Agile

Manifesto of Agile Software Development (2001):

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas

Principles of Agile Software

- ...
- Working software is primary measure of progress
- ...

Agile-menetelmistä

- Väite: Agile-menetelmät ovat epämuodollisia, suunnittelemattomia ”cowboy” -menetelmiä

- Höpö höpö. XP:n (Extreme Programming) käytänteet muodostavat kurinalaisen ”common sense” -menetelmän
  $\rightarrow$ **cowboy coding** $\neq$ **extreme programming**
- Tunnetuimmat Agile-menetelmät: XP ja SCRUM

Lyhyet iteraatiot

- Asiakkaalle toimitetaan aikaisessa vaiheessa, ja sen jälkeen lyhyin aikavälein, toimiva sovellus
- Inkrementaalinen ja iteratiivinen ohjelmistonkehitys

→ pakottaa suunnittelemaan ja seuraamaan projektia
→ pakottaa paljastamaan asiakkaalle työtahdin
→ puolin ja toisin realistiset odotukset
→ motivoi!
→ mahdollistaa oppimisen (reflektointi) projektin aikana
→ konkretian esille tuonti tarkentaa asiakasvaatimuksia

Vaatimusten valinta iteraatioon

- Asiakas määrittelee ja priorisoi vaatimukset (yhdessä kehittäjien kanssa)
- Harris & Cohn:
  - aikaista vaatimuksia, jotka lisäävät tietämystä
  - siirrä myöhemmäksi vaatimuksia, joiden oletettu muutoskustannus (expected cost of change) on suuri
  - hyödynnä opittua usein (suunnittelu on osa prosessia ei prosessin tulos), mutta vain päättääksesi mitä tehdä seuraavaksi

Iteraatioista

- Projektin alussa ”nollaiteraatio”
- Iteraation reunat: suunnittelu ja hyväksyntä
- Iteraation pituus
- Työrauha (SCRUM korostaa), vältettävä kaaosta
- Päivittäiset lyhyet kehittäjäpalaverit

Suunnittelu (planning game)

- Käyttäjätarinat (user stories): yksinkertainen dokumentointi esim. pahvikorteille
- Kommunikointi keskeisessä asemassa
- Suhteelliset arviot (esim. pisteet) tarinoiden toteutuksen vaativuudesta
  $\rightarrow$ vertailu aiemmin toteutettuihin toiminnallisuuksiin yksinkertaista
- Reflektointi avuksi, arviointia pystytyään parantamaan aikaisempien iteraatioiden tulosten perusteella
- Käyttäjätarinat suppeampia kuin use caset
- Välitön arviointi (cost), vrt. ”perinteisten” vaatimusten kirjoittaminen
- Cohn: A good story is independent, negotiable, valuable to users or customers, estimatable, small, and testable

Esimerkki (Cohn):

A company can pay for a job posting with a credit card.

Note: Will we accept Discover cards?

Note for UI: Don't have a field for card type (it can be derived from first two digits on the card).

”Spike”

- Tutkitaan ongelmallista toteutuksen osa-aluetta etukäteen
- Kestoltaan esim. yksi iltapäivä (kiinteä arvio)
- Perusteltua valita spike toteutettavaksi, ja story, jota spike tutkii vasta (mahdollisesti) seuraavaan iteraatioon
  → samaan iteraatioon sijoittaminen lisää epävarmuutta, koska spiken tulosta ei tunneta

Iteraatioon valittujen storyjen tarkennus

- Kehittäjät jakavat storyt ohjelmointitehtäviksi (tasks)
- Tarvittaessa tarkentavia kysymyksiä asiakkaalle
- Suhteelliset arviot taskeille
- Velocityn tarkistus → tarvittaessa uudelleen priorisointi

Iteraation hyväksyntä

- Hyväksyntää ei saa ohittaa, muuten iteratiivisyys jää näennäiseksi
- Hyväksynnän kohdistuttava siihen, mitä on sovittu tehtäväksi, vrt. työrauha
- Storyn hyväksyntätestit voidaan asiakkaan kanssa kirjoittaa esim. story-kortin kääntöpuolelle
- Pyrkimys hyväksyntätestien automatisointiin

Hyväksyntätestit (Cohn):

- Test with Visa, MasterCard and American Express (pass).
- Test with Diner's Club (fail).
- Test with good, bad and missing card ID numbers.
- Test with expired cards.
- Test with over $100 and under $100.

Release Plan

- Karkea suunnitelma toiminnallisuuksista, joita julkaistaviin sovelluksen versioihin halutaan sisällyttää.
- Esimerkiksi lista storeja, jotka kiinnitetään tiettyyn iteraatioon ja julkistukseen (release)
- Saa muuttua

Implementoinnista

- Miksi ei uskalleta aluksi tehdä toiminnallisuuksia vaan laajennettavuutta, ylläpidettävyyttä ym. ?
- Bottom-up
- Refaktorointi ei ole epäonnistumisesta aiheutuvaa ylimääräistä työtä
- Pyrkimys yksinkertaisiin ratkaisuihin
- Asiakkaalla oikeus päättää käytetäänkö resursseja välittömään toiminnallisuuksien toteutukseen vai ohjelmiston rakenteen uudelleenmuokkaukseen -> avoimuus -> ”oma pää puhtaana”
- Test Driven Development (TDD, Jonne kertoo)

Vertailua

Zhang et. al: Comparison Between Test Driven Development and Waterfall Development in a Small-Scale Project

No.	Evaluated parameters	Superioty of TDD to Waterall (percentage)
1	Productivity	10 %
2	Reliability	28 %
3	Maintainability	8 %
4	Flexibility	30 %
5	Efficiency	33 %
6	Tester Quality	10 %

Kommunikointi

- Miksi ei uskalleta luottaa kasvotusten kommunikointiin?
- Kasvotusten kommunikointi on ideaalinen vuorovaikutustapa
- Kumpi on suurempi riski: kommunikointi erilaisia kaavioita katselmoiden vai ihmisten välinen keskustelu?

Kehitystiimi

- Kommunikoi asiakkaan kanssa
- Itseorganisoituu, ei kiinteitä rooleja
- Koostuu (ideaalitapauksessa) eri alojen asiantuntijoista
- Coach, Scrum-master

Mihin Agile ei sovellu?

Esim. Beck arvelee XP:stä:

- Organisaatiohin, joiden todelliset arvot eivät vastaa XP-arvoja.
- Suuriin kehitystiimeihin (satoja ohjelmoijia).
- Sovellusympäristöihin, joiden teknologia ei tue muutosta.
- Huom. soveltuvuutta arvioitava tapauskohtaisesti

Practices of XP

- whole team
- metaphor
- the planning game
- simple design
- small releases
- customer tests
- pair programming

Practices of XP

- test-driven development
- design improvement
- collective code ownership
- continuous integration
- sustainable pace
- coding standards

Lähteet:

Manifesto for Agile Software Development: http://agilemanifesto.org/.

Principles of Agile Software: http://agilemanifesto.org/principles.html.

Cohn, M., Writing Stories (chapter 2), From ”User Stories Applied”, 2004 http://www.mountaingoatsoftware.com/system/hidden_asset/file/10/usa_sample.pdf.

Beck, K., Extreme Programming Explained (2nd Edition), The XP Series , Addison-Wesley, 2005.

Beck, K., Fowler, M., Planning Extreme Programming, The XP Series, Addison-Wesley, 2001.

Steinberg, D. H., Palmer, D. W., Extreme Software Engineering: A Hands-On Approach, Pearson Education, Inc., 2004.

Schwaber, K., Beedle, M., Agile Software Development with Scrum, Prentice Hall, 2002.

Zhang et. al, Comparison Between Test Driven Development and Waterfall Development in a Small-Scale Project, Proceedings of XP 2006.

Harris, R. S., Cohn, M., Incorporating Learning and Expected Cost of Change in Prioritizing Features on Agile Projects, Proceedings of XP 2006.

Fowler, M., Evolutionary Database Design, 2003,
`http://www.martinfowler.com/articles/evodb.html`.