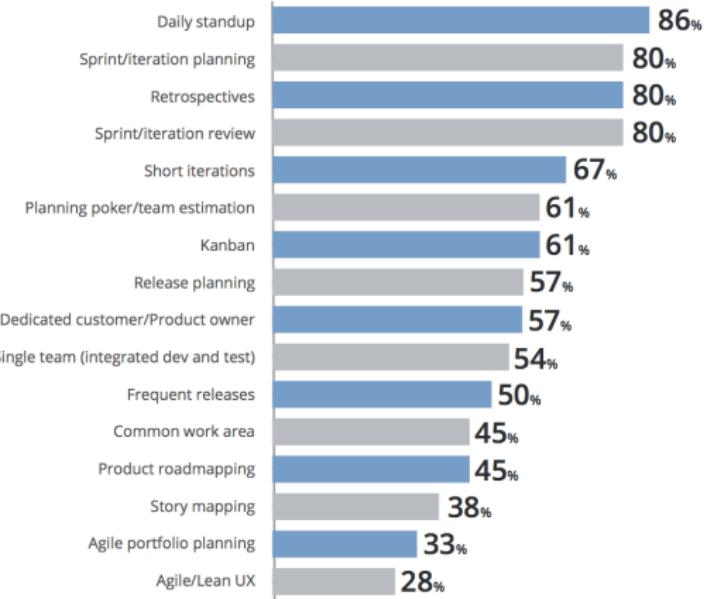
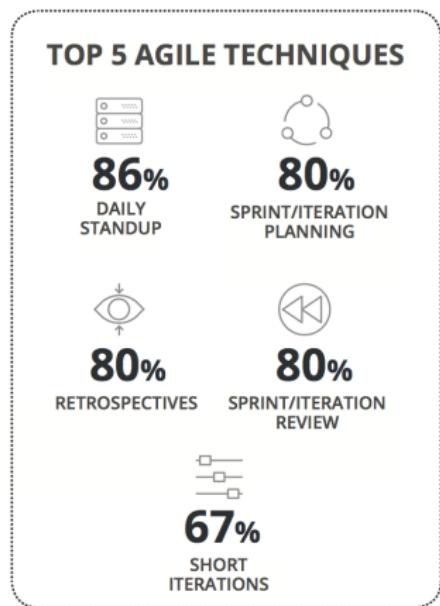


# State of Agile -raportti

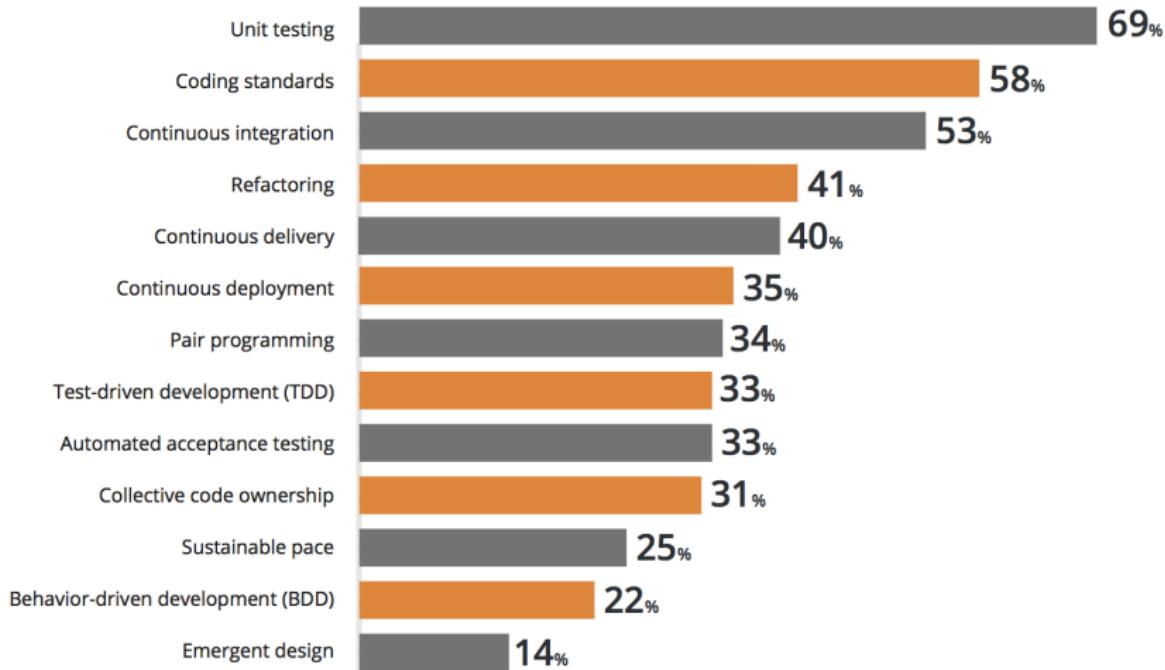
- ▶ Scrum dominoi



## ► Projektinhallintakäytänteet



## ► Tekniset käytänteet

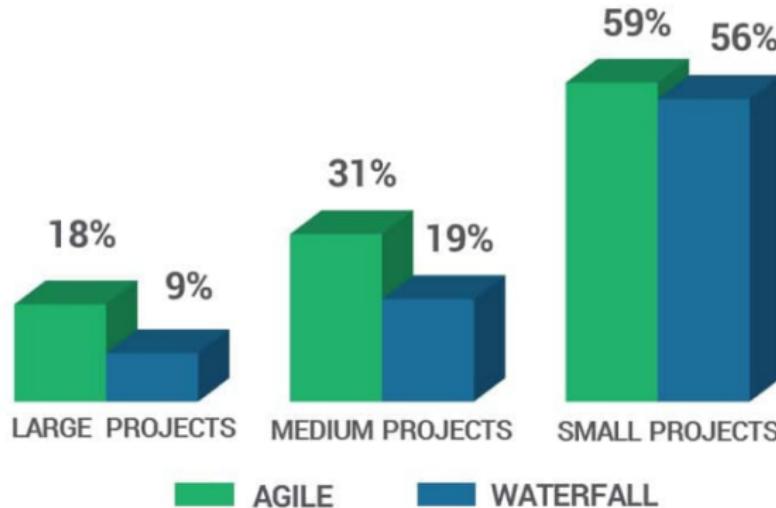


# Toimiiko ketterä ohjelmistokehitys

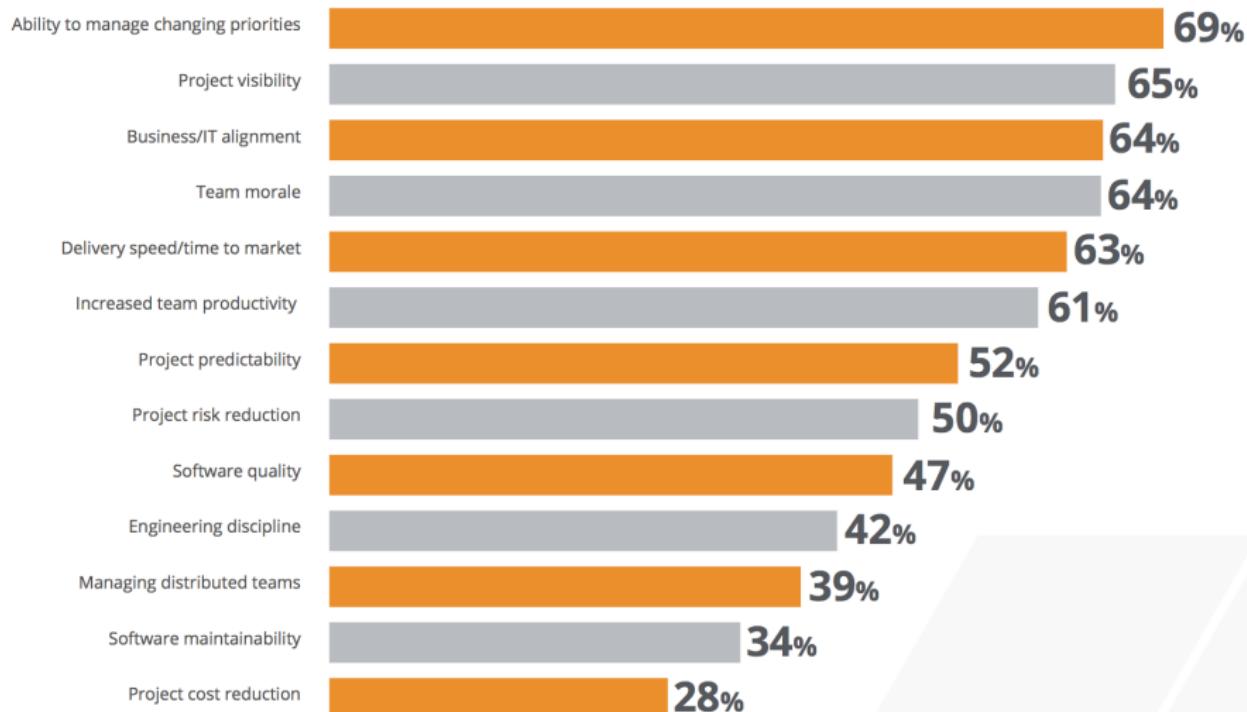
- ▶ Standish groupin *Chaos rapport*, vuodesta 1995
- ▶ 2018:



- ▶ Projektin koolla on todella suuri merkitys onnistumisen kannalta.



## ► Ketteryydellä saavutettuja hyötyjä



- ▶ Evidenssiä on, mutta...
- ▶ Kaikki edelliset olivat kyselytutkimuksia
  - ▶ käsitteitä ei ole kunnolla määritelty (Scrum vs Scrumbut)
  - ▶ osallistuneet eivät välttämättä edusta tasaisesti koko populaatiota
  - ▶ kyselyjen tekijät eivät puolueettomia menetelmien suhteen

- ▶ Evidenssiä on, mutta...
- ▶ Kaikki edelliset olivat kyselytutkimuksia
  - ▶ käsitteitä ei ole kunnolla määritelty (Scrum vs Scrumbut)
  - ▶ osallistuneet eivät välttämättä edusta tasaisesti koko populaatiota
  - ▶ kyselyjen tekijät eivät puolueettomia menetelmien suhteen
- ▶ Akateemisenkin tutkimuksen, laatu ja tulosten yleistettävyyys vaihtelee

- ▶ Evidenssiä on, mutta...
- ▶ Kaikki edelliset olivat kyselytutkimuksia
  - ▶ käsitteitä ei ole kunnolla määritelty (Scrum vs Scrumbut)
  - ▶ osallistuneet eivät välttämättä edusta tasaisesti koko populaatiota
  - ▶ kyselyjen tekijät eivät puolueettomia menetelmien suhteen
- ▶ Akateemisenkin tutkimuksen, laatu ja tulosten yleistettävyyys vaihtelee
- ▶ Ohjelmistokehityksessä liian paljon muuttuja, jotta jonkin yksittäisen tekijän vaikutusta voitaisiin mitata empiirisesti
- ▶ Menetelmiä soveltavat ihmiset, ja mittaustulos yhdellä tiimillä ei välttämättä yleisty muihin olosuhteisiin