Johdattelua, motivointia,
eli missä ollaan ja kuinka siihen on tultu

Ohjelmistotuotanto

Kuinka valmistaa *laadukkaita* ja *tehokkaita* ohjelmistoja mahdollisimman *edullisesti?*

*Ohjelmistotuotanto (*Software Engineering*)* tarkoittaa periaatteita ja käytäntöjä, joita noudattamalla tämän pitäisi onnistua!

Ohjelmistojen laatu

- Laatu on ollut ongelma koko sen ajan kun ohjelmistoja on ollut olemassa
- Ohjelmistotuotanto tutkimusalana syntyi pitkälti ohjelmistojen heikon laadun takia
- (Mitä ohjelmistojen laatu on?)

Ohjelmistotuotanto

- yhdistää tietämystä useilta aloilta
  - Tietojenkäsittelytiede
  - Projektinhallinta
  - Systems engineering (”järjestelmäsuunnittelu”, ei vakiintunutta suomennosta)

Insinööritiede: prosessi

- Painotus käytäntöön: kuinka todellisia ohjelmistoja tehdään todellisessa maailmassa
- *Ohjelmistotuotantoprosessi* (software process) on joukko toimintoja, jotka johtavat ohjelmiston valmistumiseen
  - Vaikutteita muista insinööritieteistä (systems engineering)
  - Ohjelmistojen tuottamisessa kuitenkin omat erityispiirteensä

Insinööritiede: työkalut

- Korkean tason ohjelmointikielet
- Mallinnustyökalut: Unified Modeling Language (UML)
- Versionhallintaohjelmistot
- Virheenjäljittimet
- CASE-työkalut
- ym. ym.

Human factors

- Ihmiset tekevät ohjelmistoja ihmisten käytettäväksi
- Lisäksi suuret ohjelmistot tehdään ryhmissä, *ohjelmistotuotantoprojekteissa*
- On herätty huomaamaan, että ihmisten ja ryhmien ominaisuudet (*human factors)* ovat keskeisiä tekijöitä ohjelmistotuotannossa
- Systemaattista, tutkittua tietoa siitä, kuinka *human factors* pitäisi ottaa huomioon ohjelmistotuotannossa, on hyvin vähän

Human factors: yksilö

- Human factors: yksilöön, ryhmään, organisaation, kulttuuriin liittyvät tekijät
- Yksilö
  - Taidot, osaaminen
  - Yksilöpsykologiset tekijät
    - persoonallisuus (käyttäytymiseen ja tuntemiseen liittyvät taipumukset)
    - motivaatio
    - kognitiivinen kyvykkyys
    - sosiaaliset taidot

Human factors: ryhmä

- Kuinka erilaiset ihmiset saadaan toimimaan yhdessä tavoitteen (= kaikin puolin onnistunut ohjelmisto) saavuttamisen kannalta optimaalisesti?
- Ryhmäilmiöt, ryhmädynamiikka
  - Kommunikaatio
  - Roolit
  - Johtajuus (valta)
  - Konfliktien ratkaiseminen
  - Ryhmäkoheesio, sosiaalinen integraatio

Human factors: organisaatio

- Ohjelmistot tehdään yleensä jossakin organisaatiossa, tyypillisesti yrityksessä
  - Ryhmääkin laajempi organisaatiotason ilmiöiden näkökulma tarpeen
  - Esim. organisaation arvot, johtaminen, laatujärjestelmä, palkitsemisjärjestelmä, tiedonkulku, yrityskulttuuri

Human factor: kulttuuri

- Kulttuurien väliset erot
  - Kuinka johdetaan?
  - Kuinka kommunikoidaan?
  - Globaali ohjelmistokehitys ja hajautetut tiimit lisänneet kulttuurierojen huomioimisen merkitystä

Ketterät prosessimallit

- Moderni ohjelmistokehitys siirtymässä suuntaan, jossa kommunikaatio ja muut sosiaaliset tekijät entistä keskeisemmässä roolissa
- *Ketterät (agile) prosessimallit*
  - Projektipäällikön merkitys vähentynyt, ”tasa-arvoiset” ryhmät
  - ”itseorganisoituminen” - ei ryhmän ulkopuolista roolien jakoa
  - kommunikointi ensisijaisesti kasvotusten, ei dokumenttien välityksellä

Ohjelmistotuotannon synty

- Termi *Ohjelmistotuotanto* (Software Engineering) esiteltiin ensimmäistä kertaa 1968 pidetyssä NATO:n konferenssissa. Termi määriteltiin näin:

  The establishment and use of *sound* engineering principles in order to obtain *economically* software that is *reliable* and works *efficiently* on *real machines*.

  (P. Naur, R.Randell (eds.): Software Engineering: A Report on a Conference Sponsored by the NATO Science Committee, 1968)

1950-luku

- Ensimmäiset ohjelmat kirjoitettiin konekielellä
- Ensimmäinen kaupallinen ohjelmisto 1951 Englannissa (J. Lyons Company)
- Ensimmäiset korkean tason ohjelmointikielet kehitettiin jo 50-luvulla (Fortran, Cobol, Algol)
- Alkuaikoina ohjelmistojen teko oli suoraviivaista ja ohjaamatonta
- Silti jo 50-luvulla tehtiin kohtuullisen monipuolisia ohjelmistoja

1960-luku

- 1960-luvun alussa
  - määriteltiin jo varsin moderneja asioita, kuten testattavuus ja rajapinnat
  - ei vielä voi puhua varsinaisesta ohjelmistotuotannosta
  - *Cobol* ja *PL/1* -ohjelmointikielet merkittäviä kehitysaskelia, edelleen 60-luvun lopussa kehitetty *C-kieli* (B-kielestä jatkokehitetty)

60-luku ja ohjelmistokriisi

- tietokoneiden kapasiteetti kasvoi

→ohjelmistojen koko kasvoi

→ suoraviivainen ohjelmistojen tekemisen tapa alkoi kohdata suuria ongelmia
  - laatuongelmat
  - projektien myöhästely
  - kustannusten ylittäminen
- ns. *60-luvun ohjelmistokriisi*
- johtopäätös: ohjelmistojen valmistamisessa tarvitaan muutakin kuin ohjelmointia

Ensimmäiset ohjelmistotuotantokonferenssit

- Vuosina 1968-69 NATO organisoi kaksi konferenssia
  - esiteltiin termi Software Engineering
  - määriteltiin ohjelmistotuotannon perusteet
- Konferenssit eivät ehkä ratkaisseet ohjelmistokriisiä
- Esitellyt ideat rakensivat perustan, jolle nykyaikainen ohjelmistotuotanto on rakennettu

1970-luku ja prosessimallit

- 1970-luvulle tultaessa määriteltiin ensimmäinen ohjelmistotuotannon *prosessimalli*
- Prosessimalli on mahdollisimman yleisesti sovellettavissa oleva ohjeisto ohjelmistojen tuottamiseen
- Ensimmäinen prosessimalli oli *vesiputousmalli*
- Sen pohjana käytettiin insinööritieteiden yleistä *systems engineering* -mallia (”järjestelmäsuunnittelumallia”, vakiintunutta suomennosta termille ei ole)

Systems engineering –prosessi (vesiputousmalli)

- Vaatimusten määrittely
- Suunnittelu
- Toteutus ja yksikkötestaus
- Integroiminen ja järjestelmätestaus
- Käyttöönotto ja ylläpito

Systems engineering -malli

- Systems engineering -mallin avulla voidaan suunnitella vaikkapa siltoja
- Malli kopioitiin suoraan ohjelmistotuotantoon
- Mutta onko siltojen suunnittelu samanlaista kuin ohjelmistojen suunnittelu?

- Pian huomattiin, että vesiputousmalli ei sovi kaikenlaisten ohjelmistojen tekoon

→ 1970- ja 80-luvulla esiteltiin lukuisia muita prosessimalleja

1980-luku ja oliopohjaisuus

- Merkittävin uusi asia *oliopohjaisuus*
- Ensimmäinen oliokonferenssi OOPSLA pidettiin 1986.
- Oliomaailman periaatteet oli esitelty aiemmin
- Vasta oliokielet kokosivat ne yhteen (ensimmäinen oliokieli SIMULA kuitenkin jo 1967)
- Ohjelmointikielistä oliopohjaisuus levisi vähitellen koko ohjelmistotuotantoon

1990-luvun alku ja henkilökohtainen tietojenkäsittely

- 1980-luvulla alkanut *henkilökohtainen tietojenkäsittely* (jokaisella oma tietokone) mullisti ohjelmistojen tarpeen, mutta ei niiden valmistuksen tekniikkaa
- 1990-luvulle tultaessa ohjelmistotuotanto melko vakiintunutta
- *työkalujen aika:* CASE-työkalut kehittyivät (Computer-Aided Software Engineering)

1990-luvun ilmiöitä

- 1990-luvulla Internetin merkityksen räjähdysmäinen kasvu
- FOSS (Free/Open Source Software), avoimen lähdekoodin ohjelmistot
  - Lähdekoodin vapaus
  - Free software: GNU project 1983-
    - Vakaiden versioiden yhteydessä julkistetaan lähdekoodi
    - Näiden välillä vain rajatun kehittäjien joukon saatavilla

FOSS

- Linux 1991-
  - vapaa lähdekoodi
- Myöhemmin lukuisia muita
- *Open Source Initiative* (OSI) vuonna 1998 (organisaatio edistämään vapaata/avointa ohjelmistokehitystä)
- *Eric S. Raymond: The Cathedral and the Bazaar* (1997): avoin ohjelmistokehitys internetin kautta toimivan yhteisön kautta

Avoimen ohjelmistokehityksen ideoita

- Vapaus, copyright-rajoitusten purkaminen
- Jakaminen, kommunikaatio, kaikkien mahdollisuus osallistua (”basaarimalli”)

→ Sosiaalisten tekijöiden korostuminen

- Ohjelmistojen tekeminen itsessään arvokasta, ei vain taloudellinen hyöty
- Ei eksplisiittistä ohjelmistotuotantoprosessia
- Ohjelmoinnista kiinnostuneiden yhteisö: hauskaa
- *Linus Torvalds: Just for fun (2001)*

”*The architecture of Linux, the Internet, and the World Wide Web are such that users pursuing their own "selfish" interests build collective value as an automatic byproduct”*

- Tim O'Reilly: Architecture of Participation (2004)

1999 ja ketterät prosessimallit

- Merkittävä ohjelmistotekniikan murros tapahtui 1999
- esiteltiin ensimmäinen *ketterä prosessimalli* Extreme Programming (XP)
- Myöhemmin monia muita, esim. Crystal, Scrum
- Ketterät prosessimallit saavuttivat nopeasti erityisesti ohjelmoijien ja PK-yritysten suosion, sillä *mallit suosivat ohjelmointia suunnittelun sijaan*

- Ketterien menetelmien käyttöä edistää Agile Alliance, joka on voittoa tuottamaton järjestö
- Manifesto for Agile Software Development
- ”vapautusliike” dokumentoinnin ja kontrollin ”orjuudesta”
- Ihmisten ja tiimin korostaminen, vuorovaikutuksen korostaminen
- ”itseorganisoituvat tiimit”

“I think that ultimately, Extreme Programming has mushroomed in use and interest, not because of pair-programming or refactoring, but because, taken as a whole, the practices define a developer community freed from the baggage of Dilbertesque corporations”

- Jim Highsmith, for the Agile Alliance, 2001