Arkkitehtuurikeskeisyys sovelluskehitysprosessissa

Jouko Poutanen,
Rational Software Finland Oy

Taustaa

Sovelluskehitys on muuttumassa entistäkin haastellisemmaksi. Sovellusten rooli on muuttunut yritysten liiketoimintaa tukevasta roolista keskeiseksi menestystekijäksi.

Sovelluksia on nykyisin kaikkialla. Riippumatta sovelluskehityksen luonteesta - kehitetäänkö sulautettuja, yrityksen sisäisiä tai kaupallisia järjestelmiä – kohdataan sama haaste: entistä kompleksisempi järjestelmä on kehitettävä nopeammin ja samalla täytettävä korkeat laatuvaatimukset. Lisäksi itse liiketoiminta muuttuu nopeammalla syklillä – nämä muutokset on huomioitava luonnollisesti myös toimintaa tukevissa tietojärjestelmissä.

Miten voimme vastata näihin kasvaviin haasteisiin? Keskeisessä roolissa on käytettävä sovelluskehitysprosessi, ihmiset ja heidän osaamisensa sekä prosessia tukevat työkalut. Edelläkuvatuista haasteista on johdettavissa kehitysprosessilta vaadittavia ominaisuuksia:

- systemaattinen vaatimusten hallinta
- muutosten hallinta
- jatkuva laadunvarmistus
- arkkkitehtuurikeskeisyys ja komponenttiarkkitehtuurien käyttö
- visuaalinen mallinnus

Tyypillisesti sovelluksen elinkaaresta yli 70% on tuotantokäyttö- ja ylläpitovaihetta. Tänä pitkänä aikana ilmenee kuinka ylläpidettävä ja laajennettava sovellus todella on - ja kuinka suuriksi sen kokonaiskustannukset muodostuvat. Sovellukset on siis rakennettava muutossietoisiksi ja laajennettaviksi.

Kun tarkastellaan muutossietoisuutta kehitetyn sovelluksen kannalta, keskeiseen osaan nousee sovellusarkkitehtuuri: millainen on sovelluksen sisäinen rakenne ja rakenne-elementtien välinen toiminnallinen vastuunjako. Laajoissa järjestelmissä on lukuisia 'liikkuvia osia', jolloin yksittäisten järjestelmäosien (luokat, komponentit) onnistunut suunnittelu ei riitä, vaan niistä on muodostettava selkeitä kokonaisuuksia. Juuri tämä on sovellusarkkitehtuurin tehtävä.

Mitä sovellusarkkitehtuuri on?

Sovellusarkkitehtuuri (myöhemmin lyhyesti arkkitehtuuri) käsittää joukon strategisia suunnittelupäätöksiä joiden mukaan sovelluksen sisäinen rakenne määräytyy. Näitä päätöksiä ovat:

- niiden keskeisten rakenteellisten elementtien ja rajapintojen valinta joista sovellus muodostetaan
- toiminnallisuuden määrittäminen edellämainittujen elementtien välisenä vuorovaikutuksena
- keskeisten elementtien kokoamisen laajemmiksi osajärjestelmiksi
- kokonaisuuden muotoutumista ohjaavan arkkitehtuurityylin valinta.

Nämä strategiset päätökset ohjaavat ja rajoittavat tarkempaa suunnittelua ja sitä kautta myös toteutusta. Toteutus muotoutuu arkkitehtuurin mukaiseksi. Strategisten linjausten muuttaminen toteutuksen aikana lisää merkittävästi ylimääräistä työtä ja kustannuksia.

Arkkitehtuuri on siis muutakin kuin vain muoto johon suuunitteluelementit on järjestetty. Arkkitehtuuri sisältää myös elementtien välisen vuorovaikutuksen kuvaamisen, sekä ennenkaikkea *perustelut* tehdyille päätöksille. Arkkitehtuuri voitaisiin kiteyttää yhtälöön: *Arkkitehtuuri = elementit + muoto + perustelut*. Perustelut ovat oleellisia arvioitaessa tehtyjen ratkaisujen kestävyyttä. Perusteluissa arkkitehtuuri on liitettävä siihen ympäristöön, jossa sovellus tulee toimimaan elinkaarensa aikana.

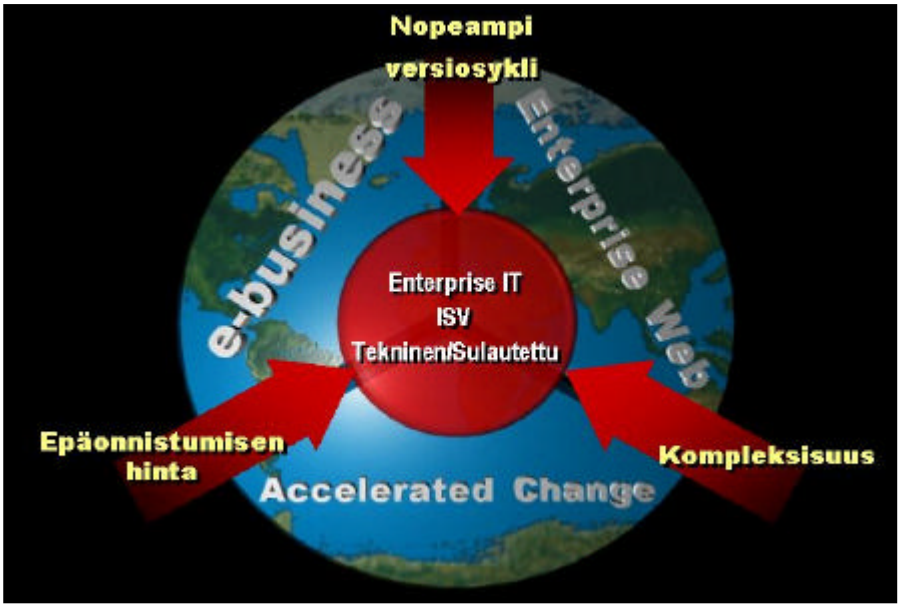
Arkkitehtuurityön yksi tehtävä on useiden vastakkain vaikuttavien tekijöiden huomioiminen ja tasapainoittaminen. Esimerkiksi sovelluksen kompleksisuus vaikeuttaa korkeaan laatuun yltämistä. Kehitystiimin arkkitehtuuri- ja suunnittelutaidot saattavat rajata käytettävissä olevaa teknologiaa, jne.

Arkkitehtuurikeskeinen prosessi

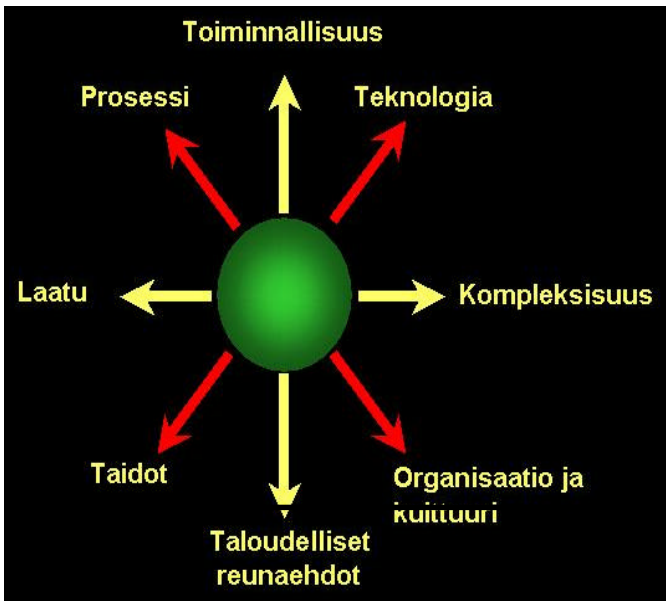
Arkkitehtuurikeiskeinen prosessi huomioi ja tunnnustaa arkkitehtuurin keskeisen merkityksen onnistuneessa sovelluskehityksessä. Arkkitehtuurikeskeisyys ei käsitä pelkkiä suunnitteluperiaatteita vaan arkkitehtuuri huomioidaan kokonaisuutena johon kuuluvat mm. työn sijoittaminen kehityselinkaarelle, lopputuotteet sekä tarvittavat roolit ja vastuut.

Arkkitehtuurikeskein prosessi etenee yleensä seuraavin vaihein ja periaattein:

- sovelluksen haluttu toiminnallisuus kuvataan joukolla skenaarioita
- rakennetaan ja testataan arkkitehtuuri, joka käyttää hyväkseen edelläkuvatuista skenaarioista löydettyjä yhteisiä kaavoja (*pattern*)
- sovelletaan inkrementaaalista ja iteratiivista kehitystä, joka mahdollistaa kehitysprojektin aikana esiintulevien uusien vaatimusten huomioimisen



Kuva 1: Sovelluskehityksen muuttuva ympäristö



Kuva 2: Arkkitehtuuriin vaikuttavat tekijät

Hyvään lopputulokseen pääseminen vaatii näiden kaikkien kolmen periaatteen noudattamista.

Miksi iteratiivisesti ?

Arkkitehtuurin rakentaminen iteratiivisesti on perusteltua tehtävän vaativuuden vuoksi. Valmiita reseptejä, joilla saadaan suoraan vaatimuksista toimiva arkkitehtuuri, ei ole olemassa. Iteratiivinen kehitys; suunnittele vähän - koodaa vähän – testaa vähän; on käytännössä ainoa toimiva tapa. Lisäksi iteratiivisus mahdollistaa pienemmissä paloissa tapahtuvan testauksen.

Seuraavassa on esitetty iteratiivisuuden soveltamista arkkitehtuurin osalta projektissa, jonka elinkaarivaiheet ovat aloitus, valmistelu, rakennus ja käyttöönotto. Aloitusvaiheen tavoitteena on määrittää tavoitteet projektille, valmisteluvaiheen tavoitteena on saada määrittely siihen pisteeseen, että tarkempi suunnittelu ja toteutus voidaan aloittaa.

Aloitusvaiheen lopussa (kohdassa 1), on suoritettu alustava riskianalyysi, hankittu ymmärrys arkkitehtuuriin liittyvistä vaatimuksista sekä hahmotettu arkkitehtuuria koskevat rajoitukset ja olettamukset.

Valmisteluvaiheen 1. iteraation (kohta 2) tavoitteena on tuottaa prototyyppi arkkitehtuurista. Toteutusalusta, kehitysympäristöt ja mahdolliset sovelluspalvelimet on valittu. Keskeiset toiminnalliset vaatimukset on purettu auki skenaarioina ja ne on sovitettu valittuun toteutusalustaan. Arkkitehtuurista luodaan prototyyppi, jonka avulla voidaan testata keskeisten vaatimusten täyttyminen. Tässä vaiheessa on aika reagoida, jos havaitaan jotain korjattavaa.

Valmisteluvaiheen lopussa (kohta 3) on arkkitehtuurin kaikkien keskeisten elementtien ja mekanismien oltava valmiina. Tässä vaiheessa on suunniteltu ja toteutettu kaikkein riskialttein toiminnallisuus, tämä voi olla vain n. 5 – 10% koko sovelluksesta. Nyt laajamittainen toteutus ja tarkempi suunnitelu voi alkaa, sillä toimiva perusinfrastruktuuri on valmiina. Todennäköisyys sille, että perusteellisia linjamuutoksia tarvitsisi jatkossa tehdä on oleellisesti pienempi.

Rakennusvaiheessa arkkitehtuurityö on jo huomattavasti vähäisempää. Löydettyjä puutteita tä ja virheitä korjataan ja lopun suunnittelun sekä toteutuksen tuloksena mahdollisesti esiintulevat arkkitehtuurin parannusehdotukset otetaan harkinnan mukaan toteutukseen. Tässä vaiheessa arkkitehtuuri ’viilataan’ lopulliseen kuntoon.

Arkkitehtuurikeskeisyydellä saavutettavat hyödyt

Etenkin verkkokeskeisissä sovelluksissa teknologiariskit ovat kehitysprojekteissa suuria. Iteratiivisuus sekä arkkitehtuurin aikaisessa vaiheessa tapahtuva rakentaminen ja testaaminen auttavat kohtaamaan ja eliminoimaan keskeiset riskit vaarantamatta koko projektin aikataulua.

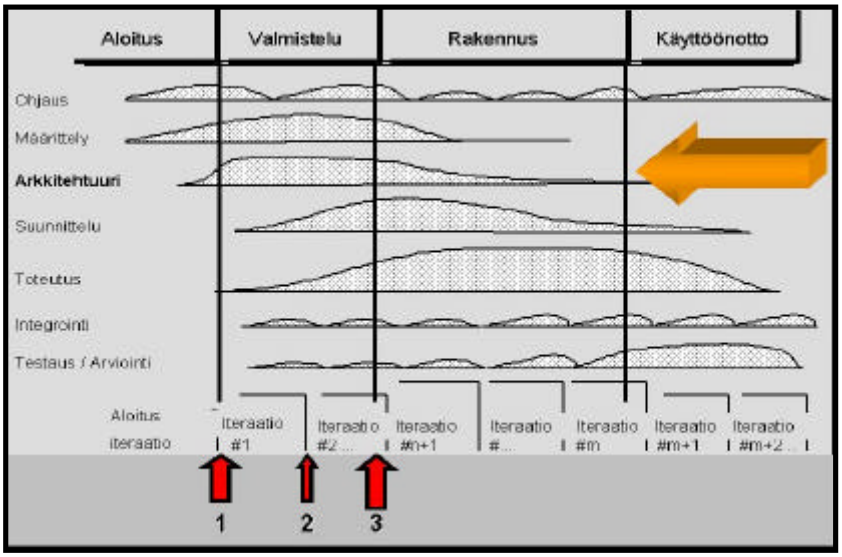
Arkkitehtuuri auttaa meitä rakentamaan muutossietoisia sovelluksia. Abstrahoinnin, kapseloinnin ja oliokeskeinen suunnittelun avulla voimme luoda arkkitehtuureja, joissa muutosvaikutukset ovat mahdollisimman paikallisia. Modulaarinen osajärjestelmärakenne mahdollistaa tehokkaamman työnjaon projektisssa sekä järjestelmäosien testattavuuden parantumisen.

Vaikka kehitysprojektissa kustannukset painottuvat alkuvaiheisiin, voidaan projektin aikana saavuttaa merkittäviä säästöjä toimivan perusinfrastruktuurin ansiosta. Otettaessa myös ylläpitovaihe mukaan tarkasteluun, säästetään myös jatkossa paremman muutossietoisuuden ja ylläpidettävyyden ansiosta.

Yhteenveto

Nopea kehityssykli sekä korkeat laatuvaatimukset edellyttävät kehitysprosessilta myös arkkitehtuurityön korostamista. Riskienhallinnan kannalta on tärkeää suunnitella, toteuttaa sekä testata arkkitehtuuri etupainoitteisesti ennen laajamittaisen toteutustyön aloittamista.

Jouko Poutanen,
Senior Consultant,
Rational Software Finland
jouko.poutanen@rational.com



Kuva 3: Arkkitehtuurityö aikajanalla