Ohjelmistoarkkitehtuurin suunnittelu

Luento 6

2. osa

Oppimistavoitteet

- Rationaalinen ja empiirinen suunnittelumenetelmä
- ”Walking skeleton”

Miten suunnittelijat suunnittelevat?

RATIONAALINEN JA EMPIIRINEN SUUNNITTELUMENETELMÄ

Suunnittelusta

- Tarkastellaan ohjelmistoarkkitehtuurin suunnittelumenetelmiä eli metodologioita
  - Miten asiantuntija ratkoo suunnitteluongelmia?
  - Millaisia suunnittelutekniikoita tai -strategioita hän käyttää ajattelutyössään?
  - Miten työ etenee ongelman määrittelystä sen ratkaisuun?
- Tämä osuus perustuu pääasiassa seuraaviin lähteisiin

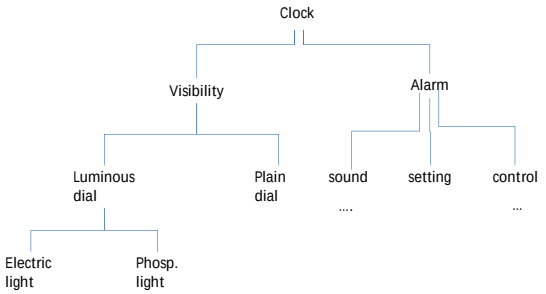
  Brooks, F. P. JR.: *The Design of Design.* Addison Wesley, Pearson Education, 2010.

  Buschmann, F.: Learning from Failure, Part 2: Featuritis, Performitis, and Other Diseases. *Software, IEEE* , vol.27, no.1, pp.10,11, Jan.-Feb. 2010

Suunnittelun rationaalinen menetelmä

- Lähtökohtana on oletus, että suunnitteluongelma ja ongelman ratkaisun vaatimukset, tavoitteet ja rajoitteet voidaan *määritellä riittävän tarkasti etukäteen*
- Itse suunnittelutyö on *etsintää*, jossa järjestelmällisesti
  1) generoidaan mahdollisia ratkaisuja
  2) evaluoidaan niitten arvo
  3) valitaan paras
- Ratkaisu kuvataan puumaisena rakenteena (design tree), jossa kukin solmu vastaa jotain suunnittelupäätöstä ja sen alapuolella oleva puun osa tämän päätöksen jälkeen jäljelle jääviä mahdollisia suunnittelupäätöksiä
  - Kaikki mahdolliset päätöspuut muodostavat yhdessä ratkaisuavaruuden, josta optimaalinen ratkaisu (puu) etsitään

Herätyskellon osittainen päätöspuu



Rationaalinen suunnittelumetodi

- Goal (primääritavoite)
- Desiderata (sekundääriset tavoitteet)
- Utility function (ratkaisun hyödyllisyyden ja hyvyyden mittaava arvofunktio)
- Constraints (budget, resource allocations)
- Design tree decisions
  - UNTIL ("good enough") or (time runs out)
    - DO another design (to improve utility function)
      - UNTIL design is complete // design == tree)
        - » WHILE design remains feasible, make another design decision // constraints
        - » END WHILE
        - » Backtrack up design tree
        - » Explore a path not searched before
      - END UNTIL
    - END DO
    - Take best design
  - END UNTIL

Rationaalinen menetelmä

- Rationalisti uskoo, että saatuaan oikean koulutuksen, kasvattavaa kokemusta ja tekemällä riittävän paljon tarpeeksi huolellista ajatustyötä, suunnittelija voi tehdä virheettömän suunnitelman
- Suunnittelumetodi tähtää virheettömyyden saavuttamiseen suunnitteluvaiheessa

Rationaalisen menetelmän kritiikkiä[1]

- Tavoitteet ja vaatimukset ovat harvoin riittävän tarkasti selvillä etukäteen ja ne muuttuvat
  - "We don't really know the goal when we start"
    - Koskee erityisesti tuotesuunnittelua
- Suunnittelupuun rakenne on ongelmallinen
  - Puun rakennetta ei yleensä tunneta etukäteen, vaan se löytyy suunnittelutyön aikana; päätökset avaavat uusia aiemmin tuntemattomia mahdollisuuksia
  - Kombinatorinen räjähdys: suunnittelupäätösten järjestys voi vaikuttaa radikaalisti puun rakenteeseen; yksittäisillä päätöksillä saattaa riippuvuuksien ja sivuvaikutusten kautta olla globaalia vaikutusta

[1]Brooks, F. P. JR.: *The Design of Design. Addison Wesley, Pearson Education, 2010.*

Rationaalisen menetelmän kritiikkiä

- Hyvyysfunktion (utility function) inkrementaalinen evaluointi ei aina ole mahdollista
  - Sen voi evaluoida vasta kun koko päätöspuu on valmis lehtiä myöten
- Suunnittelijat eivät vain tee työtään näin…
  - Käytännössä suunnittelutyö näyttää oskilloivan eri suunnitteluongelman alueiden ja niiden (osa-) ratkaisujen välillä sekä osaratkaisujen koostamisen välillä (top-down & bottom-up vuorotellen)
  - Aloittelevalle suunnittelijalle rationaalinen malli voi kuitenkin antaa jonkinlaisen lähtöpisteen työhönsä

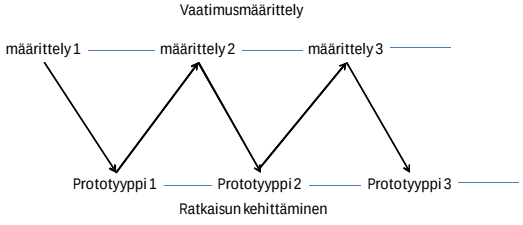
Empiirinen menetelmä[1]

- Ongelmia ei pystytä määrittelemään täysin etukäteen eikä ratkaisuehdotuksia evaluoimaan ilman *konkretiaa*
  - Ihminen *ei pysty* virheettömän suunnitelman tuottamiseen
  - suunnittelun viat löydetään *kokeellisesti*
- *Suunnittelu* ja *toteutus* etenevät rinnakkain (iteroiden)
  - Ratkaistaan ensin joitakin ongelmia
  - Testataan ratkaisujen toimivuus
  - Katsotaan, mihin uusiin avauksiin ja mahdollisuuksiin tehdyt ratkaisut johtavat
- *"Design isn't simply selecting from alternatives, but also realizing their existence"*

[1]Brooks, F. P. JR.: *The Design of Design. Addison Wesley, Pearson Education, 2010.*

Empiirisiä menetelmiä

- Co-evoluutio[1]



[1] Maher M., Tang H.-H.: Co-evolution as a computational and cognitive model of design. Research in Engineering Design Volume 14, Issue 1, 2003, pp 47-64.

Empiirisiä menetelmiä

- Open Source –kehitysmenetelmät (Raymondin ”Bazaar”)
  - Evolutiivinen, ohjelmiston kasvattamiseen pyrkivä malli
  - Kuka tahansa jonkin tarpeen näkevä voi tarjota siihen ratkaisua
  - Avoin kilpailu vaihtoehtoisten ratkaisujen välillä (ála evoluutio)
  - Joukkotestaus, paremman bugikorjaukset
  - Toimii hyvin, kun suunnittelijat ja toteuttajat ovat itse myös käyttäjiä (vaatimukset, ratkaisujen evaluointikriteerit)

Empiirisiä menetelmiä

- Spiral model (Boehm)
  - Käytiin läpi aikaisemmalla luennolla
  - Inkrementaalisuus & kasvattaminen, prototyypit, riskien hallinta

Suunnittelua tiimityönä?

- Brooks korostaa käsitteellisen eheyden (conceptual integrity) merkitystä
  - Sama asia tehdään vain yhdellä ja samalla tavalla eri puolilla järjestelmää
- Hänen mukaansa yhteistoiminnallinen reaaliaikainen (kollaboratiivinen) suunnittelu ei toimi, mutta *parityöskentelyssä* on taikaa
  - Suunnittelu vaatii itsenäistä miettimistä ja keskittymistä
  - Reaaliaikainen kollaboraatio toimii suunnitelmien katselmoinnissa, mutta ei itse suunnittelussa
- Jossain määrin ongelmallinen asia *emergenssiä* korostaville ketterille menetelmille
  - Ei yleensä erillistä arkkitehdin roolia, mutta tarvitaan kuitenkin vahvoja, suunnittelupäätöksiin kykeneviä yksilöitä tiimissä

Esimerkki - ”Walking skeletons”

- Frank Buschmann selostaa empiirisen koulukunnan näkemyksiä noudattavan lähestymistavan ohjelmistoarkkitehtuurin suunnitteluun *Pragmatic Architect* kolumnissaan[1]
  - Tavoitteena erityisesti arkkitehtuurin tarkoituksenmukaisuus ja eräiden ei-toivottujen ilmiöiden välttäminen
  - Ei mene suunnitteluprosessin yksityiskohtiin, mutta antaa suuntaviivat

Buschmann, F.: Learning from Failure, Part 2: Featuritis, Performitis, and Other Diseases. *Software, IEEE* , vol.27, no.1, pp.10,11, Jan.-Feb. 2010

Vältettäviä asioita

- *Featuritis* – toiminnallisuuden toteuttamisen korostaminen laatuominaisuuksien kustannuksella
- *Flexibilitis* – arkkitehtuurin kuormittaminen tarpeettoman laajoilla laajennos-, sovitus- ja konfigurointimekanismeilla
- *Performitis* – projektin loppusuoralla suorituskyky havaitaan huonoksi, mikä johtaa laajamittaiseen paikalliseen optimointiin ja häkkäykseen globaalin strategian puuttuessa
  - ongelmien perimmäisenä syynä usein kaksi edellistä ”tautia”

Luuranko

- Walking skeleton[1] – ohjelmiston *perusarkkitehtuuri* (baseline), joka tukee
  - Tärkeimpien arvoa tuottavien toiminnallisuuksien toteuttamista (business case)
  - Haastavimpien laatuvaatimusten toteuttamista
  - Ohjelmistotuoteperheen rakentamista (variability)
- Nämä ovat *arkkitehtuurin kannalta merkittävimpiä vaatimuksia* (Architecturally Significant Requirements)

[1] http://alistair.cockburn.us/Walking+skeleton

…joka kävelee

- Ketterässä kehityksessä luuranko on suoritettavaa koodia eli se ”kävelee”
- Empirismi
  - Testaus
  - Ominaisuuksien mittaus
  - Vaatimusten validointi

Sovellusalueen merkitys

- Sovellusalueen käsitteet ja ilmiöt ohjaavat merkittävällä tavalla perusarkkitehtuurin suunnittelua
  - Arkkitehtuurissa suunniteltujen komponenttien vastuiden ja yhteistoiminnan pitää heijastaa sovellusalueen käsitteitä
- Tämä mahdollistaa vaaditun toiminnallisuuden järkevän toteuttamisen (esim. riittävät tietomallit)
  - Sovellusalueen paikalliset muutokset jäävät todennäköisemmin paikallisiksi myös ohjelmistossa

Käyttötapaukset

- Koko ohjelmiston ”läpi menevät” (end-2-end) käyttötapaukset ja skenaariot ohjaavat perusarkkitehtuurin suunnittelua kattamaan sovellusalueen eri osat
  - Suppea mutta edustava joukko tärkeimmät suunnitteluhaasteet esiin tuovia käyttötapauksia
  - Laatuvaatimusten tulisi tulla esille skenaarioissa
    - Sovellustoiminnallisuuden suhde laatuvaatimukset toteuttavaan infrastruktuuriin

Suunnittelutyön kulku

- Suunnittelu lähtee liikkeelle arkkitehtuuristen vaatimusten (ASR) kannalta relevanttien käyttötapausten peruskuluista (main success & failure scenarios)
- Vasta kun peruskulut toteuttava arkkitehtuuri on *vakaa ja se toimii*, aletaan ottaa harkiten mukaan näiden variaatioita ja laajennoksia seuraavissa iteraatioissa
  - Piirremalleja voidaan käyttää apuna (myöhemmin kurssilla)
- Luurangon ympärille kasvatetaan lihaksia, ei läskiä
- Sama perusajattelu on nähtävissä RUP-prosessikehyksessä, jossa *elaboration* -vaihe tuottaa ”suoritettavan” arkkitehtuurin, eli järjestelmän todistettavasti toimivan luurangon (tosin se voi olla vain malli)

Yhteenveto

- Rationaalinen malli ei käytännössä yleensä toimi
  - Voi soveltua kuitenkin rajattuihin, matemaattisessa mielessä hyvin määriteltyihin ongelmiin
- Empiiriset menetelmät (iteroivat ja evolutiiviset) sopivat paremmin reaalimaailman projekteihin, joissa erehtyväiset ihmiset toimivat epätäydellisen tiedon varassa
  - Vuorottelu ongelma- ja ratkaisuavaruuden välillä, prototyypit ja validointi
  - Ratkaisujen rakentaminen mahdollista sekä *top-down* (osittamalla, divide & conquer) että *bottom-up* (osaratkaisuista kokoamalla)