



Sovellusalustan optimointi

*Pasi Mäkinen, Microsoft Oy
v. 1.0, toukokuu 2007*

Sovellusalusta siihen liittyvine käytäntöineen ja prosesseineen on isommassa organisaatiossa kokonaisuus, jonka kehittämisestä vastaaminen on haastavaa.

- Miten varmistaa että liiketoiminnan tarvitsemat uudet ratkaisut ovat joustavasti ja kustannustehokkaasti toteutettavissa?*
- Onko sovellusalusta teknisesti ajan tasalla?*
- Miten sovellusten elinkaarta pitäisi tukea?*
- Onko käyttäjäkokemus huomioituna sovelluskehitysprosessissa?*

Sovellusalusta on omia järjestelmiä toteuttavalle tai teettävälle organisaatiolle strateginen päätös. Sovellusalustan kehittämiseen kannattaa siis suhtautua suunnitelmallisesti ja pitkäjänteisesti. Tässä dokumentissa esitellään malli sovellusalustan optimoinnille ja analysoidaan päätöksentekoon liittyvä näkökulmia.

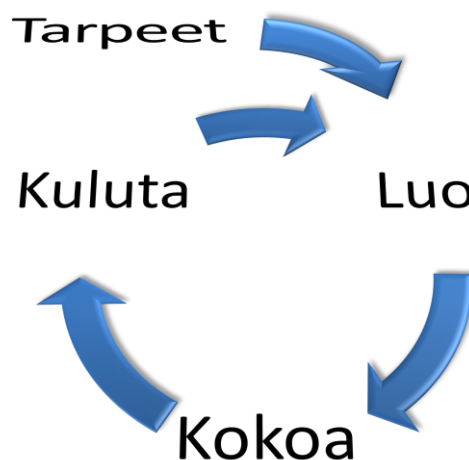
Sovellusalan vaatimukset

Organisaation toimintatapojen muuttaminen vaatii muutoksia toiminnan mittareihin, prosesseihin, työntekijöiden rooleihin, käytettävissä olevien tietojen valikoimaan ja työvälineisiin. Useimmat edellä luetelluista osatekijöistä pohjautuvat nykypäivänä tietotekniisiin sovelluksiin. Toiminnan muuttaminen ja tehostaminen vaatii muutoksia sovelluksiin joita käytämme työvälineinä ja tietolähteinä. Kyky muuttaa ja sopeuttaa sovelluksia toiminnan mukaan on edellytys nopealle ja tehokkaalle organisaatiolle.

Nopea muutосkyky ei sinällään ole mikään haaste jos resursseja on käytössä määrätömästi. Reaalimaailmassa joudumme kuitenkin toimimaan rajallisten taloudellisten ja teknisten resurssien puitteissa. Sovellusalan kehittäminen edellyttää taitoa sovittaa yhteen ristiriitaisia liiketoimintasovellusten tarpeita, tiukkoja taloudellisia rajoitteita ja näkemystä teknologian kehityksestä teknisten umpikujien välttämiseksi.

Sovellusalan kehittämisen käynnistäjänä toimii usein halu uudistaa alustaa meneillään olevan arkkitehtuurityylin mukaiseksi. Tällä hetkellä yleisin tavoite on palveluarkkitehtuuri (Service Oriented Architecture - SOA). Palveluarkkitehtuurin soveltamiseen Microsoft suosittelee vaiheittaista liiketoimintatarpeen mukaan tapahtuvaa etenemistä jota kuvataan Real World SOA dokumentissa.¹ Palveluarkkitehtuuria kannattaa rakentaa käytännön liiketoimintahankkeissa erillisten teknislähtöisten projektien tai liiketoiminnan kokonaisvaltaiseen muuttamiseen tähtäävien mammuttiprojektien sijaan. Palveluarkkitehtuurin rakentaminen on toistuva sykli, jossa ensin luodaan palveluja nykyjärjestelmiä kuorruttamalla tai uusia rakentamalla, koostetaan palveluista ylemmän tason palveluita ja prosesseja ja tuodaan prosessit käyttäjille heidän kannaltaan tarkoituksenmukaisesti.

Vaiheittainen toteutustapa perustelee ja rahoittaa palveluarkkitehtuurin selvillä liiketoiminnan hyödyillä. Jokaisen iteraatiokierroksen jälkeen myös aikaisemmin toteutettujen palvelujen arvo kasvaa niiden käytön lisääntyessä. Syntyy verkkoefekti, jossa tietyn kynnyksen jälkeen kokonaisuuden tuottama hyöty kasvaa voimakkaasti.

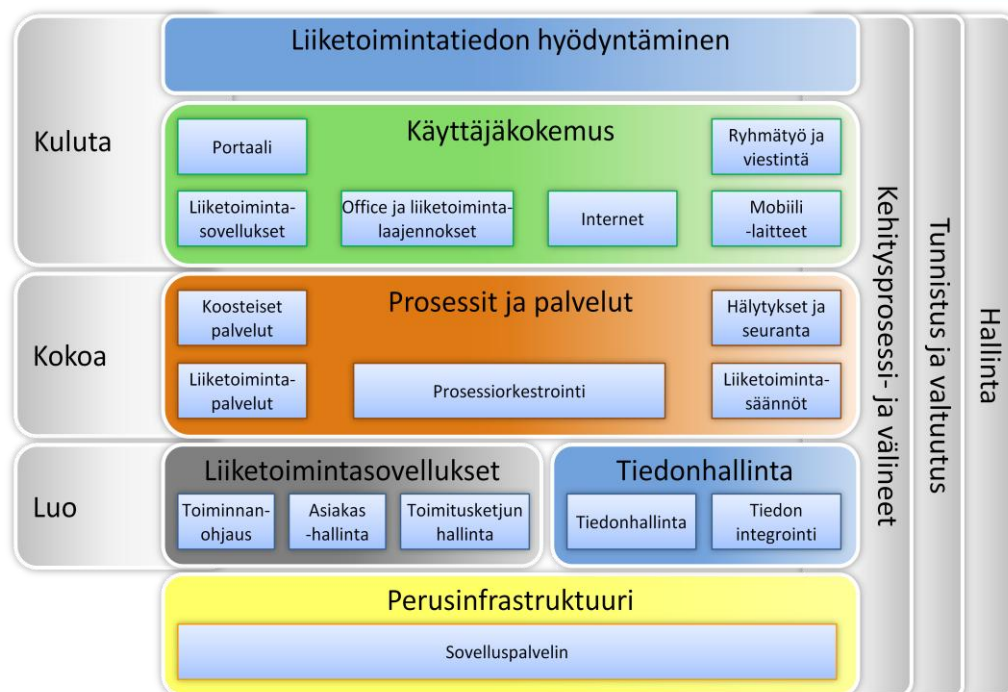


Kuva 1 Real World SOA prosessi

Sovellusalan määritelmä

Sovellusalan voi tarkoittaa suppeimmillaan .NET tai J2EE ajoympäristöä. Open Group määrittelee sovellusalan palvelukokonaisuudet laajemmin, mutta hyvin teknologialähtöisesti.² Organisaation sovelluskokonaisuuden ja arkkitehtuurin kannalta tarkasteltuna sovellusalan kannattaa määritellä kuitenkin hieman liiketoimintalähtöisemmin.

Sovellusalan on se teknologiasta, ohjeistuksesta, käytännöistä ja ihmisistä koostuva systeemi, jonka avulla organisaatio kehittää, käyttää ja ylläpitää sovelluksia.



Kuva 2 Palvelupohjainen sovellusalan

Käyttäjäkokemus ratkaisee viimekädessä sovellusinvestoinneista ja prosessien tehostamisesta saatavan hyödyn. Jos palvelut ja prosessit on organisoitu käytännön työntekijöiden kannalta väärin tai tarjotaan hankalan ja vaikeasti opittavan käyttöliittymän kautta, menetetään osa tavoitelluista hyödyistä. Samalla kasvatetaan tarpeettomasti sovellusten koulutus- ja tukikustannuksia.

Palvelujen käytön näkökulmasta sovellusalan määrittelee ne käyttöliittymäteknologiat joita sovelletaan eri käyttäjäryhmille ja kanaville palveluja tarjottaessa. Palvelujen käyttöä helpottaa jos käyttöliittymät kyetään tarjoamaan käyttäjäroolin mukaan sovitettuina ja mahdollisimman tutulla käyttöliittymällä.

Hyvä sovellusalan tarjoaa kattavan kirjon käyttöliittymätyylejä ja mahdollisuuden nopeaan ja kustannustehokkaaseen käyttöliittymien kokoonpanoon.

Liiketoimintatiedon hyödyntäminen on päätöksenteon näkökulma prosesseihin ja palveluihin. Tiedon ei pitäisi olla pelkästään ylimmän johdon ja asiaan vihkiytyneiden analyytikoiden saatavilla. Tarjoamalla raportteja ja analysointityökaluja laajalle joukolle työntekijöitä, maksimoidaan perusjärjestelmiin kerrytetyn tiedon hyöty päätöksenteossa eri organisaatioilla.

Prosessien integrointi ja palveluarkkitehtuuri mahdollistavat järjestelmäratkaisujen muuttamisen samassa tahdissa liiketoiminnan muutostarpeen kanssa. Oleellimmat ominaisuudet sovelluslustralle ovat tuki XML-muotoisen tiedon käsittelylle ja web services hajautustekniikoille. Prosessien orkestroinnin kannalta alustan on syytä tukea monipuolisesti erilaisia prosessimalleja kuten sekventiaalisia prosesseja, tilakoneita ja liiketoimintasääntöjä.

Tiedonhallinta varmistaa liiketoimintatiedon käytettävyyden ja turvallisuuden sekä mahdollistaa tiedon joustavan integroinnin ja käytön eri palveluissa ja prosesseissa.

Kehitysprosessi ja välineet on oleellinen näkökulma niin sovelluksia itse tekeväille kuin sovelluskehitystä tilaajan näkökulmasta tarkastelevalle. Myös tilaajan kannattaa arvioida sovelluksen elinkaaren hallintaa ja toimittajan projektitiimin kanssa kommunikointia helpottavat ratkaisut. Sovelluskehitysvälineet ja -kehikot ratkaisevat niin reagoitavuuden liiketoiminnan muutoksiin, kehitysprojektien tuottavuuden ja riskitömyyden kuin kehittäjäresurssien saatavuuden.

Tietoturva niin mekanismeina, tuotteina kuin prosesseina on huomioitava osana sovelluslustrua turvallisen kokonaisuuden takaamiseksi.

Hallittavuus vaikuttaa eniten sovelluslustrun elinkaarikustannuksiin. Sovelluslustrun käyttöä ollessa vähintään 10 vuotta, kannattaa kiinnittää huomiota sovelluslustrun hallintavälineisiin ja tukeen avoimille hallintastandardeille.

Sovelluslustrun kypsyysmalli

Sovelluslustrun siihen liittyvine käytäntöineen ja prosesseineen on vähänkään isomassa yrityksessä kokonaisuus, jonka kehittämisestä vastaaminen on haastavaa. Sovelluslustrun kehittämiseen hyvän viitekehyksen tarjoaa Microsoftin sovelluslustrun optimointimalli Application Platform Infrastructure Optimization (APIO)³. Sovelluslustrun optimointimalli tähtää parempaan kykyyn tukea liiketoimintaa. Optimointi mahdollistaa prosessien ja sovellusten paremman integroitavuuden, kehittämisen tuottavuuden kasvattamisen ja investointien liiketoimintahyötyjen mitattavuuden.



Kuva 3 Sovellusalan kypsyydet

Sovellusalan optimointimalli kuvaa neljä kypsyydetasoa.

Perustasolla sovellusala ja ratkaistaan tapauskohtaisesti ja työvälit valitaan usein projektikohtaisesti. Tyypillisesti organisaatiolla ei ole yhtenäistä sovelluskehitysprosessia ja jokainen kehittäjä toimii kaikissa eri rooleissa. Projektien yli tapahtuvaa arkkitehtuuriohjausta ei yleensä ole ja järjestelmät integroidaan tapauskohtaisesti. Liiketoiminnan näkökulmasta sovellusala on pelkkä kustannuskomponentti.

Standardoidulla tasolla sovellusalustoja ohjataan standardein ja kustannuksissa ollaan perustasoa tehokkaampia. Raportointiin on rakennettu yksikkökohtaisia mittaristoja ja tietovarastoja. Prosessiautomaatiota ja -integraatiota on toteutettu yksiköiden sisällä. Käytössä on moderneja sovelluskehitysvälineitä ja sovelluskehitysprosessi kuvaa selkeästi eri roolit.

Edistyneellä tasolla prosesseja automatisoidaan ja integroidaan koko organisaation tasolla. Raportointi on integroitu käyttäjien näkökulmasta yhdeksi kokonaisuudeksi ja käytössä on yhteinen tietovarastointiratkaisu. Sovelluspalvelimet ovat virtualisoitu resurssien käytön tehostamiseksi ja tiedonhallinnalle on yhteiset standardit ja käytännöt.

Dynaamisella tasolla liiketoiminnan raportointi on upotettu sovelluksiin ja käytössä on ajantasaisen reagoimisen mahdollistava Business Activity Monitoring -ratkaisu. Arkkitehtuuria suunnitellaan ja ohjataan järjestelmällisesti. Liiketoiminnan vaatimat järjestelmämuutokset voidaan toteuttaa nopeasti ja kustannustehokkaasti palvelupohjaisen arkkitehtuurin avulla. Tiedonhallinta on integroitu yli yksiköiden Master Data Management -ratkaisulla. Sovelluskehitystiimien käytössä on sovelluskehitysprosessin mukaan ohjautuvat välineet jotka automatisoivat ja tehostavat projektin viestintää ja auttavat hallitsemaan riskejä.

Sovellusalan optimoinnin lisäksi Microsoftilla on vastaavat mallit infrastruktuurin (Core Infrastructure Optimization⁴) ja tuottavuusvälineiden (Business Productivity Infrastructure Optimization⁵) optimointiin.

Optimointimallista on Internetissä selaimella käytettävä itsearviointityökalu⁶, jonka avulla on voi arvioida nykytilannetta ja suunnitella millä kypsyydetasolla sovellusalan tulisi olla. Työkalu koostuu kokoelmasta kysymyksiä, joiden perusteella sovellusalan nykytila kartoitetaan. Työkalu on teknologiariippumaton, joten sitä kannattaa kokeilla vaikka ei Microsoft teknologiaa tällä hetkellä hyödynnä.

Itsearviointityökalu arvioi sovellusalan neljän osa-alueen mukaan:

- Ohjelmistokehitys (Development)
 - sovelluskehitysprojektien merkitys ja laajuus, kehitysvälineet, prosessit, roolit
- Prosessien hallinta ja palveluarkkitehtuuri (BPM, SOA)
 - arkkitehtuuri, integrointi, web services, liiketoimintaprosessit
- Tiedonhallinta (Data Management)
 - käytännöt, roolit ja standardilinjaukset
- Liiketoimintatiedon hyödyntäminen (Business Intelligence)
 - tiedon saatavuus, mittarit ja loppukäyttäjät palvelut

Itsearviointityökalu ei tällä hetkellä kata sovellusalan käyttäjäkokemuksen ja käyttöliittymien (User Experience) osa-alueita.

Arvioinnissa sovellusalan kypsyys jaotellaan edellä kuvattujen mukaisesti perus, standardoitu, edistynyt ja dynaaminen -tasoihin. Kunkin osa-alueen tasolta seuraavalle siirtymisen osalta on kuvattu toimenpiteet kypsyystason nostamiseksi ja nostamisesta saatavat hyödyt. Kuvauksista pääsee edelleen porautumaan Microsoftin menetelmiin, tuotteisiin ja teknologioihin, joita muutoksessa voi hyödyntää.

Sovellusalan valintakriteerit

Millaisia kriteereitä kannattaisi sitten arkkitehdin näkökulmasta käyttää sovellusalan ja sen komponenttien valinnassa? Aina ei tietysti ole mahdollista eikä toivottavaakaan ennalta lukita organisaation soveltamia teknologioita. Perustellut liiketoimintatarpeet menevät luonnollisesti teknologiarajausten edelle.

Käyttövarmuus

Käyttövarmuus rakentuu korkean käytettävyyden laite- ja ohjelmistoratkaisujen lisäksi oikeista tietotekniikan suunnittelun, johtamisen ja tuotannon toimintatavoista.

Korkeaa käytettävyyttä tukevia sovellusalan ratkaisuja ovat mm. Windows Server 2003 palvelinten työkuormien ryvästyminen, SQL Server 2005 tietokantojen peilaus, BizTalk 2006 ympäristön monistaminen, IIS v.6.0 web-palvelinten monistaminen ja ISA Serverin käyttö kuormanjakoon monistetussa ympäristössä. Windows Server 2003 -pohjaisilla ratkaisulla on saavutettu Stratus Technologiesin vikasietoisilla palvelinlaitteilla tilastollisesti merkittävästä määrästä asiakkaita 99.9998 % käyttöaste.⁷

Parhaita käytäntöjä Microsoft-pohjaisen ratkaisun käyttövarmuuden varmistamiseen on koottu infrastruktuurin osalta Windows Server System Reference Architecture⁸ (WSSRA) -malliin ja toimintatapojen osalta ITIL-pohjaiseen Microsoft Operations Framework⁹ -malliin. WSSRA referenssiarkkitehtuuri sisältää dokumentoituja parhaita käytäntöjä sovellusalan perusinfrastruktuurin rakentamiseen ja Operations Framework määrittelee prosessit ja toimintatavat joilla sovellusalan palvelukyky pystytään pitämään halutulla tasolla.

Arkkitehtuuriohjeistusta käyttövarmuuden huomioinnista hajautetuissa .NET-pohjaisissa ratkaisuissa löytyy mm. Architecture Journalin artikkelista Reliability in Connected Systems¹⁰.

Lisää kolmansien osapuolien ja asiakkaiden näkemyksiä käyttövarmuudesta löytyy Get The Facts-sivuston osiosta Reliability¹¹.

Yhteentoimivuus

Yhteentoimivuuden arviointi alustaa valittaessa varmistaa sen hyvän integroituvuuden muihin alustoihin ja väistämättä eteen tuleviin tapauskohtaisiin ratkaisuihin. Yhteentoimivuuteen ei riitä pelkkä standardien rajapintojen soveltaminen, vaan yhteentoimivuus on otettava huomioon jo tuotteiden suunnitteluvaiheessa ja yhteentoimivuutta on testattava.

Käytännön yhteentoimivuuden testausta Microsoftilla tehdään mm. Bill Hilfin¹² johtamassa Open Source Software Interoperability Lab -yksikössä, joka ylläpitää Port25¹³-sivustoa open source yhteentoimivuudesta. Microsoft osallistuu myös aktiivisesti Web Services -standardien kehittämiseen ja niiden yhteentoimivuuden varmistamiseen.

Yhteentoimivuutta edistetään myös saattamalla kehitettyjä rajapintoja, tietomuotoja, lähdekoodia ja patenteja asiakkaiden, kumppanien ja kilpailijoiden käyttöön. Hyvä esimerkki tästä on Microsoft Open Specification Promise¹⁴, joka määrittelee ne web service, virtualisointi, tietoturva ja Office XML standardit, joiden soveltamisen ja implementoinnin suhteen Microsoft takaa vapaan oikeuden patenttiansa hyödyntämiseen.

Lisätietoja yhteentoimivuuden varmistamiseksi tehtävästä työstä löytyy Microsoftin Interoperability¹⁵ -verkkosivuilta. Kolmansien osapuolien ja asiakkaiden näkemyksiä yhteentoimivuudesta löytyy Get The Facts-sivuston osiosta Interoperability¹⁶.

Käytännön esimerkki yhteentoimivuudesta .NET- ja suurkoneympäristön kesken on Eläke-Fennian yritysportaali¹⁷. Palveluja kehitetään Microsoft .NET -arkkitehtuurissa, mutta ne toimivat yhteistyössä IBM-suurkonemaailman ja Java-arkkitehtuurin kanssa. Kytkentä syntyi varsin kevyesti, koska molempia maailmoja yhdistää sama web services -hajautusteknologia.

Suorituskyky

Suorituskykyä ja skaalattavuutta voidaan mitata monin tavoin. Palvelinkäyttöjärjestelmälle ja palvelinohjelmistoille on tärkeää niin yksittäisen palvelimen resurssien tehokas käyttö, moniprosessoriarkkitehtuurien hyödyntäminen kuin palvelinkomponenttien monistettavuus. Windows Server 2003 sisältää monia suorituskykyä parantavia ominaisuuksia¹⁸. Ohessa on muutama esimerkki toteutettujen ratkaisujen tapahtumakäsittelyn läpäisykyvystä sekä tietokantojen koosta ja rivimääristä.

Tapahtumakäsittelyn suorituskyvyltään suuria toteutuksia ovat:

- Yhdysvaltain teknologiapörssi NASDAQ:n Market Data Dissemination System järjestelmä käsittelee 5000 tapahtumaa sekunnissa ja 100 000 päivittäistä kyselyä tietokantaa vasten. Järjestelmä testattiin 64 000 tapahtumaa sekunnissa kuormala¹⁹
- Lontoon pörssin Infolect markkinatietojärjestelmä käsittelee 3500 tapahtumaa sekunnissa.²⁰

Lisätietoja Windows Server 2003 suorituskyvystä ja skaalautuvuudesta on koottu Windows Server 2003 Performance and Scalability -sivustolle.²¹

Winter Corporationin viimeksi vuonna 2005 tekemän TopTen Surveyn²² mukaan SQL Serverillä pyörii:

- Rivimäärältään maailman kolmanneksi suurin tapahtumatietokanta.
- Kannan koolla mitattuna 6., 7. ja 10. suurimmat tapahtumatietokannat.
- Datamäärältään maailman 8. suurin tietovarastointiratkaisu.
- Kahdeksan kymmenestä rivimäärältään suurimmasta tapahtumatietokannasta Windows palvelinalustalla.

Tietokannan koolla mitattuna referenssejä suurista toteutuksista on koottu omalle sivustolle.²³ Lisää kolmansien osapuolien ja asiakkaiden näkemyksiä suorituskyvystä löytyy Get The Facts-sivuston osiosta Performance.²⁴

Turvallisuus

Laajasti käsitettynä turvallisuus sisältää niin käyttövarmuuden ja toimittajariskin kuin tietoturvan ja yksityisyyden suojan. Nämä neljä näkökulmaa sisältyvät Microsoftin Trustworthy Computing²⁵ -ohjelmaan, joka ohjaa Microsoftin toimintaa ja tuotekehitystä. Ohjelma määrittelee mm. tietoturvaan liittyvät Security Development Lifecycle²⁶ -käytännöt tuotteen koko elinkaarelle suunnittelusta ylläpitoon. Elinkaarimalli aloittaa tietoturvan suunnittelun jo tuotteen vaatimusmäärittelystä ja vaatii suunnitteluvaiheessa kattavan uhkamallinnuksen sekä määrittelee tarkistuspisteet, joissa tuotteen etenemistä seuraavaan tuotekehitysprosessin vaiheeseen kontrolloidaan tietoturvanäkökulmasta.

Microsoft on myös julkaissut asiakkaidensa ja kumppaniensa käyttöön uhkamallinnustyökalun, joka sisältää valmiin uhkakirjaston vastakeinoineen.²⁷ Lisää kolmansien osapuolien ja asiakkaiden näkemyksiä käyttövarmuudesta löytyy Get The Facts-sivuston osiosta Security.²⁸

Toimittajariski

Toimittajariskillä tarkoitetaan teknologiaan tai tuotteeseen liittyviä sen toimittaneesta organisaatiosta johtuvia riskejä. Onko organisaatio vielä olemassa viiden vuoden kuluttua? Onko toimittaja sitoutunut tuen suhteen johonkin konkreettiseen aikaan?

Microsoftin yritys- ja ohjelmistokehityskäyttöön tarkoitettujen tuotteiden tuki noudattaa yhtenäistä elinkaarimallia²⁹. Yritys- ja ohjelmistokehitystuotteita tuetaan vähintään 10 vuotta. Ensimmäiset viisi vuotta ovat mainstream-tukijaksoa, jonka puitteissa tuotteisiin toimitetaan tietoturvapäivitykset, muut korjaukset sekä erikseen sovittaessa maksullista lisätukea. Seuraavat viisi vuotta kuuluvat extended-tukijaksoon, jonka aikana toimitetaan edelleen tietoturvapäivitykset sekä tukisopimuksen solmineille asiakkaille muut korjauspäivitykset. Luonnollisesti extended-tukijakson ajalle voi erikseen sopia maksullisesta lisätuesta. Itsepalveluperiaatteella toimivat tuotteiden online-tukipalvelut ovat käytettävissä vähintään samat 10 vuotta. Online-tukea artikkeleiden ja ohjeiden muodossa löytyy edelleen esimerkiksi SQL Server 6.0:lle, jonka mainstream-tukijakso loppui vuonna 1999 sekä Windows for Workgroups 3.1:lle, joka julkaistiin vuonna 1992.

Taloudelliset perusteet

Yksittäisen tietojärjestelmäinvestoinnin taloudellisia perusteita mietittäessä kannattaa keskittyä investoinnin tuoton laskentaan. Kustannusvertailu ilman niistä saatavien hyötyjen rahallista arviointia johtaa kurjuuden maksimointiin. Sovellusalan valinnassa rahallisia hyötyjä on vaikeampi arvioida. Todelliset hyödyt syntyvät rakennettaessa alustan päälle liiketoimintaratkaisuja.

Elinkaarikustannusten minimointi

Sovellusalan kustannuksia kannattaa tarkastella elinkaarikustannusmallin (TCO - Total Cost of Ownership) avulla. Tyypillisesti järjestelmän elinkaarikustannuksista alle 10 % riippuu laitekustannuksista ja vastaava osuus ohjelmistolisenssien kustannuksista. Suurin kustannustekijä on kuitenkin järjestelmän käytön aikaiset henkilökustannukset. Hankala ja monista erillisistä palasista rakentuva hallintatyökalujen kokonaisuus nostaa elinkaarikustannuksia monesti enemmän kuin säästö lisenssikustannuksissa.

Microsoft pyrkii tehostamaan järjestelmien hallittavuutta tarjoamalla eri palvelintuotteisiin standardoidut järjestelmähallintarajapinnat ja tekemällä integraation omiin järjestelmähallinnan ratkaisuihin mahdollisimman valmiiksi. Räätelöityjen yrityssovellusten hallittavuutta parannetaan sovelluskehitys- ja hallintatyökalujen kesken yhteisellä System Definition Model -kuvauskielellä, joka mahdollistaa hajautetun yrityssovelluksen rakenteen ja toiminnan kommunikoinnin suoraan kehitysprojektin työvälineistä tuotannon järjestelmähallinnan työvälineisiin.

Kolmansien osapuolien laatimia TCO-arvioita ja ohjeita sekä asiakkaiden näkemyksiä Microsoft-alustan elinkaarikustannuksista löytyy Get The Facts-sivuston osiosta Total Cost of Ownership.³⁰

Tuoton maksimointi

Arkkitehdin perusosaamista on kyky kommunikoida ratkaisu eri sidosryhmille heidän omalla kielellään. Taustastamme johtuen teknisille sidosryhmille viestiminen on useimmiten helpompaa kuin organisaatiossa ylöspäin liiketoimintahyödyistä ja rahasta kiinnostuneelle johdolle. Arkkitehdin perusosaamiseen tulisikin kuulua investoinnin tuottolaskelmien (Return of Investment, ROI) tekeminen. Strukturoitu ja järjestelmälli-

nen lähestymistapa on laadukkaan ja uskottavan laskelman edellytys. Osaltaan laatua voi parantaa soveltamalla valmiita ROI-laskimia.

Nucleus Researchiltä löytyy kattava kokoelma valmiita ROI-laskimia mm. Visual Studio Team Systemille, BizTalk Serverille, SQL Serverille ja Dynamics tuotteille.³¹ Laskinten soveltamisesta löytyy myös runsaasti ohjeistusta. Hyvä aloitusohje on Quick Reference Guide³², joka sisältää monia yleispäteviä ohjeita laskelmien tekemiseen.

Ennen Excelin avaamista laskelman tekemiseksi, kannattaa arvioida investoinnin tuottopotentiaalia muutaman peukalosäännön avulla:

- **Laajuus** – mitä suurempi käyttäjämäärä on, sitä suurempi on tuottomahdollisuus.
- **Toistettavuus** – usein tarvittava ratkaisu tuottaa enemmän hyötyä.

Tuottolaskelmassa kannattaa maksimoida hyötyjä kustannusten minimoinnin sijaan. Muita hyvää kannattavuutta ennustavia tekijöitä ovat:

- **Nykykustannukset** – nykyisen toimintatavan suuret kustannukset merkitsevät hyvää tehostamispotentiaalia.
- **Tiedon uudelleenkäyttö** – saman tiedon uudelleenkäyttö kertaa hyötyä.
- **Yhteistoiminta** – yhteistyön automatisointi ja tehostaminen ihmisten välillä parantaa tuloksia ja vähentää aikaa vieviä palavereja.

Kun varsinaista investoinnin tuottolaskelmaa päästään tekemään, hankkeen kustannukset ovat yleensä helppoja arvioida. Tulokinnanvaraa on yleensä vain infrastruktuurin ja jaettujen resurssien jyvittämisessä eri hankkeille. Yleensä riittää että miettii hankit- tiinko resurssi tämän hankkeen vuoksi.

Hankkeen hyötyjen arviointi aiheuttaa tyypillisesti enemmän keskustelua. Selkeät tuotannolliset tuottavuushyödyt ovat uskottavia. Usein hyödyt ovat kuitenkin henkilötyön tuottavuuden kasvua. Hyvä periaate on että ajansäästön voi laskea täysimääräisesti tuotoksi vain tiukasti johdettujen ja puhtaasti suorittavien tehtävien osalta. Muiden roolien osalta kannattaa perussääntönä soveltaa 50 % hyötyastetta. Henkilötyön tuottavuushyödyn laskemista vastustetaan usein sillä argumentilla, ettei ketään kuitenkaan irtisanota ja hyöty on siis näennäistä.

Työn tuottavuuden kasvu on kuitenkin teknologiainvestointien päähyöty. Jos ei usko työn tuottavuuden kasvuun, miksi siis edes hankkia tietokoneita työntekijöille?

Jos tuottolaskelma näyttää ensimmäisellä yrittämällä toivottomalta, kannattaa tarkistaa onko olettamissa käytön laajuudesta ja toistettavuudesta virheitä tai onko kustannukset arvioitu epärealistisen suuriksi. Lievästi alakanttiin olevan tuottolaskelman optimointiin kannattaa kokeilla seuraavia keinoja:

Kustannusten aikataulutus – jos hankkeen kaikki kustannukset on kohdistettu laskelman ensimmäiselle vuodelle, tarkista voidaanko osa kustannuksista siirtää eteenpäin.

Neuvottele hinnasta – hankkeesta riippuen pienikin kustannusten alenema voi parantaa tuottolaskelmaa huomattavasti.

Rytmitä kustannuksia käytön mukaan – aikatauluttamalla investointi ratkaisun käyttäjämäärän kasvun mukaan voidaan osa kuluista kattaa tulorahoituksella.

Vaiheista hyödyn perusteella – vaiheistamalle hanke siten että suurimman hyödyn tuottavat ratkaisun osat otetaan käyttöön mahdollisimman aikaisin, ratkaisu ehtii keräämään enemmän tuottoa laskelman aikajänteellä.

Tarkista korjauskertoimesi – jos olet käyttänyt korjauskertoimia hyötyjen arviointiin, tarkista etteivät ne ole liian varovaisia. Korjauskertoimen voi muuttaa olettamasta mitatuksi suureeksi tekemällä siitä kyselyn organisaatiossa.

Järjestelmällisellä laskelmalla varmistat ettei organisaatiosi jätä hyödyntämättä kannattavaa teknologiaa.

Yhteenveto

Sovellusalan osien valinnan eri kriteereistä voisi jokaisesta kirjoittaa sivukaupalla tarinaa. Tämä tiivistelmä toimii lähinnä katsauksena peruskriteereihin. Valintatilanteessa arvioitavana on myös joukko organisaatiokohtaisia vaatimuksia, joiden läpikäynti ei tämän tyyppisessä yleisessä tarkastelussa ole mahdollista.

Sovellusalan kehittäminen vaatii laaja-alaista osaamista liiketoiminnan tarpeista ja teknologioista, kykyä sovittaa toisiinsa nähden ristiriitaisia vaatimuksia sekä taitoa perustella linjauksia niin taloudellisin kuin teknisin argumentein. Toivottavasti tästä yhteenvedosta ja lisätietolinkeistä on apua sovellusalan kehittämiseen ja Microsoftin sovellusalan sopivuuden arviointiin.

Viitteet

¹ Reaalimaailman SOA whitepaper,

http://www.mspost.fi/microsoft/Reaalimaailman_SOA.pdf

² Open Group TOGAF Application Platform Taxonomy,

http://www.opengroup.org/public/arch/p3/trm/trm_dtail_intro.htm

³ Application Platform Infrastructure Optimization,

<http://www.microsoft.com/business/peopleready/appplat/default.aspx>

⁴ Core Infrastructure Optimization,

<http://www.microsoft.com/business/peopleready/coreinfra/default.aspx>

⁵ Business Productivity Infrastructure Optimization,

<http://www.microsoft.com/business/peopleready/bizinfra/default.aspx>

⁶ Sovellusalan itsearviointityökalu,

<http://www.microsoft.com/business/peopleready/appplat/ac/apio.aspx>

⁷ Stratus ftServer line closes in on perfect reliability, Stratus Technologies, lokakuu 2003,

<http://www.codix.gr/downloads/STRATUS20030804.pdf>

- ⁸ Windows Server System Reference Architecture,
<http://www.microsoft.com/technet/solutionaccelerators/wssra/raguide/default.aspx>
- ⁹ Microsoft Operations Framework,
<http://www.microsoft.com/technet/solutionaccelerators/cits/mo/mof/default.aspx>
- ¹⁰ The Architecture Journal, Numero 8, Heinäkuu 2006, Reliability in Connected Systems,
<http://msdn2.microsoft.com/en-us/arcjournal/bb245677.aspx>
- ¹¹ Get the Facts, Reliability -sivut,
<http://www.microsoft.com/windowsserver/facts/topics/reliability.mspix>
- ¹² Microsoft, Open Source and Interoperability, Bill Hilfin esitystaltiointi, TechEd IT Forum,
<http://www.microsoft.com/emea/itshowtime/sessionh.aspx?videoid=347>
- ¹³ Microsoft Open Source Lab ylläpitämä Port25 –sivusto,
<http://port25.technet.com/>
- ¹⁴ Microsoft Open Specification Promise,
<http://www.microsoft.com/interop/osp/default.mspix>
- ¹⁵ Microsoft Interoperability -sivut,
<http://www.microsoft.com/interop/>
- ¹⁶ Get the Facts, Interoperability -sivut,
<http://www.microsoft.com/windowsserver/facts/topics/interoperability.mspix>
- ¹⁷ Eläke-Fennian yritysportaali,
<http://www.microsoft.com/finland/business/casestudies/net2apr03.mspix>
- ¹⁸ Windows Server 2003 Performance and Scalability, lokakuu 2004,
<http://download.microsoft.com/download/5/8/7/587312a9-d421-4f07-9d9e-b6515720082b/perfscal.doc>
- ¹⁹ Case Study: NASDAQ Stock Market,
<http://www.microsoft.com/casestudies/casestudy.aspx?casestudyid=49271>
- ²⁰ Case Study: London Stock Exchange,
<http://www.microsoft.com/windowsserver/facts/casestudies/lse.mspix>
- ²¹ Windows Server 2003 Performance and Scalability -sivusto,
<http://www.microsoft.com/windowsserver2003/evaluation/performance/perfscaling.mspix>
- ²² Winter Corporationin tutkimus SQL Server tietokannan skaalautuvuudesta,
<http://www.microsoft.com/sql/prodinfo/compare/wintercorp-survey.mspix>
- ²³ SQL Server 2005 Enterprise Capabilities -sivusto,
<http://www.microsoft.com/sql/bigdata>
- ²⁴ Get the Facts, Performance –sivut,
<http://www.microsoft.com/windowsserver/facts/topics/performance.mspix>
- ²⁵ Microsoft Trustworthy Computing -ohjelma,
<http://www.microsoft.com/mscorp/twc>
- ²⁶ Microsoft Trustworthy Computing Security Development Lifecycle,
<http://msdn2.microsoft.com/en-us/library/ms995349.aspx>
- ²⁷ Microsoft Threat Analysis & Modeling v2.1.2 työkalu,
<http://www.microsoft.com/downloads/details.aspx?FamilyID=59888078-9daf-4e96-b7d1-944703479451&displaylang=en>
- ²⁸ Get the Fact, Security –sivut.
<http://www.microsoft.com/windowsserver/facts/topics/security.mspix>
- ²⁹ Microsoftin yritystuotteiden elinkaarimalli,
<http://support.microsoft.com/lifecycle/>
- ³⁰ Get the Facts, Total Cost of Ownership,
<http://www.microsoft.com/windowsserver/facts/topics/tco.mspix>
- ³¹ Microsoft ROI Support Center, Nucleus Research,
http://www.nucleusresearch.com/msfsupport_txt.html
- ³² Research Note, Measuring Return on Investment Quick Reference Guide, 2007 Nucleus Research
<http://www.nucleusresearch.com/research/b20.pdf>