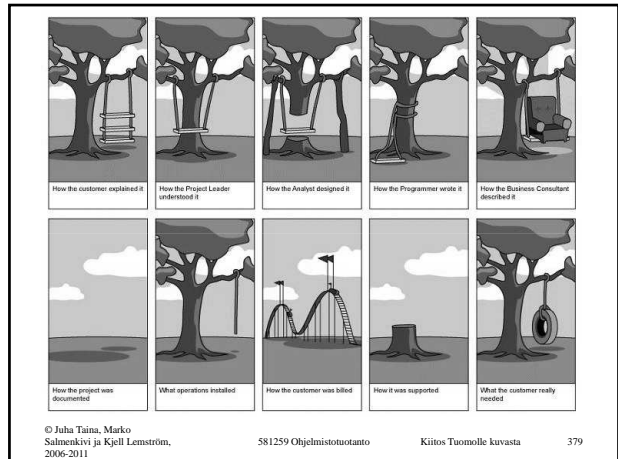


Yhteenvetoa, pieniä laajennuksia, tulevaisuuden haasteita

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 378



Ohjelmistotuotannon perustehtävät

- **projektinhallinta:** *kuka tekee, milloin* tehdään, mitä huomioidaan tehdessä
- **ongelman määrittely:** *mitä* tehdään
- **ohjelmistosuunnittelu:** *miten* tehdään
- **ohjelmiston toteutus:** tehdään
- **verifiointi ja validointi:** varmistetaan, että tehdään se mitä *pitää* tehdä
- **ylläpito:** pidetään ohjelmisto käyttökelpoisena

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 380

Perustehtävät

- Tehtäviä ei välttämättä suoriteta em. järjestyksessä eikä edes peräkkäin
- Ohjelmistotuotannossa on tarvetta erilaisille prosesseille ja prosessimalleille
- Vaikka jokainen osa-alue on tärkeä, ehkä tärkein on projektinhallinta
 - Suunnittelu ei ala tuotteesta, vaan prosessista
- Ilman suunnittelua ei synny laadukasta tuotetta

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 381

Suunnitelmakeskeiset prosessit

- Ennen ketteriä prosessimalleja kehitettyjä prosesseja kutsutaan *suunnitelmakeskeisiksi*
 - Tehdään paljon valmistelevaa työtä ennen toteutusta
- Vesiputousmalli ja V-malli ovat lineaarisia suunnitelmakeskeisiä malleja
- Suunnitelmakeskeisyys on eri asia kuin lineaarisuus
 - On olemassa iteratiivisia ja suunnitelmakeskeisiä malleja, kuten *spiraalimalli*

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 382

Vesiputousmalli

- Lineaariset vaiheet. Edellisen vaiheen tulos on seuraavan vaiheen syöte. Nykyisin mukana jonkin verran iteratiivisuutta.
- Perustuu huolelliseen dokumentaatioon ja tuotosten tarkastamiseen
- Selkeä ja helposti omaksuttava.
- Teoria- ja työkalutuki. Toimiva laadunvarmistus ja prosessin parannus.
- Ei vaadi projektin osallistujilta erityistaitoja. Sopii jos jatkuva ryhmätyö ei miellytä. Sopii hierarkkisiin organisaatioihin.
- Selkeän ja ymmärrettävän kustannusarvio projektin alussa.
 - Vahinko vaan, että se voi heittää jopa 4x verrattuna todellisuuteen
- Kiireisen asiakkaan ei tarvitse osallistua projektityöhön

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 383

Vesiputousmallin haitat

- Vaatimukset eivät saa muuttua
- Asiakas näkee valmista vasta projektin päättyessä
- Aikataulun lipsuessa testaus kärsii
- Dokumentointiin ja tarkastuksiin turhaudutaan helposti
- Malli ei suosi luovuutta
- Väärille urille joutuneen projektin saattaminen oikeaan suuntaan on vaikeaa
- Myöhässä olevaa projektia on vaikeaa saada takaisin aikatauluun
- Iso osa vesiputousmallin mukaisista projekteista epäonnistuu tästä syystä: projektit eivät valmistu aikataulussa

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011

581259 Ohjelmistotuotanto

384

V-malli

- Puhdasta vesiputousmallia kehittyneempi lineaarinen malli, joka yleistyi 1990-luvun alkupuolella
- V: *Suunnittelu ja toteutus + Testaus*
- Suunnittelu- ja toteutusvaiheessa määritellään vastaavan testausvaiheen testejä
 - Vaatimusten kartoitus -> Hyväksymistestit
 - Vaatimusten analyysi -> Järjestelmätestit
 - Arkkitehtuurisuunnittelu -> Integroititestit
 - Komponenttisuunnittelu -> Komponenttitestit
 - Koodaus ja yksikkötestaus

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011

581259 Ohjelmistotuotanto

385

Iteratiiviset ja IID prosessit

- Iteratiivisia prosesseja oli käytössä jo 1960-luvulla
 - Jokainen iteraatio on pieni projekti, jossa ovat mukana ohjelmistotuotannon perustehtävät
 - Ohjelmiston elinkaari muodostuu peräkkäisistä iteraatioista
- *Lisäävässä ohjelmistokehityksessä* (incremental development) tuotteeseen lisätään sykleittäin uusia ominaisuuksia
- IID (iterative and incremental development)
- *Kaikki* nykyiset ketterät prosessit ovat IID-variantteja

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011

581259 Ohjelmistotuotanto

386

Julkaisut

- Iteraation tuloksena saadaan *julkaisu*, lopullisen järjestelmän toimiva osajoukko (vrt. prototyyppi)
- Kaikki julkaisut eivät ole julkisia eli loppukäyttäjälle tarkoitettuja: sisäiset julkaisut ovat kehitystiimin ja asiakkaan käytössä
- Nykyisin kehitystyö jatkuu yleensä tuotteen elinkaaren ajan: tuotteesta tehdään useita julkisia julkaisuja
- Jos projektissa on monta kehitystiimiä, niin julkaisuun integroidaan kaikkien kehitystiimien tuotokset

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011

581259 Ohjelmistotuotanto

387

Iteraation pituus

- IID ei määrittele iteraation pituutta, mutta moderneissa iteratiivisissa prosessimalleissa iteraation suositeltava pituus on yhdestä kuuteen viikkoa.
- Lyhyt sykli tarkoittaa nopeaa reagoitua muutokseen. Toisaalta se myös tarkoittaa, että tehtävä ohjelmisto on ositettava hyvin pieniin toteutettaviin osiin.
- Iteraatioissa perustehtävien painotukset vaihtelevat sen mukaan, missä kohdassa tuotteen elinkaarta ollaan.
- IID:ssä varaudutaan muuttuviin vaatimuksiin, mutta ei koska tahansa. Muutoksiin varaudutaan iteraatioiden välillä, ei niiden sisällä.

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011

581259 Ohjelmistotuotanto

388

Riskilähtöinen ja asiakaslähtöinen iteraatio suunnittelu

- Iteraatioissa toteutettavat ominaisuudet voidaan valita *riskilähtöisesti* (risk-driven) tai *asiakaslähtöisesti* (client-driven).
 - Riskilähtöisyys: iteraatioon valitaan ne tehtävät, jotka ovat vaikeimmat ja riskialteimmat toteuttaa
 - Asiakaslähtöisyys: iteraatioon valitaan ne tehtävät, jotka asiakas asettaa korkeimmalle prioriteetille
- Riskilähtöinen kehitystyö palkitaan projektissa vasta myöhemmin
- Asiakas ei välttämättä pidä lähestymistavasta, koska toiminnallisuuden parantuminen ei yhtä suoraviivaista
- Asiakaslähtöinen kehitystyö ei saisi tapahtua arkkitehtuurisuunnittelun kustannuksella

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011

581259 Ohjelmistotuotanto

389

Timeboxing

- IID:ssa iteraation kesto kiinnitetään etukäteen: *timeboxing*-tekniikka
- Ketterät prosessimallit vähintään suosittelevat vahvasti tai vaativat timeboxing-tekniikan käyttöä
- Kestoaikea ei joustaa
 - Tiimi itse voi vähentää iteraatiossa tehtäviä, jos timeboxing ei muuten toteudu
- Kaikkien syklien pituuden ei tarvitse olla sama
- Iteraation käynnistymisen jälkeen ulkoiset sidosryhmät eivät vaikuta syklin sisältöön.
 - Jos iteraatio on menossa todella pahasti metsään, niin se voidaan keskeyttää ja aloittaa uusi iteraatio etuajassa

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 390

IID ja vaatimusmäärittely

- Useimmissa projekteissa *suurin osa* vaatimuksista on melko stabiileja: nämä tunnistetaan (ja toteutetaan) varhaisissa sykleissä
- Aina asiakas ei osaa kertoa, mitä tarvitaan tai ongelmakenttä muuttuu. Nämä tunnistetaan ja toteutetaan projektin edetessä.
- *Laatutekijöihin liittyvät vaatimukset* pitäisi tunnistaa mahdollisimman aikaisin arkkitehtuurivalinnat räätälöidään näiden tarpeiden mukaan

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 391

IID ja aikataulu

- IID:n luonteen takia varsinkin projektin alussa on vaikea tehdä edes alustavaa aikataulua
- Jo muutaman iteraation jälkeen käsitys on paljon helpompi muodostaa
 - Suunnitelmakeskeisten projektien alussa tehdyt aikataulut ovat myös hyvin alustavia
- Ongelman ratkaisu on
 - siirtää yleisen aikataulun tekoa projektin alusta muutaman iteraation päähän ja
 - tehdä yksityiskohtaista aikataulusuunnittelua korkeintaan muutaman iteraation verran eteenpäin.

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 392

Ketterät vs. suunnitelmakeskeiset

- Ketterien vs. suunnitelmakeskeisten prosessien sopivuuteen vaikuttavat
 - Millaista järjestelmää ollaan kehittämässä:
 - kuinka iso projekti
 - kuinka muuttuvat vaatimukset
 - kuinka pian tarvitaan toimiva versio
 - kuinka paljon tarvitaan dokumentaatiota
 - kuinka muodollinen laadunvarmistus tarpeen
 - Kehittäjätiimin kokemus ja taidot
 - Kehittäjäorganisaation organisaatiokulttuuri

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 393

Prototyypimalli

- Ensimmäisenä muuttuvien vaatimusten ongelmaan pyrittiin vastaamaan *prototyypimallilla*
 - Prototyyppi on toteutus, jossa ulkoiset liittymät ovat kunnossa mutta sisäinen logiikka on vajaa
 - Paperiprototyyppi tai suoritettava ohjelmisto
 - Loppukäyttäjät antavat palautetta prototyypistä
 - Varsinaisen ohjelmiston kehitystyö alkaa, kun loppukäyttäjät ovat tyytyväisiä prototyyppiin
- Tällä hetkellä tehdään enemmän iteratiivista ohjelmistokehitystä, mutta prototyyppejäkin käytetään
- Prototyyppien käytön voi yhdistää eri prosessimalleihin
- Toimii parhaiten pienillä (osa)järjestelmillä

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 394

Prototyyppien käyttö

- Ehdoton vaatimus on prototyyppien teon helppous
 - Käytännössä on voitava käyttää mahdollisimman paljon uudelleenkäytettäviä komponentteja
 - Sopiva sovelluskehitin auttaa samoin prototyyppien teossa
 - Prototyypeissä keskitytään ulkoisiin rajapintoihin
- Erikoistapaus: *käyttöliittymäpohjaiset prosessimallit*
 - Toiminnallisuuden sijaan suunnittelu aloitetaan käyttöliittymästä
 - Asiakkaalle tehdään kokeiltavaksi yksi tai useampi käyttöliittymämalli, jonka käytettävyyttä tämä arvioi

© Juha Taina, Marko Salmenkivi ja Kjell Lemström, 2006-2011 581259 Ohjelmistotuotanto 395

Plussia ja miinuksia

- Asiakkaalle pelkkää tekstidokumentaatiota selkeämpi kuva ohjelmiston toiminnasta ja valmiin tuotteen malli
- Prototyypit selventävät rajapintojen käyttöä
- Prototyyppien avulla ongelmakentästä saattaa löytyä käsittelemättömiä alueita
- Lisäkustannukset, kustannusarvion ja aikataulun teon vaikeus
- Miksi proto ei ole luovutettava tuote? Koska prototyypeissä oidaan toteutuksessa - se ei täytä lopulliselta tuotteelta vaadittavaa laatua
- Joskus on vaikea sanoa, milloin on tehty riittävästi protoja ja on aika ruveta varsinaiseen kehitystyöhön
- Vaatii hyvät työkalut ja paljon uudelleenkäytettävää koodia

© Juha Taina, Marko
Salmenkivi ja Kjell
Lemström, 2006-2011

581259 Ohjelmistotuotanto

396

Tulevaisuuden haasteita

- Epäyhtenäisyyden haaste
- Toimitusajan haaste
- Luottamuksen haaste

- Haasteet eivät ole riippumattomia:
voivat olla jopa keskenään ristiriidassa

© Juha Taina, Marko
Salmenkivi ja Kjell
Lemström, 2006-2011

581259 Ohjelmistotuotanto

397

Epäyhtenäisyyden haaste

- Ohjelmistopohjaisten järjestelmien täytyy toimia hyvin erilaisissa ympäristöissä eri järjestelmien kanssa
- Ohjelmistojen pitää myös toimia vanhojen järjestelmien kanssa
- Vanhoihin järjestelmiin täytyy usein kehittää uutta toiminnallisuutta
- Vanhoja järjestelmiä tekehengitetään

© Juha Taina, Marko
Salmenkivi ja Kjell
Lemström, 2006-2011

581259 Ohjelmistotuotanto

398

Toimitusajan haaste

- Ohjelmistojen teko ohjelmistotuotannon menetelmin vaatii paljon aikaa
- Aika on kallis resurssi
- Liiketoiminnassa täytyy olla dynaaminen ja valmis muutoksiin
- Ohjelmistojen pitäisi mukautua muutoksiin nopeasti, mutta laatu ei saisi kärsiä

© Juha Taina, Marko
Salmenkivi ja Kjell
Lemström, 2006-2011

581259 Ohjelmistotuotanto

399

Luottamuksen haaste

- Ohjelmistot vaikuttavat jatkuvasti enemmän elämäämme
- Sitä enemmän ihmisten pitäisi voida luottaa niihin
- Laadunhallinnan kehittäminen siksi jatkuvana haasteena

© Juha Taina, Marko
Salmenkivi ja Kjell
Lemström, 2006-2011

581259 Ohjelmistotuotanto

400