Jari Alamaa

Palvelukeskeisen arkkitehtuurin suunnittelumenetelmät

Tietojärjestelmätieteen
kandidaatintutkielma
28.2.2009

Jyväskylän yliopisto
Tietojenkäsittelytieteiden laitos
Jyväskylä

TIIVISTELMÄ

Alamaa, Jari Markus

Palvelukeskeisen arkkitehtuurin suunnittelumenetelmät / Jari Alamaa
Jyväskylä: Jyväskylän yliopisto, 2008, 42 s.

Kandidaatintutkielma

Nykypäivän nopeasti muuttuva liiketoiminnan tehtäväkenttä asettaa tietojärjestelmät uudenlaisien haasteiden eteen. Tietojärjestelmien vaatimustaso on noussut, ja järjestelmän joustavuudesta sekä tehokkuudesta on tullut merkittävä kilpailutekijä. Tilanteen ratkaisemiseksi on oman vastauksensa antanut palvelukeskeinen arkkitehtuuri, joka on teknologiariippumaton ajatusmalli arkkitehtuurin toteuttamiseen. Palvelukeskeinen arkkitehtuuri mahdollistaa tietojärjestelmän toimintojen ja vanhojen perinnejärjestelmien uudelleenkäytön sekä tietojärjestelmien integroinnin.

Tämä tutkielma pyrkii vastaamaan kysymykseen, mitä menetelmiä palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistämiseen voidaan käyttää sekä verrata menetelmiä keskenään. Tutkielman alussa käydään läpi peruskäsitteet palvelukeskeisestä arkkitehtuurista ja liiketoimintaprosessien hallinnasta. Tämän jälkeen tarkastellaan palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistämistapoja ja verrataan menetelmiä keskenään. Tutkimus osoitti, että palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistämisellä voi organisaatio saavuttaa huomattavia etuja verrattuna siihen, että tekniikoita käytettäisiin erikseen. Yhdistämiseen ei löytynyt yhtä selvää lähestymistapaa joka toimisi aina, vaan mallien käyttö on hyvin tilannekohtaista.

AVAINSANAT: Palvelukeskeinen arkkitehtuuri, liiketoimintaprosessien hallinta, SOA, BPM, ylhäältä alas – malli, alhaalta ylös – malli, yhdistämismalli

Ohjaaja: Petri Maaranen

Tietojenkäsittelytieteiden laitos

Jyväskylän yliopisto

Tarkastaja: Petri Maaranen

Tietojenkäsittelytieteiden laitos

Jyväskylän yliopisto

SISÄLLYSLUETTELO

1 JOHDANTO

Liiketoimintavaatimusten nopea muuttuminen, voittojen kasvattaminen ja kulujen optimointi ovat viemässä tietotekniikkaa yhä lähemmäksi yrityksen liiketoimintaa. Samalla myös informaatioteknologiasta on tullut yksi avaintekijöistä luotaessa yritykselle liiketoiminnallista arvoa. Nykypäivän liiketoiminta vaatii yrityksen tietojärjestelmiltä joustavuutta ja ketteryyttä, jotta pystytään vastaamaan liiketoiminnan muuttuvaan luonteeseen. Tämä on saanut yritykset miettimään tietojärjestelmiään ja panostamaan parempien ja toiminnallisempien liiketoimintaprosessien luomiseen, jotka tukevat yrityksen liiketoimintoja paremmin (Josuttis, 2007).

Palvelukeskeinen arkkitehtuuri (*Service Oriented Architecture, SOA*) on yksi tämän hetken keskeisimmistä ja puhutuimmista aiheista kokonaisarkkitehtuurin saralla. SOA tarjoaa kehyksen helpommin integroitaviin järjestelmiin, jotka vastaavat ja täydentävät liiketoiminnan tarpeita. Ydinajatuksena toimivat löyhän kytkennän periaatteella toimivat palvelut, jotka mahdollistavat perinnejärjestelmien hyödyntämisen uudella tavalla. Vanhojen järjestelmien ja niihin tehtyjen investointien uudelleen käyttö onkin myös yksi suurimmista motivaattoreista palvelukeskeiseen arkkitehtuuriin siirtymiseen. Palvelukeskeinen arkkitehtuuri mahdollistaa myös eri tietojärjestelmien ja ohjelmistojen yhdistämisen siten, että ne saadaan palvelemaan yhtä aikaa samoja liiketoimintamalleja (Erl, 2005). Liiketoimintajärjestelmien täytyy kyetä tuottamaan informaatiota ja palveluita oikeassa muodossa ja oikeaan aikaan, jotta ne kykenevät tukemaan jokapäiväisiä liiketoimintatarpeita. Liiketoimintaprosessien hallinta on tuonut helpotusta tähän ongelmaan, sillä se antaa mahdollisuuden liiketoimintaprosessien täysimittaiseen hyödyntämiseen ja kehittämiseen. Liiketoimintaprosessien hallinta mahdollistaa mm. järjestelmän joustavan ja nopean reagoinnin muuttuviin liiketoimintavaatimuksiin, pyrkien samalla tukemaan organisaation jatkuvaa kehittymistä (Chang, 2006).

Palvelukeskeinen arkkitehtuuri ja liiketoimintaprosessien hallinta nähdään usein asioina, jotka on tehty täydentämään toisiaan. Vaikka liiketoimintaprosessien hallintaa pystytään hyödyntämään myös erikseen varsin hyvin ja tehokkaasti, on kuitenkin huomattu, että yhdistämällä se osaksi palvelukeskeistä arkkitehtuuria saadaan laajempi hyöty irti molemmista. Yhdistämiseen voidaa soveltaa kolmea eri lähestymistapaa, jotka ovat hyvin riippuvaisia useista eri tekijöistä kuten resurssit, lähtökohta ja tavoite. Tämän tutkielman tarkoituksena on selvittää mitä palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistämisellä voidaan saavuttaa ja minkälaisia menetelmiä yhdistämiseen voidaan soveltaa. Lisäksi selvitetään mitä ovat eri suunnittelumenetelmien vahvuudet ja heikkoudet sekä verrataan menetelmiä keskenään. Tutkielma on toteutettu kirjallisuuskatsauksena, joka koostuu alan kirjallisuudesta, tieteellisistä artikkeleista ja ohjelmistotoimittajien omasta dokumentaatiosta.

Tutkielma on rakenteeltaan jaettu kahteen osaan, siten että luvussa 2 käydään läpi palvelukeskeistä arkkitehtuuria ja liiketoimintaprosessien hallintaa koskevat käsitteet jä määritelmät. Luvussa 3 käsitellään palveukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistämistä käymällä läpi yhdistäminen liiketoiminnan ja tietotekniikan näkökulmasta. Viimeisessä kappaleessa käydään läpi tutkielman perusteella tehdyt johtopäätökset sekä pohditaan jatkotutkimusaiheita.

2 KÄSITTEET JA MÄÄRITELMÄT

Tässä kappaleessa on tarkoitus käsitellä palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan määritelmiä sekä niihin liittyviä keskeisiä käsitteitä.

2.1 Palvelukeskeinen arkkitehtuuri

Palvelukeskeinen arkkitehtuuri (*Service Oriented Architecture, SOA*) on ollut hyvin ajankohtainen ja puhuttu aihe tietotekniikan saralla parin viime vuoden aikana. Termin SOA merkitykseen ja sisältöön on liittynyt paljon väärinymmärryksiä, sekavuutta ja virheellistä tietoa joten kiistellyn aiheen ympärillä on riittänyt kritiikkiä ja puolestapuhujia. Kritiikki on kohdistunut paljolti siihen, että SOA ei sinänsä tuo mitään uutta tietotekniikkaan, vaan käyttää vanhoja arkkitehtuureja pohjana uudelle ajattelutavalle. Samaan hengenvetoon on myös puhuttu, että SOA on vain toinen termi web-palvelutekniikalle (*Web Services*). Puolestapuhujat painottavat SOA:n oikein ymmärtämiseen, kun se ymmärretään ja toteutetaan oikein, saadaan siitä todellinen hyöty irti. SOA on ennemmin tietotekniikan seuraava kehitysaskel, kuin aivan täysin uudelleen keksitty tekniikka (Erl, 2005).

Vähemmän teknisestä näkökulmasta SOA on kokoelma liiketoiminta, organisointi ja hallinnointi menetelmiä, jotka on tehty luomaan suorituskykyä, tehokkuutta ja joustavuutta liiketoimintaympäristöön. Teknisestä näkökulmasta SOA on tapa standardisoida ja parantaa sovelluskehitystä ja -toteutusta käyttämällä hyväksi helposti saavutettavia ja uudelleenkäytettäviä palveluita, jotka ovat itsenäisiä ja teknologiariippumattomia (Ballard, ym., 2006). Palvelukeskeisessä arkkitehtuurissa on kysymys enemmänkin eräänlaisesta jalostetusta ajattelutavasta, näkökulmasta tai arkkitehtuurimallista, kuin suoranaisesta työkalusta joka johtaa arkkitehtuuritoteutukseen. SOA on teknologia- ja alustariippumaton

arkkitehtuurimalli, jossa tietojärjestelmien toiminnot ja prosessit on suunniteltu toimimaan itsenäisinä, joustavina ja avoimina palveluina. Tavoitteena on tehostaa tietojärjestelmän joustavuutta ja mukautuvuutta tuoden tietotekniikkaa ja liiketoimintaa lähemmäksi toisiaan. Palvelukeskeisen arkkitehtuurin toteutukseen ei ole olemassa mitään hopea luotia minkä pystyy ostamaan suoraan kaupan hyllyltä, vaan kysessä on pitkä aikainen projekti, joka rakentuu pala palalta kokonaisuudeksi (Josuttis, 2007).

Josuttis määrittelee SOA:n eräänlaiseksi arkkitehtuuriseksi paradigmaksi, jossa liiketoimintaprosessit on hajautettu laajalti vanhojen ja uusien järjestelmien kesken, joita hallitsevat useat eri toimijat (Josuttis, 2007, 24).

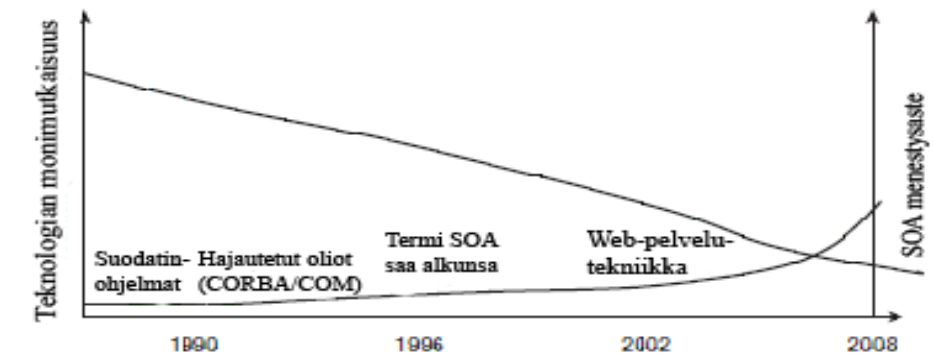
Dr. Hao Henin määritelmän mukaan SOA on arkkitehtuurityyli, jonka pyrkimyksenä on saavuttaa sovellusten välille kommunikointi mahdollisimman löyhin sidoksin, sekä poistamaan liiketoiminnan kannalta epärelevantteja sidoksia ja riippuvuuksia. Tarkoituksena on saavuttaa liitos palveluntarjoajan ja asiakkaan välillä (Hen, 2003). Tässä tutkielmassa on tarkoitus käyttää Dr. Hao Henin määritelmää palvelukeskeisestä arkkitehtuurista ja edetä hänen määrittämällään tiellä.

2.1.1 Palvelukeskeisen arkkitehtuurin historia

Termi SOA on katsottu saaneen alkunsa entiseltä Gartnerin analyytikolta vuonna 1994. Kirjassaan SOA in Practice, Josuttis kertoo, että termin SOA nykyäänkin käytetyt pääperiaatteet ovat saaneet alkunsa vuonna 1994 Gartnerin analyytikko Alexander Pasikilta. Pasik keksi termin *Server Orientation*, koska vanha määritelmä asiakas- ja palvelintoteutukselle (*client/server*) oli menettänyt merkityksensä. Tämä johtui siitä, että uuden tyyppisissä asiakas - palvelintoteutuksissa asiakas tyypillisesti vastasi lähinnä liiketoimintalogiikasta ja palvelin tietokantayhteyksistä. Välttääkseen sekaannuksen uuden ja vanhantyppisen asiakas – palvelintoteutuksen välillä

keksi hän termin *Server Orientatio.* Termiä hän käytti rohkaistakseen kehittäjiä suunnittelemaan palvelukeskeisempiä liiketoimintasovelluksia (Josuttis 2007,7).

Ensimmäisen julkaisun palvelukeskeisestä arkkitehtuurista julkaisivat Gartnerin analyytikot Roy Schulte ja Yefim Natis vuonna 1996. Alkutaipaleet SOA meni käsi kädessä yhdessä CORBA[1]-arkkitehtuurin kanssa, mutta vuoden 2003 aikoihin alkoi SOA erota omaksi osakseen. Tämä johtui siitä, että suuret IT-yritykset alkoivat paketoida SOA:a erilliseksi osaksi CORBA:sta. Tästä eteenpäin palvelukeskeinen arkkitehtuuri alkoi ottaa jalansijaa tietotekniikan maailmassa omana terminään (Josuttis, 2007) (Footen & Faust, 2008). Alla oleva Kuva 1 havainnollistaa SOA:n kehitysaskelia.



Kuva 1: SOA aikajana (Rosen;Lublinsky;Smith;& Balcer, 2008)

2.1.2 Palvelukeskeisen arkkitehtuurin keskeiset toteutusteknologiat

Tässä kappaleessa keskitytään kolmeen keskeiseen palvelukeskeisen arkkitehtuurin toteutusteknologiaan (palveluväylä, palvelurekisteri ja palvelutietokanta), jotka ovat hyvin tärkeitä palvelukeskeisen arkkitehtuurin toteutuksessa.

---

[1] CORBA (*Common Object Request Broker Architecture*) on asiakas-palvelin arkkitehtuuriin perustuva standardi, jossa palvelinoliot tarjoavat palveluita asiakasoliolle.

- Palveluväylä (*Enterprise Service Bus, ESB*) on avoin standardipohjainen integrointialusta, jonka tarkoituksena on helpottaa palvelukeskeisen arkkitehtuurin palveluiden välistä viestintää. Käytännön tasolla ESB on asiakas ja palvelin moduulin väliin sijoitettava kerros, joka vastaa kommunikointi- ja integraatiologiikasta niiden välillä (Papazoglou & Jan van den Heuvel, 2007). ESB on web-standardeja hyödyntävä väliohjelmisto, jonka tarkoitus on tukea ohjelmistojen välistä älykästä kommunikaatiota. ESB:n perusajatuksena on, että se muodostaa järjestelmään integraatiokerroksen, jonka ensi sijaisena tarkoituksena on vastata sanomien välittämisestä ja reitittämisestä palveluiden välillä (Josuttis, 2007). ESB on tärkeä osa palvelukeskeistä arkkitehtuuria, sillä se mahdollistaa löyhempiä yhteyksiä järjestelmäintegroinissa sekä kykenee hajottamaan integraatiologiikan pieniin helposti hallittaviin palasiin, mikä taasen helpottaa ylläpitoa. Avoimena standardipohjaisena viestinvälitysväylänä ESB on suunniteltu mahdollistamaan palvelukeskeiseen arkkitehtuuriin pohjautuvien ratkaisujen helpomman toteutuksen, käyttöönoton ja hallinnan (Papazoglou & Jan van den Heuvel, 2007).

  Palveluväylä toimii eräänlaisena selkärankana palvelukeskeiselle arkkitehtuurille mahdollistaen palveluiden välisen löyhän kytkennän reitittämällä ja muuttamalla viestejä käyttäen hyväksi eri viestinvälitys protokollia (*http udp, soap, smtp, ftp, yms*.) (Liu;Gorton;& Zhu, 2007). Löyhät kytkennät ovat mahdollisia ESB avulla, koska se vastaa viestin välityksestä eikä palveluiden itse tarvitse huolehtia siitä. Tämän johdosta palveluiden ei tarvitse olla riippuvaisia toisista palveluista taikka tietää toisten olemassaolosta, vaan palveluille tulevat kutsut ja tehtävät reititetään ESB:n kautta (Sward & Whitacre, 2008).

- Palvelurekisteri (*Service Registry*) voidaan ajatella eräänlaisena palveluiden metatieto luettelona ja listana, joka mahdollistaa palveluiden

hallinnan, ylläpidon ja tehokkaan hyödyntämisen. (Kulkarni, 2007). Rekisteri helpottaa palveluiden uudelleenkäytettävyyttä, sillä uusia palveluita kehitettäessä rekisteristä löytyvät olemassa olevat palvelut, joita voidaan hyödyntää uudelleen joko sellaisenaan tai uudelleen modifioiden (Rosen;Lublinsky;Smith;& Balcer, 2008). Löyhät kytkennät palveluiden ja käyttäjien välillä SOA-ympäristössä vaativat hyvää palvelutietojen ylläpitoa. Palvelurekisteri on tärkeässä ylläpidon roolissa, sillä se mm. ylläpitää tietoa palveluista, palveluiden sijainnista, palveluiden tilasta sekä tuottaa tarvittavaa metatieto palveluiden yksilöintiin (Juric;Loganathan;Sarang;& Jennings, 2007). Palvelurekisterin avulla sovellukset pystyvät etsimään käytettävän palvelun ja herättämään sen toimintaan. Palvelukeskeisen arkkitehtuurin palvelurekisterit käyttävät usein UDDI-rekisteri (*Universal Description, Discovery and Integration*), joka on XML-pohjainen rekisteri yritysjärjestelmän tarpeisiin (Papazoglou & Jan van den Heuvel, 2007).

- Palvelutietokanta (*Service repository*) tallentaa nykyistä ja vanhaa tietoa palveluista. Tietokanta säilöö mm. palveluihin liittyvät tekniset dokumentaatiot, raportit ja toimintakuvaukset (Kulkarni, 2007). Palvelutietokanta antaa mahdollisuuden tuottaa parempaa liiketoiminnan näkyvyyttä organisaation sovelluksille, pitämällä huolta ja hallinnoimalla sovellusten tietoja, suhteita ja itsenäisyyksiä. Tehokas tietokanta pitää kirjaa tekijöistä jotka parantavat vaikutusanalyysiä ja optimoivat uudelleen käytettävyyttä. Liiketoimintaprosessien hallinnan yhteydessä palvelutietokanta on iso merkitys käytettävien tuotteiden varastoinnissa ja hallinnoinnissa (Rosen;Lublinsky;Smith;& Balcer, 2008).

  Siinä missä palvelurekisterin tehtävänä on säilöä metatietoa palveluista, niin palvelutietokannan tehtävänä on säilyttää kaikki muu tieto palvelukeskeisen arkkitehtuurin palveluista. Palvelutietokanta sisältää

palveluiden tiedot lähtien toiminnallisista vaatimuksista päättyen toteutuksen ja toiminnan aikana tuleviin tietoihin.

2.2 Liiketoimintaprosessien hallinta

Liiketoimintaprosessien hallinnalla (*Business Process Management, BPM*) on merkittävä osa palvelukeskeisen arkkitehtuurin onnistumisessa ja hyödyntämisestä, sillä ilman liiketoimintapuolta ei olisi olemassa todellista palvelukeskeistä arkkitehtuuria. Nykypäivän muuttuva yhteiskunta ja liiketoimintatrendit vaativat organisaatiolta jatkuvaa liiketoimintaprosessien kehittämistä ja täydellistä hallintaa. Liiketoimintaprosesseilta vaaditaan jatkuvaa uusien toimintatapojen etsimistä ja uudistumista, jotta organisaation kilpailukykyä pystytään ylläpitämään.

Di Nitton ja kumppanien (Di Nitto;Ghezzi;Metzger;Papazoglou;& Pohl, 2008) määrittelmän mukaan liiketoimintaprosessien hallinta on tietotekniikan mahdollistamaa hallinnointia, joka käsittää liiketoimintaprosessien arvioinnin, suunnittelun ja parantamisen.

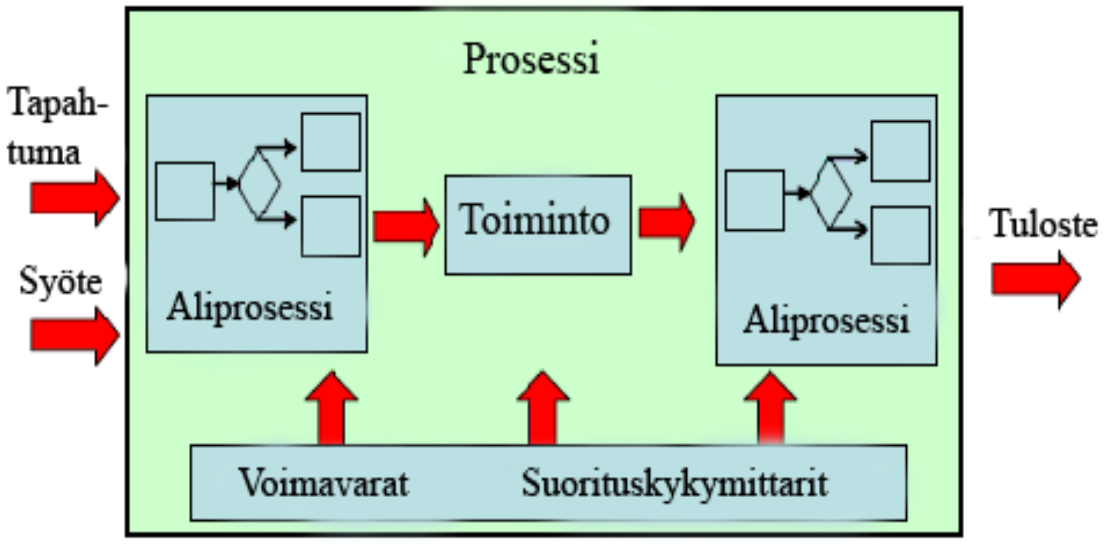
Wesken määrityksen mukaan liiketoimintaprosessien hallinta on selkeä esitysmalli liiketoimintaprosessin toiminnoista, suorituksista ja tuotoksista. Liiketoimintaprosessien hallinta pitää sisällään käsitteet, menetelmät ja tekniikat liiketoimintaprosessien suunnittelun tukemiseen, hallinnointiin, konfigurointiin, suorittamiseen ja analysointiin (Weske, 2007). Tässä tutkielmassa edetään Wesken tekemän määrityksen mukaan.

Liiketoimintaprosessien hallinta nähdään usein pitkäjänteisenä työnä, jota tehdään liiketoiminnalle asetettujen päämäärien saavuttamiseksi. Usein toiminnot jotka muodostavat lopullisen prosessin kulkevat läpi organisaatio- ja järjestelmärajojen, minkä vuoksi BPM nähdään usein hyvin haastavana asiana kuin myös äärimmäisenä vahvuutena organisaatiolle. Liiketoiminta – ja kilpailuedun saavuttaminen vaatii kykyä toimia oikein kriittisten

liiketoimintaprosessien suhteen, minkä vuoksi myös BPM ja sen työkalut, jotka helpottavat hallitsemaan liiketoimintaprosesseja ovat avain asemassa tämän päivän liiketoiminnassa. Helppo hallinta ja kattava teknologia orientoituminen tukevat jokaista BPM:n elinkaaren (analysointi ja mallintaminen, suunnittelu ja toteutus, suorittaminen ja käyttö, mittaaminen ja optimointi) askelta (BEA, 2006). Liiketoimintaprosessien hallinta mahdollistaa järjestelmän joustavuuden ja nopean reagoinnin muuttuviin liiketoimintavaatimuksiin optimoimalla ja automatisoimalla liiketoimintaprosessit (kuva 2):

- havaitsemaan ja poistamaan päällekkäisyydet ja pullonkaulat
- parantamaan siirrettävyyttä ja pienentämään kustannuksia teollisuusstandardeja hyödyntäen
- minimoimaan manuaalisesti toteutettavat tehtävät
- jalkauttamaan nopeasti liiketoiminnan uudet määritykset ja prosessit
- valvomaan ja hallitsemaan prosessin suorituskykyä, käyttämällä suorituskykyilmaisimia hyväkseen.

(Ballard, ym., 2006)



Kuva 2: Liiketoimintaprosessin elementit (Ballard, ym., 2006)

Liiketoimintaprosessien hallinnassa yhdistyy prosessit, informaatio ja tietotekniikka, jotka antavat tukensa organisaation pääresursseille: ihmiset, informaatio, teknologia ja prosessit. BPM:n avulla saadaan reaaliaikaista tietämystä niin liiketoiminnan tilasta, resurssien käytöstä kuin tietojärjestelmien suorituskyvystä. BPM:n soveltaminen mahdollistaa myös liiketoiminta-informaation nopeamman saatavuuden ja reagoinnin muuttuviin markkina-trendeihin, kilpailuedun saavuttamisen sekä liiketoiminnan saavuttamaan parempaa tulosta. Tavoitteena on luoda prosessien jatkuva kehittyminen määrittämällä, mittaamalla ja kehittämällä prosesseja. Liiketoimintaprosessien hallinnalla pyritään optimoimaan organisaation liiketoimintaprosessit ja tavoittelemaan organisaation jatkuvaa kehittymistä (Ballard, ym., 2006).

2.2.1 Liiketoimintaprosessin hallinnan hyödyt

Liiketoimintaprosessien hallinta mahdollistaa yritystä tulemaan joustava-mmaksi ja samalla helpommin integroitavaksi liiketoimintapäämääriin ja -tavoitteisiin. Järjestelmä mahdollistaa uusien tuotteiden ja palveluiden nopeam-man markkinoille tuonnin, ja auttamaan yritystä reagoimaan nopeammin

muutoksiin (Ballard, ym., 2006). Liiketoimintaprosessien hallinta tuo mukanaan etuuksia, jotka mm.:

1. Lisäävät tuottavuutta, tehokkuutta ja hallintaa. Liiketoimintaprosessien hallinta pitää huolen, että työskentely tapahtuu aina tärkeimpien tuotteiden parissa ja jatkuvan valvonnan alaisuudessa. Tämä edesauttaa tärkeimpien prosessien nopeampaa läpivientiä sekä parempaa tuottavuutta, koska prosessit saadaan toimimaan parhaalla kapasiteetilla.

2. Mahdollistavat joustavampaa liiketoimintaa, joka kykenee vastaamaan paremmin liiketoiminnan nopeisiin muutoksiin. Liiketoimintaprosessien hallintajärjestelmät on kehitetty UML tyyppisiksi visuaalisiksi prosessimalleiksi, mikä helpottaa muutoksien tekemistä.

3. Parantavat voimavarojen hyödyntämistä, suunnittelua ja prosessinäkyvyyttä. Prosessitapahtumat ovat paremmin näkyvillä ja hallittavissa, mikä tekee tilanneraportoinnista helpompaa. Raporttien nopea saatavuus tekee myös suunnittelun helpommaksi.

4. Pienentävät kuluja ja lyhentävät tuotantoaikoja. Tämä on mahdollista, koska BPM varmistaa prosessin sujuvuuden mahdollisimman pienin voimavaroin sekä pitäen samalla huolen tuotannon laadusta.

5. Mahdollistavat tyytyväisemmän asiakaskunnan. BPM mahdollistaa tuotteiden ja palveluiden tuottamisen asiakkaiden tarpeisiin nopeasti ja korkealla laadulla.

(Cumberlidge, 2007)

3 PALVELUKESKEISEN ARKKITEHTUURIN JA LIIKETOIMINTAPROSESSIEN HALLINNAN YHDISTÄMINEN

Etsiessäni tietoa tutkielmani aiheeseen, tuli usein vastaan englanninkielinen lause: "*SOA makes great BPM, but BPM makes crappy SOA*", eli palvelukeskeinen arkkitehtuuri mahdollistaa hyvää liiketoimintaprosessien hallintaa, mutta liiketoimintaprosessien hallinta tekee huonoa palvelukeskeistä arkkitehtuuria. Toisin sanoen pitäisikö palvelun tulla ensin vai prosessin? Rosen, Lublinsky, Smith ja Balcer (Rosen;Lublinsky;Smith;& Balcer, 2008, 106) mainitsevatkin kirjassaan, että yksi yleisemmistä erimielisyyksistä käydään näkökulmien kesken siitä, toimiiko SOA mahdollistajana BPM:lle vai toisin päin. Tämän erimielisyyden selvittelyyn perehdytään tässä kappaleessa, eli selvitetään mitä eroja yhdistämismenetelmillä on ja verrataan niitä keskenään.

Liiketoimintaprosessit ovat ydinosa palvelukeskeistä arkkitehtuuria ja usein päämääränä on saavuttaa mahdollisimman ketterästi toteutettu ympäristö, joka pystyy automatisoidusti vastamaan muutoksiin (Erl, SOA: Principles of Service Design, 2008). Liiketoimintaprosessien hallinta on luonnollinen tekijä täydentämään palvelukeskeistä arkkitehtuuria, mikäli halutaan tavoitella korkeantason liiketoimintaa. BPM on mahdollista toteuttaa toimivasti ja tehokkaasti ilman SOA:a, mutta näiden kahden lähestymistavan yhteistyöllä voidaan saavuttaa molemmanpuolisia etuja (BEA, 2006, 7). Vaikka palvelukeskeinen arkkitehtuuri ja liiketoimintaprosessien hallinta omaavatkin erilaiset lähtökohdat, toisen ollessa enemmän tietotekniikka painotteinen ja toisen liiketoimintalähtöinen saa niiden yhdistäminen paljon hyvää aikaiseksi. Yhdistämällä SOA:n ja BPM:n saadaan synergia etuja, joiden avulla pystytään vastaamaan muuttuvaan liiketoimintaympäristöön paremmin. SOA ja BPM voidaan ajatella olevan saman kolikon molemmat puolet. Toisella puolella on SOA, joka tasoittaa tietä BPM:n räjähdysmäiselle kasvulle auttamalla sitä annetuilla lupauksilla järjestelmän paranevasta joustavuudesta ja ketteryydestä.

Kolikon toisella puolella on BPM, joka tuottaa SOA:n vahvoja liiketoimintatapauksia ja pyrkii tuomaan liiketoimintaa ja tietotekniikkaa lähemmäksi toisiaan. SOA:n ja BPM:n yhdistäminen mahdollistaa ketterämpää liiketoimintaa ja parempaa joustavuutta vastata muutoksiin (Kamoun, 2007).

Organisaatio voi valita palvelukeskeisen arkkitehtuurin ja nauttia sen tarjoamista eduista, kuten joustavuudesta, uudelleen käytettävyydestä ja sen sopeutumiskyvystä. Viemällä palvelukeskeistä arkkitehtuuria vastaamaan liiketoiminnan haasteita BPM avulla, mahdollistaa se mitattavissa olevan tuoton investoinneille (*ROI, Return On Investment*) ja suuremman suoran vaikutuksen liiketoimintaan (BEA, 2006). Käyttämällä palvelukeskeistä arkkitehtuuria ja liiketoimintaprosessien hallintaa yhdessä voi organisaatio saavuttaa parempia tuloksia kuin toteutukset yksinään. Siinä missä SOA parantaa uudelleen käytettävyyttä ja tekee liiketoimintaprosesseista joustavampia, tuottaa BPM rikkaampaa ja korkeamman tason liiketoimintaprosesseja palveluiden avulla. SOA soveltuukin parhaiten erityisesti pitkän elinkaaren omaaviin liiketoiminnan osa-alueisiin, joita eivät muutokset koettele lyhyellä aikavälillä (Kamoun, 2007). Alla näkyvä kuvio (Kuva 3) havainnollistaa kuinka SOA:n ja BPM:n vuorovaikutussuhde toimii.

Kuva 3: SOA:n ja BPM:n vuorovaikutussuhde (Noel, 2005)

Palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistämiseen on olemassa kaksi tutumpaa lähestymistapaa ja yksi yhdistelmämalli (**Kuva 7**, sivu 32). Yhdistämisprosessia voidaan lähteä viemään eteenpäin ylhäältä alas – mallin (*top-down*) mukaan tai alhaalta ylös – mallia (*bottom-up*) hyödyntäen tai kolmatta vaihtoehtoa käyttäen eli yhdistelmämallia (*middle out, meet in the middle*). Yhdistelmämallin yhteydessä usein puhutaan ketterästä strategiasta, jonka tarkoituksena on luoda tasapaino ylhäältä alas – mallin liiallisen suunnittelun ja alhaalta ylös - mallin hajanaisen toteutuksen välillä (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005). Teoria usein johdattaa käyttämään palveluiden toteutukseen ylhäältä alas – mallia, jossa prosessit ja palvelut suoritetaan yrityksen toimintamallia silmällä pitäen. Tätä pidetäänkin yleisesti liiketoimintaihmisten mallina. Ylhäältä alas – mallin projektit usein alkavat korkean tason liiketoimintaprosessien mallinnuksella ja kattavalla tietojärjestelmien analysoinnilla. Mikäli taustalla on jo kokemusta palveluiden toteutusprosessista tai halutaan hyödyntää perinnejärjestelmiä ja olemassa olevia toimintoja, johtaa se usein alhaalta ylös – mallin hyödyntämiseen (Rosen;Lublinsky;Smith;& Balcer, 2008).

3.1 Ylhäältä alas - malli (top-down)

Ylhäältä alas – mallissa voidaan puhua eräänlaisesta analysointilähtöisestä ajatusmallista, jossa prosessit tulevat ensin. Malli ottaa enemmän yrityslähtöisemmän ja strategiapohjaisemman näkökulman asioihin, jossa prosessit määräävät tahdin. Mallissa liiketoimintaprosessit määrittelevät palvelut, joita tarvitaan prosessien läpivientiin ja toteutukseen. Ylhäältä alas - malli huolehtii koko yrityksen nykyisistä ja tulevaisuuden liiketoimintavaatimuksista sekä vastaa siitä, kuinka toiminnalliset ratkaisut sopeutuvat yrityksen strategiaan ja taktisiin vaatimuksiin. Ylhäältä alas–malli huolehtii kokonaisvaltaisemmin yritystoiminnan vaatimuksista pitkällä aikavälillä (Rosen;Lublinsky;Smith;& Balcer, 2008). Ylhäältä alas – mallin elinkaaren aikana kerätään liiketoimintavaatimukset, luodaan

liiketoimintaprosessit, ja varmistetaan että tietojärjestelmän toiminnot on asetettu vastaamaan vaatimuksiin (Bouillet;Feblowitz;Liu;Ranganathan;& Riabov, 2008).

Kuva 7 osoittaa kaksi eriteltyä ylhäältä alas - mallin lähtökohtaa: yritysjärjestelmä analyysin ja liiketoimintaprosessimallin (Rosen;Lublinsky;Smith;& Balcer, 2008).

1. Yritysjärjestelmä analyysi (*Enterprise System Analysis*) pyrkii ymmärtämään ja keräämään järjestelmävaatimuksia. Tyypillisesti aloitetaan liiketoiminnan tavoitteista ja tekijöistä, ja analysoidaan yrityksen liiketoiminta-arkkitehtuuri ja viestintämallit. Suunnitelman perusteella tehdään kooste tarvittavista palveluista ja luodaan suunnitelma niistä kokonaisuuksista ja palveluista, joita tarvitaan liiketoiminnan tavoitteiden tukemiseen tulevina vuosina. Suunnitelman tulisi myös pitää sisällään palveluiden toteutuksen priorisoinnin, jossa kriittisimmät tai laajemmin käytetyt palvelut luodaan ensin.

2. Toinen ylhäältä alas – mallin lähtökohta on kehittää liiketoimintaprosessimallit (*Business Process Models*) osaksi yksityiskohtaisia liiketoimintavaatimuksia. Tässä lähestymistavassa liiketoimintaprosessimallin tehtävät toteutetaan liiketoimintapalveluina. Liiketoimintaprosessimalli voidaan nähdä pelkkänä lisäarvoa tuottavana laajennuksena yritysjärjestelmäanalyysiin, jotka tuovat uusia yksityiskohtia liiketoiminta-arkkitehtuuriin.

(Rosen;Lublinsky;Smith;& Balcer, 2008)

Näiden kahden lähtökohdan avulla saadaan ylhäältä alas – malli hajotettua kahteen pienempään osakokonaisuuteen, jolloin palveluiden suunnittelua on helppo lähteä viemään eteenpäin. Jaottelun avulla ylhäältä alas – malliin saadaan mukaan niin yritysnäkökulma kuin myös strateginen liiketoiminta

näkökulma. Aluksi analysoidaan nykyinen yritysjärjestelmä ja tehdään johtopäätökset tarvittavista kokonaisuuksista. Tämän jälkeen käydään läpi liiketoimintaprosessimallit ja kehitetään niitä vastaamaan liiketoimintavaatimuksia.

Thomas Erl määrittelee ylhäältä alas – mallin strategiaksi, joka vaatii liiketoimintaprosesseja tulemaan palvelukeskeisiksi, mutta myös ylentämään organisaation rakenteen liiketoimintamalliksi. Yleistä mallille onkin, että se luo useita uudelleenkäytettäviä liiketoiminta- ja sovelluspalveluita. Malli koostuu seitsemästä eri vaiheesta, jotka Kuva 4 osoittaa (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005).

Vaihe 1. Pitää sisällään yrityksen oleellisten ontologioiden määrittämisen, jonka tuloksena saadaan koostettua vaadittava informaatio organisaatiosta. Saaduista tiedoista kootaan informaatiojoukkoja ja pyritään määrittämään niiden väliset suhteet. Malli vaatii liiketoiminnan vaatimuksien olevan jo tiedossa ja määriteltynä, ja niiden tukevan yrityksen kokonaisontologiaa.

Vaihe 2. Asetetaan oleelliset liiketoimintamallit tukemaan uutta tai uudistettua ontologiaa. Liiketoimintamallit täytyy tässä vaiheessa sovittaa tai luoda vastaamaan uutta määritystä. Liiketoimintamalli kokonaisuudet ovat erityisen tärkeitä määrittää kunnolla, sillä niitä tullaan käyttämään myöhemmin pohjana liiketoimintapalveluille.

Vaihe 3. Käydään läpi analyysivaihe, jossa määritetään mitä palveluita on tarve rakentaa ja mitä sovelluslogiikkaa tullaan missäkin palvelussa käyttämään. Tämä vaihe vie yleensä eniten aikaa, ja on tärkeässä osassa strategian onnistumista, sillä palveluiden määrittämisen lisäksi vaihe pitää sisällään myös päätöksenteon kriittisimmistä asioista, kuten esimerkiksi mitä palvelukerroksia on tarkoitus rakentaa ja miten ne vaikuttavat palvelukeskeisen ympäristön

rakenteeseen. Kolme tärkeintä palvelukeskeisen arkkitehtuurin palvelukerrosta ovat sovelluspalvelukerros, liiketoiminta palvelukerros ja orkestrointi palvelukerros (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005).

Sovelluskerros pystyy käsittelemään tietoa uudessa tai olemassa olevassa sovellusympäristössä, jolloin palvelut eivät ole riippuvaisia jostain tietystä teknologiasta. Liiketoiminta palvelukerros on yhteydessä liiketoimintapalveluihin, jotka edustavat palvelukeskeisen arkkitehtuurin elintärkeitä liiketoiminta-palvelumalleja. Kerros pitää huolen liiketoimintapalveluista ja niiden liiketoimintalogiikasta. Orkestrointi palvelukerros pitää huolen palveluiden järjestyksestä ja logiikasta. Kerroksen luomat prosessipalvelut yhdistävät liiketoiminta- ja sovelluspalveluita parantaen joustavuutta ja uudelleen käytettävyyttä (An;Lee;& Jin, 2007).

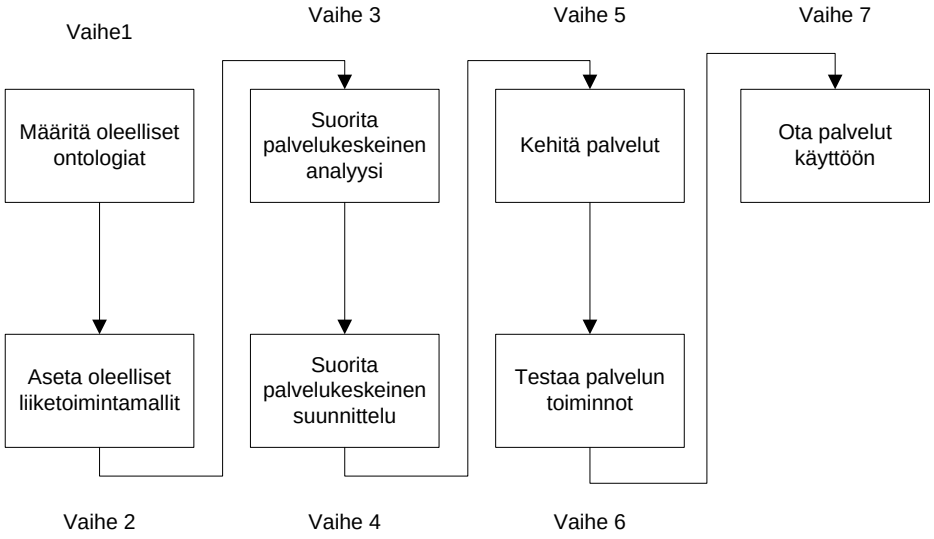
Vaihe 4. Suoritetaan itse palvelukeskeisen arkkitehtuurin suunnittelu, jossa eri palvelukerrokset määritetään osaksi palvelukeskeistä suunnitteluprosessia ja tehdään kuvaukset palveluille. Vaiheen päämääränä on muuttaa aiemman vaiheen palveluehdokkaat fyysisiksi palvelusuunnitelmiksi. Tämän määrityksen pohjalta fyysisistä palvelusuunnitelmista tehdään abstrakteja rakenteita, jotka toteuttavat liiketoimintaprosesseja.

Vaihe 5. Kehitetään palvelut kunkin oman määrityksen ja vaiheessa neljä saadun kuvauksen mukaan.

Vaihe 6. Tehdään testejä palveluille ja niiden operaatiolle niin kauan, että vaadittava toimivuus ja laatu saavutetaan.

Vaihe 7. Malli on tehty valmiiksi käyttöönottoa varten ja oikein toteutettuna luonut tulevaisuutta ajatellen helposti uudelleenkäytettäviä palveluita.

(Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005)



Kuva 4: Ylhäältä alas - mallin vaiheet (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005)

Ylhäältä alas – mallissa on tärkeää ymmärtää yritysympäristö, liiketoiminnan suuntaviivat sekä kyky integroida ne prosessipohjaiseen palvelusuunnitteluun. Tämä luo usein paineita järjestelmäprojekteissa, sillä osallistujilta vaaditaan tietämystä yrityksen liiketoiminnallisista vaatimuksista, joka on usein puutteellista (Rosen;Lublinsky;Smith;& Balcer, 2008).

Ylhäältä alas – mallissa palveluiden tunnistus perustuu joko liiketoimintaprosesseihin tai liiketoiminta-tapahtumiin (Kohlmann & Alt, 2009). Mallissa lähdetään liikkeelle prosessilähtöisesti liiketoimintanäkökulmaa silmälläpitäen ja se soveltuukin paremmin lähtökohdaksi projekteihin, jossa kehitys aloitetaan puhtaalta pöydältä. Ylhäältä alas – mallin prosessi alkaa yleisistä hankkeen tavoitteista ja päämääristä. Enemmän koordinointia vaativana mallina pystytään sen avulla saavuttamaan nykyaikaista ja

johdonmukaista palvelukeskeistä arkkitehtuuria (Schepers;Iacob;& Van Eck, 2008). Ylhäältä alas – malli huolehtii siitä, että liiketoimintavaatimukset kerätään, liiketoimintaprosessit tulevat luoduiksi, ja että tietojärjestelmän toiminnot on tehty tukemaan liiketoimintavaatimuksia. Mallin sisältämä analysointi mahdollistaa myös hyvää työnkulkun[2] (*workflow*) suunnittelua ja prosessien kehittämistä (Bouillet;Feblowitz;Liu;Ranganathan;& Riabov, 2008). Ylhäältä alas – malli pyrkii kehittämään yrityksen yleisiä liiketoimintamalleja priorisoitujen tarvevaatimusten mukaan. Priorisoinnin avulla voidaan saavuttaa korkeamman tason omaavaa palvelukeskeistä arkkitehtuuria (An;Lee;& Jin, 2007). Ylhäältä alas – mallin palveluiden kehittämisen tärkeimpänä etuna on sovellusten ristiriidattomuus ja integraatiomekanismi, joiden avulla palvelukeskeisten ratkaisujen kehittäminen on helppoa samalla kun teollisuus kehittyy (Papazoglou & van den Heuvel, 2006). Ylhäältä alas - malli mahdollistaa korkean laadun omaavaa palvelukeskeistä arkkitehtuuria, koska palveluiden suunnittelutyö on pitkäjänteisempää, kuin alhaalta ylös – mallissa. Ylhäältä alas - malli huolehtii enemmän yrityksen tämän hetken ja tulevaisuuden vaatimuksista, minkä johdosta se vaatii myös huomattavasti aikaa analyysi- ja suunnittelutyöhön (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005).

Ylhäältä alas - mallin heikkoutena on usein työnkulku joka on suunniteltu etukäteen ja palvelut jotka on kehitetty sopimaan näihin työnkulkuihin. Etukäteen suunnittelu aiheuttaa sen, että malli ei ole tarpeeksi joustava käsittelemään uusia tilanteita ja uusia prosessipäämäärä, jotka vaativat eri työnkulkuja (Bouillet;Feblowitz;Liu;Ranganathan;& Riabov, 2008). Malli vaatii myös enemmän koordinointia kuin esimerkiksi alhaalta ylös – malli ja voi aiheuttaa muutosvastarintaa, koska kysessä on yleensä laajempi ja kokonaisvaltaisempi ratkaisu (Schepers;Iacob;& Van Eck, 2008). Mallin vaatima

[2] Työnkulussa voidaan määritellä tietojärjestelmän automaattisesti toteutettavat tehtävät, liiketoimintaprosessit sekä ihmisten vastuut.

analysointi- ja suunnittelutyö on aikaa vievää, mikä vaatii isompaa rahallista investointia, joka ei välttämättä näytä tuloksia lyhyellä aikavälillä (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005). Aikaa vievänä toteutusmallina myös prosessin todellinen valmiusaste voi olla hankalaa hahmottaa, mikä vaikeuttaa raportointia (Juric;Loganathan;Sarang;& Jennings, 2007).

Ylhäältä alas – malli soveltuu hyvin projekteihin jossa liiketoiminta on merkittävässä osassa. Mallia voisi soveltaa palvelukeskeisen arkkitehtuurin palveluiden suunnitteluun esimerkiksi B2B projektissa, jossa halutaan luoda yhteinen järjestelmä organisaation ja sen asiakkaiden ja kumppanien välille.

3.2 Alhaalta ylös - malli (bottom-up)

Lähestymistapa kannustaa luomaan palveluita, jotka täydentävät olemassa olevien sovellusten vaatimuksia ja hyödyntävät nykyisiä perinnejärjestelmiä. Ensisijainen motivaatio alhaalta ylös – mallin käyttöön on integrointi, jossa pystytään hyödyntämään nykyistä perinnejärjestelmää ja sen viestintä viitekehyksiä sekä käärimään palvelut helpommin osaksi järjestelmää (Erl, 2005). Alhaalta ylös- malli alkaa palveluiden kokoamisella, joka lähtee liikkeelle kasaamalla sovellukset yhteen ja hajottamalla ne palveluiksi. Tässä vaiheessa voidaan luoda uusia palveluita, joiden katsotaan oleva hyödyksi tulevaisuudessa. Malli käyttää lähtökohtanaan perinnejärjestelmiä mikä mahdollistaa sen, että järjestelmäkuvaukset on jo valmiiksi toteutettu. Näitä valmiita kuvauksia voidaan myös helpommin käyttää pohjana palveluiden uudelleenkäytölle (Rosen;Lublinsky;Smith;& Balcer, 2008).

Kuva 7 esittää kaksi tavallista palvelukeskeisen arkkitehtuurin alhaalta ylös-mallin lähestymistapaa: hyötypalvelut ja palvelun aktivointi. (Rosen;Lublinsky;Smith;& Balcer, 2008)

1. Hyötypalveluissa (*Utility Services*) palvelukeskeinen arkkitehtuuri esitellään organisaatiolle usein tietohallinto-osaston kehittäjien teknologiasaavutuksena. Palvelukeskeisen arkkitehtuurin uskotellaan olevan se teknologinen tekijä, joka tulee tuomaan tarvittavan eron kilpailijoihin ja näin tuomaan kilpailuetua. Tämä johtaa usein siihen, että todellisuudessa hankkeella ei ole olemassa mitään liiketoimintalähtöistä motivaatiota palvelukeskeisen arkkitehtuurin hyödyntämiseen. Liiketoiminta näkökulman puuttuminen johtaa siihen, että voidaan ainoastaan saavuttaa osittaisia hyötyjä uudelleen käytettävyydestä, pienenemistä kustannuksista, paremmasta laadusta ja tuottavuudesta.

2. Toisessa alhaalta ylös – mallin lähestymistavassa palvelukeskeinen arkkitehtuuri nähdään palvelun aktivoijana (*Service Enabling*), jota pidetään seuraavana EAI:n (järjestelmäintegrointi, *Enterprise Application Integration*) korvaajana. Alhaalta ylös – mallia hyödyntäen palvelukeskeinen arkkitehtuuri nähdään keinona yhdistää data ja toiminnot entisistä perinnejärjestelmistä. SOA ratkaisua tarjoavat yritykset mainostavat, että SOA on helppo ja vaivaton tapa muuttaa nykyisten järjestelmien sisältö palvelu rajapinnoiksi. Tämä on kuitenkin väärä syy palvelukeskeiseen arkkitehtuuriin siirtymiseen. Menetelmä on kyllä nopea ja helpommin toteutettava, mutta sitä käyttämällä yhdistämiseen ei rajapintoja saada vastamaan yrityksenvaatimuksia (joustavuus ja laajennettavuus). Liiketoimintavaatimukset jäävät taka-alalle tässä menetelmässä.

(Rosen;Lublinsky;Smith;& Balcer, 2008)

Lähtökohdat tarjoavat alhaalta ylös – mallin hyödyntämisen helpommin ymmärrettävissä olevana kokonaisuutena. Alhaalta ylös – mallia käytettäessä halutaan hyödyntää olemassa olevat investoinnit ja näin ollen motivaatio on teknologialähtöinen. Tämä käy myös ilmi lähestymistavan jaeotteluissa, sillä

aluksi palvelukeskeinen arkkitehtuuri myydään läpi teknologiana, joka tulee johtamaan teknologisen kilpailuedun saavuttamiseen. Tämän jälkeen osoitetaan, kuinka helppoa palvelukeskeiseen arkkitehtuuriin siirtyminen on hyödyntämällä olemassa olevia järjestelmiä ja investointeja. Helppoa ja yksinkertaista, mutta ei pitkä kantoista toimintaa sillä liiketoimintaa ei ole huomioitu juuri ollenkaan.

Thomas Erl määrittelee alhaalta ylös - mallin strategiaksi, joka kannustaa luomaan palveluita, jotka täydentäisivät nykyisiä sovellusvaatimuksia ja hyväksi käyttävät olemassa olevia perinnejärjestelmiä. Malli koostuu viidestä eri vaiheesta, jotka Kuva 5 osoittaa (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005).

Vaihe 1. Tuottaa määritelmän sovellusvaatimuksille, joka pitää sisällään kiintopisteet palvelukeskeisen arkkitehtuurin ja vanhan perinnejärjestelmän välillä. Vaatimuksiin myös määritellään mitä kommunikointiteknologiaa käytetään. Malli tuottaa hyvin palvelukeskeisiä ratkaisuja, joihin mallinnetaan mukaan tietyt liiketoimintalogiikat ja säännöt.

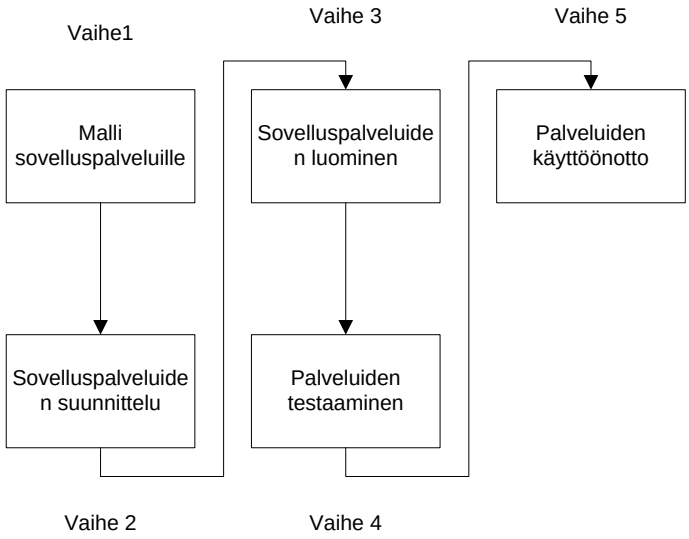
Vaihe 2. Suunnitellaan vaadittavat sovelluspalvelut, joita käytetään. Vaiheen tarkoituksena on mm. päättää mitä perinnejärjestelmän toiminnallisuuksia tullaan käärimään palveluiksi (*wrapper service*), ja mitä toiminnallisuuksia mukautetaan välityspalvelinpalveluiksi (*proxy service*). Palvelurajapintojen toteutusjärjestyksen määrittäminen on tärkeää tässä vaiheessa, jotta voidaan saavuttaa korkean laadun omaavaa palvelukeskeistä arkkitehtuuria.

Vaihe 3. Luodaan tarvittavat sovelluspalvelut käyttäen hyväksi palvelukuvauksia ja sopivia suunnittelumäärityksiä.

Vaihe 4. Testataan palveluiden toimivuus käytettävään ympäristöön ja perustana olevaan perinnejärjestelmään. Tällä tavoin varmistetaan, että asetetut vaatimukset pystytään täyttämään. Testivaiheessa usein mitataan kuinka käärityt palvelut suoriutuvat perinnejärjestelmän kanssa.

Vaihe 5. Ratkaisut ja sovelluspalvelut otetaan käyttöön, kun testaus on mennyt hyväksyttävästi läpi.

(Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005)



Kuva 5: Alhaalta ylös - mallin vaiheet (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005)

Historia on näyttänyt, että projektit joita on lähdetty toteuttamaan alhaalta ylös – mallilla, luovat ennemminkin pitkäaikaisia kustannuksia kuin pitkänaikavälin tuottoja. Hyvien palveluiden suunnittelu ja toteuttaminen vie enemmän aikaa, kuin huonojen. Tämä pitäisi muistaa, kun projektia lähdetään viemään eteenpäin kovien aikavaatimusten saattelemana. Vaikka alhaalta ylös – malli on nopeammin toteutettavissa kuin ylhäältä alas – malli, niin liian kovaa toteutusaikataulua ei sillekään kannattava suunnitella

(Rosen;Lublinsky;Smith;& Balcer, 2008). Alhaalta ylös – malli on yleisesti tunnettu malli ja helppo toteuttaa, mutta sitä käyttämällä ei pystytä saavuttamaan edistyksellistä palvelukeskeisyyttä tai nykyaikaista palvelukeskeistä arkkitehtuuria (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005).

Alhaalta ylös – mallissa lähdetään liikkeelle olemassa olevien sovellusten pohjalta, jossa palvelut on kääritty täyttämään olemassa olevan perinnejärjestelmän sovellusvaatimuksia. Alhaalta ylös – mallin avulla palveluiden tuottaminen on helpompaa ja nopeampaa, kuin ylhäältä alas – mallissa (An;Lee;& Jin, 2007). Mallissa uudet palvelurajapinnat on kehitetty vanhojen sovellusten pohjalta, mikä takaa vakaan pohjan ja vähentää työmäärää. Malli soveltuu erityisen hyvin järjestelmäympäristöön, joka pitää sisällään useita heterogeenisiä teknologioita ja alustoja tai ympäristöön joka käyttää hyväksi nopeasti kehittyviä teknologioita (Papazoglou & van den Heuvel, 2006). Alhaalta ylös – malli mahdollistaa helposti uudelleenkäytettäviä yksittäisiä ja yhdistettyjä palveluita, joita pystytään hyödyntämään eri konteksteissa. Myös uudet työnkulut voidaan helpommin yhdistää vastamaan uusia loppukäyttäjävaatimuksia tai muita suoritusvaatimuksia (Bouillet;Feblowitz;Liu;Ranganathan;& Riabov, 2008). Alhaalta ylös – mallissa palveluiden suunnittelu ja kehittäminen lähtee liikkeelle tunnistamisprosessilla, jonka perustana toimivat käyttäjien asettamat palveluvaatimukset ja tämän vuoksi palvelut saadaan hyvin vastaamaan käyttäjien tarpeita (Schepers;Iacob;& Van Eck, 2008). Isona etuna mallilla on myös, että se mahdollistaa teknologian ja taustajärjestelmien suorituskykyisemmän ja tehokkaamman käytön, koska lähtökohtana toimivat nykyiset teknologiaratkaisut (Kohlmann, Service Identification and Design – A Hybrid Approach In Decomposed Financial Value Chains, 2008). Mallin pyrkimyksenä on suunnitella nykyiset järjestelmät osaksi uutta alustaa, uudelleen käyttämällä olemassa olet investoinnit nykyisessä laajuudessaan. Palvelut pyritään luomaan

vähäisellä työmäärällä, yhteen sovittamalla tai hajottamalla nykyisiä sovelluksia (Shan, 2004). Perinnejärjestelmiä hyödyntävänä menetelmänä myös vahva alusta tekniikoille ja parhaille käytännöille on jo kehitetty (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005). Vanhojen järjestelmien hyväksikäyttö mahdollistaa myös, että järjestelmäkuvaukset on jo valmiiksi toteutettu, joten niitä voidaan käyttää pohjana uudelleenkäytölle (Rosen;Lublinsky;Smith;& Balcer, 2008).

Alhaalta ylös - mallin heikkoutena on, ettei se pysty luomaan nykyaikaista ja edistyksellistä palvelukeskeistä arkkitehtuuria. Mallia vaivaa myös kapeakatseisuus ja arvioinnin puute laajemmissa kokonaisuuksissa (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005). Malli keskittyy enemmän tekniikkaa ja toteutusta koskeviin näkökulmiin, jolloin liiketoiminta jää vähemmälle huomiolle (Kohlmann & Alt, 2009). Alhaalta ylös – malli ei myöskään mahdollista yhtä joustavia liiketoimintamalleja ja nopeaa markkinoille saanti aikaa (*time-to-market*), kuin ylhäältä alas – malli (Kohlmann, Service Identification and Design – A Hybrid Approach In Decomposed Financial Value Chains, 2008). Koska mallin pääpaino on olemassa olevissa sovelluksissa, jää liiketoimintavaatimukset usein vähälle huomiolle eikä yhtymäkohtia liiketoiminannan kanssa välttämättä löydy (Rosen;Lublinsky;Smith;& Balcer, 2008).

Alhaalta ylös – malli soveltuu siis hyvin projekteihin jossa liiketoiminalla ei ole niin suurta merkitysta, vaan halutaan nopeasti vakaa, uudelleen käytettävä ja toimiva järjestelmä. Alhaalta ylös – malli sopii menetelmäksi esimerkikisi asiakkaiden tilauskantapalvelun tekemiseen, joka hyödyntää web-sovellustekniikkaa.

3.3 Yhdistämismalli (meet in the middle, middle out)

Yhdistämismallia joka tunnetaan myös ketteränä mallina, pidetään usein parhaana vaihtoehtona yhdistämiseen. Yhdistämismalli mahdollistaa ylhäältä

alas – mallin ja alhaalta ylös - mallin parhaat puolet, ja näin luo tasapainon mallien välille. Yhdistämismalli kykenee hyödyntämään olemassa olevan perinnejärjestelmän sovellusinfrastruktuurin palvelukeskeisen arkkitehtuurin tavalla ja mahdollistaa enemmän prosesseja hyödyntäviä funktioita, jotka palvelevat liiketoiminnan tarpeita. Malli koostuu seitsemästä eri vaiheesta, jotka Kuva 6 osoittaa (Erl, 2005).

Vaihe 1. Alkaa ylhäältä alas analyysilla, joka painottuu palvelukeskeisen arkkitehtuurin avaintekijöihin (uudelleen käytettävyys, joustavuus, riippumattomuus ja avoimuus) ja yrityksen liiketoimintakokonaisuuksiin. Vaihe etenee kuten normaali ylhäältä alas – mallin analyysi, mutta hieman kapeammalla fokuksella.

Vaihe 2. Ylhäältä alas analyysin edetessä riittävän pitkälle alkaa palvelukeskeisen analyysin suorittaminen. Samalla kun palvelukeskeistä analyysia suoritetaan on ensimmäinen vaihe käynnissä taustalla. Mitä pidemmälle ensimmäisen vaiheen ylhäältä alas analyysi etenee, sitä enemmän kolmannen vaiheen palveluiden suunnittelu siitä hyötyy, sillä näin saadaan liiketoimintapalvelut paremmin vastaamaan liiketoimintamalleja.

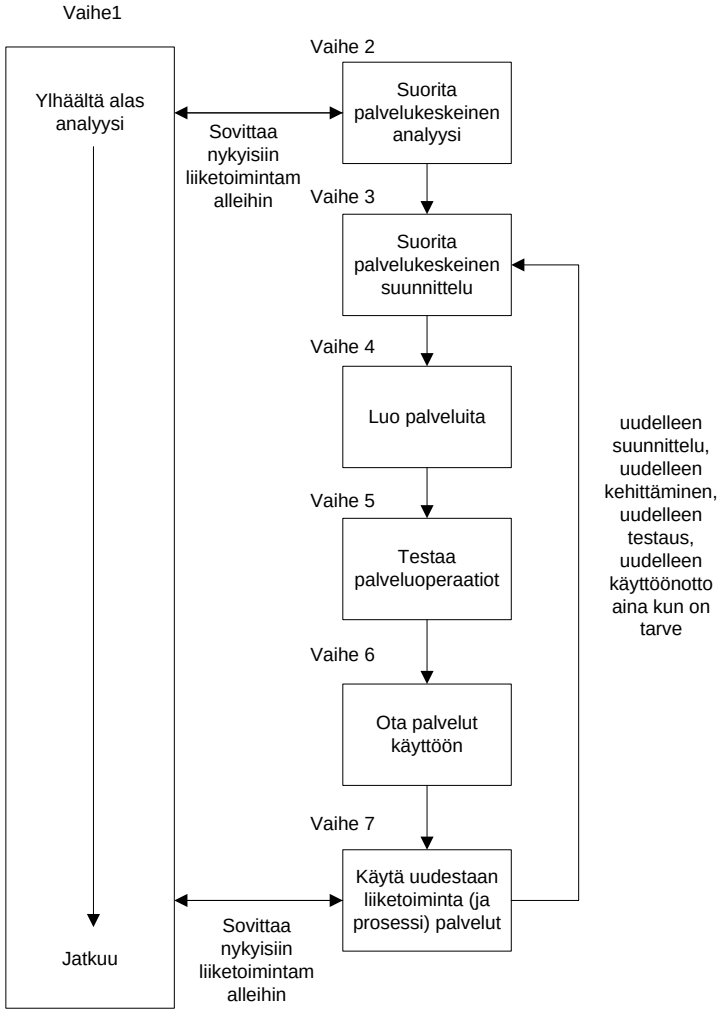
Vaihe 3. Suoritetaan palvelukeskeinen suunnittelu, jonka avulla määritellään ja suunnitellaan valitut palvelutasot ja yksittäiset palvelut osaksi palvelukeskeistä suunnitteluprosessia.

Vaihe 4, 5 ja 6. Luodaan, testataan ja otetaan palveluita käyttöön suorittamalla standardi testejä ja eri käyttöönottomenetelmiä.

Vaihe 7. Ensimmäisen vaiheen ylhäältä alas analyysi jatkaa etenemistään ja tässä vaiheessa tehdään katselmus liiketoimintapalveluihin. Katselmuksen tarkoituksena on verrata vaiheessa kolme tehtyjä suunnitelmia nykyisten liiketoimintamallien tilaan. Mikäli

eroavaisuuksia suunnittelun ja nykytilan välillä löytyy, hypätään takaisin vaiheeseen kolme niiden palveluiden osalta jotka eivät vastanneet suunnitelman tavoitteita.

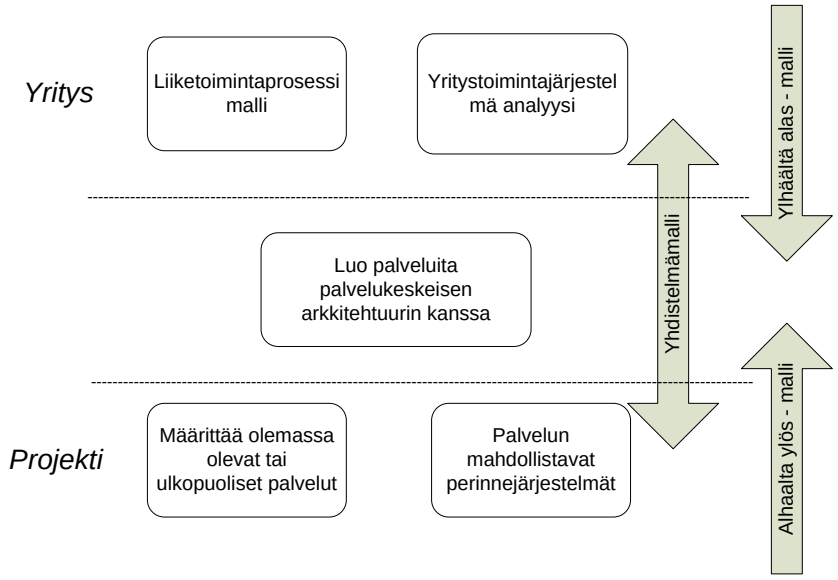
(Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005)



Kuva 6: Esimerkki yhdistämismallista (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005)

Yhdistämismalli mahdollistaa yritys- ja liiketoimintanäkökulman hyödyntämisen palvelukeskeisessä arkkitehtuurissa. Lähestymistapa tuottaa korkea tason liiketoimintaa sekä suunnittelee ja työstää kehitettyjä palveluita (Rosen;Lublinsky;Smith;& Balcer, 2008). Mallia alusta asti seuraava analysointi pitää huolen, että palvelut ovat muutettavissa, täydennettävissä ja käytettävissä

aina kun tarve vaatii (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005). Mallin toteutus ei kuitenkaan ole helppoa, sillä se vaatii suuren työmäärä, koska tavoitteena on sovittamaan yhteen kahden eri lähtökohdan omaavaa menetelmää (ylhäältä alas – malli ja alhaalta ylös - malli). Yhdistelmämalli vaaditaan saumatonta yhteistyötä liiketoiminta-analyytikkojen, sovellus arkkitehtien ja perinnejärjestelmä asiantuntijoiden kesken, jotta järjestelmästä saadaan toimimaan molempiin suuntiin (Rosen;Lublinsky;Smith;& Balcer, 2008).



Kuva 7: SOA projektin toteutusmallit (Rosen;Lublinsky;Smith;& Balcer, 2008)

Yhdistämismalli on menetelmä, joka yhdistää ylhäältä alas – mallin ja alhaalta ylös – mallin yhdeksi kokonaisuudeksi. Yhdistämismallin erityisenä vahvuutena on palveluiden suunnitteluprosessi, joka mahdollistaa jatkuvan ylhäältä alas analysoinnin sekä alhaalta ylös – mallin joustavuuden ja uudelleen käytettävyyden (An;Lee;& Jin, 2007). Yhdistämismallin soveltama jatkuva analysointi helpottaa toteutettujen palveluiden muuttamista, täydentämistä ja käyttämistä aina kun on tarve. Mallin avulla kehitettävä järjestelmä saadaan täyttämään organisaation lyhyenaikavälin sekä pitkänaikavälin vaatimusia paremmin (Erl, Service-Oriented Architecture: Concepts, Technology, and

Design, 2005). Yhdistämismalli kartoittaa ja kehittää samanaikaisesti sekä liiketoimintaprosesseja ja tavoitteita, että palveluiden ominaisuuksia ja suorituskykyä (Maurizio;Sager;Jones;Corbitt;& Girolami, 2008). Liiketoiminnan analysointi tuottaa palveluita, jotka omaavat korkean laadun ja pitkän eliniän. Alhaalta ylös – malli mahdollistaa nykyisten tietojärjestelmäinvestointien täysi mittaisen hyödyntämisen (Ramollari;Dranidis;& Simons, 2007).

Yhdistämismalli on varsin houkuttelevan kuuloinen vaihtoehdolta, mutta sen heikkoutena on kallis toteutus. Yhdistämismallin toteuttaminen vaatii huomattavasti enemmän aikaa ja rahaa, kuin että käyttäisiin ylhäältä alas – mallia tai alhaalta ylös – mallia. Yhdistämismalli on lisäksi monimutkaisempi toteuttaa kuin ylhäältä alas ja alhaalta ylös – malli, sillä sen pitää pystyä täyttämään kahden erilähtökohdan omaavan menetelmän vaatimukset (Erl, Service-Oriented Architecture: Concepts, Technology, and Design, 2005). Yhdistämismalli vaatii myös tarkempaa huomiointia, että liiketoiminta tavoitteet vastaavat ja sopivat jo olemassa olevaan kokonaisarkkitehtuuriin (Kohlmann, Service Identification and Design – A Hybrid Approach In Decomposed Financial Value Chains, 2008).

Yhdistämismalli voidaan soveltaa hyvin projekteihin missä suunnittelu lähtee liikkeelle liiketoiminnan ehdoilla, mutta halutaan myös hyödyntää nykyiset järjestelmäinvestoinnit. Yhdistämismalli on suositeltava ja kannattava menetelmä palveluiden suunnitteluun mikäli projektin aikataulu ei ole kireä eikä raha ole ongelmana. Mikäli aikaa toteutukseen on, niin tuottaa yhdistelmämalli pitkäikäistä ja laadukasta palvelukeskeistä arkkitehtuuria sekä joustavia ja uudelleen käytettäviä palveluita.

Alla olevaan taulukkoon olen kerännyt palveluiden suunnittelumenetelmien ominaisuuksia vertailun helpottamiseksi.

Taulukko 1: Palveluiden suunnittelumenetelmien vertailu

Ominaisuus	Ylhäältä alas - malli	Alhaalta ylös - malli	Yhdistämismalli
Soveltuvuus	Soveltuu hyvin projekteihin, jotka alkavat puhtaalta pöydältä tai mikäli aiotaan käyttää olemassa olevia palveluita uutta tarkoitusta varten.	Soveltuu menetelmäksi, mikäli halutaan hyödyntää olemassa olevia sovelluksia ja järjestelmä investointeja.	Lähtökohdasta riippumaton yhdistämismenetelmä, joka pystyy hyödyntämään ylhäältä alas – mallin ja alhaalta ylös – mallin ominaisuuksia.
Prosessi	Prosessissa koostuu seitsemästä vaiheesta. Analysointi ja suunnittelu ovat tärkeä osa prosessia.	Prosessissa koostuu viidestä vaiheesta ja on hyvin suoraviivainen.	Prosessissa koostuu seitsemästä vaiheesta. Mahdollistaa jatkuvan analysoinnin ja palveluiden muokkaamisen sekä täydentämisen.
Palvelut	Tuottaa liiketoiminta- ja sovelluspalveluita. Jokainen tuotettu palvelu on tarkkaan suunniteltu ja analysoitu.	Palvelut kääritään osaksi perinnejärjestelmää, jotka täydentävät sovellusvaatimuksia.	Mahdollistaa sekä ylhäältä alas – mallin että alhaalta ylös – mallin mukaisia palveluita.
Vahvuudet	Liiketoiminta näkökulma on vahvasti esillä. Sisältää paljon koordinointia, mikä mahdollistaa nykyaikaista ja johdonmukaista palvelukeskeistä arkkitehtuuria. Malli huolehtii, että tietojärjestelmän toiminnot on tehty tukemaan liiketoi-	Lähtee liikkeelle olemassa olevien sovellusten ja ratkaisujen pohjalta. Perinnejärjestelmään tehtyjen investointien hyödyntäminen on helppoa ja tehokasta. Palveluiden tuottaminen on helppoa ja nopeaa	Pystyy hyödyntämään ylhäältä alas – mallin ja alhaalta ylös – mallin parhaita ominaisuuksia. Palveluiden suunnitteluprosessi, joka mahdollistaa ylhäältä alas – mallin jatkuvan analysoinnin sekä alhaalta ylös – mallin jousta-

	mintavaatimuksia. Mahdollistaa hyvää työnkulun suunnittelua ja prosessien kehittämistä. Pyrkii kehittämään yrityksen yleisiä liiketoimintamalleja. Sovellusten ristiriidattomuus ja integrointimekanismi helpottavat palveluiden kehittämistä. Mahdollistaa korkean laadun omaavaa palvelukeskeistä arkkitehtuuria. Huolehtii yrityksen nykyisistä ja tulevaisuuden vaatimuksista.	Soveltuu ympäristöön, jossa teknologian kehitys on nopeaa. Malli mahdollistaa helposti uudelleen käytettäviä yksittäisiä ja yhdistettyjä palveluita. Työnkulut voidaan helposti yhdistää vastaamaan uusia vaatimuksia. Mahdollistaa teknologian ja taustajärjestelmien suorituskykyisen ja tehokkaan käytön, koska lähtökohtana toimivat nykyiset teknologiaratkaisut. Vahva alusta tekniikoille ja parhaille käytännöille on jo kehitetty.	vuuden ja uudelleen käytettävyyden Mallin avulla järjestelmä saadaan vastaamaan organisaation lyhyenaikavälin ja pitkänaikavälin vaatimuksiin paremmin. Kartoittaa ja kehittää samanaikaisesti sekä liiketoimintaprosesseja ja tavoitteita, että palveluiden ominaisuuksia ja suorituskykyä. Tuottaa korkean laadun ja pitkän eliniän omaavia palveluita Mahdollistaa nykyisten tietojärjestelmä investointien hyödyntämisen.
Heikkoudet	Työnkulku suunnitellaan etukäteen ja palvelut kehitetään sopimaan näihin työnkulkuihin, jolloin malli ei ole välttämättä tarpeeksi joustava käsittelemään uusia tilanteita ja prosessipäämääriä jotka vaativat eri työnkulkuja.	Ei pysty luomaan nykyaikaista ja edistyksellistä palvelukeskeistä arkkitehtuuria. Mallia vaivaa kapeakatseisuus ja arvioinnin puute laajemmissa kokonaisuuksissa. Liiketoiminta näkökul-	Vaatii huomattavasti aikaa ja rahaa. Monimutkainen toteutus. Vaatii tarkempaa huomiointia, että liiketoiminta tavoitteet vastaavat ja sopivat jo olemassa olevaan koko-

	Vaatii enemmän koordinointia kuin alhaalta ylös – malli. Voi aiheuttaa muutosvastarintaa, koska kyseessä on suurempi kertamuutos. Malli on aikaa ja rahaa vievä. Valmiusaste voi olla hankala hahmottaa.	ma jää vähälle huomiolle. Ei mahdollista yhtä joustavia liiketoimintamalleja ja nopeaa markkinoille saanti aikaa, kuin ylhäältä alas – malli. Liiketoimintavaatimukset jäävät usein vähälle huomiolle, eikä yhtymäkohtia liiketoiminnan kanssa välttämättä löydy.	naisarkkitehtuuriin.
Mihin	Esim. B2B-sovelluksen tekemiseen, jossa halutaan luoda yhteinen järjestelmä organisaation asiakkaiden sekä kumppanien välille.	Esim. WEB-sovellustekniikkaa hyödyntävän tilauspalvelun tekemiseen.	Liiketoiminta ratkaisuihin ja perinnejärjestelmä investointien hyödyntämiseen.

4 YHTEENVETO

Nykypäivän liiketoiminnan tehtäväkenttä on hyvin laaja-alaista, moninaista ja paljon päätöksentekoa vaativaa. Tämän myötä myös tietotekniikasta on tullut tärkeä päätöksenteon tuki ja turva. Järjestelmien yhteiskäyttö mahdollistaa erilaisten tietojen keräämisen laajalti organisaatiosta ja niiden esittämisen ja tulkitsemisen erilaisten mittareiden ja analyysien kautta. Tätä helpotusta ja kilpailuetua myös palvelukeskeinen arkkitehtuuri ja liiketoimintaprosessien hallinta tarjoavat.

Palvelukeskeinen arkkitehtuuri on kokonaisarkkitehtuuri- ja ajatusmalli, joka tarjoaa suuntaviivoja tietojärjestelmien suunnitteluun ja toteutukseen. Ydinajatuksena palvelukeskeisessä arkkitehtuurissa on, että tietojärjestelmien toteutetut palvelut tulisi olla riippuvaisia toisistaan mahdollisimman löyhin kytköksin ja riittävän yleisiksi toteutettuja. Tällä tavoin niiden uudelleenkäytöstä saadaan tehtyä mahdollisimman yksinkertaista. Palvelukeskeisessä arkkitehtuurissa pyritään kokonaisuuden parempaan hallintaan, sillä se mahdollistaa organisaation erilaisten tietojärjestelmien yhdistämisen, saaden ne toimimaan yhdessä samoja palveluja hyödyntäen. Liiketoimintaprosessien hallinnalla haetaan liiketoimintaprosessien jatkuvaa kehittymistä ja täyden mittakaavan hallintaa. Ei ole ihme että kehitystä halutaan, sillä liiketoimintaprosesseista on tullut yritysten tärkeintä omaisuutta ja kilpailukykyä ylläpitävä voimavara. Liiketoimintaprosessien hallinta näyttelee myös tärkeää osaa palvelukeskeisen arkkitehtuurin täysivaltaisessa hyödyntämisessä. Liiketoimintaprosessien hallinta mahdollistaa liiketoiminnan ja tietotekniikan nitoutumisen lähemmäksi toisiaan yhteisten päämäärien ja tavoitteiden saavuttamiseksi.

Palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistäminen voidaan toteuttamaan kolmen eri yhdistämismenetelmän avulla, jotka hieman eroavat toisistaan ja joilla jokaisella on omat hyvät ja huonot

puolensa. Ensimmäinen menetelmä on ylhäältä alas – malli. Mallissa lähdetään liikkeelle käyttäen hyväksi valmiita standardeja ja prosessimalleja palveluiden suunnittelussa. Ylhäältä alas – malli on hyvä vaihtoehto yhdistämismenetelmäksi, sillä sen avulla palvelut saadaan toteutettua alusta asti yhdenmukaisesti rajapintamääritysten kanssa. Malli vaatii kuitenkin alusta lähtien tarkkaa analysointia ja suunnittelua liiketoiminnan näkökulmasta, minkä myötä myös liiketoimintaprosessit saadaan paremmin hyödynnettyä. Vaikka tämä on yhdistämismenetelmänä paremman kuuloinen sikäli, että se antaa enemmän vapauksia suunnitteluun ja toteutukseen, voi se helposti johtaa tilanteeseen että massiivinen kertamuutos ei toteudu kunnolla tai määräaikojen puitteissa. Malli vaatii myös erityistä tarkkuutta uudelleenkäytön ja vanhojen sovellusten suhteen. Ylhäältä alas – malli sopii hyvin yhdistämismalliksi, mikäli aikaa ja rahaa riittää. Näin saataisiin toteutettua liiketoimintaa tukeva tietojärjestelmä, joka pystyttäisiin määrittämään alusta saakka omanlaiseksi.

Toisessa lähestymistavassa, eli alhaalta ylös – mallissa lähdetään liikkeelle olemassa olevista sovelluksista, joihin lähdetään suunnittelemaan ja toteuttamaan uusia rajapintoja. Tässä nopeammassa lähestymistavassa vaaditaan olemassa olevien järjestelmien ja ratkaisujen hyvää tuntemusta, sillä muuten toteutuksesta voi tulla monimutkainen ja hajanainen mikä tulisi kalliiksi pitkällä aikavälillä. Vaikka malli tuottaa tarkkaa ja uudelleenkäytettävää palvelukeskeistä arkkitehtuuria jättää se liiketoiminnan vähemmälle huomiolle, sillä kokonaisvaltainen liiketoiminnan analysointi ja suunnittelu puuttuvat prosessista. Näin ollen malli jättää pois yhden palvelukeskeisen arkkitehtuurin vahvuuksista, eli liiketoiminnan ja tietotekniikan onnistuneen yhdistämisen.

Kolmas, eli yhdistelmämalli on menetelmä joka hyödyntää kahden aiemman yhdistämismallin parhaita puolia keräten ominaisuuden yhden sapluunan alle. Se ottaa palveluiden luomisessa huomioon niin liiketoiminnan kuin nykyiset jo olemassa olevat järjestelmät. Tätä menetelmää hyödyntäen saavutetaankin

yleensä parhaiten halutut toiminnot, jotka miellyttävät kaikkia. Tuntuisi että yhdistelmämallia voisi soveltaa aina toteutukseen ja sillä saadaankin useimmiten laadukasta tulosta aikaiseksi. Kuitenkin heikkouksiakin mallista löytyy, sillä yhdistelmämallin toteutus on kallista, aikaa vievää ja monimutkaista.

Tutkimuksessani perehdyin selvittämään palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistämismenetelmiä. Yhdistämismenetelmät olivat ennestään jo jonkin verran tuttuja, mutta syvempää tuntemusta menetelmien käytöstä palvelukeskeisen arkkitehtuurin ja liiketoimintaprosessien hallinnan yhdistämiseen ei ollut. Tämän vuoksi jouduin nojautumaan vahvasti aiheeseen liittyvään kirjallisuuteen. Kirjallisuustarjonta osoittautui sikäli hankalaksi, että palvelukeskeisestä arkkitehtuurista ja liiketoimintaprosessien hallinnasta itsenäisinä osina löytyi varsin hyvin kirjallisuutta, mutta itse yhdistämisestä ja niiden yhteistoiminnasta hyvin rajoitetusti.

Tutkimuksen tuomista tuloksista sanoisin, että mitään suoranaista ohjenuoraa yhdistämismallin valintaan ei ole. Yhdistämismenetelmää valittaessa kysymys on enemmänkin tilannekohtaisesta harkinnasta, lähtökohdasta ja tavoiteltavasta päämäärästä, jotka sanelevat mitä menetelmää kannattaa milloinkin käyttää. Yhdistäminen vaatii hyvää alkuvalmistelua ja kurinalaista toteuttamista, että se tuottaa hyötyä organisaatiolle myös pitkällä aikavälillä. Jatkotutkimusta ajatellen erityisesti kiinnostaisi viedä yhdistämistä vielä askeleen eteenpäin paneutuen käyttöönottoon ja toteutukseen käytännössä sekä hyötyjen näkemiseen kunnolla organisaation näkökulmasta.

LÄHDELUETTELO

An, M.-J.;Lee, H.-C.;& Jin, H.-J. (2007). Design of the material control system based on service oriented architecture. *International Conference on Control, Automation and Systems 2007*, (ss. 978 - 983).

Ballard, C.;Abdel-Hamid, A.;Frankus, R.;Hasegawa, F.;Larrechart, J.;Leo, P.;ym. (2006). *Business Intelligence and Business Process Management.* IBM Redbooks.

BEA, W. P. (2006). *Extending the Business Value of SOA Through Business Process Management.* BEA Systems, Inc.

Bouillet, E.;Feblowitz, M.;Liu, Z.;Ranganathan, A.;& Riabov, A. (2008). A Faceted Requirements-Driven Approach to Service Design and Composition. *IEEE International Conference on Web Services* , (ss. 369 - 376).

Chang, J. F. (2006). *Business process management systems : strategy and implementation.* Auerbach Publications.

Cumberlidge, M. (2007). *Business Process Management with JBoss jBPM.* Packt Publishing.

Di Nitto, E.;Ghezzi, C.;Metzger, A.;Papazoglou, M.;& Pohl, K. (2008). A journey to highly dynamic, self-adaptive service-based applications. *ACM Automated Software Engineering* (15 , Issue 3-4), 313-341.

Erl, T. (2005). *Service-Oriented Architecture: Concepts, Technology, and Design.* Prentice Hall.

Erl, T. (2008). *SOA: Principles of Service Design.* The Prentice Hall.

Footen, J.;& Faust, J. (2008). *The Service-Oriented Media Enterprise.* Focal Press.

Hen, H. (2003). *What Is Service-Oriented Architecture?* Haettu 24. 2 2009 osoitteesta http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html

Josuttis, N. M. (2007). *SOA in Practice.* O'Reilly.

Juric, M. B.;Loganathan, R.;Sarang, P.;& Jennings, F. (2007). *SOA Approach to Integration.* Packt Publishing Ltd.

Kamoun, F. (2007). *A Roadmap towards the Convergence of Business Process Management and Service Oriented Architecture* (Vuosik. Volume 8). ACM.

Kohlmann, F. (2008). Service Identification and Design – A Hybrid Approach In Decomposed Financial Value Chains. *Proceedings of the 2nd Int'l Workshop on Enterprise Modelling and Information Systems Architectures - Concepts and Applications (EMISA'07).*

Kohlmann, F.;& Alt, R. (2009). Aligning Service Maps - A Methodological Approach from the Financial Industry. *Hawaii International Conference on System Sciences* (ss. 1 - 10). IEEE Computer Society.

Kulkarni, N. (18. 1 2007). *Infosys*. Haettu 28. 2 2009 osoitteesta http://www.infosysblogs.com/soa/2007/01/soa_terminologies_defined.html

Liu, Y.;Gorton, I.;& Zhu, L. (2007). Performance Prediction of Service-Oriented Applications based on an Enterprise Service Bus. *Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International , Volume 1, 24-27*, 327 - 334.

Maurizio, A.;Sager, J.;Jones, P.;Corbitt, G.;& Girolami, L. (2008). Service Oriented Architecture: Challenges for Business and Academia. *HICSS '08: Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences.* IEEE Computer Society.

Noel, J. (2005). *BPM and SOA: Better Together.* IBM Corporation.

Papazoglou, M. P.;& Jan van den Heuvel, W. (2007). *Service oriented architectures: approaches, technologies and research issues.* (V. 1. The VLDB Journal, Toim.) Springer-Verlag.

Papazoglou, M. P.;& van den Heuvel, W.-J. (2006). Service-oriented design and development methodology. *International Journal of Web Engineering and Technology , Volume 2* (4), 412 - 442.

Ramollari, E.;Dranidis, D.;& Simons, A. J. (2007). A Survey of Service Oriented Development Methodologies. *2nd European Young Researchers Workshop on Service Oriented Computing.* Leicester: Software Engineering & Service-Oriented Technologies research group.

Rosen, M.;Lublinsky, B.;Smith, K. T.;& Balcer, M. J. (2008). *Applied SOA.* Wiley Publishing.

Schepers, T. G.;Iacob, M. E.;& Van Eck, P. A. (2008). A lifecycle approach to SOA governance. *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing* (ss. 1055-1061). ACM.

Shan, T. C. (2004). Building a Service-Oriented eBanking Platform. *SCC '04: Proceedings of the 2004 IEEE International Conference on Services Computing* (ss. 237 - 244). IEEE Computer Society.

Sward, R. E.;& Whitacre, K. J. (2008). A multi-language service-oriented architecture using an enterprise service bus. *Ada Lett. , 28* (3), 85 - 90.

Weske, M. (2007). *Business Process Management: Concepts, Languages, Architectures.* Springer.