Arviointitekniikat

Arviointitekniikat

Ohjelmistoon kuuluvan työmäärän ja kustannusten arviointiin voidaan käyttää kahta erityyppistä tekniikkaa

1. *Kokemuspohjaiset tekniikat*
   - Perustuu arvioijan kokemukseen aikaisemmista projekteista ja sovellusalueesta.
     - Arvioija tekee tietoisen päätöksen koskien oletettua työmäärän tarvetta.
2. *Algoritmi-pohjainen kustannusten mallintaminen*
   - Käytetään kaavamaista lähestymistapaa projektin työmäärän arvioimisen laskentaan
     - Tuotteen ominaisuudet, kuten koko, ja
     - Prosessin ominaispiirteet, kuten henkilöstön kokemus.

Kokemuspohjaiset lähestymistavat

- Arviot perustuvat kokemuksiin aikaisemmista projekteista ja näissä projekteissa ohjelmiston kehittämiseen kuluneesta työmäärästä.
- Tunnistetaan projektissa tuotettavat tuotokset sekä erilaiset ohjelmistokomponentit tai järjestelmät, mitkä tulee kehittää.
  - Nämä dokumentoidaan taulukkoon, arvioidaan yksilöllisesti ja lasketaan vaadittu kokonaistyömäärä.
- Työmäärän arvioinnissa on ottaa mukaan ryhmä ihmisiä ja pyytää jokaista ryhmän jäsentä selittämään oma arvionsa.

Kokemuspohjaisen lähestymistavan haasteet

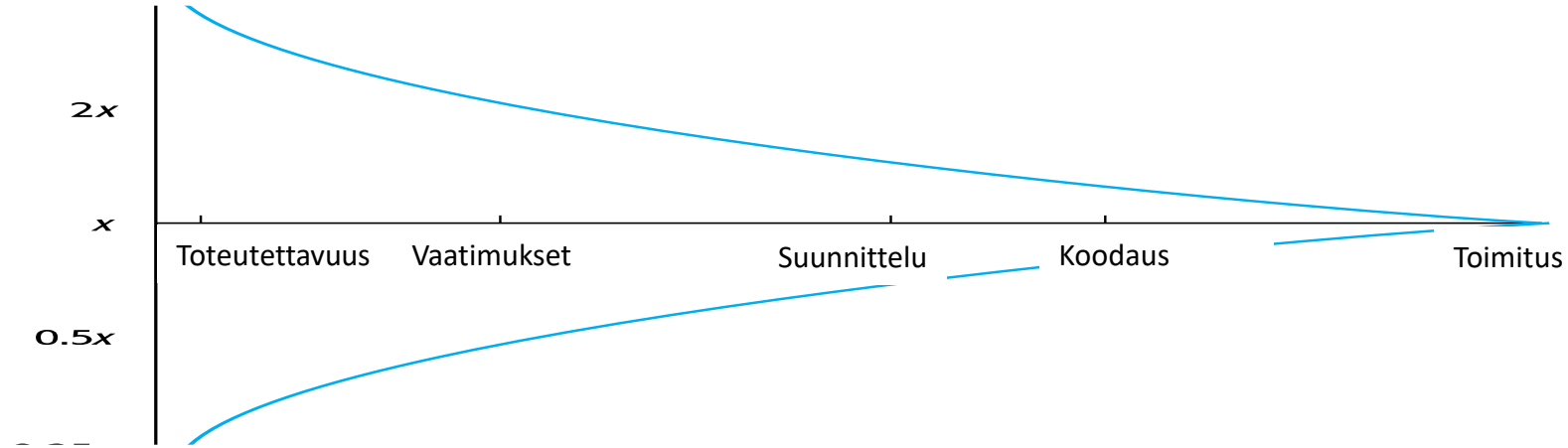
- Kokemuspohjaisten tekniikoiden haasteena on se, että uudella ohjelmistoprojektilla ei välttämättä ole paljon yhteistä aikaisempien projektien kanssa.
- Ohjelmistokehitys muuttuu nopeasti ja projektissa käytetään usein vähemmän tuttuja tekniikoita
  - Mikäli arvioija ei ole työskennellyt näiden tekniikoiden parissa, aikaisemmasta kokemuksesta ei ole hyötyä arvioitaessa vaadittua työmäärää.

Algoritmi-pohjainen kustannusten mallintaminen

- Kustannukset estimoidaan arvioijan antamien tuotteen, projektin ja prosessin ominaisuuksien matemaattisena funktiona:
  - $\text{Työmäärä} = A \times \text{Koko}^B \times M$
    - A on organisaatiosta-riippuvainen vakio,
    - B ottaa huomioon suurten projektien suhteettoman suuren työmäärän ja
    - M on kerroin, mikä ottaa huomioon tuotteen, prosessin ja ihmisten ominaisuudet.
- Useimmiten käytetty tuotteen ominaisuus kustannusten arvioimiseen on ohjelmistokoodin koko.
- Useimmat mallit ovat samankaltaisia, mutta ne käyttävät erilaisia arvoja A:lle, B:lle ja M:lle.
- Nämä mallit ovat monimutkaisia ja vaikeita käyttää.
  - Arvojen arvioinnissa on huomioitava useita ominaisuuksia ja niihin liittyy huomattava määrä epävarmuutta.
  - Algoritmi-pohjaista kustannusten mallintamista ovat käytännössä soveltaneet hyvin rajallinen määrä suuria yrityksiä, mitkä toimivat pääasiassa puolustus- ja ilmailujärjestelmien suunnittelussa.

Arvioinnin tarkkuus

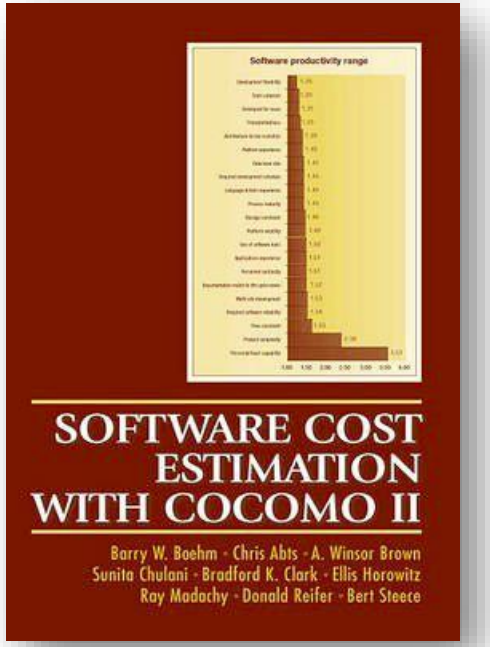
- Ohjelmistojärjestelmän koon pystyy tietämään tarkasti vasta kun järjestelmä on valmistunut.
- Lopulliseen kokoon vaikuttavat useat tekijät
  - Järjestelmien ja komponenttien uudelleenkäyttö;
  - Ohjelmointikieli;
  - Järjestelmän hajauttaminen.
- Arviot B:hen and M:ään vaikuttavista kriteereistä vaihtelevat, koska arviot ovat subjektiivisia ja riippuvaisia arvioitsijan käyttämistä kriteereistä.
- Kehitysprosessin edetessä arviot koosta tarkentuvat.



Cone of uncertainty

COCOMO kustannusmalli

- Empiirinen malli, mikä perustuu projekteissa kerättyyn kokemukseen.
- Tarkasti dokumentoitu, ‘itsenäinen’ malli, mikä ei ole kytköksissä mihinkään tiettyyn ohjelmiston tuottajaan.
- Pitkä historia, alustava versio (COCOMO-81) julkaistiin vuonna 1981. Sen jälkeen, useiden versioiden myötä on kehitetty COCOMO 2.
  - Tunnetuin malli
- COCOMO 2 huomioi erilaisia ohjelmistonkehityksen lähestymistapoja, kuten uudelleenkäyttö, jne.



By legendary
Prof. Barry Boehm et al.

https://www.youtube.com/watch?v=5sxKi-QsIOU

Varhaisen suunnittelun malli (osa COCOMO II mallia)

- Arvioita voidaan tehdä heti, kun vaatimuksista on päästy yhteisymmärrykseen.
- Perustuu algoritmi-pohjaisten mallien standardikaavaan
  - PM = A × Koko^{B} × M missä
    - M = PERS × RCPX × RUSE × PDIF × PREX × FCIL × SCED;
    - A = 2.94 alustavassa kalibroinnissa,
    - Koko KLOC-asteikolla,
    - B vaihtelee 1.1 ja 1.24 välillä, riippuen projektin uutuudesta, kehityksen joustavuudesta, riskienhallintalähestymistavoista ja prosessin kypsyydestä.
  - Kertoimet viittaavat kehittäjien kapasiteettiin, ei-toiminnallisiin vaatimuksiin, kehitysalustan perehtyneisyyteen, jne.
    - RCPX – Tuotteen luotettavuus ja monimutkaisuus;
    - RUSE – Vaadittu uudelleenkäyttö;
    - PDIF – Alustan vaikeus;
    - PREX – Henkilöiden kokemus;
    - PERS – Henkilöiden kyvykkyys;
    - SCED – Edellytetty aikataulu;
    - FCIL – Tiimin tukitoiminnot.

https://csse.usc.edu/tools/COCOMOII.php