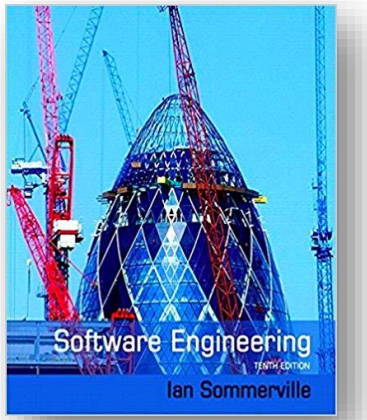
Extreme Programming ja Test Driven Development

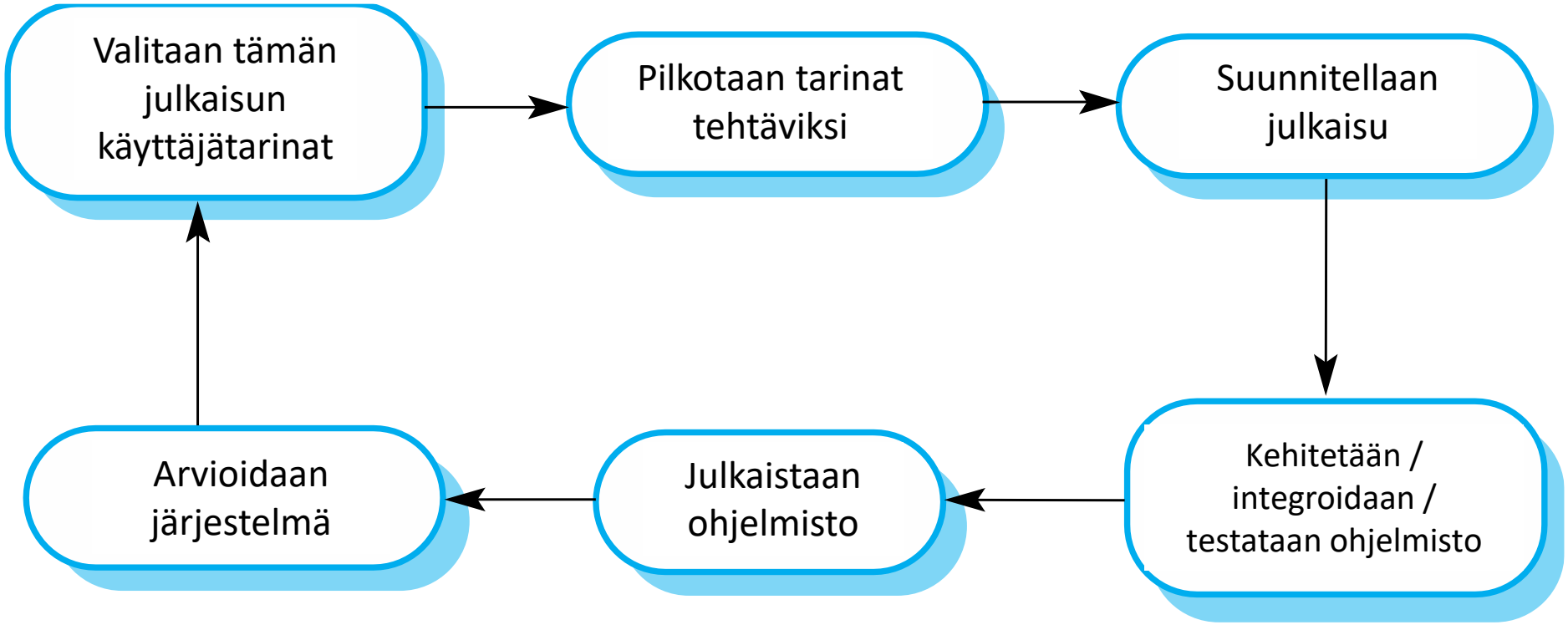


UNIVERSITY OF OULU

Extreme programming XP

- 1990- luvun lopulla kehitetty ketterä menetelmä, joka sisältää joukon ketteriä kehitystekniikoita.
- Extreme Programming (XP) omaksuu ‘extreme’ eli äärimmäisen lähestymistavan iteratiiviseen kehitykseen.
  - Uusia versioita voidaan tehdä useita kertoja päivässä;
  - Tuoteversiot toimitetaan asiakkaalle kahden viikon välein;
  - Kaikki testit on suoritettava joka versiolle ja versio hyväksytään vain jos testit suoritetaan onnistuneesti.

The extreme programming julkaisusykli



XP ja ketterät periaatteet

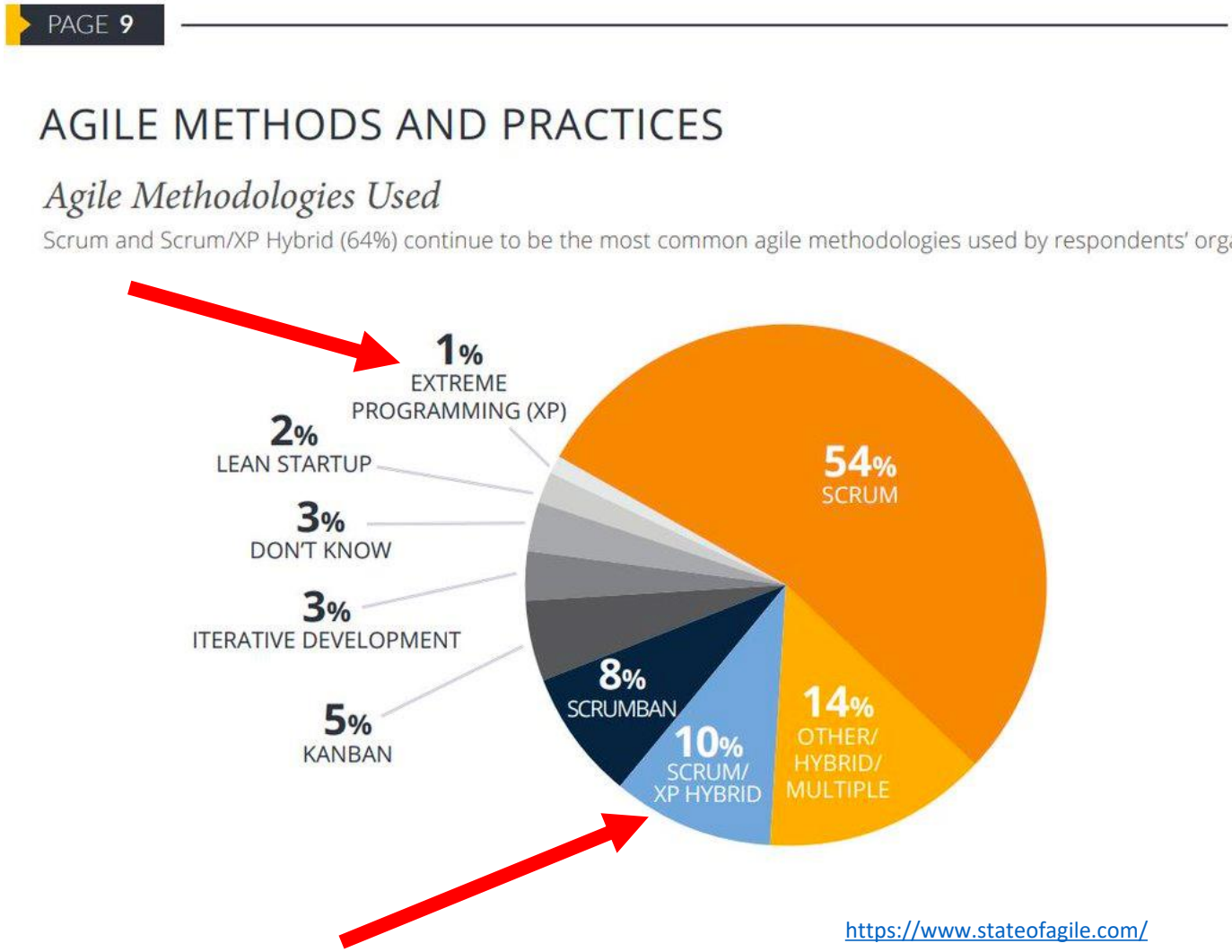
- Inkrementaalista, laajenevaa kehitystä tuetaan pienillä, usein tapahtuvilla järjestelmien julkaisuilla.
- Asiakkaan osallistuminen tarkoittaa kokopäiväistä asiakkaan sitoutumista kehittäjätiimiin.
- Ihmiset ennen prosessia –periaatetta toteutetaan
  - pariohjelmoinnilla,
  - yhteisomistajuudella ja
  - prosessilla, jossa vältetään pitkiä työpäiviä.
- Muutosta tuetaan säännöllisillä järjestelmän julkaisuilla.
- Ylläpidetään yksinkertaisuutta korjaamalla koodia jatkuvasti refaktoroinnin avulla.

https://commons.wikimedia.org/wiki/File:Pair_programming_1.jpg

Pariohjelmointi

Merkittävät XP käytännöt

- XP on teknisesti painottunut ja useimmissa organisaatioissa sitä ei ole helppo yhdistää johtamiskäytäntöihin.
- Näin ollen, vaikka ketterä kehitys käyttää XP:n käytäntöjä, alkuperäinen menetelmä ei ole käytössä laajalti.
- Avainkäytännöt
  - Käyttäjätarinoiden käyttö määrittelyä varten
  - Refaktorointi
  - Testaus-ensin kehitys
  - Pariohjelmointi



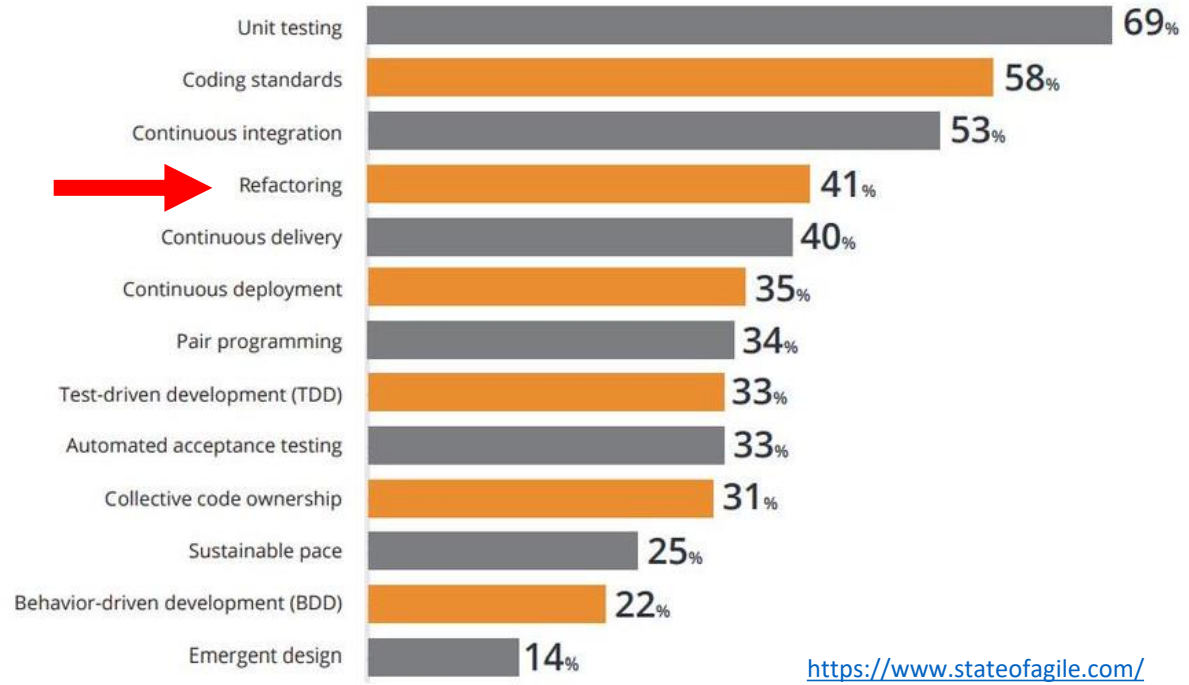
https://www.stateofagile.com/

Käyttäjätarinat vaatimuksissa

- XP:ssä asiakas tai käyttäjä on osa XP-tiimiä ja on vastuussa vaatimuksia koskevien päätösten tekemisessä.
- Käyttäjävaatimukset ilmaistaan käyttäjätarinoina tai skenaarioina.
- Käyttäjätarinat on kirjoitettu korteille ja kehitystiimi jakaa ne toteutettaviksi tehtäviksi. Näiden tehtävien mukaan tehdään aikataulu ja kustannusarvio.
- Asiakas valitsee tarinat sisällytettäväksi seuraavaan julkaisuun tärkeysjärjestyksensä ja aikatauluarvioidensa perusteella.

Refaktorointi

- Ohjelmointitiimi etsii mahdollisia ohjelmistoparannuksia ja tekee nämä parannukset jopa silloin kun niille ei olisikaan mitään välitöntä tarvetta.
- Niillä parannetaan ohjelmiston ymmärrettävyyttä ja siten vähennetään dokumentointitarvetta.
- Hyvin jäsenneltyyn ja selkeään koodiin on helpompi tehdä muutoksia.
- Jotkut muutokset vaativat kuitenkin arkkitehtuurin refaktorointia ja tämä on paljon kalliimpaa.

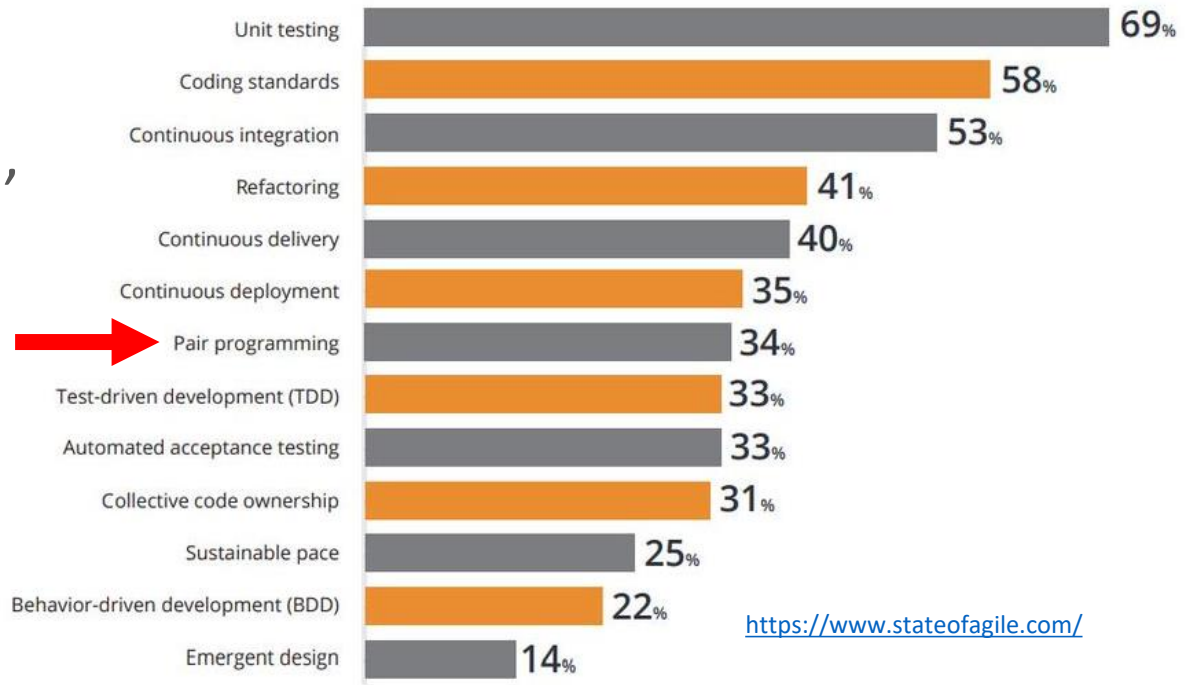


https://www.stateofagile.com/

Pariohjelmointi 1/2

- Pariohjelmoinnissa ohjelmoijat työskentelevät pareittain kehittäen koodia yhdessä.
  - Ohjelmoijat istuvat yhdessä saman tietokoneen ääressä kehittääkseen ohjelmistoa.
- Tämä edistää koodin yhteisomistusta ja levittää tietoa koko tiimille.
- Se toimii epävirallisena tarkistusprosessina, koska joka koodirivin tarkistaa useampi kuin yksi henkilö.
- Järjestelmäkoodin parannukset kannustavat refaktorointiin, koska koko tiimi hyötyy niistä.

https://commons.wikimedia.org/wiki/File:Pair_programming_1.jpg

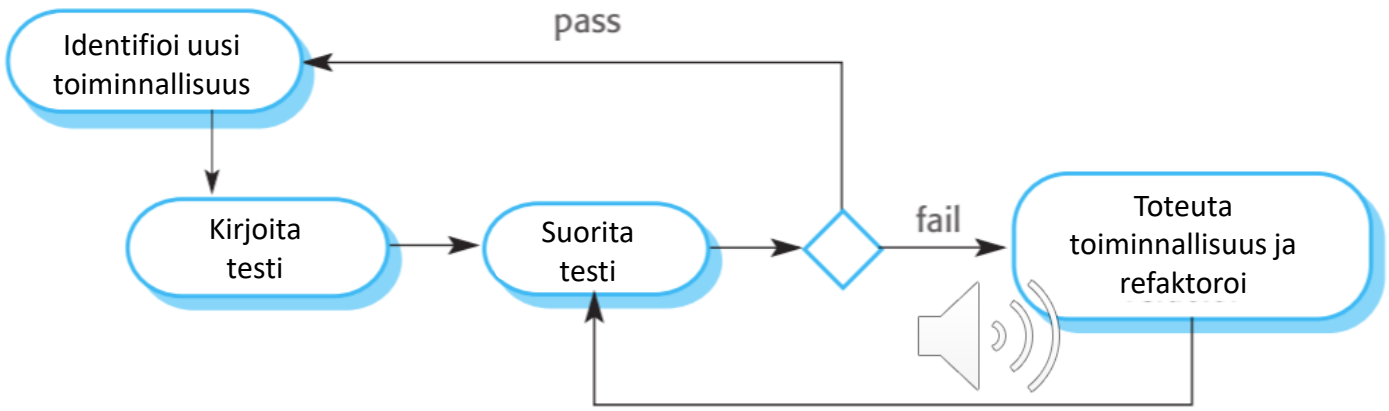


https://www.stateofagile.com/

Pariohjelmointi 2/2

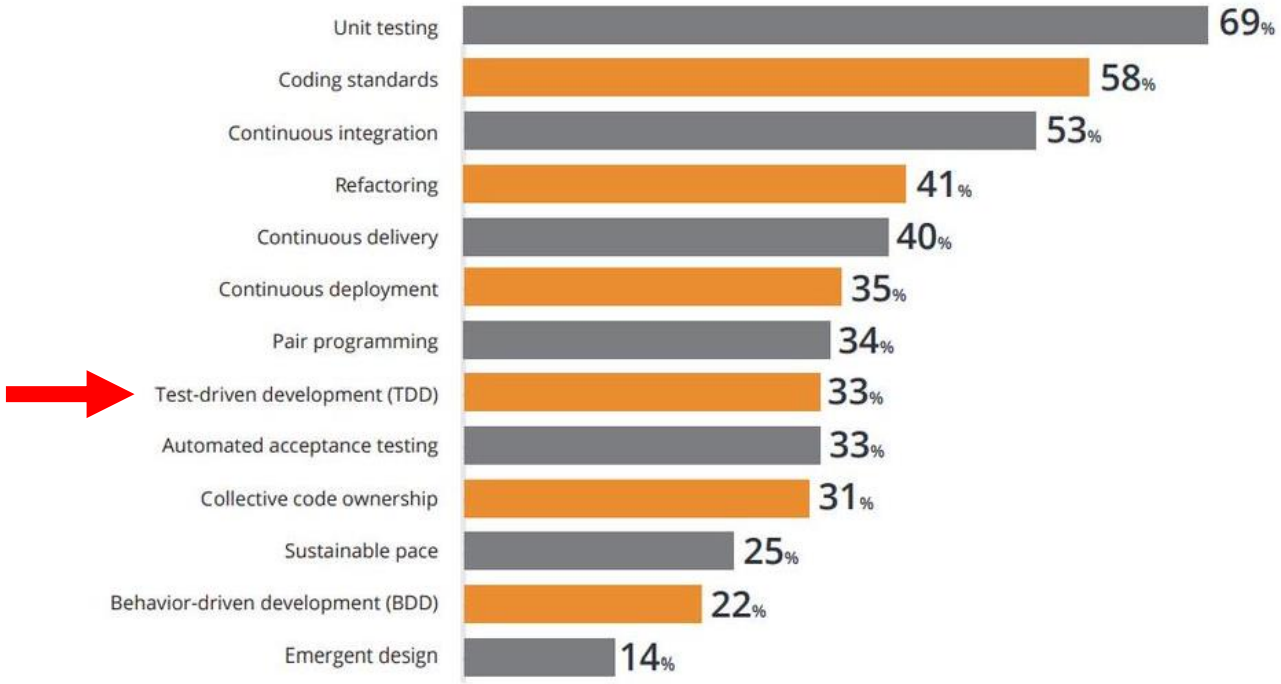
- Parit muodostetaan vaihtelevasti, jotta kaikki tiimin jäsenet työskentelisivät keskenään kehitysprosessin aikana.
- Pariohjelmointiin liittyvä jaettu tieto vähentää projektiin liittyvää riskiä siinä tapauksessa, jos joku tiimin jäsenistä lopettaa projektissa.
- Tutkimustulokset menetelmän hyödyistä eivät ole selvät. Jotkut tutkimukset osoittavat alhaisempaa tuottavuutta, toiset korkeampaa.

Test-driven development TDD
Testivetoinen kehitys



- Testien kirjoittamisen tarkoitus ennen koodin kirjoittamista on selventää toteutettavaksi tarkoitettuja vaatimuksia.
- Testit kirjoitetaan ohjelmina eikä tietoina, jotta ne voidaan suorittaa automaattisesti.
  - Testissä on mukana tarkistus siitä, että se on suoritettu oikein.
- Kaikki aiemmat ja uudet testit suoritetaan automaattisesti aina kun lisätään uusi toiminnallisuus.
  - Näin tarkistetaan, että uuteen toiminnallisuuteen ei ole tullut virheitä.

Test-driven development TDD
Testivetoinen kehitys



https://www.stateofagile.com/