Ohjelmistoarkkitehtuurit

Arkkitehtuurin arviointi

Luento 13

Oppimistavoitteet

- Arkkitehtuurin arviointi
  - Mitä, miksi, milloin?
- Arviointimenetelmiä
  - Skenaariopohjaiset menetelmät, suunnittelupäätöksiin kohdistuva menetelmä
- ATAM
- Ketterä arkkitehtuurin arviointi

ARKKITEHTUURIN ARVIOINTI

Arkkitehtuurin arviointi (architectural analysis)

- Arkkitehtuurin arvioinnin tarkoituksena on muodostaa perusteltu käsitys kehitettävän ohjelmiston tärkeistä ominaisuuksista
  - Arkkitehtuuri = tärkeimmät suunnittelupäätökset
  - Nyt ja *tulevaisuudessa*
- Arvioinnin rooli korostuu *suunnitelmallisuutta vaativissa kehitysprojekteissa*, joihin liittyy huomattavia riskejä
  - Kaikki keinot käyttöön riskien minimoimiseksi
- Voi olla myös osa ohjelmiston laajempaa evaluointia
  - kun ollaan hankkimassa ohjelmistoratkaisua
  - kun olemassa oleva ohjelmisto tulee elinkaarensa johonkin käännekohtaan

Arkkitehtuurin arviointi (architectural analysis)

- Täsmällisemmin tavoitteina voi olla saavuttaa
  - aikainen arvio järjestelmän koosta, monimutkaisuudesta ja kustannuksista
  - käsitys ohjelmistoon kohdistuvien vaatimusten (sekä toiminnallisten että erityisesti laadullisten) toteutumisesta
  - käsitys suunnitteluohjeiden ja sääntöjen noudattamisesta
  - käsitys dokumenttien ja toteutuksen vastaavuudesta, jne…
- Arvioinnin kohteena
  - Komponentit ja niiden väliset kytkennät
  - Järjestelmä kokonaisuutena ja osajärjestelmät
  - Tiedot ja tiedonkulku osajärjestelmien välillä
  - Arkkitehtuurit
    - *Vaihtoehtoiset / suhde referenssiarkkitehtuuriin*

Arkkitehtuurin arviointi

- Arvioinnissa päähuomio on laadullisissa vaatimuksissa
- Jos jotain laadullista ominaisuutta ei voida arkkitehtuurin (kuvauksen) perusteella arvioida, arkkitehtuuri(kuvaus) on tältä osin puutteellinen
- Poikkeuksia
  - Käytettävyys riippuu pitkälti käyttöliittymän yksityiskohdista, jotka eivät näy arkkitehtuurikuvauksessa
  - Toteutustason tietorakenne- ja algoritmiratkaisut vaikuttavat suorituskykyyn
- Arkkitehtuurin on kuitenkin mahdollistettava laatuvaatimusten täyttyminen tunnettujen toteutustapojen puitteissa

© Harri Laine, A-P Tuovinen

Arkkitehtuurin arviointi

- Arvioitavat yleiset laatuvaatimukset on aina täsmennettävä ja priorisoitava
  - esim. muunneltavuus → minkä asioiden suhteen järjestelmän tulee olla muunneltava
  - esim. suorituskykyvaatimusten priorisointi keskenään ja suhteessa muunneltavuusvaatimuksiin
- Myös toiminnallisia vaatimuksia voidaan arvioida arkkitehtuurin pohjalta
  - Ovatko halutut toiminnot toteutettavissa arkkitehtuurin komponenttien välisellä vuorovaikutuksella?

Arkkitehtuurin arviointi

- Arviointi auttaa *ennustamaan* järjestelmän evoluutiota ja ylläpitokustannuksia
  - Hyödyllistä tietoa johdolle resursointia varten
- Arviointi auttaa suunnittelijoita *ymmärtämään* järjestelmää paremmin ja näkemään sen ongelmakohtia
  - Hyödyllistä myös jälkikäteen tehtynä
  - Auttaa jakamaan hiljaista tietoa ratkaisujen perusteluista, jota ei ole ehkä koskaan kirjattu mihinkään
  - Arviointi voi toimia koulutuksena uusille tekijöille

Arkkitehtuurin arviointi

- Arkkitehtuurin arviointi osana ohjelmistoprosessia
  - Pitäisi olla rutiininomainen osa normaalia ohjelmistokehitysprosessia (riskiarvion perusteella)
    - *vaativuudeltaan erilaisia arviointeja projektin ja riskien mukaan*
  - Arkkitehtuuri sisältää kriittisimmät suunnitteluratkaisut
    - *arviointi erityisen tärkeää*
  - Arkkitehtuurikuvaus on järjestelmän ensimmäinen arvioinnin mahdollistava kuvaus
    - *havaitut puutteet pystytään vielä korjaamaan suhteellisen vähillä kustannuksilla*
  - Arviointi on edullinen tapa testata arkkitehtuuri
    - *prototyypin rakentaminen on kalliimpaa (entä agile, Walking Skeleton?)*
    - *arkkitehtuurin testaus vasta järjestelmätestauksen yhteydessä on kallein tapa*

Arkkitehtuurin arviointi

- Arkkitehtuurin arvioinnin tulokset ovat aina suhteessa arkkitehtuurikuvauksen (arkkitehtuurin mallin) laatuun: jos kuvaus/malli on epätarkka, myös arvioinnin tulokset ovat epätarkkoja
- Erilaisia arkkitehtuuriarviointeja
  - Arvioidaan heti ensimmäinen arkkitehtuurin luonnos
  - Arvioidaan valmis arkkitehtuurikuvaus
  - Arvioidaan valmiin järjestelmän arkkitehtuuri

Arkkitehtuurin arviointi

- Arvioidaan heti ensimmäinen arkkitehtuurin luonnos
  - Luonnokselle vain pienimuotoinen arviointi
  - Analysoidaan vaatimukset arkkitehtuurin kannalta: voiko mikään arkkitehtuuri täyttää vaatimukset
  - Tuloksena realistinen, priorisoitu vaatimusjoukko sekä alustava arkkitehtuuriratkaisu niiden täyttämiseksi
  - Alustavassa arkkitehtuurin arvioinnissa tulisi olla läsnä henkilöitä, joilla on valtuus määrätä vaatimuksista

Arkkitehtuurin arviointi

- Arvioidaan valmis arkkitehtuurikuvaus (malli)
  - Tähän kannattaa käyttää enemmän resursseja
  - Tulisi tapahtua ennen kuin toteutus aloitetaan, koska arvioinnin tuloksena arkkitehtuuri saattaa muuttua
    - *Ongelma: toteutus on resursoinnin ja aikataulun takia aloitettava usein jo arkkitehtuurityön aikana*
    - *Ongelma: BDUF – kuinka toteuttamiskelpoinen ennakkosuunnittelun tuottama arkkitehtuuri lopultakaan on?*
    - *Osittaisarviointi*

Arkkitehtuurin arviointi

- Arvioidaan valmiin järjestelmän arkkitehtuuri
  - Voi olla järkevää, jos halutaan esim. lisätä tietoa järjestelmän laatuattribuuteista (esim. onko järjestelmä siirrettävissä uuteen ympäristöön)
    - *Osana jotain suurempaa muutosta*
    - *Koulutustarkoituksessa tai tiedon jakamiseksi tehty arviointi*
  - Jos ei ole ajan tasalla olevaa arkkitehtuurikuvausta, sellainen voidaan tuottaa (osittain) automaattisesti takaisinmallinnustyökalujen avulla

Arkkitehtuurin arviointi

- Kysymyksiä:
  - Sopiiko suunniteltu arkkitehtuuri järjestelmälle?
    - *Järjestelmän (laatu)vaatimusten täyttyminen tietyllä arkkitehtuurilla*
    - *Jos ei sovi, niin vastauksen pitäisi kertoa myös miksi*
  - Mikä vaihtoehtoisista arkkitehtuureista sopii parhaiten järjestelmälle ja miksi?
    - *Arkkitehtuurille on esitetty useita vaihtoehtoja*
    - *Vastaus kertoo, mikä on paras ja miksi*
  - Miten hyvä tulee olemaan järjestelmän jokin tietty laadullinen ominaisuus, olettaen järkevä toteutustapa?
    - *Määrälliset arviot joistain laatuominaisuuksista*
    - *Vastaus voi esimerkiksi kertoa, kuinka kalliiksi järjestelmän ylläpito todennäköisesti tulee*

ARVIOINTIMENETELMÄT

Arviointimenetelmät

- Arvioinnin perusongelma:
  - tulevan järjestelmän tiettyyn laatuominaisuuteen vaikuttavien kohtien (eli suunnittelupäätösten) poimiminen arkkitehtuurin kuvauksesta
- Asiantuntija-arvio on yleisin tapa tehdä arviointi:
  - tulos riippuu täysin arvioijan kokemuksesta ja taidoista
  - Siksi on kehitetty menetelmiä, jotka ainakin periaatteessa ovat yksilön mielipidettä objektiivisempia

Arviointimenetelmät

- Eri tyyppistä arviointia
  - Arkkitehtuuriin tai arkkitehtuurin suunnitteluprosessiin liittyvät tarkistuslistat ja kysymyslomakkeet
    - *esim. "onko käyttöliittymä erotettu sovelluslogiikasta?"*
    - *yleisiä tai sovellusaluekohtaisia*
    - *voidaan yhdistää skenaariopohjaisiin menetelmiin*
  - Mallipohjainen arviointi
    - *Edellyttää täsmällisiä yksityiskohtaisia malleja (tarkasteltavan ominaisuuden suhteen)*
    - *Sopii huonosti laajojen kokonaisuuksien arviointiin*
    - *Staattisiin ominaisuuksiin perustuva mallipohjainen arviointi*
      - *Sopii erityisesti sisäisten yksittäisten rakenteellisten piirteiden arviointiin: syntaktinen oikeellisuus, tyylin noudattaminen, lukkiutumattomuus, kuormitus, jne (tila-automaatit, jonoverkot, riippuvuusmatriisit, ...)*
    - *Dynaamisten ominaisuuksien simulointipohjainen arviointi*
      - *Sopii yksittäisten käyttäytymispiirteiden arviointiin*

Arviointimenetelmät

- Skenaarioihin perustuva arviointi
  - *Esitetään **konkreettisia** esimerkkitilanteita, joissa tietyt laatuominaisuudet tulevat esiin*
  - *Tutkitaan, miten arkkitehtuuri sopii kyseiseen skenaarioon*
  - *Etuna skenaarioiden suhteellisen helppo löytäminen, konkreettisuus ja ymmärrettävyys*
  - *Pyritään saamaan kaikki sidosryhmät osallistumaan skenaarioiden määrittelyyn omasta näkökulmastaan*

Arviointimenetelmät

- Skenaarion luonne riippuu sitä, mitä laatuominaisuutta halutaan tarkastella
  - *Jos tarkastellaan muutettavuutta, skenaario käsittelee jotakin muutostarvetta järjestelmän evoluutiossa*
  - *Jos tarkastellaan suorituskykyä, skenaario käsittelee jotakin suorituskykykriittistä tilannetta järjestelmän käytössä (tämä on lähellä käyttötapauksen käsitettä)*
  - *Jos tarkastellaan turvallisuutta, skenaario käsittelee jotakin uhkatilannetta järjestelmän käytössä*
  - *Jos tarkastellaan uudelleenkäytettävyyttä, skenaario käsittelee jonkin uuden sovelluksen tekemistä uudelleen käyttämällä järjestelmää (= tuoterunko)*
  - …

Arviointimenetelmät

- Skenaariopohjaisia arviointimenetelmiä
  - SAAM (Software Architecture Analysis Method)
    - *Varhaisimpia skenaariopohjaisia menetelmiä*
    - *Kehitetty SEI:ssä (Software Engineering Institute, Carnegie-Mellon University)*
    - *Tarkoitettu lähinnä muunneltavuuteen liittyvien laatuominaisuuksien arviointiin, mutta sillä voidaan arvioida myös toiminnallisuutta*
  - ATAM (Architecture Tradeoff Analysis Method)
    - *SAAM-menetelmästä edelleen SEI:ssä johdettu yleisempi menetelmä myös muihin laatuominaisuuksiin*
  - MPM (Maintenance Prediction Method)
    - *Jan Boschin kehittämä ylläpidettävyyteen keskittyvä arviointimenetelmä*

Arviointimenetelmät

- Suunnittelupäätöksiin perustuva arviointi
  - DCAR – Decision Centric Architecture Review http://www.dcar-evaluation.com/
  - Kevyempi menetelmä skenaariopohjaisiin arviointeihin verrattuna
  - Tiivis, yhden päivän kestävä istunto; pähkinänkuoressa:
    1. *Liiketoiminnan tavoitteiden, rajoitteiden ja tärkeimpien vaatimusten läpikäynti suunnitteluun vaikuttavien voimien tunnistamiseksi (architectural drivers)*
    2. *Arkkitehtuurin läpikäynti*
    3. *Suunnittelupäätösten tunnistaminen ja priorisointi*
    4. *Tärkeimpien suunnittelupäätösten dokumentointi*
    5. *Dokumentoitujen suunnittelupäätösten arviointi: mahdolliset riskit ja muut huomioitavat asiat sekä äänestys (liikennevalot)*
    6. *Retrospektio ja raportointi*

ATAM -ARVIOINTI

ATAM

- Architectural trade-off analysis
  - Rick Kazman, Mark Klein, Paul Clements: **ATAM: Method for Architecture Evaluation** (http://www.sei.cmu.edu/reports/00tr004.pdf)
- Ensisijaisesti riskien kartoitukseen kehityksen alkuvaiheessa
- Keskittyy laadullisiin ominaisuuksiin
- Määrittelee prosessin ja käytännöt
- Prosessi luonteeltaan formaalin katselmuksen kaltainen
  - Lähtökohtana (liike)toimintatavoitteet ja järjestelmän arkkitehtuuri
  - Suorittajana arviointiryhmä, jossa eri sidosryhmien edustaja

ATM-prosessi

- ***Vaihe 1:***
  - ***Esittelyosiossa*** käydään läpi ATAM-menetelmä, järjestelmän arkkitehtuuri sekä liiketoimintatavoitteet
  - ***Analyysiosiossa***
    - *etsitään laatuominaisuuksiin vaikuttavat arkkitehtuuriratkaisut*
    - *analysoidaan järjestelmän laatuvaatimukset*
    - *kuvataan laatuvaatimuksiin liittyvät skenaariot*
    - *identifioidaan skenaarioiden avulla arkkitehtuurin kriittiset kohdat*
- Vaihe 2:
  - ***Testausosiossa*** täydennetään skenaarioita järjestelmän käyttäjien näkökulmasta ja analysoidaan arkkitehtuuri uutta skenaariojoukkoa vasten
  - ***Raportointiosiossa*** esitetään arvioinnin tulokset

ATAM-prosessi

- Esittely- ja analyysiosiot muodostavat menetelmän ensimmäisen vaiheen, johon osallistuvat arviointiryhmän lisäksi tuoteprojektin vastuuhenkilöt
- Testaus- ja raportointiosio muodostavat arvioinnin toisen vaiheen, johon osallistuu ensimmäisen vaiheen osallistujien lisäksi muiden sidosryhmien edustajia (esim. asiakkaan)
- Koko prosessi kestää noin kolme päivää
  - Kahdessa jaksossa (1+2), välissä parin viikon tauko
- Toisen vaiheen aluksi käytetään päivä ensimmäisen vaiheen tulosten kertaamiseen uusille osallistujille

ATAM-prosessi

- Esittelyosio:
  - (1) ATAM-menetelmän esittely (arvioinnin johtaja selostaa)
    - *Arviointiprosessi, roolitus, käytettävät tekniikat ja tulosten muoto*
    - *Tärkeää: kaikille realistiset odotukset prosessista*
  - (2) Järjestelmän olennaiset vaatimukset, liiketoiminta-tavoitteet ja toimintaympäristö (projektipäällikkö)
  - (3) Arkkitehtuuri, sen tekninen toimintaympäristö ja rajapinnat muihin järjestelmiin (pääarkkitehti)
    - *Käytetään näkymiä*
    - *Erityisesti arkkitehtuurin kannalta keskeiset näkökulmat*
      - *Tulisi olla esiteltävissä tunnissa*

ATAM-prosessi

- Analyysiosa:
- (1) Arkkitehtuuriratkaisujen ja niiden laatuvaikutusten tunnistus
  - arkkitehtuurityylit ja patternit listataan

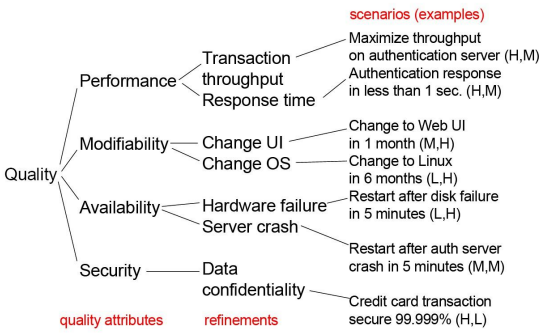
ATAM-prosessi

- Analyysiosa:
- (2) Skenaarioiden määrittely
  - Eri tyyppisiä skenaarioita riippuen siitä, mitä laatuominaisuutta tarkastellaan
    - *Käyttötapaukset*
      - *Kuinka käyttäjät käyttävät*
    - *Kasvuskenaariot*
      - *Kuvaavat suunniteltuja tai ennakoitavia muutoksia ohjelmistossa*
    - *Rajoja kartoittavat skenaariot*
      - *Etsivät järjestelmän rajoja: esim, kuormitusmuutoksia, alustan vaihto, palvelimen rikkoutuminen*
    - *Kaikkiin skenaarioihin voidaan liittää suoritusaika/-määrätavoitteita*

ATAM-prosessi

- ***Laatupuun*** (utility tree) laatiminen jäsentämään skenaarioita (ks. Seuraava kuva)
  - Puussa kuvataan laatuattribuutit ja niiden jako alikohtiin
- Lehdiksi ***skenaarioita***, jotka testaavat kyseistä ominaisuutta
  - *Skenaariot painotetaan tärkeydellä ja vaikeudella*
  - *Asteikkona esim (H=high, M=medium, L=low)*
    - *(painotus kyettävä perustelemaan)*
  - *Painopari esim (H,L)*
- Laatupuu on järjestelmäkohtainen, vaikka samantyyppisillä järjestelmillä on yleensä samantyyppiset laatupuut
  - Eri järjestelmissä voidaan tarkastella eri osajoukkoja eri painotuksin (esim. ylläpidettävyys vs. siirrettävyys)
    - *painot puun haaroille*

Laatupuu



ATAM

- (3) Painotetun laatupuun & skenaarioiden liittäminen arkkitehtuuriratkaisuihin
  - Skenaarioihin liitetään niitä tukevat arkkitehtuuriratkaisut
    - *Tarkastelu aloitetaan tärkeimmistä ja vaikeimmista (H,H)-luokka, vähempiarvoiset saattavat jäädä vähäiselle käsittelylle*
    - *Arkkitehtuurityylit ja muut skenaarioon vaikuttavat ratkaisut*
  - Identifioidaan ***riskit***, ***turvalliset ratkaisut***, ***herkkyyskohdat*** ja ***tasapainottelukohdat*** puun avulla
    - *Tunnistuksen perustana laatuattribuuttikohtaiset kysymykset*

ATAM

- *Riski (risk)*
  - Suunnittelupäätös tai arkkitehtuurin piirre, joka voi johtaa jonkin laatuominaisuuden huononemiseen
  - Kuvataan kertomalla kyseinen päätös/ominaisuus, siitä mahdollisesti aiheutuva laatuongelma ja ongelman syy
- *Turvallinen ratkaisu (non-risk)*
  - Järjestelmän laatuominaisuuksia parantava ja arkkitehtuurissa voimassa oleviin olettamuksiin perustuva päätös (jos arkkitehtuurioletukset muuttuvat, ratkaisu ei välttämättä pysy turvallisena)
  - Kuvataan kertomalla taustaoletukset, suunnittelupäätös, siitä seuraavat laatuedut sekä syyt näille seurauksille

ATAM

- *Herkkyyskohta (sensitivity point)*
  - Suunnittelupäätös tai arkkitehtuurin piirre, joka on kriittinen jonkin laatuominaisuuden kannalta
  - Jos tätä ominaisuutta muutetaan tai päätöksestä luovutaan, jokin laatuominaisuus on vaarassa huonontua
  - Miksi järjestelmä saavuttaa tietyn laatuominaisuuden?
    - *Kuvataan kertomalla kyseinen suunnittelupäätös ja siitä riippuva laatuominaisuus*

ATAM

- *Tasapainottelukohta (trade off point)*
  - Herkkyyskohta, joka vaikuttaa useampaan kuin yhteen laatuominaisuuteen (esim. edullisesti johonkin ja epäedullisesti toiseen)
  - Esim. jonkin patternin soveltaminen voi lisätä muunneltavuutta, mutta heikentää suorituskykyä
  - Kuvataan selostamalla kyseinen suunnittelupäätös ja sen vaikutus laatuominaisuuksiin

ATAM

- Testausosio:
  - (1) Muut kiinnostusryhmät kuin arkkitehdit tai arviointiryhmä (esim. testaajat, ylläpitäjät, asiakkaat, johto) ideoivat skenaarioita omista lähtökohdistaan
    - *Esim. käyttötapauksia, odotettavissa olevia muutoksia, …*
    - *Skenaariot priorisoidaan esim. äänestämällä*
    - *Tavoitteena on herättää keskustelua eri kiinnostusryhmien välille ja saavuttaa yhteinen näkemys laatuominaisuuksista*

ATAM

- Testausosio jatkuu:
  - (2) Skenaarioita verrataan jo löydettyihin skenaarioihin
    - *Jos nämä täsmäävät, kaikki on hyvin*
    - *Jos skenaariota ei löydy puusta, mutta sille on puussa sopiva laatuhaara, siitä tulee uusi lehti puuhun (tai useita, jos se liittyy useisiin laatunäkökohtiin)*
    - *Jos skenaariolle ei ole sopivaa haaraa puussa, se joko ei liity lainkaan laatuun tai se liittyy uuteen laatuominaisuuteen*
    - *Jälkimmäisessä tapauksessa puuhun lisätään uusi haara, johon skenaario liitetään lehdeksi (tällainen skenaario paljastaa yleensä järjestelmästä uusia riskejä)*

ATAM

- Raportointi:
  - ATAM-prosessin tulokset esitetään arvioinnin lopuksi koko arviointiin osallistuneelle ryhmälle ja laaditaan arviointiraportti
  - Raportti voidaan organisoida esim. skenaarioittain
    - *skenaariota tukevat arkkitehtuuriratkaisut*
    - *skenaarioon liittyvät riskikohdat, turvalliset ratkaisut, herkkyyskohdat ja tasapainokohdat (listataan erikseen)*
    - *laatuominaisuudet, joita skenaario testaa*
    - *kuvaus olosuhteista, joissa skenaario tapahtuu*

ATAM

- Raportointi:
  - Saadaan tietoa, joka auttaa ymmärtämään arkkitehtuuria ja hallitsemaan sen riskejä: miten arkkitehtuuri liittyy laatuun
  - Tuloksena voi joskus tulla myös parannusehdotuksia arkkitehtuuriin (ei kuitenkaan ole ATAM:in varsinainen tavoite)
  - Arvioinnin tuloksiin kootaan myös nk. riskiteemoja
    - *riskien ryhmiä, joiden taustalla on sama yleinen syy*
    - *esim. tiedon häviäminen laitteisto- tai ohjelmistohäiriöissä → riskiteema "riittämätön tiedon varmistus"*
    - *Tarkoituksena on liittää riskit liiketoimintatavoitteisiin ja tehdä ne ymmärrettävämmiksi johdolle*

Arvioinnin ongelmia

- Arvioinnin raskaus pienille järjestelmille & organisaatioille
  - Hyvien skenaarioiden laatiminen voi osoittautua hankalaksi!
  - Voidaan tehdä osittainen tai kevennetty katselmointi
  - Tarkastuslista keskeisimmistä arkkitehtuurilta vaadittavista ominaisuuksista arvioinnin minimivaatimus
- Arvioinnin ajoitus
  - Aikaisessa vaiheessa arviointi voi tuntua tarpeettomalta, sillä järjestelmästä tiedetään suhteellisen vähän
  - Toteutusvaiheen alkaessa arviointi voidaan nähdä uhkana järjestelmän toteutusaikataululle (löydettyjen vikojen korjaaminen!)
  - Toteutuksen jälkeinen arviointi puolestaan voidaan nähdä turhana, koska järjestelmä on jo toteutettu ja toimii
  - Näistä syistä arviointi pitäisi kiinnittää johonkin ohjelmistoprosessin etappiin

Arvioinnin ongelmia

- Usein kuullun huomion mukaan arkkitehdit tietävät arkkitehtuurin ongelmista jo etukäteen, ja siksi arviointi voi vaikuttaa turhalta
- Tällöin korostuu arvioinnin rooli tiedon välityksen työkaluna
  - Arviointia voi esim. hyödyntää opetustilaisuutena kutsumalla arviointiin mukaan ohjelmistosuunnittelijat, jotka lopulta toteuttavat arkkitehtuurin mukaisen järjestelmän
  - Tietämyksen siirto pitkään käytössä olleen ja jatkuvasti kehittyvän järjestelmän eri "kehittäjäsukupolvien" välillä (jatkuvuus!)
- Suunnittelupäätöksiin perustuva arviointi on prosessina kevyempi ja se keskittyy tiedettyihin konkreettisiin asioihin (tehtyihin päätöksiin) ja niiden seurauksiin
  - Voi olla helpompi lähtökohta osallistujille kuin laatuskenaarioiden (vaatimusten) riittävän tarkalle tasolle viety määrittely

KETTERÄ ARKKITEHTUURIN ARVIOINTI

Ketterä arviointi

- Lähde: Veli-Pekka Eloranta, **Techniques and Practices for Software Architecture Work in Agile Software Development**. PhD Thesis, Tampere University of Technology, Pub. #1293, 2015.
  - http://dspace.cc.tut.fi/dpub/handle/123456789/22918
- Väitöskirjassa kehitellään mm. kevyitä menetelmiä arkkitehtuuritietämyksen hallintaan ketterässä kehityksessä (VK:n 9. Luku, Kai Koskimies artikkelin toinen kirjoittaja)
- Avainroolissa ovat DCAR-tyyppiset arkkitehtuurin arvioinnit

Ketterä arviointi

- Eloranta ja Koskimies ovat osallistuneet pariin kymmeneen kattavaan ATAM-tyyppiseen arkkitehtuurin arviointiin
  - Yritysten edustajat pitivät tyypillisesti tärkeimpänä arvioinnin antina organisaatiossa lisääntynyttä arkkitehtuuritietämystä
- Arvioinnin aikana nousee esiin suuri määrä tietoa
  - Suunnittelupäätöksiä, liiketoimintatavoitteita, arkkitehuuriin vaikuttavia vaatimuksia ja rajoitteita jne.
  - 'Hiljaista tietoa', joka tyypillisesti tulee esiin arviointikeskusteluissa
- Tieto on koodattava helpolla ja lukijoille käyttökelpoisella tavalla (lightweight Architecture Knowledge Management)
  - Tarjottava eri sidosryhmille heidän tarpeidensa mukainen näkymä tietoon
  - Dokumentaation on synnyttävä lähes automaattisesti, kehitystyön sivuvaikutuksena, ilman huomattavaa lisätyötä

Ketterä arviointi

- ATAM on kuitenkin melko raskas menetelmä
  - Sidosryhmien edustajien kokoaminen useaksi päiväksi samaan paikkaan on yleensä vaikeaa
  - Skenaarioiden kehittely vie suhteellisesti paljon aikaa, eikä suurinta osaa skenaarioista kuitenkaan käytetä (priorisointi)
- Skenaarioissa on kysymys vaatimusten täsmentämisestä ja keskustelu kiertyy usein järjestelmän tarkoitukseen
  - Vaatimusten validointi – Are we building the RIGHT system?
  - Ketterässä kehityksessä taas validointi on jo muutenkin jatkuvaa (lyhyet iteraatiot, inkrementaalinen kehitys, walking skeleton)

Ketterä arviointi

- ATAM:ia ei ole tarkoitettu jatkuvaan arviointiin vaan kerran suoritettavaksi
  - Mutta milloin ketterässä projektissa on 'oikea aika' arvioinnin suorittamiseksi?
  - *Iteraation aikaisten arkkitehtuuripäätösten arviointi* on luontevampi tulokulma ketterässä projektissa, jossa muutenkin tehdään retrospektiota
- DCAR sopii siis paremmin ketterän projektin arviointityypiksi
  - Vaatii tyypillisesti puoli päivää ja noin 15-20 henkilötyötuntia kaiken kaikkiaan
  - Voidaan tehdä inkrementaalisesti, jos käydään läpi suunnittelupäätösten jokin osajoukko kerrallaan
  - Vaatii vain päätöksiin osallisten läsnäolon
  - Suunnittelupäätösten välisten riippuvuuksien kirjaaminen näkyviin tukee emergenssiä

Ketterä arviointi

- Eloranta ja Koskimies tekivät kyselytutkimuksen arkkitehtuuri-informaation tuottamisesta ja käyttämisestä Scrum-projekteissa teollisuudessa (kaksi yritystä, yhteensä kolme tiimiä)
- Käytössä oli kolme eri tapaa arkkitehtuurityöhön
  - BDUF architecting
  - Sprint-Zero architecting
  - In-Sprints architecting
  - (Separated architecture team ei ollut käytössä)

Ketterä arviointi

- Kyselyn tuottaman tiedon perusteella Eloranta ja Koskimies esittelevät kaksi tapaa käyttää arviointeja tuottamaan ajantasaista arkkitehtuuritietämystä
  - BDUF / Sprint-Zero architecting
    - *0-Sprintin yhteydessä ATAM + DCAR osana retrospektiivia niissä sprinteissä, joissa on tehty uusia/muutettu tehtyjä arkkitehtuuripäätöksiä*
  - In-Sprint architecting
    - *DCAR osana jokaisen sprintin retrospektiivia*
- Tärkeässä roolissa on AIR – keskitetty arkkitehtuuritietämyksen tietovarasto, johon arviointien tuottama tieto+dokumentaatio automaattisesti tallentuu arvioinnissa käytetyn kirjanpitotyökalun kautta

"Lightweight AKM manifesto"

1. "The producing and consuming of architectural information in AKM should not require extra effort. There should be no activities that are related only to AKM itself during the software system lifecycle.
2. AKM should be invisible from the viewpoint of information producers and consumers. The producers and consumers should not need to be aware of AKM. AKM should be seamlessly integrated with usual knowledge-sharing activities like documentation, reviews and project meetings.
3. Only potentially useful architectural information should be stored in AKM. All stored information is expected to be used in the context of a probable development or evolution scenario. The burden of useless information surpasses the possible benefit of coincidental usage."

Veli-Pekka Eloranta, **Techniques and Practices for Software Architecture Work in Agile Software Development**. PhD Thesis, Tampere University of Technology, Pub. #1293, 2015.
http://dspace.cc.tut.fi/dpub/handle/123456789/22918