Ohjelmistotuotannon massaräätälöinnistä tietojärjestelmätuotannossa

Antti Sand

Tampereen yliopisto
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
Pro gradu -tutkielma
Ohjaaja: Mikko Ruohonen
Joulukuu 2012

Tampereen yliopisto
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
Tekijän Nimi: Antti Sand
Pro gradu -tutkielma, 56 sivua
Joulukuu 2012

Massaräätälöinti yhdistää massatuotannon kustannustehokkuuden ja nopeat valmistusajat räätälöityjen tuotteiden yksilöllisyyteen. Tietojärjestelmien massaräätälöinti voi auttaa saavuttamaan paremman laadun, nopeamman projektin läpivientiajan, sekä kustannussäästöjä. Monet tietojärjestelmien massaräätälöinnin piirteet myös auttavat tuottavaa yritystä reagoimaan nopeammin toimintaympäristön muutoksiin.

Tietojärjestelmien kysynnän jatkuva kasvu, kilpailun koventuminen ja ohjelmistoprojektien korkea epäonnistumisprosentti lisäävät paineita kehittää uusia tuotantotapoja alati muuttuvaan toimintaympäristöön.

Tarkastelen tässä Pro Gradu –työssä tietojärjestelmien massaräätälöintiä käsitteellisellä analyysillä.

Avainsanat ja -sanonnat: tietojärjestelmien massaräätälöinti, massaräätälöinti, tuotantolinja-ajattelu.

Sisällys

1. Johdanto

Tietojärjestelmien tuottajien on pystyttävä palvelemaan asiakkaitaan tehokkaammin. Useiden projektien laajuus vaatii suunnittelijoiltaan suurta tarkkuutta ja kykyä nähdä suuria kokonaisuuksia. Projektien kasvaessa laajuudessaan myös aikataulupaineet kovenevat. Tietojärjestelmätyötä on paljon, mutta alan nuoruuden ja tunnusomaisen nopean muutoksen vuoksi parhaat toimintatavat ja uudet ratkaisumallit laahaavat usein perässä. Tästä kertoo surullista kieltään ohjelmistoprojektien melko korkea epäonnistumisprosentti.

”Ohjelmistoprojektit ovat edelleen myöhässä, yli budjetin ja vaikeita ennustaa. Välillä koko projekti epäonnistuu ennen kuin saadaan mitään tuotetta aikaiseksi.” Näin kirjoittaa Trident Data Systemsin John S. Reel [1999]. Hän jatkaa kuvaten sitä, miten noin viisikymmentävuotisen ohjelmistotuotannon historian aikana on kuljettu läpi ainakin neljän ohjelmointikielisukupolven ja kolmen ison tuotantoparadigman. Vaikka nykyään pidetäänkin lukemattomia seminaareja ja opiskelijoille tarjotaan useita kursseja ohjelmistotuotannosta ja vaikka monia standardeja on tuotu työpaikoille, emme silti pysty onnistuneesti ja toistuvasti tuomaan ideoitamme ohjelmistotuotteiksi.

Reel mainitsee myös surullisen tilaston. Vaikka ohjelmistoprojektien epäonnistumisprosentti onkin laskenut, suuriin vaikeuksiin ajautuvien projektien määrä on noussut liki puolella.

Esimerkiksi tarkan aikataulun ja kustannusarvion antaminen on ongelmallista, sillä lähes kaikkiin ohjelmistoprojekteihin liittyy piirteitä, joita ei voida etukäteen tietää. Monet tutkimukset antavatkin ohjelmistoprojektien onnistumisprosentiksi vain noin 20 prosenttia, eli noin 80 prosenttia ohjelmistoprojekteista epäonnistuu saavuttamaan ainakin jonkin sille asetetun tavoitteen. [Armour, 2007]

Etenkin Suomen kohdalla tilanne on merkittävä, sillä meillä ICT (*Information and communications technology)* ja sen liitännäisalat olivat 1990-luvun puolivälistä 2000-luvun alkuun nopeimmin kasvanut talouden sektori ja kokonaisuudessaan suurin yritysklusteri [Hernesniemi, 2010]. Toisin sanoen ohjelmistoprojektien onnistumisella on suuri merkitys Suomen taloudelle. Tietojärjestelmätuotannon ja ohjelmistotuotannon kasvu onkin tunnistettu kansalliseksi tavoitteeksi Suomessa [Arjen tietoyhteiskunnan neuvottelukunta, 2010].

Samalla yrityksistä on tullut yhä enemmän riippuvaisia tieto- ja viestintätekniikasta [Virtanen, 2011]. Tällöin tietojärjestelmäprojektin epäonnistumisella voi olla suuri vaikutus organisaation toimintakykyyn.

Vastatakseen ajan haasteisiin tietojärjestelmätutkimus pyrkii kehittämään uusia toimintamalleja projektien läpimenoajan lyhentämiseksi ja laadun varmistamiseksi. Tässä

Pro Gradu –työssä tarkastelen tietojärjestelmien massaräätälöintiä vastauksena näihin ongelmakohtiin.

Massaräätälöinti tarkoittaa tuotevariaatioiden luomista yhdistettynä massatuotannon tehokkuuteen [Ahoniemi *et al.*, 2007]. Massaräätälöintiä on käytetty tuotantolinja-tuotannossa, esimerkiksi autotehtaissa, jo pitkään [mm. Teresko, 1994], mutta viime aikoina sitä on alettu hyödyntämään myös tietojärjestelmätuotannossa [mm. Highsmith and Cockburn, 2001].

Internetistä voi esimerkiksi tilata tietokoneita, vaatteita ja mainoslahjatuotteita tarpeidensa ja mittojensa mukaan. Esimerkiksi Dell ja Apple tarjoavat verkkokaupoissaan mahdollisuuden kustomoida tilaamaansa tietokonetta esimerkiksi lisäämällä siihen muistia, tai valitsemalla perinteisen kiintolevyn tilalle nopeamman SSD – aseman [Apple, 2012; Dell, 2012].

Kuluttajien vaatimuksen alemmista hankinta- ja käyttökustannuksista ohjelmistotuotteille ovat tavallaan ymmärrettäviä. Kuluttajat ovat tottuneet siihen, että viisi vuotta vanhan tietokoneen tilalle saa usein monin verroin tehokkaamman mallin jopa halvemmalla. Valmistuskustannusten kulurakenteiden eroja tietokonekomponenttien tuotantolinja-massatuotannon ja ohjelmistojen henkilötyön välillä ei ilmeisesti ymmärretä, mahdollisesti siitä syystä, että koko tietokone ohjelmistoineen saatetaan ymmärtää yhdeksi kokonaisuudeksi, joka saapuu samasta kaupasta usein samassa paketissa.

Tarkastelen tässä Pro Gradu –työssä tietojärjestelmien massaräätälöintiä noudattaen osin Ahoniemi *et al.* kirjassa *Massaräätälöinnillä* kilpailukykyä [2007] esiteltyjä piirteitä. Kyseisessä kirjassa tarkastellaan massaräätälöintiä suomalaisten teknologiateollisuusyritysten näkökulmasta, mutta sama jaottelu ja samanlaiset huomioitavat asiat nousevat paljolti esiin myös tietojärjestelmätuotannon parissa.

Käyn läpi massakustomoinnin motivaattoreita, vaatimuksia, odotuksia ja haasteita suomalaisen toimijan näkökulmasta globaalissa kilpailukentässä.

Pyrin myös esittämään mahdollisimman kattavasti tämänhetkiset tutkimukset massakustomoinnista ja tarjoamaan hyvän näköalan viimeisimpiin julkaisuihin alalta.

Otan tutkimuksessani myös kriittisen asenteen massaräätälöintiä kohtaan. Uskon, että massaräätälöintiä, niin kuin useita muitakin kustannustehokkuutta mahdollisesti parantavia malleja, saatetaan joissain yrityksissä pitää kertaratkaisuna yrityksen ongelmiin, ”hopealuotina”, joka korjaa kerralla kaikki yrityksen ulkoiset ja sisäiset ongelmakohdat.

Tulen tarkastelemaan massaräätälöinnin suuria alkukustannuksia, asiakkaalle mahdollisesti aiheutuvaa massahämmennystä, sekä massaräätälöinnin rajoitteita.

Tutkimusalueena tietojärjestelmien massaräätälöinti on laaja. Jo kysyttäessä millä tavoin ja kuinka räätälöityvä tuote on, tarkasteltavia piirteitä on monia. Kaikkia tietojärjestelmien massaräätälöintiin liittyviä kysymyksiä en tule tarkastelemaan ja

niissäkin kysymyksissä, joita tarkastelen, voitaisiin esittää monia lisäkysymyksiä ja mennä asian tarkastelussa vieläkin syvemmälle.

2. Tutkimuskysymys, tutkimusmenetelmä ja aikaisemmat tutkimukset

2.1. Tutkimuskysymys

Tutkimuksessa pyrin tarkastelemaan tietojärjestelmien massaräätälöinnin teorioita alan tutkimusjulkaisujen ja kirjallisuuden pohjalta etsien samalla yhtymäkohtia ja eroavaisuuksia teollisuuden massaräätälöinnin teorioihin.

Tietojärjestelmien massaräätälöinti on verrattain uusi ilmiö tutkimusmielessä. Malleja, jotka *a posteriori* voitaisiin katsoa toteuttavan tietojärjestelmien massaräätälöintiä, on luotu jo vuosikymmeniä sitten, mutta tuolloin niiden motivaattorina oli työskentelyn tehostaminen ja kustannustehokkuuden parantaminen.

Sittemmin tietojärjestelmien massaräätälöintiä on alettu tarkastelemaan teollisuudesta lainatun massaräätälöinnin mallin pohjalta.

Toimialojen eroavaisuuksien vuoksi voidaan olettaa, että vaikka monia yhteneväisyyksiä teollisuuden massaräätälöinnin ja tietojärjestelmien massaräätälöinnin välille voidaankin löytää, saattaa niistä osa olla väkinäisiä, yleistäviä, tai jopa virheellisiä. Immateriaalituotanto on monilta kohdin kuitenkin niin erilainen pelikenttä, kuin perinteinen tuotanto, että suorien analogioiden vetäminen on liian kapeakatseista.

Noiden eroavaisuuksien vuoksi tietojärjestelmien massaräätälöinnissä on sellaisia erityiskysymyksiä, joihin jo olemassa oleva teollisuuden massaräätälöinnin malli ei sellaisenaan pysty suoraan vastaamaan.

Tietojärjestelmien massaräätälöinnin tutkimukselle on tarvetta. Tätä kirjoittaessani tietojärjestelmien massaräätälöinnin tutkiminen on vielä melko vähäistä ja todella nuorta – mallit saattavat olla jo vakiintuneita ja yleisesti hyväksyttyjä, mutta niiltä puuttuu vuosien aktiivisen tutkimisen suoma kokemuksellinen ja kokeellinen pohja.

Monesti lisäksi kokeellinen pohja ja mittaustulokset tuodaan erityiskenttään viittaamalla suoraan teollisuuden massaräätälöintiin, jolloin todellinen validius jää vielä saavuttamatta.

2.2. Tutkimusmetodi

Tutkimusmenetelmänäni on käsiteanalyyttinen metodi, joka on valittu Järvisen ja Järvisen *Tutkimustyön metodeista* [2011] –kirjan pohjalta. Tutkimusmetodinani on etsiä yhtäläisyyksiä toisaalta teollisuuden massaräätälöinnistä ja toisaalta tietojärjestelmien massaräätälöinnistä.

Tutkimusta varten olen koonnut viimeisen kymmenen vuoden aikana julkaistuja tutkimuksia, artikkeleita ja oppikirjoja massaräätälöinnistä, tuotantolinja-ajattelusta, sekä yrityskulttuurin muutoksesta omaksuttaessa massaräätälöinnin ideologiaa.

Tutkimuksen alkupuoli keskittyy tarkastelemaan teollisuuden massaräätälöinnin oppien yhteneväisyyksiä ja eroja verrattuna tietojärjestelmien massaräätälöintiin. Sen jälkeen käsittelen puhtaasti tietojärjestelmien massaräätälöinnistä sekä ohjelmistojen massaräätälöinnistä tehtyjä tutkimuksia ja teorioita nähdäkseni paremmin ne eroavaisuudet, joita tämä erityisala olettamukseni mukaan tarvitsee.

Tulen käyttämään rinnakkain termejä tietojärjestelmätuotanto ja ohjelmistotuotanto. Vaikka ohjelmisto onkin vain yksi osa sähköisestä tietojärjestelmästä, kohdennan tutkimukseni ohjelmistotuotannon massaräätälöintiin. Tarkastelen ohjelmistojen rinnalla kuitenkin myös tietojärjestelmiä.

2.3. Aikaisemmat tutkimukset

Lähden rakentamaan tutkimustani Ahoniemen *et al.* [2007] kirjan *Massaräätälöinnillä kilpailukykyä* tarjoaman viitekehyksen pohjalta. Kirja on kohdistettu teollisuuden massaräätälöintiin ja sen on Teknologiateollisuus ry:n julkaisuja. Tämän kirjan tarjoaman lähestymistavan mukana vertaan tietojärjestelmätuotannon eroavaisuuksia perinteiseen teollisuuteen ja pyrin löytämään analogioita ja toisaalta myös eroavaisuuksia.

Tulen viittaamaan toistuvasti Frank Pillerin julkaisuihin. Piller on Aachen yliopiston teknologian ja innovaatioiden johtamisen ryhmän professori ja on kirjoittanut paljon massaräätälöinnistä niin perinteisemmän teollisuuden, kuin uusien teknologioidenkin parissa. Pilleriä voitaisiinkin pitää koko massaräätälöinnin tutkimusalueen johtavana asiantuntijana.

Massaräätälöinnin rajoitteista on kirjoittanut mm. Paul Zipkin, tuotantolinja-ajattelusta tietojärjestelmätuotannossa puolestaan Charles W. Krueger.

3. Lyhyt johdatus tutkimuksen keskeisiin osa-alueisiin

3.1. Tietojärjestelmät

Puhuttaessa tietojärjestelmistä ihmiset usein tulkitsevat sen tarkoittavan samaa kuin tietokoneohjelma. Todellisuudessa tietojärjestelmä ja tietokoneohjelma eivät kuitenkaan ole synonyymejä toisilleen.

Tietokoneohjelma on hyvin selkeästi rajoittuva ohjelma, joka toimii jossain tietyssä järjestelmässä noudattaen jotain tiettyä ohjeistusta. Puhuttaessa pelkästään tietokoneohjelmasta emme ota huomioon sen käyttöympäristöä tai käyttäjää.

Tietojärjestelmä puolestaan Business Dictionaryn [2012] mukaan kattaa huomattavasti suuremman kokonaisuuden sisältäen ihmiset, jotka sitä käyttävät, tiedonkäsittelylaitteet, tiedonsiirtolaitteet, sekä itse tietokoneohjelman.

Tietokoneohjelma voi toimia itsenäisesti omassa ympäristössään ilman ihmistä ja ilman tietosyötettä. Se voidaan kerran käynnistää määrätyssä ympäristössä ja ohjata toimimaan itsenäisesti niin kauan, kunnes toimintaympäristö muuttuu niin, ettei toiminta enää ole mahdollista. Esimerkiksi ohjelma voidaan jättää suorittamaan silmukkaa (engl. *loop*) ilman mitään päättöehtoa. Tällöin ohjelman toiminta jatkuu, kunnes toimintaympäristön muutos, esimerkiksi sähkökatkos, poistaa toimintaedellytyksen.

Kuvatunlaisessa tietokoneohjelmassa ihmisen vuorovaikutukselle (engl. *interaction*) ei ole suorituksen aikana mitään tarvetta, kuten ei myöskään ole tarvetta uudelle tiedolle tiedonsiirtolaitteiden kautta.

Yleisenä esimerkkinä tietojärjestelmästä voitaisiin käyttää kirjastoa. Tässä tapauksessa tietojärjestelmään voidaan katsoa kuuluvan virkailijan, asiakkaan, tietokoneita, palvelimia (engl. *server*), tietokantoja (engl. *database*), ja tiedonsyöttövälineitä, kuten näppäimistö ja nykypäivänä erilaiset verkkosivuilla olevat sähköiset lomakkeet.

Asiakas voi tarkastella kirjaston valikoimaa tietokoneellaan Internetissä toimivan verkkosivun kautta. Verkkosivu saa sisältönsä palvelimelta, joka kysyy kirjaston kulloisenkin valikoiman tietokannasta.

Asiakas voi sitten löydettyään haluamansa kirjan tiedot tehdä kyseiselle kirjalle varauksen käyttämällä verkkosivulla olevaa sähköistä lomaketta. Kirjaston virkailija saa sitten tiedon esimerkiksi sähköpostiviestin kaltaisella sähköisellä ilmoituksella ja voi noutaa kirjan hyllystä odottamaan hakijaansa. Samalla, riippuen toteutustavasta, virkailija voi merkata tietokoneohjelmaan kyseisen opuksen olevan tallessa, mutta poissa hyllystä ja tietokoneohjelma voi päivittää kirjan tilanteesta merkinnän tietokantaan.

Tietojärjestelmät ovat siis laajempi kokonaisuus kuin tietokoneohjelmat ja vaikka nykypäivänä tietojärjestelmiin kuuluvatkin olennaisena osana tietojenkäsittelylaitteet, (sähköiset)tietokannat ja (sähköiset)tiedonsyöttölaitteet, voidaan myös vanhanaikaista

kortistoa pitää tietojärjestelmänä. Siinä tietokanta on korttihyllykkö, tiedonsyöttövälineenä voidaan käyttää kynää ja tiedonhaku tapahtuu ihmislähtöisesti noudattaen lajitteluperusteita.

Tässä tutkielmassa keskityn kuitenkin nykyaikaisiin, sähköisiin tietojärjestelmiin, jota kautta myös pääpaino on tietokoneohjelmissa ja tietokannoissa, koska esimerkiksi tiedonsyöttölaitteista valinta painikkeiden, kosketuspinnan, eleiden ja puheentunnistuksen välillä on siirtynyt paljolti osaksi käyttäjän preferenssejä sen sijaan, että vain yhtä niistä voisi käyttää.

On kuitenkin hyvä huomioida jo varhaisessa vaiheessa se, että massaräätälöinnillä tarkoitetaan lähdemateriaalissa laajempaa kokonaisuutta, kuin vain tuotetta tai tuotantotapaa kapeassa kontekstissa. Käsittelen lähdemateriaalin pohjalta myös yrityskulttuurin, laajemman kontekstin toimintatapojen ja esimerkiksi asiakas- ja yhteistyösuhteiden roolia massaräätälöinnin onnistumisessa.

3.2. Tietojärjestelmätutkimus

Tarkastelen tässä tutkimuksessa tietojärjestelmiä suurimmaksi osaksi ohjelmistotuotannon kautta. Ohjelmistojen tuotantolinja-ajattelusta ja tietojärjestelmien massaräätälöinnistä löytyy monia tieteellisiä julkaisuja ja julkaisumäärissä mitattuna nämä aiheet ovat puhuttaneet enemmän vasta viimeisen kahdenkymmenen vuoden ajan. Kumpikaan ajattelutapa ei kuitenkaan ole pohjimmiltaan uusi eikä kumpaakaan voida ajatella informaatioteknologian itse synnyttäminä uusina ajatuksina, sillä ne lainaavat paljon muilta aloita.

Ohjelmistoarkkitehtuurin varttuessa se on usein omaksunut toimintatapoja kypsemmiltä aloilta. Ohjelmistoarkkitehtuurin toimintatavoista ja ratkaisumalleista löytyy samankaltaisuuksia muilta, vanhemmilta aloilta, joissa niihin on päästy ehkä vuosisatojen yrityksen ja erehdyksen kautta.

Esimerkiksi rakennusarkkitehtuuri tehosti rakentamisen nopeutta ja kustannustehokkuutta tuomalla modulaariset kappaleet rakennustyömaille. Samalla tavalla kokonaisuuksien jakaminen yhdisteltäviin ja yhteisen rajapinnan toteuttaviin ohjelmistokomponentteihin on vakiintunut ohjelmistoarkkitehtuurin perusajatukseksi.

Tietojärjestelmätutkimus on tärkeää tietojärjestelmätuotannon projektien onnistumisprosentin parantamiseksi. Kuten johdannostakin ilmeni, tietojärjestelmätuotanto on ilmeisen ongelmallinen ja riskialtis toimiala. The Standish Group on yksi tietojärjestelmähankkeiden onnistumisen seuraaja ja raportoija.

The Standish Group julkaisi ensimmäisen CHAOS raporttinsa vuonna 1994. Sen tilastojen mukaan tietojärjestelmähankkeista täysin onnistuneita oli vain 16 prosenttia kaikista tutkituista tapauksista. Toiset 53 prosenttia tietojärjestelmähankkeista toimitettiin

joko puutteellisilla toiminnallisuuksilla, myöhässä aikataulusta, yli budjetin tai kaikkia näitä. [Standish Groug, 2003]

Raporttia on sen jälkeen päivitetty ja sen validiutta on kyseenalaistettu useasti [mm. Eveleens ja Verhoef, 2008]. Kuitenkin heidän julkaisemansa luvut ovat hyvin linjassa esimerkiksi Armourin [2007] julkaisemien lukujen kanssa. Samalla nähdään suoraan, ettei onnistumisprosentti ole noussut 13 vuodessa kuin noin 4 prosenttiyksikköä.

Ottamatta kantaa The Standish Groupin CHAOS raportin esitettyihin puolueellisuuksiin, voidaan kuitenkin todeta, että tietojärjestelmätuotannossa, kuten kaikessa muussakin tuotannossa, on vielä varaa parantaa.

3.3. Massaräätälöinnistä

Da Silveira *et al.* [2001] määrittelevät massaräätälöinnin liittyvän tapaan tarjota yksilöllisesti suunniteltuja tuotteita ja palveluita jokaiselle asiakkaalle korkean prosessien joustavuuden ja integraation avulla.

Massaräätälöinnin lyhyt kuvaus voisi olla vaikkapa toimintatapa, jossa pyritään yhdistämään massatuotannon kustannustehokkuus ja nopeus käsityötuotannon uniikkeihin, räätälöityihin tuotteisiin.

Molemmissa toimintatavoissa itsessään on omat selkeät hyötynsä ja vastaavasti omat voimakkaan rajoitteensa.

Massatuotannossa pystytään valmistamaan selkeällä raaka-ainevarastolla tiettyä tuotetta nopeasti ja kustannustehokkaasti. Sen kilpailuetu tulee suurista valmistemääristä, nopeasta läpivientiajasta ja melko hyvästä laadusta.

Kun massatuotantoa harjoittava yritys tilaa raaka-aineita, se voi suurten homogeenisten tilausmäärien vuoksi neuvotella itselleen edullisemman hinnan kuin perinteiset käsityöyritykset. Vakiintuneet toimintatavat ja hiotut prosessit puolestaan parantavat tuotteen laatua ja vähentävät virheellisten tuotteiden suhteellista määrää.

Vaikka massatuotannossa valmis tuote pystytäänkin valmistamaan verrattain nopeasti, menee tuotantolinjan pystyttämiseen ja sen asetuksiin usein paljon enemmän aikaa. Lisäksi tuotantoa on usein pystyttävä pyörittämään katkeamatta, jotta toiminta olisi kannattavaa. Tällöin työnseisaukset ja laitehuollot voivat tuottaa yritykselle huomattavia ansionmenetyksiä.

Massatuotannolle on luonteenomaista prosessien jatkuva kehittäminen. Vaikka lopputuote pyritäänkin pitämään samankaltaisena, kohdistetaan tuotantoprosessiin muutoksia ja parannuksia toiminnan virtaviivaistamiseen ja tuottamattoman työn minimointiin. Tässä tuotantomallissa markkinat määrittelevät hintatason ja kilpailukyky on pystyttävä synnyttämään omien prosessien tehostamisella. [Ahoniemi *et al.*, 2007]

Perinteisessä käsityössä puolestaan voidaan valmistaa asiakkaalle hänen haluamansa tuote asiakkaan toiveiden mukaisesti. Kilpailuetuna on uniikki tuote, joka vastaa

paremmin asiakkaan toiveisiin. Etenkin yksityishenkilöt ovat usein halukkaita maksamaan enemmän juuri heidän tarpeisiinsa tehdystä tuotteesta ja monet pitävät ainutlaatuisuuttakin merkittävänä lisäarvon tuottajana.

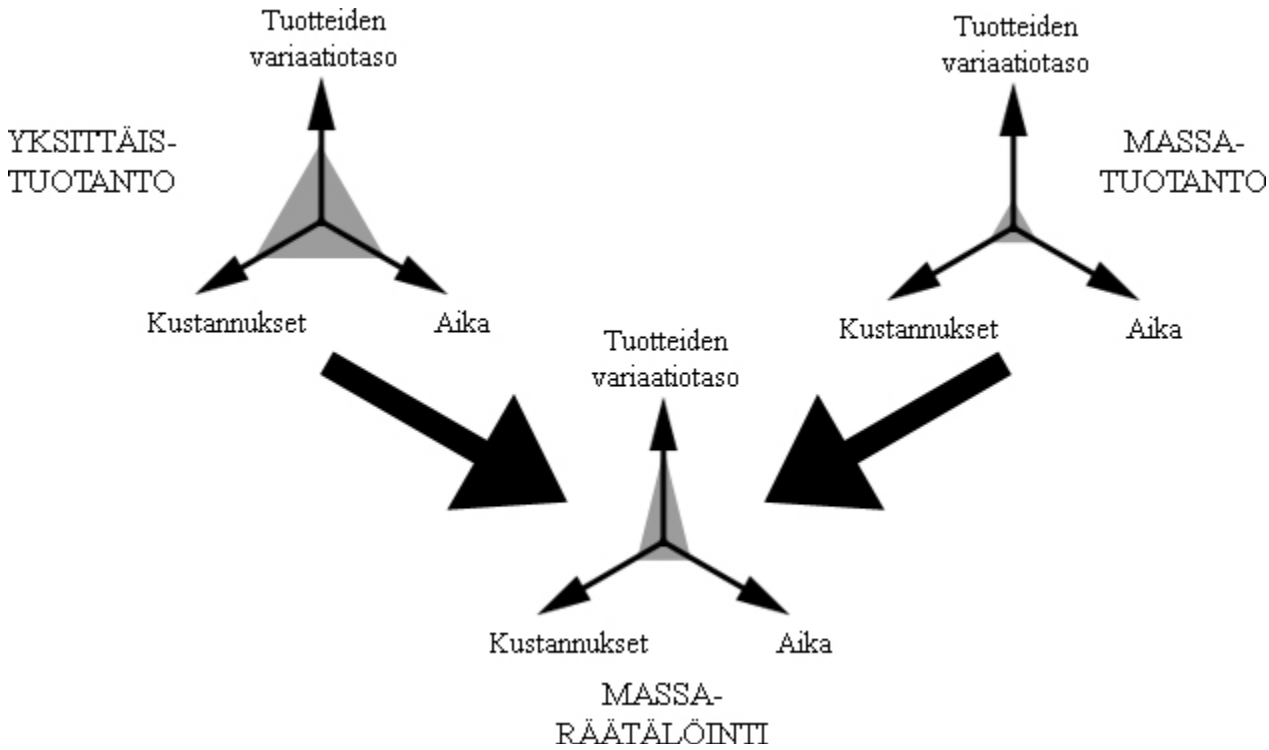
Käsityössä rajoitteeksi kuitenkin käytännössä aina muodostuu tuotteen korkea hinta ja verrattain pitkä valmistusaika. Usein myös asiakkaan omaa osallistumista vaaditaan, jotta tuotteesta tulee vaatimusten mukainen. Otetaan esimerkiksi räätälöity vaate, jonka oikean istuvuuden varmistamiseksi voidaan vaatia useita mittauksia ja koesovituksia.

Oikean kohdan löytäminen massatuotannon ja käsityön kuvitteellisella janalla on kuitenkin vaikeaa. Todellista uniikkituotantoa voitaisiin katsoa olevan se, että tietty vaatekappale suunnitellaan alusta pitäen asiakasta varten. Tällöin se voidaan katsoa olevan aidosti asiakkaalle tehtyä käsityötä.

Toisaalta, useammin kuitenkin vaate vain räätälöidään asiakkaan mittojen mukaiseksi noudattaen kuitenkin jotain olemassa olevaa designia. Jos asiakas valitsee tietystä materiaalivalikoimasta ja tietystä leikkausvalikoimasta, on käsityötä vain mittojen mukaan valmistaminen ja tuotannon voidaan katsoa sijoittuvan hieman lähemmäs massaräätälöintiä, kuin uniikkituotantoa. Massaräätälöinnistä vaateteollisuudessa seuraa lisää myöhemmässä kappaleessa, jossa tarkastelemme massaräätälöinnin käyttöä perinteisissä liiketoiminnoissa.

Uniikille käsityölle on luonteenomaista tuotteiden ja tuotantoprosessien runsas muunteleminen. Tällöin voidaan tuottaa ainutlaatuisia tuotteita vaihtelevien tuotantoprosessien tuloksena. [Ahoniemi *et al.*, 2007]

Massaräätälöinnin sijoittumista yksittäistuotannon ja massatuotannon välimaastoon havainnollistaa hyvin seuraava kuva (kuva 1).



Kuva 1. Massaräätälöinnillä yhdistetään massatuotannon tehokkuus ja yksittäistuotannon tuotevariaatiotaso [Forza and Salvador, 2007; Ahoniemi *et al.*, 2007]

Vuosisatojen ajan asiakaskohtaisesti räätälöidyille tuotteille on löytynyt kysyntää, mutta korkeat kustannukset ovat rajoittaneet markkinoita. Esimerkiksi muotokuvien maalaamisessa rajoittavina tekijöinä ovat olleet osaavien maalareiden suhteellisen pieni määrä ja siten toimitusaikojen kasvu.

Kun valmistuksessa siirrytään kompleksisempiin tuotantotapoihin, kohoavat myös uniikkituotannon tuotantokustannukset. Tuotantoprosessia ei ole kannattavaa toistuvasti muutella jokaista asiakasta varten. Esimerkiksi monia elektronisia laitteita voidaan valmistaa kuluttajahinnoin vain suurten homogeenisten valmistemäärien myötä.

Massaräätälöinti toteutuukin kannattavasti yleensä vain tilanteessa, jossa asiakkaiden yksilöllisiä tarpeita vastaavia tuotteita kyetään valmistamaan kustannustehokkaasti, mikä puolestaan on mahdollista vain joustavien tuotantojärjestelmien kautta. [Ahoniemi *et al.*, 2007]

Massaräätälöintiä toimintatapana on visioitu kohta jo puoli vuosisataa, mutta vasta viimeisten parin vuosikymmenen nopea tieto- ja viestintäteknologioiden kehitys on tehnyt osaltaan mahdollisiksi asiakaslähtöisten massaräätälöintistrategioiden toteuttamisen käytännössä [Ahoniemi *et al.*, 2007].

Massaräätälöinti on suuri murros teollisuuden alalla. 1900-luvun vaihteessa tuotanto mullistui siirryttäessä käsityöstä tehdastuotantoon, massatuotantoon. Nyt voimme kokea vastaavanlaisen suuren mullistuksen siirtymällä massatuotannosta massaräätälöintiin. On kuitenkin huomattava, että massatuotannossa on ollut jo pitkään piirteitä, joiden voidaan

katsoa edeltäneen varsinaista massaräätälöintiä, joten siirtyminen todelliseen massaräätälöintiin ei tapahtunut hetkessä. Tällaisia piirteitä ovat muun muassa komponenttipohjainen valmistus ja myöhäinen variaatio.

John Teresko mainitsi kesällä 1994 julkaistussa *Industry Week* –lehdessä, miten hänen mielestään massaräätälöintiä maalailtiin jo 1970 –ilmestyneessä Alvin Tofflerin kulttikirjassa *Future Shock*. Nimensä massaräätälöinti sai Tereskon ja Pinen mukaan Stan Davisin vuoden 1987 kirjasta *Future Perfect*. Käytännössä nykyisenmuotoisen massaräätälöinnin isänä pidetään kuitenkin B. Joseph Pine II:ta ja hänen vuonna 1993 julkaisemaansa kirjaa *Mass Customization, The New Frontier in Business Competition* (Harvard Business School Press, 1993) massaräätälöintitutkimuksen alkuunpanijana [Teresko, 1994; Pine, 1993].

Massaräätälöinti on myös monimutkainen kokonaisuus, jonka kustannushyöty ja muut tavoitteet maksimoituvat ymmärtämällä ja toteuttamalla sitä mahdollisimman laajasti. Se ei siis ole yksinkertainen tai nopea muutos. Kuitenkin muutoksen edetessä sillä on yhä suuremmat vaikutukset kilpailukykyyn ja kannattavuuteen.

Teresko varoitti jo vuonna 1994 siitä, ettei massaräätälöintiä saa nähdä vain räätälöinnin ja massatuotannon liittona, vaan koko yritystoiminnan perustuksia ravistelevana uutena toimintatapana maksimaalisen kilpailuhyödyn saavuttamiseksi [Teresko, 1994].

Uuteen toimintamalliin siirtyminen ei siis ole yksinkertaista. Täyden hyödyn saavuttamiseksi massaräätälöintiin tuleekin siirtyä kokonaisvaltaisesti. Muutos vaikuttaa ja vaatii uusiutumista monilta organisaation eri osa-alueilta aina suunnittelusta markkinointiin ja tuotannosta ylläpitopalveluihin asti. Samaten myös eri yritysten verkostoyhteistyö tarvitsee muutosta massaräätälöintiin siirryttäessä. [Ahoniemi *et al.*, 2007]

Massaräätälöintiin siirtyminen ei ole yritykselle pelkästään tuotannon haaste, vaan kokonaisvaltainen haaste [Ahoniemi *et al.*, 2007]. Tämän tutkimuksen puitteissa keskityn kuitenkin lähinnä tuotannon kysymyksiin.

3.4. Esimerkkejä massaräätälöinnistä

Mainoslahjapuolella esimerkiksi kyniä oman yrityksen logolla on mahdollista hankkia valitsemalla ensin tuotepohjan sarjavalmisteisesta varastosta ja sitten täyttämällä verkkolomakkeeseen haluamansa mainostekstin. Samaten esimerkiksi käyntikortteja voidaan tilata valitsemalla yrityksen tarjoamasta korttipohjasta haluamansa ja sitten lisäämällä siihen omat yhteystietonsa, sen sijaan, että graafinen suunnittelija tekisi koko työn alusta loppuun jokaisen asiakkaan kohdalla. Samanlaisia toimintoja tarjotaan myös ohjelmistopuolella. Esimerkiksi palvelut, kuten Kotisivukone, tarjoavat käyttäjilleen

valittavaksi verkkosivuilleen valmiin graafisen teeman ja työkalut sivun tekstien ja kuvien muokkaamiseen [Kotisivukone, 2012].

Tämän kaltainen tuotanto mahdollistaa nopeammat toimitusajat ja kustannussäästöjä, sillä palvelun tarjoava yritys voi ylläpitää tuotevarastoa, joka ei ole suoraan sidoksissa yhden asiakkaan tarpeisiin.

On ilmeistä, että tällä tavalla voidaan ylläpitää geneerisempää tuotevarastoa, sillä tuotteen viimeistely asiakkaan toiveiden mukaisesti voidaan toteuttaa ATO (*Asseble to order*, tuote kootaan tilauskohtaisesti), tai STO (*Ship to order,* tuote toimitetaan tilauskohtaisesti) tyylillä, eli vasta tilauksen saavuttua. Varastohyödyn vastapainoksi tämän kaltainen massakustomointi tuo mukanaan kuitenkin toisen ongelman, nimittäin tuotteiden palauttamisen. Voiko yritys ottaa takaisin tuotteita, joihin on jo painettu jonkin yrityksen logo ja/tai yhteystiedot? Lisähaasteen tuo myös laadullisen vastuun jaottuminen myös myyjälle. Jos tuote on viallinen, syy voi olla valmistajan, mutta jos esimerkiksi kahvikuppiin painettu yrityksen logo himmenee pesun yhteydessä, on tuotteen myyjä ja painaja vastuullinen. Tässä tilanteessa myös seuraamukset ovat verrattain suuret, sillä samalla kertaa asiakkaalle voidaan joutua hyvittämään satoja kappaleita kyseistä tuotetta, jotka tämän jälkeen ovat käytännössä arvottomia. Ja koska niiden myyminen tuskin onnistuu, täytyy myyjän lisäksi järjestää niiden hävittäminen pois varastotilaa viemästä.

Henkilökohtaisten tietokoneiden (engl. *Personal computer, PC*) nopea yleistyminen on tehnyt niistä tavallisia hyödykkeitä, joihin useimmilla on varaa. Tietokonekomponenttien suuri kysyntä on johtanut niiden tehokkaaseen massatuotantoon moderneissa tuotantolaitoksissa, joka on laskenut huomattavasti kappalekohtaisia valmistuskustannuksia. Tämä puolestaan on vaikuttanut kuluttajahintojen todella huomattavaan laskuun viimeisten kahdenkymmenen vuoden aikana. Tietojärjestelmätuotannon ja ohjelmistotuotannon puolella tuotantoprosessit ja toimintatavat eivät kuitenkaan ole vielä kehittyneet niin tehokkaiksi, että samankaltaisia kustannussäästöjä saavutettaisiin.

3.5. Tietojärjestelmien massaräätälöinnistä

Massakustomointi on tällä hetkellä pinnalla tietojärjestelmätutkimuksen kentällä. Sen kiinnostavuus tutkimuskohteena niin yliopistoissa kuin yrityksissäkin on helposti ymmärrettävissä; massaräätälöinnillä haetaan samalla kertaa kustannussäästöjä ja laadun parannusta.

Massaräätälöinnin avulla pyritään vakinaistamaan yleisesti tarvittavia ohjelmistokomponentteja ikään kuin rakennuspalikoiksi, joita voidaan vähällä työllä yhdistellä tietojärjestelmän tarpeiden mukaan, tuottaen näin asiakkaan tarpeisiin vastaava ohjelmistotuote kierrättämällä mahdollisimman paljon jo aiemmin tehtyjä osia.

Tämän kaltaisella toiminnalla ohjelmistotuotteen valmistamiseen käytettävää aikaa voidaan lyhentää ja samoja, jo testattuja, osia käyttämällä voidaan ylläpitää korkeampaa laatua. Laadun ja läpimenoajan hyötyjen lisäksi usein syntyy myös kustannussäästöjä.

Puhuttaessa ohjelmistojen massaräätälöinnistä on tarkasteltava asiakkaan osuutta tuotteen määrittelyyn. Yritys voi omaan osaamiseensa ja omiin tutkimustuloksiinsa pohjaten tarjota markkinoille massaräätälöityä tuotetta, mutta yritys voi myös määritellä tuotteen yhdessä asiakkaan kanssa. Tuotteen suunnittelu vaikuttaa massaräätälöitymiseen alusta asti, sillä jo suunnitteluvaiheessa tehtävät valinnat heijastuvat siihen millä tavoilla räätälöityvä tuote voidaan toteuttaa.

3.6. Tuotantolinja-ajattelu

Tuotantolinja-ajattelussa on paljon päällekkäisyyksiä massaräätälöinnin tuotannollisen näkökulman kanssa tietojärjestelmien alueella ja joskus voikin olla vaikea eritellä näitä kahta käsitettä täsmällisesti. Tässä tutkielmassa näillä käsitteillä on voimakkaita päällekkäisyyksiä ja tietojärjestelmien massaräätälöinnin tuotannollisesta osasta puhuttaessa molempia puolia tuleekin käsitellä reaalimaailman tilanteita tutkittaessa.

Krueger on tutkinut ja julkaissut paljon asiaan liittyen ja esittää tuotantolinja-ajattelun voimakkaan sidoksen ohjelmistotuotantoon käytännön pakon mukaan. Krueger nostaa esiin sen, miten ohjelmistotuotannon työkalut ja käytännöt keskittyvät yksittäisten tuotteiden kehittämiseen ja miten toisaalta yritykset joutuvat käytännössä toteuttamaan tuotantolinjoja, joissa kootaan yhteen ominaisuuksiltaan ja variaatioiltaan samankaltaisia tuotteita tuoteportfolioihin yksittäisten tuotteiden sijaan, sekä sen, miten tämä ristiriita on nostanut esiin tarpeen erityisesti tuotantolinja-ajatteluun keskittyville työkaluille ja toimintatavoille [Krueger, 2006b].

Kruegerin mukaan [2006a] jo ensimmäiset ohjelmistojen tuotantolinjoja tutkineet *case*-tutkimukset 1980- ja 1990-luvuilta kuvailivat rakenteita, jotka jälkeenpäin havaittiin noudattavan ohjelmistojen tuotantolinja-ajattelua. Vaikka nämä rakenteet luotiin aikanaan tehostamaan ohjelmistotuotantoa uudelleenkäytön kautta, olivat ne kaikki itsenäisesti osaltaan luoneet mallin, jota nyt kutsutaan ohjelmistojen tuotantolinja-ajatteluksi.

Tuotantolinja-ajattelussa pyritään kokoamaan yhtenäisiä tuotteita, joita voidaan yhdistellä ja määritellä tarpeiden mukaan. Määrittelyllä tässä kohtaa tarkoitetaan tuotteen sisäisten muuttujien asettamista tarpeen, yleensä vaihtelevan käyttöympäristön, mukaan. Esimerkkinä sisäisten muuttujien määrittelemisestä voidaan ottaa Linux-käyttöjärjestelmä, jossa voidaan asennuksen aikana määritellä monia sisäisiä asetuksia vastaamaan käyttöympäristöä, lähinnä siis alustan laitteistokonfiguraatiota. Samaa tuotetta voidaan asentaa toimimaan hyvin erilaisiin tietokoneisiin ja laitteisiin abstraktion kautta. Tämä tarkoittaa sitä, että esimerkiksi kohdelaitteen verkkosovittimelle on valmiina useita eri matalan tason ajureja, joista valitaan asennuksen yhteydessä käytettäväksi vain

yksi, joka kuitenkin toteuttaa yhtenevät rajapinnat, joiden kautta käyttöjärjestelmä voi ohjata laitetta ottamatta kantaa siihen, mikä ajuri juuri tähän koneeseen valittiin.

Laajemmin käsiteltäessä ohjelmistojen tuotantolinjana voitaisiin ajatella yksittäisen Linux julkaisun (engl. *distribution*) eri variaatioita eri suoritinmallien erilaisille käskykannoille, esimerkiksi IA-32 (x86-suoritinperheen 32-bittinen käskykanta, usein käytetään myös termiä i386), sen 64-bittinen versio IA-64, PowerPC ja ARM. Erilaisten käskykantojensa ja arkkitehtuuriensa vuoksi tietylle suoritinarkkitehtuurille tehdyt sovellukset eivät suoraan toimi toisilla suorittimilla, mutta esimerkiksi Linuxin tapauksessa päällisin puolin sama käyttöjärjestelmä on tuotantolinjan kautta saatavilla kullekin hyvin erilaiselle suoritinarkkitehtuurille.

Ilman tuotantolinjaa kokonaisen käyttöjärjestelmän kehittäminen täysin itsenäisesti kullekin suoritinarkkitehtuurille olisi työlästä ja kallista, mutta erityisesti turhaa päällekkäistä työtä.

Tämä on kuitenkin hyvin suppea esimerkki tuotantolinja-ajattelusta. Reaalimaailmassa ohjelmistotuotteiden tuotantolinjat muodostuvat enemmän heterogeenisistä tuotteista, joilla on vain tietty määrä homogeenisiä ominaisuuksia, joiden perusteella niitä voidaan koota tuoteportfolioihin. Monesti nämä homogeeniset ominaisuudet ovatkin teknisiä ratkaisuja, loppukäyttäjälle näkymättömiä tekniikoita, jotka kuitenkin ratkaisevat yleisiä sovellusten tarpeita. Esimerkkeinä näistä voitaisiin ottaa vaikkapa tietovarannot tai piirtotavat. Monet tietojärjestelmät toimivat tietokantojen yhteydessä ja tarvitsevat siten hyvin samankaltaisia metodeja kommunikointiin tietokannan kanssa. Nämä metodit voidaan nähdä yhdistävinä tekijöinä useiden ohjelmistotuotteiden kanssa ja samaa ohjelmatoteutusta voidaankin hyödyntää monissa ohjelmistoissa. Samaten esimerkiksi tietty käyttöliittymäelementtien piirtotapa voidaan ottaa yhdistäväksi tekijäksi.

Joka tapauksessa rajanveto tuoteportfolioon ottamisen ja pois jättämisen välillä on kunkin yrityksen oman harkinnan, kokemuksen ja intressien varassa ja saattaa olla vaikea perustella täsmällisesti.

4. Massaräätälöinti suomalaisessa liiketoiminnassa globaalissa ympäristössä

Suomalainen teollisuus on aina keskittynyt korkeaan teknologiseen osaamiseen ja asiakaskohtaisesti räätälöityihin ratkaisuihin. Tämä lähestymistapa on perinteisesti tuonut menestystä suomalaiselle teollisuudelle kautta vuosien. Kuitenkaan muualla maailmalla toteutettavaa laajamittaista massatuotantoa ei olla Suomessa juurikaan harjoitettu. Syynä tähän on riittämättömän pienet kotimaiset markkinat. Tämä on pakottanut suomalaiset yritykset erikoistumaan ja ottamaan vastaan lähinnä vain ne vaikeimmat tilaukset, joita muut eivät halua tai pysty toteuttamaan. [Ahoniemi *et al.*, 2007]

Globalisaation myötä aina vain useammat suomalaiset yritykset toimivat voimakkaassa kansainvälisessä kilpailutilanteessa. Tässä liiketoimintaympäristössä menestymisen ehdoksi on muodostunut kyky jatkuvasti kehittää omaa kilpailukykyään. Tutkimuskirjallisuudessa ja yritysten haastatteluissa onkin nostettu massaräätälöinnin eri toteutusmuotojen käyttö tärkeäksi kilpailukykytekijäksi. [Ahoniemi *et al.*, 2007]

On kuitenkin tärkeää huomata, että globaaleilla markkinoilla toimivat suomalaiset yritykset ovat nostaneet itselleen tärkeimmiksi toimintatavoiksi massaräätälöinnistä lähinnä tavat, joissa pyritään yhdistämään asiakaskohtainen räätälöinti kustannustehokkuuteen, vaikka massaräätälöinti käsitteenä sisältääkin laajemman kokonaisuuden ja kohdistaa vaatimuksia yhtälailla kuuteen eri yritystoiminnan osa-alueeseen. [Ahoniemi *et al.*, 2007]

Ahoniemen *et al.* [2007] mukaan nämä kuusi eri yritystoiminnan osa-aluetta ovat tuotekehitys, tuotannon kehitys, verkostoyhteistyö, ohjauksen kehittäminen, organisatoriset ja muut tekijät sekä asiakasläheisyys.

Tässä vertailussa täytyy kuitenkin ottaa huomioon se, ettei suomalainen teollisuus välttämättä vastaa rakenteeltaan sellaista, mitä ulkomaisissa aihetta käsittelevissä julkaisuissa tarkastellaan. On kysyttävä, onko suomalainen teollisuus esimerkiksi yhtä massatuotantopainottunutta kuin mitä jokin ulkomainen tuotanto saattaa olla.

On ilmeistä, että asiakkaalle räätälöity tuote tuottaa ainutlaatuista lisäarvoa. Tämä siis tarkoittaa sitä, että asiakas on valmis maksamaan enemmän juuri hänen tarpeisiinsa räätälöidystä tuotteesta. Ja jos asiakas on valmis maksamaan enemmän, on myös yrityksellä mahdollisuus ansaita enemmän tuotteen valmistamisesta, kunhan vain sen valmistuskustannukset saadaan pysymään maltillisina. Tämä puolestaan välittyy liiketoiminnan kannattavuudeksi ja kilpailukyvyksi.

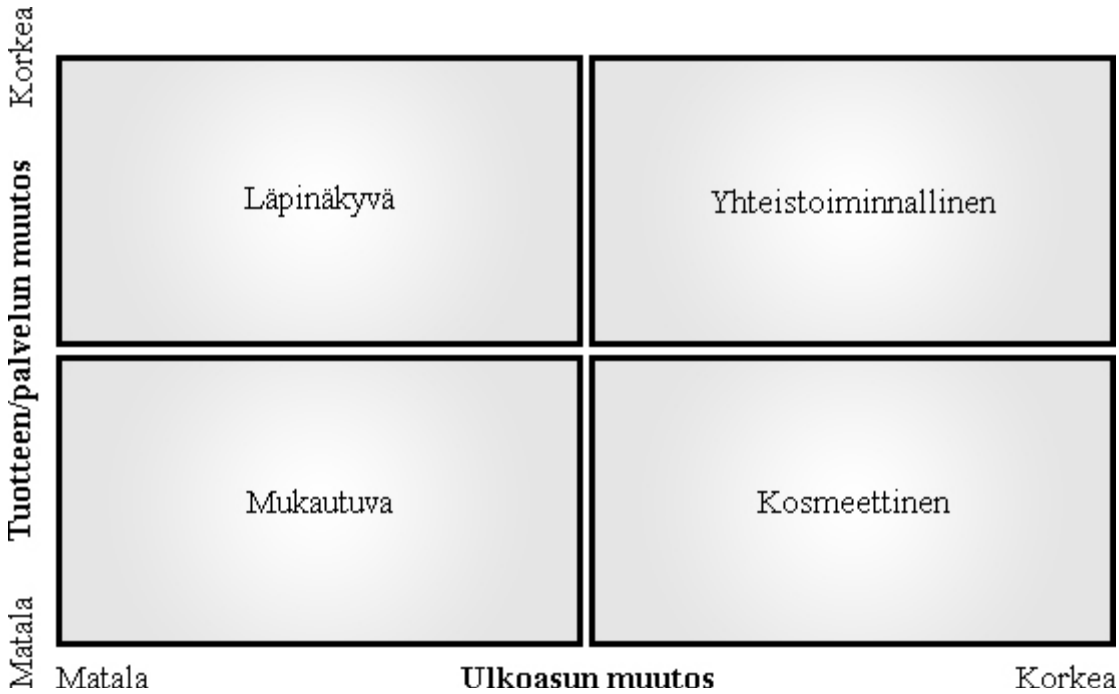
Suomalaisille yrityksille massaräätälöinti on suuri mahdollisuus, sillä pienten markkinoiden ja korkeiden henkilöstökulujen maassa hintaan perustuva, etenkin globaali kilpailu on hyvin haastavaa.

On myös huomattava, että korkea teknologinen kehitysaste, joka on massaräätälöintistrategialle elinehto, on suomalaisissa tuotantoympäristöissä ja teollisuuskulttuurissa paljolti jo valmiina.

5. Massaräätälöintitapojen nelikenttä

Gilmore and Pine [1997] jakavat erilaisiin tuotteisiin kohdistuvan räätälöintitarpeen nelikenttään tuotteeseen kohdistuvien muutosten määrän ja laadun suhteen asiakkaiden ja asiakasryhmien vaatimusten pohjalta. Massaräätälöintiä voidaan luonnehtia (ja sen perusteella jakaa) *mukautuvaksi*, *kosmeettiseksi*, *läpinäkyväksi* tai *yhteistoiminnalliseksi* sen perusteella, miten räätälöinti vaikuttaa tuotteen varsinaisiin ominaisuuksiin ja sen ulkomuotoon.

Ruohonen *et al.* [2006] esittää tämän nelikentän yhtäältä tuotteen tai palvelun muutoksen ja toisaalta ulkoasun muutoksen mukaan seuraavanlaisessa nelikentässä (kuva 2).



Kuva 2: Massaräätälöinnin eri vaihtoehdot asiakassuhteen näkökulmasta [Ruohonen *et al.*, 2006; Ahoniemi *et al.*, 2007]

Käyn seuraavaksi läpi nelikentän eri vaihtoehdot, jotta saamme paremman kuvan nimenomaan tietojärjestelmiin liittyvistä muutoksista. Massaräätälöitäessä tietojärjestelmiä muuttuvia asioita voivat olla esimerkiksi ohjelmiston käyttöliittymä, tiedonsyöttölaitteet ja käyttäjäkokemus (engl. *user experience*).

Massaräätälöitävien ominaisuuksien voidaan katsoa myös sisältävän teknisiä ratkaisuja, kuten tietokantojen sijainnin (eriytetty vs. paikallinen) ja ohjelman käyttöympäristön, esimerkiksi paikallinen asennus tai *SAAS* (software-as-a-service).

5.1. Mukautuva massaräätälöinti

Ensimmäisenä kenttänä matalalla ulkoasun muutosvaatimuksella ja matalalla tuotteen muutoksella on mukautuva massaräätälöinti. Mukautuvan massaräätälöinnin ajatuksena on tuotemoduulit, joita yhdistelemällä voidaan tarjota asiakkaalle hänen haluamaansa kokonaisuutta. Massaräätälöinti tapahtuu siis jo siinä vaiheessa, kun mahdolliset tuotekonfiguraatiot (engl. *product configuration*) määritellään. Mukautuva massaräätälöinti on voimakkaasti yleistynyt etenkin sähköisen liiketoiminnan mahdollisuuksien myötä ja käyttääkin näitä mahdollisuuksia vahvasti hyväkseen.

Esimerkkinä mukautuvasta massaräätälöinnistä voidaan pitää tietokoneiden konfigurointia asiakkaan tarpeiden mukaiseksi. Tilaaja voi valmistajan tarjoamilla verkkosovelluksilla valita haluamansa tietokoneen tarjotuista vaihtoehdoista. Kaikkien valittavissa olevien vaihtoehtojen tulee olla etukäteen määriteltyjä ja näin kaikkien valittavissa olevien konfiguraatioiden määrä tiedossa. Mukautuvassa massaräätälöinnissä modulaarisuus on keskeistä.

Osa liittimistä, kannoista ja sovittimista päivittyy muutaman vuoden välein niin paljon, etteivät komponentit ole keskenään yhteensopivia. Esimerkiksi suorittimien kehittyessä niiden kannat tietokoneen emolevyllä muuttuvat. Käytännössä siis vaihdettaessa uudempaan suoritinarkkitehtuuriin täytyy myös emolevy vaihtaa sitä tukevaksi. Myös keskusmuistin parissa tehdään sellaisia teknisiä uudistuksia, etteivät vanhemmat emolevyt muuttumattomasta liittimestä huolimatta voi hyödyntää uutta muistitekniikkaa. Keskusmuistien parissa tosin on jonkin verran alaspäin yhteensopivuutta, jolloin uudempikin emolevy pystyy vielä käyttämään hieman vanhempia muisteja.

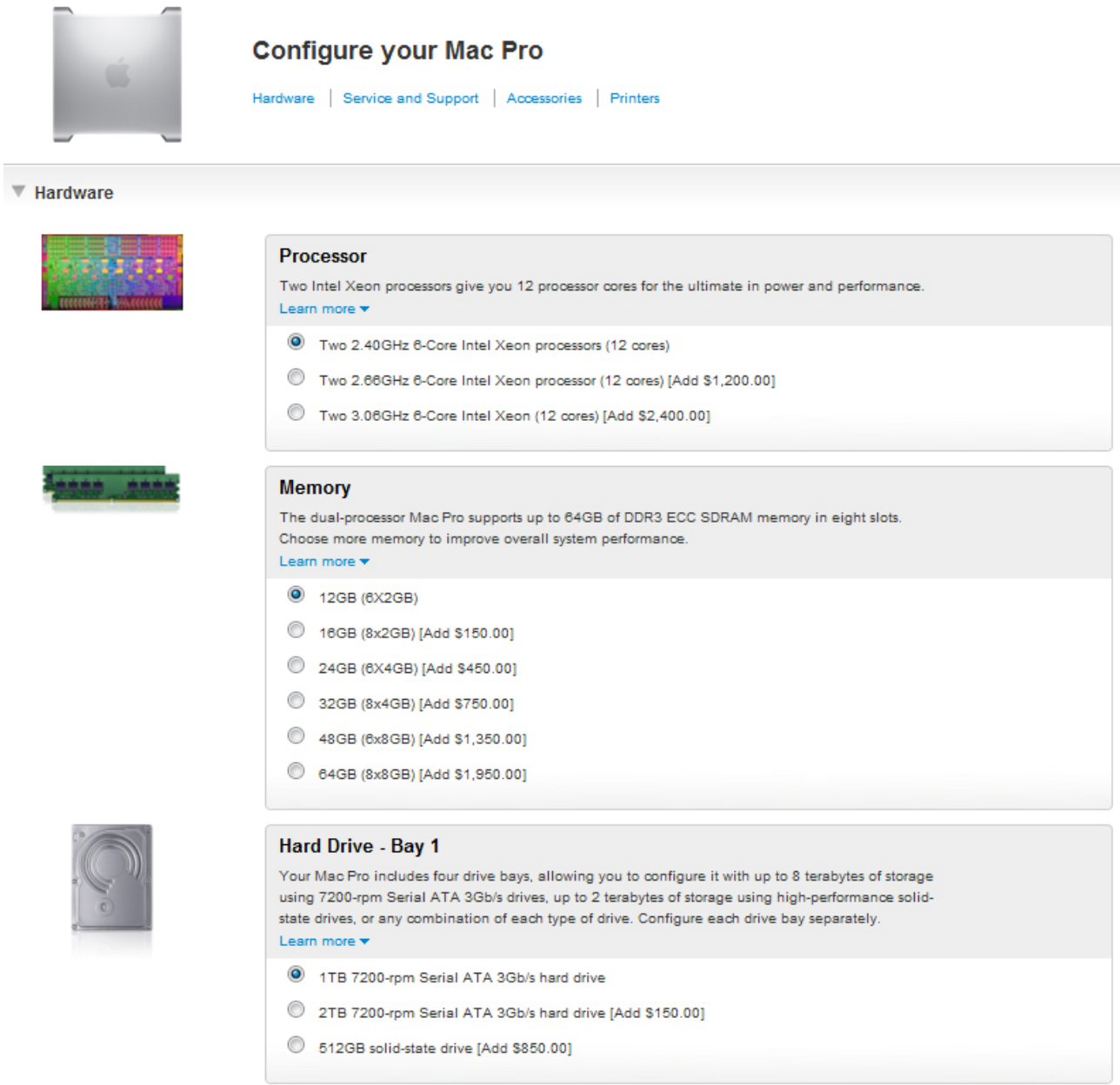
Nämä rajoitteet eivät kuitenkaan muodostu käytännön ongelmaksi valmistietokonekaupassa, sillä valmistaja voi päättää yhtenäisen arkkitehtuurin suunnittelunsa pohjaksi ja tarjota mukautuvaa massaräätälöintiä yhden tuotearkkitehtuurin puitteissa.

Esimerkiksi samaan suoritinkantaan voidaan liittää useita saman suoritinarkkitehtuurin suorittimia. Näin asiakas voi valita haluamansa määrän suoritinytimiä ja kellotaajuutta. Myös keskusmuistin määrän valitseminen on hyvin helppoa toteuttaa, sillä emolevyn muistiliittimiin voidaan asentaan monenlaisia konfiguraatioita määrän ja kuormanjakamisen tarpeisiin.

Myös kiintolevyjen ja SSD -massamuistien (Solid State Drive, flash -muistipiireihin perustuva kiintolevyn kaltainen tallennusväline) valitseminen on yksinkertaista harvemmin päivittyvän liittimen ja standardoidun koon vuoksi. Kahden ja puolen tuuman paikkaan SATA -liitimeen (Serial AT Attachment) kiinni voidaan laittaa halutulla kapasiteetilla varustettu ja haluttua tekniikkaa käyttävä massamuisti nopeasti ja helposti.

Nämä kolme ja näytönohjain lienevätkin yleisimmin käytetyt variaatiot tietokoneita konfiguroitaessa.

Samoin kuin autoteollisuudessa, myös haluamansa tietokoneen voi tilata sähköisen liiketoiminnan tarjoamista verkkosovelluksista helposti ja nopeasti vaikkapa kotisohvaltaan käsin (kuva 3).



Kuva 3: Apple Inc.:n tarjoama verkkopalvelu mukautuvaan massaräätälöintiin tietokonekaupassa [Apple, 2012]

5.1.1. Tietojärjestelmät ja mukautuva massaräätälöinti

Mukautuvassa massaräätälöinnissä massaräätälöintiä toteutetaan tietojärjestelmätuotannossa melko laajalti asiakkaan tietämättä. Monet tietokoneohjelman osat ovat moduuleita, joita sama yritys on saattanut käyttää jossain toisessakin projektissa. Järkevällä ohjelmistoarkkitehtuurilla samoja komponentteja voidaan uusiokäyttää ja niistä voidaan koostaa geneerisiä kirjastoja, joita yritys voi hyödyntää useissa projekteissaan. Tällainen toimintatapa luo pitkässä juoksussa usein kustannussäästöjä ja parantaa ohjelmakoodin laatua useampien testauksien kautta.

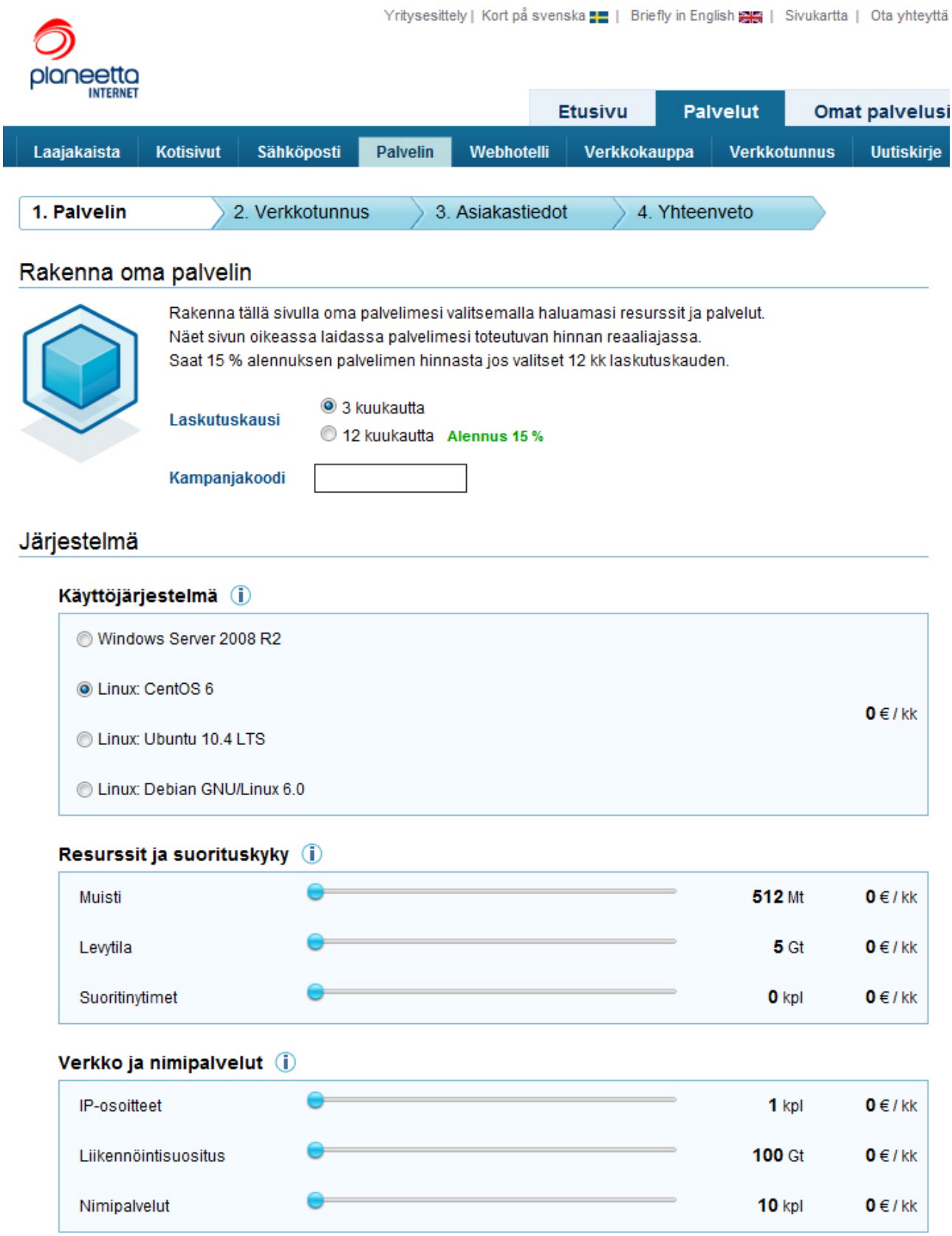
Koskimies ja Mikkonen esittivät kirjassaan *Ohjelmistoarkkitehtuurit* [2005], että ohjelmistokehyksen break-even piste, eli hetki jolloin saavutetut säästöt ylittävät alkuinvestoinnin tapahtuisi suomalaisella pelifirmalla kolmannen tuotteen kohdalla. Siihen asti siis ohjelmistokehyksen tuottamiseen kulutetut resurssit ovat suuremmat, kuin mitä tavallisen tuotteen valmistamiseen ilman ohjelmistokehystä olisi kulunut.

Tämän suppean esimerkkiaineiston valossa läpinäkyväksi massaräätälöinniksi katsottava tuotteen koostaminen käyttämällä valmiita ohjelmistokehyksiä vaikuttaa olevan taloudellisesti järkevää.

Toinen mukautuvan massaräätälöinnin piirre tietojärjestelmätuotannossa on joidenkin tekniikoiden valitseminen. Monet ohjelmistokehykset tukevat esimerkiksi SQL-pohjaisten tietokantojen käsitteellistämistä (engl. *abstraction*) sovitin –suunnittelumallien kautta. Tällöin asiakkaalle voidaan tarjota usein kustannus- ja suorituskykyvaatimusten mukaan montaa eri tietokantamoottoria, vaikka itse tietokantakyselyt pysyvätkin hyvin samankaltaisina. Parhaassa tapauksessa koko tietojärjestelmä voidaan siirtää yhdeltä tietokantamoottorilta toiseen kesken tuotantokäytön vain ohjelmakonfiguraatiota muuttamalla ilman tarvetta ohjelmointityölle.

On kuitenkin huomattavaa, että esimerkin tietokantamoottorin vaihtaminen kesken tuotantokäytön on harvoin asiakkaalle oikeasti tarpeellinen ominaisuus ja sellaisen tarve kertoo ehkä enemmän arkkitehtuurisuunnittelun lyhytnäköisyydestä tai virheellisistä olettamuksista.

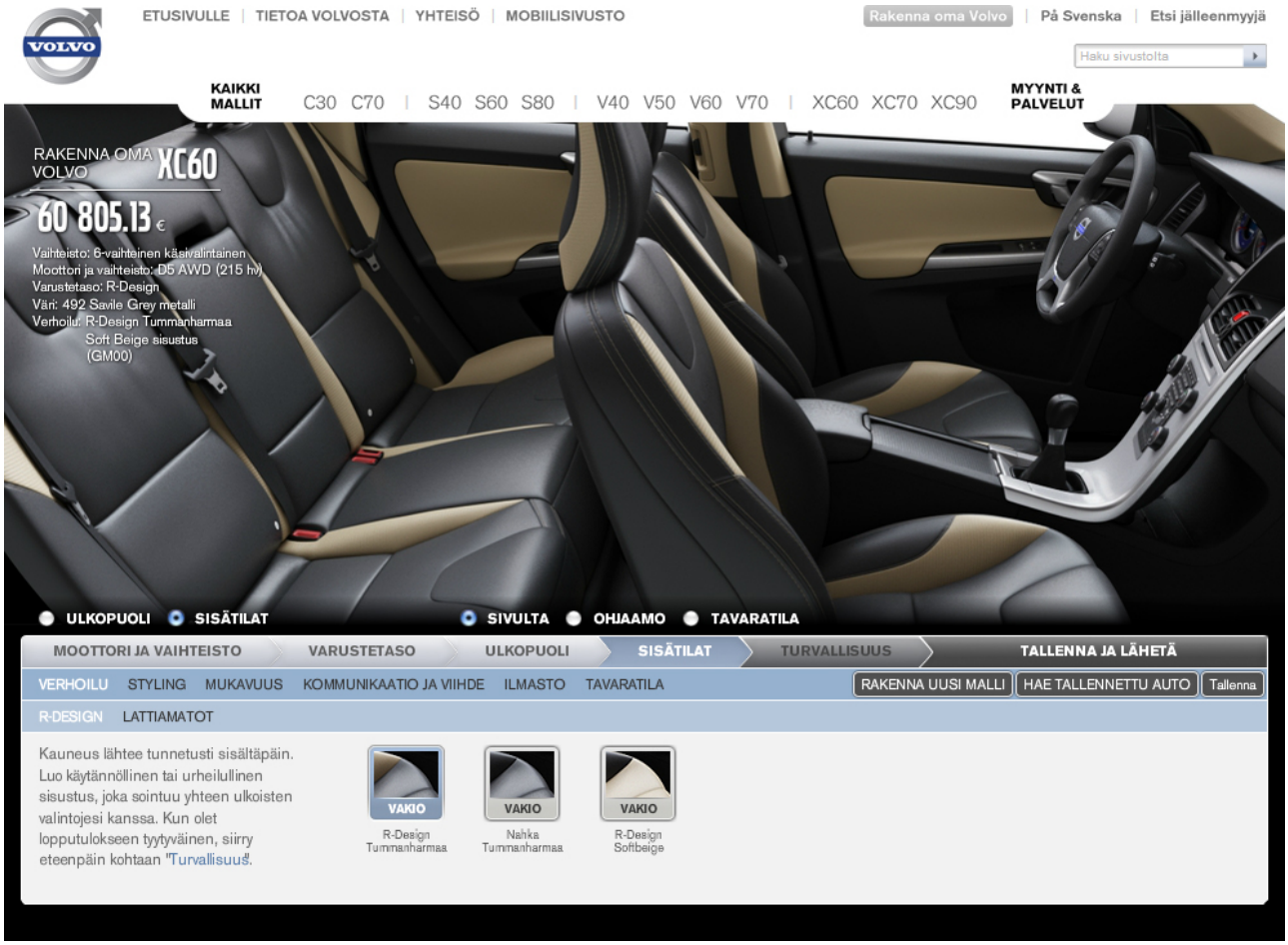
Mukautuvaa massaräätälöintiä käytetään paljon myös hosting –palveluiden tarjonnassa. Niissä asiakas voi valita tarvitsemansa ominaisuudet ja palvelun hinta muodostetaan niiden perusteella. Esimerkiksi Planeetta Internet Oy tarjoaa verkkosivuillaan dynaamisen sähköisen lomakkeen, jonka avulla asiakas voi valita haluamansa ominaisuuden ja toimintaympäristön, sekä nähdä reaaliaikaisesti, miten hänen valintansa vaikuttava kuukausihinnan muodostumiseen (kuva 4).



Kuva 4: Planeetta Internet Oy: tarjoama verkkosovellus hosting -palvelun mukautuvaan massaräätälöintiin [Planeetta Internet Oy, 2012]

5.2. Kosmeettinen massaräätälöinti

Kosmeettisessa massaräätälöinnissä räätälöitävät asiat ovat lähinnä ulkonäöllisiä. Useissa tapauksissa kosmeettinen massaräätälöinti voidaan toteuttaa hyvin viivästettynä variointina, esimerkiksi liikelahjatilauksessa haluttu yrityksen logo voidaan painattaa muuten jo täysin valmiiseen tuotteeseen. Perinteisesti kosmeettista massaräätälöintiä ollaan jo useiden vuosien ajan hyödynnetty esimerkiksi autoteollisuudessa. Monet merkit tarjoavat verkkosivuillaan asiakkaan käyttöön tuotekonfiguraattorin, verkko-ohjelman tai dynaamisen verkkosivun, jonka avulla asiakas voi ”rakentaa” haluamansa ajoneuvon virtuaalisesti valitsemalla haluamansa moduulit. Esimerkki tällaisesta verkkosovelluksesta löytyy Volvo Car Corporationin verkkosivuilta (kuva 5).



Kuva 5: Esimerkki kosmeettisen massaräätälöinnin käytöstä verkkosovelluksen kautta [Volvo Car Corp., 2012]

Riippuen moduulien määrästä ja niiden eksklusiivisuudesta (engl. *exclusive, poissulkeva*), mahdollisten tuotekonfiguraatioiden määrä voi kasvaa jyrkästi. Tällöin ei ole mahdollista ylläpitää valmista tuotevarastoa jokaiselle konfiguraatiolle, joten tuotteiden valmistuksen on oltava build-to-order –tyyppistä. Kuitenkaan esimerkiksi autoja ei voida järkevästi valmistaa alusta loppuun vasta tilauksen tultua; ei ainakaan järkevällä

toimitusajalla. Siksi tässäkin esimerkissä valmistajan on pyrittävä tuottamaan moduulit massatuotannolla ja takaamaan viivästetty variointi modulaarisella rakenteella. Tehtaat voivat tuottaa autoaihioita muutamilla variaatioilla ja lopullinen räätälöinti suoritetaan usein vasta kohdemaassa asentamalla hyvin modulaarisia komponentteja.

Autoteollisuudessa modulaarisuus on aina hieman kompleksisista, sillä jotkin osat vaativat aihion muokkaamista. Esimerkiksi peruutustutkan asentaminen vaatii usein reikien tekemistä puskuriin. Tästä huolimatta valittavissa olevat räätälöintivaihtoehdot ovat pääsääntöisesti kosmeettisia, kuten auton maalipinnan väri, sisustuksen materiaali ja tehdasasenteiset lisävarusteet. Jotkin vaihtoehdot ovat kuitenkin enemmän mukautuvaa massaräätälöintiä kuin kosmeettista massaräätälöintiä. Esimerkiksi auton moottorin ja vaihteiston valinta voidaan katsoa kuuluvan autoaihion muokkaamiseksi ennemmin kuin kosmeettisiksi muutoksiksi. Onkin hyvä huomioida, että massaräätälöinnin nelikenttä ei jaa räätälöintitapoja tiukkoihin kategorioihin, vaan vaihtoehdot voivat olla, ja usein ovatkin, sekoitus useampaa massaräätälöintitapaa.

5.2.1. Tietojärjestelmät ja kosmeettinen massaräätälöinti

Kosmeettista massaräätälöintiä esiintyy tietojärjestelmätuotannossa hyvin rajallisesti. Pääasiassa kosmeettinen massaräätälöinti ilmenee ohjelmistotuotteen personointina asiakasyrityksen yritysilmeeseen. Esimerkiksi verkkokauppasovelluksia tarjoavat yritykset tarjoavat usein myös mahdollisuuden ohjelmiston värimaailman ja joidenkin grafiikoiden kustomointiin asiakkaan toiveiden mukaisiksi.

Kosmeettisen massaräätälöinnin käyttöä rajoittaa kuitenkin useiden ohjelmistotuotteiden tarjoama mahdollisuus asiakkaan omatoimiseen kosmeettiseen räätälöintiin etenkin kuluttajatuotteissa. Vaikka ohjelmistoa tuottava yritys voi tarjota tuotettansa yritysasiakkaalle jälleenmyyntiin asiakkaan omalla logolla ja nimellä varustettuna, kuluttaja-asiakkaalla ei useinkaan ole tahtoa maksaa kyseisenkaltaisesta räätälöinnistä.

Esimerkkinä kosmeettisesta massaräätälöinnistä ohjelmistotuotteissa voidaan ottaa F-Secure Oyj:n Internet Security 2012 –tietoturvaohjelmisto, jonka mm. Elisa Oyj:n laajakaistaliittymäasiakkaat saavat käyttöönsä nimellä Saunalahti Turvapaketti ja Sonera Oyj:n laajakaistaliittymäasiakkaat nimellä Sonera Tietoturva (kuva 6).

Kuva 6: Kosmeettinen massaräätälöinti F-Secure Oyj:n ohjelmistotuotteessa

On kuitenkin hyvä huomata, että kuluttajille suunnatun kosmeettisen massaräätälöinnin tarpeen ovat osaltaan poistaneet ohjelmistotuotteiden valmistajat itse. Monissa ohjelmistotuotteissa on mahdollisuus vaikuttaa ohjelmiston ulkoasuun esimerkiksi valitsemalla jonkin valmiin väriteeman, asentamalla ulkoasua muuttavia tiedostoja, skinejä (engl. *skin, nahka*), tai joissain tapauksissa muokkaamalla eri asioihin käytettyjä väriarvoja ja grafiikoita suoraan.

Kosmeettinen massaräätälöinti kuluttajille suunnatuissa tietojärjestelmissä on pääsääntöisesti eriytetty varsinaisesta ohjelmakoodista, tehden ulkoasulliset muutokset mahdollisiksi toteuttaa hyvin myöhäisessä vaiheessa, useimmiten itse ohjelmistotuotteen jo valmistuttua.

Yritysohjelmistoissa käyttäjälle tarjottavien toimintojen määrä saattaa olla niin suuri, ettei keskimääräiseen näyttölaitteeseen mahtuisi toimintopainikkeiden lisäksi juurikaan muuta sisältöä.

Monipuolisten yritysohjelmistojen käyttöliittymät saattavatkin sisältää mahdollisuuden kustomoida työkalupainikkeiden sijainteja ja piilottaa itselle tarpeettomia työkaluvalikoita näkymästä. Esimerkiksi Blender Foundationin avoimen lähdekoodin 3D-mallinnusohjelma Blender tarjoaa valmiita työkalujen asetteluprofiileja erilaisiin tilanteisiin, mutta tarjoaa käyttäjälle myös mahdollisuuden luoda itse uusia työkalujen asetteluprofiileja sekä jakaa näitä toisille käyttäjille [Blender Foundation, 2012].

Kuva 7: Valmiina tarjottu työkaluprofiili "Model" [Blender Foundation, 2012]

Kuva 8: Käyttäjän luoma työkaluprofiili "temp" [Blender Foundation, 2012]

Näillä työkaluprofiileilla sama ohjelma saadaan tarjoamaan kohdennetumman käyttäjäkokemuksen piilottamalla kyseisessä työvaiheessa tai kyseiselle käyttäjälle tarpeettomia painikkeita ja asettelemalla työkaluvalikoita eri sijainteihin. Sama ohjelmistotuote voi näyttää kahdella eri käyttäjällä hyvin erilaiselta.

Oma lukunsa olisi lisäksi yritys- tai käyttäjäkohtaiset kustomoidut käyttöliittymät yhteen tai useampaan yritysohjelmistoon, mutta en käsittele niitä tässä tutkimuksessa.

5.3. Läpinäkyvä massaräätälöinti

Läpinäkyvä massaräätälöinti on hyvin pitkälti tuottavan yrityksen omaa räätälöintiä. Siinä asiakasta ei vaivata ominaisuusmäärittelyillä ja kyselyillä. Asiakkaalta ei siis suoraan hankita tietoa hänen haluamistaan ominaisuuksista, vaan nuo tarpeet kartoitetaan eri palvelukanavista kerätyllä asiakastuotetiedoilla. [Ahoniemi *et al.*, 2007]

Yrityksellä itsellään on siis vastuu kattavan asiakastuntemuksen hankkimisesta ja toisaalta yritys saa käyttää omaa ammattitaitoaan ja asiantuntemustaan monien ratkaisujen tekemiseen. Läpinäkyvän massaräätälöinnin ei tulisi kuitenkaan unohtaa asiakasta kokonaan kehitystyössä, vaan tietämystä pyritään hankkimaan uusilla tiedonmuodostustekniikoilla, kuten tiedon louhinnalla (engl. *data mining*) ja tietovarastotekniikoilla (engl. *data warehousing*). Näiden tekniikoiden avulla pystytään tunnistamaan myös sellaisia asiakaskäyttäytymismalleja, joita aiemmilla tekniikoilla ei vielä pystytty tunnistamaan. [Ahoniemi *et al.*, 2007]

Läpinäkyvää massaräätälöintiä voidaan katsoa toteutettavan monissa yrityksissä CRM –ohjelmien (customer relationship management, asiakkuudenhallinta) käytön kautta. Tällöin samankaltaista asiakaspalvelua voidaan räätälöidä asiakaskohtaiseksi asiakkaasta kerättyjen tietojen kautta ilman suoraa yhteistoiminnallista räätälöintiä.

5.3.1. Tietojärjestelmät ja läpinäkyvä massaräätälöinti

Tietojärjestelmien kehittämisen on pitkään katsottu olevan hyvin kehittäjälähtöistä, jossa yritys havaitsee tarpeen ja tuottaa järjestelmän täyttämään tuon tarpeen. Asiakasta ei oteta mukaan suunnitteluun, vaan suunnitelmat viedään asiakkaalle korkeintaan käyttäjätestauksen muodossa. Tämä lähestymistapa ei vaadi asiakkaalta työtä tuotteen suunnitteluun, mutta tuote ei myöskään välttämättä vastaa täysin asiakkaan tarpeita, vaan ennemminkin markkinoilla olevan tarpeen täyttämistä. Tietojärjestelmä tuotetaan laajaa ja mahdollisesti vaihtelevaa asiakaskuntaa varten tasapainoillen useiden kohderyhmien tarpeiden välillä.

Jos asiakasta ei oteta mukaan räätälöintiin tietojärjestelmien eri versioiden tuottamisessa voidaan sen katsoa olevan aina läpinäkyvää massaräätälöintiä. Tällöin järjestelmäarkkitehti yrittää tietää käytettävissä olevan asiakastiedon pohjalta asiakkaalle mahdollisimman hyvin toimivan järjestelmän. Joissain triviaalimmissa tapauksissa tällä voidaan saavuttaa toivottu tulos. Esimerkiksi tietoverkkoja ja telekommunikaatiojärjestelmiä toteutettaessa järjestelmän suunnittelijalla voi olla tarpeeksi hyvä käsitys asiakkaan tarpeista – ja teknisissä ratkaisuissa usein parempi ymmärrys kuin asiakkaalla itsellään – jotta hän voi toteuttaa asiakkaan tarpeet täyttävän järjestelmän. Tällöin kuitenkin puhutaan enemmänkin asiakaskohtaisesta räätälöinnistä, kuin massaräätälöinnistä.

5.4. Yhteistoiminnallinen massaräätälöinti

Yhteistoiminnallinen massaräätälöinti on tässä massaräätälöintitapojen nelikentässä esitellyistä kaikkein haasteellisin, vaikein ja kallein. Siinä tuotteen tai palvelun ominaisuuksien määrittely perustuu asiakasvuorovaikutukseen. Asiakas siis otetaan mukaan suunnitteluun oman toimintansa ja tarpeensa asiantuntijana. Yhteistoiminnallista massaräätälöintiä tarvitaan silloin, kun myyjä ei voi yksin tietää, mitä asiakas lopulta haluaa, eikä asiakas osaa yksin ilmaista tai edes hahmottaa haluamaansa tuotteeseen liittyviä monimutkaisia määrittelyjä. [Ahoniemi *et al.*, 2007]

Usein yhteistoiminnallista massaräätälöintiä vaikeuttaakin se, että asiakas on ainoa oman tarpeensa asiantuntija, mutta ei ymmärrä tuotteen valmistamisprosessia, eikä valmistaja puolestaan ymmärrä asiakkaan tarpeita, koska asiakas ei osaa niitä tuoda esiin siinä muodossa, mitä tuotteen suunnittelussa vaaditaan.

Asiakas voi tarvita tuotteen, joka vaatii paljon asiakaskohtaista suunnittelua. Tuote saattaakin olla sen kaltainen, ettei valmiita moduuleita voida hyödyntää. Tällöin molemmat osapuolet, asiakas ja tuottaja, saattavat puhua täysin eri kieltä, eivätkä välttämättä ymmärrä toistensa tarpeita. Yhteistoiminnallinen massaräätälöinti saattaakin vaatia molemmilta osapuolilta paljon toisen asiantuntemusalueen ymmärtämistä ja usein tuskastuttavaa yhteisen kommunikointikielen hakemista.

Näiden vaikeuksien pienentämiseksi voidaan käyttää suunnittelukonfiguraattoreita (engl. *design configurator*). Suunnittelukonfiguraattoreilla voidaan luoda asiakasprofiileja, joita puolestaan voidaan käyttää tekemään saman asiakkaan myöhemmät tilaukset mahdollisimman helpoiksi, tehokkaiksi ja nopeiksi. Tällä puolestaan voidaan parantaa asiakkaan lojaaliutta yritystä kohtaan. Suunnittelukonfiguraattoreilla voidaan myös määritellä asiakkaalle räätälöinnin raamit ja toimintamallit, jolloin massaräätälöinnistä tulee hallittavampaa. [Mäkipää *et al.*, 2012]

5.4.1. Tietojärjestelmät ja yhteistoiminnallinen massaräätälöinti

Nykypäivänä tietojärjestelmiä tuotetaan paljolti yhteistoiminnallisesti. Tässä täytyy kuitenkin erottaa ohjelmistotuotteiden osalta niin kutsutut hyllyohjelmistot eli sellaiset ohjelmistotuotteet, jotka on tarkoitettu myytäviksi valmiina paketteina asiakkaille tiettyyn ennalta päätettyyn tarkoitukseen. Tällöin asiakasta ei tunneta ohjelmiston suunnittelun aikana, vaan asiakaskunnan tarpeet arvioidaan korkeammalla tasolla ja ohjelmisto mahdollisesti testataan oletetun asiakaskunnan edustajiksi valitulla ryhmällä.

Näissä ohjelmistoissa kehittäjä ei käytä yhteistoiminnallista massaräätälöintiä vaan massaräätälöinnin voidaan korkeintaan katsoa tapahtuneen siinä kohdassa, jossa kehittäjä on saattanut hyödyntää olemassa olevia ohjelmistokomponentteja modulaarisesti.

Käytännössä kuitenkin tietojärjestelmistä suuri osa toteutetaan tietyn asiakkaan tiettyyn tarpeeseen yhteistoiminnallisesti ja sen jälkeen tarjontaa laajennetaan useampaan samankaltaiseen tarpeeseen yhteistoiminnallisella massaräätälöinnillä jo olemassa olevia komponentteja, tuotantotapoja ja toimialueen ymmärrystä hyödyntäen.

Vaikka siis kehittäjän kannalta useassa tuotteessa tulee olemaan hyvinkin samanlaisia komponentteja ja perustekniikoita, saa kukin asiakas juuri heidän tarpeisiinsa räätälöidyn tietojärjestelmän.

Yhteistoiminnallisessa massaräätälöinnissä tietojärjestelmiä kehitettäessä asiakkaalta vaaditaan laajaa osallistumista oman toimintansa asiantuntijana. Tämä edellyttää asiakkaalta sitoutumista ja investointeja vähintäänkin työtunteina. Mutta koska asiakaskohtaisesti toteutetut tietojärjestelmät saattavat nousta yrityksen suurimmiksi rahallisiksi investoinneiksi, on asiakkaan edustajan mukanaolo kaikkien edun mukaista. Tällöin myöskään tietojärjestelmän tuottajan ei tarvitse kantaa yksin riskiä siitä, että tuotettu järjestelmä ei vastaakaan asiakkaan tarpeita, kuten on usein tapana tuotannossa, jossa asiakas ei ole mukana. Tällöinhän kehittäjäosapuoli kantaa kaiken taloudellisen vastuun. Tietojärjestelmä tuotetaan ensin ja asiakas päättää vasta sitten haluaako hän sen ostaa. Järjestelmien tuottaja ei sijoittaa työtuntejaan loputtomasti tuottamattomaan työhön, jossa valmistetaan jotain, näytetään se asiakkaalle ja joudutaan kenties yrittämään uudelleen. Valitettavasti tällaista kuitenkin monet asiakkaat odottavat.

Yhteistoiminnallisesti tuotettu tietojärjestelmä saattaa kuitenkin sisältää monia osakokonaisuuksia, jotka itsessään eivät täytä vain yhden asiakkaan tarpeita ja siten samoja osia voidaan jatkossa käyttää (yhteistoiminnalliseen) massaräätälöintiin muitakin asiakkaita varten. Esimerkkinä tällaisesta yhteistoiminnallisesta tietojärjestelmien massaräätälöinnistä voisi olla vaikkapa isojen keskusvarastojen inventaari- ja ohjausjärjestelmät. Vaikka kunkin asiakkaan järjestelmä saattaa olla hyvinkin erilainen, on kuitenkin todennäköistä, että kukin heistä käyttäisi jokseenkin samankaltaisia liukuhihnakuljettimia, robottihissejä tai ainakin sähköisiä seurantajärjestelmiä.

Avoimen lähdekoodin ohjelmistokehitys liikkuu toteutukseltaan usein melko lähellä yhteistoiminnallista massaräätälöintiä. Internetissä on useita versionhallintasivustoja, jotka tarjoavat käyttäjilleen mahdollisuuden säilyttää versionhallintatietorakennetta ja versiohallittuja tiedostoja pilvipalveluissa, joissa usein tarjotaan tiedostoihin avoin pääsy toisille kehittäjille. Näin useat kehittäjät voivat hyödyntää toistensa ohjelmakomponentteja ja jopa valita jo olemassa olevan valmiin ohjelman ohjelmakoodin oman ohjelmansa pohjaksi, tehden siihen sitten haluamansa muutokset kulloinkin valitun lisenssin asettamissa rajoissa.

Tämä mahdollistaa yhteistoiminnallista massaräätälöintiä myös siten, että muut kehittäjät voivat antaa palautetta ja parannusehdotuksia tai jopa toteuttaa korjaukset ja

uudet ominaisuudet itsenäisesti ja lähettää sitten alkuperäiselle kehittäjälle pyynnön lähdekoodien yhdistämiselle.

Myös monet yritykset voivat tarjota asiakkailleen mahdollisuuden ehdottaa uusia ominaisuuksia tai parannuksia ohjelmistoihinsa. Tämän lisäksi jotkut yritykset tarjoavat asiakkailleen mahdollisuuden lähettää automaattisesti käyttötietoja yrityksen ohjelmista, sekä virheraportteja ohjelman lopetettua toimintansa yllättävästi.

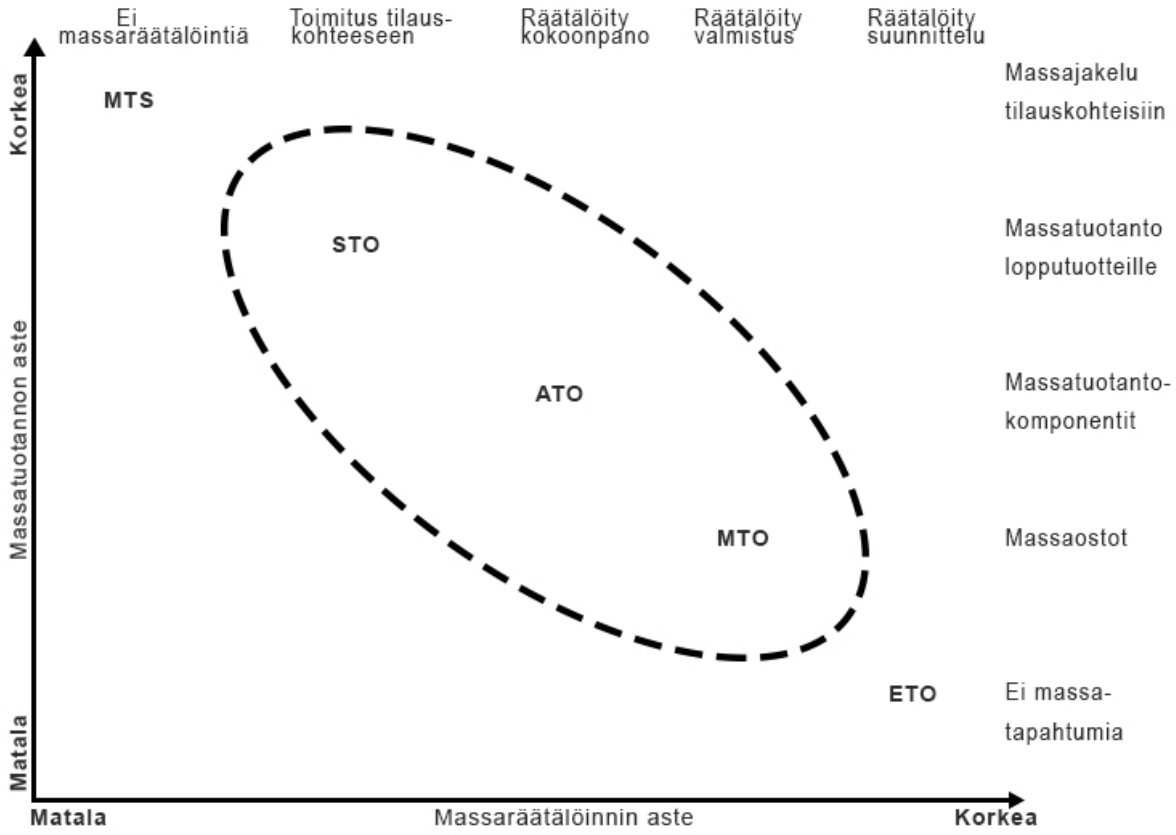
6. Valmistuksen ohjausprosessit ja ohjelmistotuotteiden massaräätälöinti

Ohjelmistotuotteiden massaräätälöinnistä puhuttaessa kaikki teollisuuden massaräätälöinnin valmistusprosessit eivät ole täysin vastaavia, mutta tiettyjä analogioita kuitenkin voidaan löytää näiden välille.

Ahoniemi *et al.* [2007] listaavat prosessin ohjaustapoja massaräätälöinnistä massatapahtumiin. Massaräätälöinnin päässä on tuotteen suunnittelu tilauskohtaisesti (ETO, *Engineering to order*), tuotteen valmistaminen tilauskohtaisesti (MTO, *Make to order*), tuotteen kokoonpaneminen tilauskohtaisesti (ATO, *Assembly to order*), tuotteen toimittaminen tilauskohtaisesti (STO, *Ship to order*), sekä kauimpana massaräätälöinnistä, massatapahtumana tapahtuva tuotteen valmistaminen varastoon (MTS, *Make to stock*).

Perinteisesti monia ohjelmistotuotteita on valmistettu varastoon, ei niinkään ohjelmistoa massavalmistamalla, vaan ohjelmistotuotetta massamonistamalla eli tuottamalla asennusmedioita myyntipaketteihin ja liikkeiden hyllyihin. Nykyään verkossa toimivat ohjelmistojen jakelukanavat ovat jonkin verran muuttaneet tätä painotusta. Latauspalvelut, kuten Applen App Store, Valven Steam ja Microsoftin Windows Store tarjoavat asiakkaalle mahdollisuuden ostaa ja ladata ohjelmistotuotteita verkon välityksellä ja tuotteet toimitetaan tilauskohtaisesti.

Todellisessa ohjelmistotuotteiden massakustomoinnissa ohjelmistotuotteet voitaisiin kokoonpanna tilauskohtaisesti olemassa olevista ohjelmistomoduuleista koostamalla, mutta käytännössä tänä päivänä ohjelmistotuotteiden massakustomointi vaatii ainakin jossain määrin myös tuotteen valmistamista tilauskohtaisesti eli ihmisen toimintaa automatisoidun tuotantolinjan yhteydessä. Tuotteen valmistaminen tilauskohtaisesti on puolestaan normaalia tilauskohtaista ohjelmistokehitystyötä, jossa kuitenkin taustalla yritys saattaa käyttää jo olemassa olevia ohjelmistokoodinpätkiä tai –moduuleita.



Kuva 9: Prosessien ohjaustavat suhteessa massaräätälöintiin [Ahoniemi et al. 2007]

Kuvassa 9 on havainnollistettu valmistuksen ohjausprosessien sijoittumista massaräätälöinnin ja massatapahtumien välille.

7. Ohjelmistotuotteiden massaräätälöintitapojen kehitys

Krueger, joka on julkaissut paljon tietojärjestelmien massakustomoinnista, esittää julkaisuissaan *New Methods in Software Product Line Practice – Examining the benefits of next-generation SPL methods* [2006] ja *New Methods in Software Product Line Development* [2006] massakustomointiin ja tuotantolinja-ajatteluun liittyviä kehitysaskelia, malleja ja käytäntöjä.

Krueger [2006b] kuvaa tuotantolinja-ajattelua prosessina, jossa kokoelmaa valmiita osia voidaan automaattisesti koostaa ja määrittää eri tavoilla luoden näin erilaisia ohjelmistotuotteita. Kaikki tuotantolinjan tuotteet koostetaan automaattisesti jonkin olemassa olevan abstraktin formaalin ominaisuusprofiilikuvauksen mukaisesti.

Verrattuna käsin manuaalisesti ominaisuuksien koostamiseen tämä korkeammin automatisoitu lähestymistapa on helpommin omaksuttavissa yrityksiin ja sen käyttöä on halvempaa jatkaa.

7.1. Massaräätälöinnin ensimmäinen sukupolvi

Alun perin tätä mallia sovellettiin jakamalla työtehtävät ohjelmiston toteuttajille (*domain engineering*) ja sovelluksen toteuttajille (*application engineering*). Ohjelmalla tässä kohtaa tarkoitetaan rajattuja ohjelmatoiminnallisuuksia ja sovelluksella lopullista ohjelmistotuotetta. Ohjelmiston toteuttajat kehittivät uudelleenkäytettäviä ydinosia, joista sovelluksen tuottajat sitten koostivat uudelleenkäytön kautta ohjelmistotuotteet.

Vaikkakin tämä lähestymistapa voisi ideaalitilanteessa toimia tehokkaasti ja vaikka tämä lähestymistapa on pohjimmiltaan hyväksi havaittua modulaarista tuotekehitystä, alkaa ongelmia tosielämässä pian ilmetä. Tosielämässä tuotantolinjat elävät ja kehittyvät, tai ainakin muuttuvat, josta alkaa nousta ongelmia tälle lähestymistavalle.

Ensimmäiseksi kehittäjät joutuvat manuaalisesti kirjoittamaan tuotekohtaisia määrittelyjä sekä ”liimakoodia” eli ohjelmakoodia, joka koostaa eri osat ja yhdistää ne toisiinsa. Tuotteet ovat usein yksilöllisiä ja koska jokainen tuote vaatii kehittäjän toteuttamaan määrittelyt ja koontikoodit, on työmäärä lineaarisessa korrelaatiossa tuotantolinjan tuotteiden määrän kanssa.

Toisena ongelmana nousee ohjelmien uniikki rajattu konteksti, jossa ydinosia käytetään. Ydinosiin on vaikea tehdä merkittäviä uudistuksia, joissa niiden rakenne saattaa muuttua, sillä muutokset pitäisi pystyä sulauttamaan takaisin tuotekohtaiseen ohjelmakoodiin jokaiseen tuotantolinjan sovellukseen.

Kolmantena ongelmana Krueger nostaa esiin vielä sosiaalisen ongelman, joka saattaa nousta työvoiman jakautumisesta ohjelmiston toteuttajiin ja sovellusten toteuttajiin. Eri ryhmien työntekijät eivät välttämättä koe olevansa osa suurempaa kokonaisuutta, vaan he saattavat ryhmittyä toisiaan vastaan kilpaileviin ryhmittymiin. Jos lopputuotteessa

ilmenee ongelmia, saattavat eri ryhmittymät etsiä syyllistä toisesta ryhmästä sen sijaan, että kaikki toimisivat yhdessä ongelman ratkaisemiseksi.

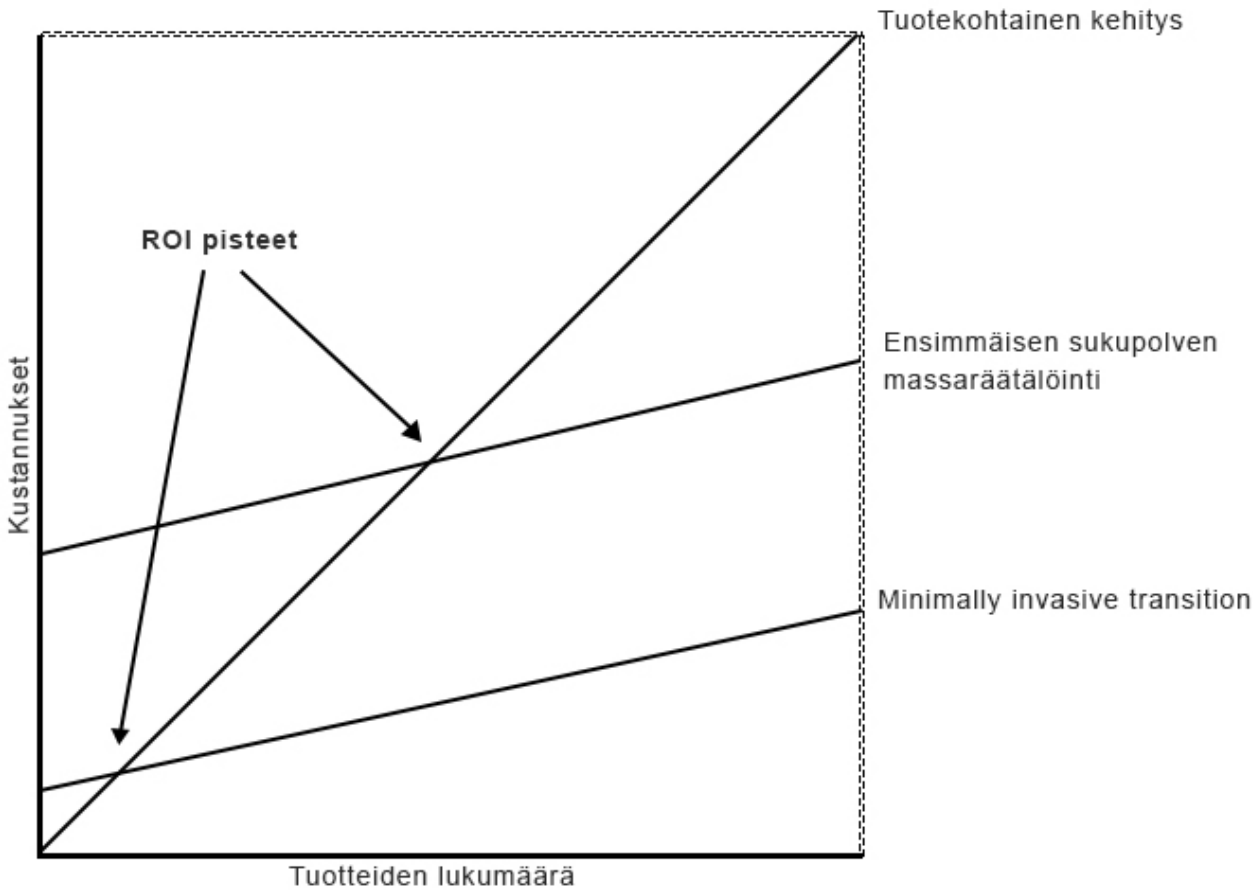
Kuitenkin automatisoitujen koontilinjojen myötä näiltä ongelmilta voidaan jossain määrin välttyä, tai ainakin niiden määrä oletettavasti laskee. Kuitenkaan täysin automatisoitua koontilinjaa, ”universal constructor”:ia, on hyvin vaikea toteuttaa ei-triviaaleissa käyttötilanteissa.

7.2. Massaräätälöinnin omaksumiskynnyksen madaltaminen

Minimally invasive transition on terminä tietoisesti johdettu kirurgisesta toimenpiteestä, jossa ongelma pyritään korjaamaan mahdollisimman vähäisellä haitalla. Tämä kuvaa mallin mahdollisimman vähäistä haittaa olemassa oleviin tuotantoaikatauluihin siirryttäessä tuotantolinjoihin ja massakustomointiin normaalista tuotekeskeisestä kehitystavasta.

Minimally invasive transitions hyödyntää voimakkaasti jo olemassa olevia ohjelmistokoodeja ja resursseja omaksuttaessa vaiheittain uudenlaista tuotantoajattelua. Tämän ajattelun taustalla on madaltaa usein varsin korkeata omaksumiskynnystä siirryttäessä pois tuotekeskeisestä kehitystavasta.

Perinteisesti tuotekeskeisesti toteutettaessa tuotteiden valmistuskustannukset ovat lineaarisessa korrelaatiossa tuotteiden lukumäärän kanssa. Massaräätälöinnin kautta kulukuvaajaa saadaan loivennettua, mutta aloituskustannukset ovat usein niin korkeat, ettei monella yrityksellä ole sellaiseen mahdollisuutta. *Minimally invasive transitions* pyrkii tuomaan siirtymän miltei huomaamatta normaalien prosessien rinnalle.



Kuva 10: Alkukustannukset ja takaisinmaksupisteet massaräätälöinnissä [Muokattu Krueger, 2006a]

Kuvassa 10 on havainnollistettu perinteisen tuotekeskeisen tuotantomallin, ensimmäisen sukupolven tuotantolinja-ajattelun, sekä *minimally invasive transition* aloituskustannukset sekä odotetut ROI -pisteet (*Return of interest, investoinnin takaisinmaksupiste)*. Vaikka jo ensimmäisen sukupolven tuotantolinja-ajattelu lupasi pitemmän päälle säästöjä tuotantokustannuksiin, oli sen aloitusinvestointi useimmille kuitenkin liian korkea.

7.3. Tuotantolinjan variaatioiden hallinta

Koostettaessa uusia ohjelmistotuotteita ydintoiminnallisuuksien ja lisäominaisuuksien sekoituksena variaatioiden määrä voi helposti karata käsistä. Krueger [2006b] kuvaa termillä *bounded combinatorics* yritystä rajoittaa variaatioiden määrää.

Laittaakseni variaatio-ongelmaa perspektiiviin voimme suorittaa yksinkertaisen laskutoimituksen. Jos ohjelmassa on 33 erilaista toimintovariaatiota, jotka voidaan toisistaan riippumatta joko ottaa mukaan tai jättää pois valmiista ohjelmistotuotteesta, näitä erilaisia tuotevariaatioita riittäisi jokaiselle maailman ihmiselle. Kaksisataakuusitoista vastaavaa ominaisuutta toisi variaatioita suunnilleen saman

määrän, kuin mitä koko maailmankaikkeudessa arvellaan olevan atomeja. Kuitenkin Krueger [2006b] mainitsee yrityksistä, jotka ovat raportoineet yli 1000 ominaisuusvarianttia. Pelkkä halutunlaisen lopputuotteen määritteleminen yli tuhannesta vaihtoehdosta on työlästä. Tämän katsotaankin johtavan alenevaan päätöksentekokykyyn ja epäolennaisten kriteerien käyttöön mahdollisuuksien rajaamisessa [Franke and Piller, 2003].

Tätä tilannetta kutsutaan alan kirjallisuudessa usein massahämmennykseksi (Huffman and Kahn, 1998, Krueger, 2006, Franke and Piller, 2003, alunperin Pine, 1994) ja se esitetään usein kritiikkinä massaräätälöinnin toteutusta kohtaan. Massahämmennystä tarkastellaan syvemmin luvussa *Massaräätälöinnin ongelmia ja kritiikkiä*.

8. Massaräätälöinnin ongelmia ja kritiikkiä

Massaräätälöinnistä on tullut suosittu tavoite monelle yritykselle oman kilpailukyvyn ja taloudellisen kannattavuuden kehittämiseksi. Osa yrityksistä saattaakin pitää massaräätälöintiä varmana tapana ratkaista kaikki tuotannolliset ja taloudelliset ongelmat pelkällä uuteen tuotantomalliin siirtymisellä. Käytännössä massaräätälöinti ei kuitenkaan ole ollut yksinkertainen ja taattu tie onneen.

Monet yritykset saattavat pyrkiä kokeilemaan massaräätälöintiä oman tuotantonsa ohessa sitoutumatta tai edes ymmärtämättä massaräätälöinnin kaikkia osa-alueita.

8.1. Massahämmennys

Yksi kenties yleisimmistä massaräätälöinnin ongelmista on asiakkaan mahdollisesti kokema massahämmennys – vaihtoehtojen määrän kasvaessa sen itselle parhaiten sopivan tuotevariaation löytyminen saattaakin vaikeutua.

Huffman ja Kahn [1998] kuvaavat yhdysvaltalaisen huonekaluliikkeen, Choise Seating Galleryn, massaräätälöintiä sohvien tarjonnassa. He mainostivat 500 tyyliä, 3000 kangasvaihtoehtoa ja 350 nahkaverhoiluvaihtoehtoa. Asiakas kuitenkin yleensä halusi vain yhden sohvan. Kyseinen 150,000 kangassohvavariantin valikoima saattoikin olla hämmentävä ja musertava hyödyllisen sijaan.

Yleensä liian laaja valikoima koetaankin turhauttavana ja ylivoimaisena, eikä sillä saavuteta asiakastyytyväisyyttä – päinvastoin [Franke and Piller, 2003].

1980-luvulla Japanilainen autovalmistaja Nissan esitteli *Intelligent Body Assembly System* –valmistusmetodinsa, jolla yhtiön mukaan voitiin sarjavalmistaa vapaavalintainen määrä vapaavalintaista korimallia kustannustehokkaasti. Tässä onnistuttiinkin hyvin ja samalla luotiin hyvä esimerkki käytännön massaräätälöintilinjastosta. Linjasto muistetaan kuitenkin paremmin siitä, että se tarjosi massaräätälöintivaihtoehtoja paljon enemmän, kuin asiakas pystyi omaksumaan tai edes käsittämään. [Teresko, 1994]

Ihmisellä on rajattu kyky ottaa vastaan informaatiota ja tyypillisesti massaräätälöintiä tarjoavien yritysten tuotevariaatiot ylittävät tämän kyvyn moninkertaisesti. Hahmottaaksemme edes jotenkin tätä variaatioiden suurta määrää, voimme tehdä summittaisia laskutoimituksia:

”Eräs urheilujalkinevalmistaja tarjoaa massaräätälöityjä urheilujalkineita. Jos he haluaisivat esittää kaikki kenkävariaationsa (noin $3*10^{21}$ kappaletta) liikkeessä, tarvittaisiin 7 000 kertaa koko maapallon pinta-ala kyseiselle valikoimalle.” [Franke and Piller, 2003]

On helppoa uskoa, ettei ihmisen rajallinen tiedonomaksumiskyky riitä tuollaisen valikoiman läpikäymiseen ja sisäistämiseen. Massahämmennys voi johtaa tuttavallisempaan informaatioähkyyn (engl. *Information Overload*).

Sohvan verhoilukankaan valinta tai urheilukengän tikkausten väri ovat isoilta osin vain makuasioita. Mutta suunniteltaessa tietojärjestelmiä, joilla voi olla satoja käyttäjiä ja hyvin erilaisia käyttötottumuksia ja joita saatetaan tulla käyttämään monien vuosien ajan alati muuttuvassa liiketoimintaympäristössä, ei valintoja voida tehdä kovin kevein perustein miettimättä pitkäaikaisia vaikutuksia. Tällöin joudutaankin miettimään asiakkaan osapuolten ja kehittäjien rooleja asiantuntijuuden kannalta.

8.2. Asiakkaan ja asiantuntijan roolit

Tietojärjestelmän kehittäjä on hyvin harvoin asiakkaan liiketoiminta-alan asiantuntija. Hän ei myöskään tavallisesti ole asiakkaan liiketoimintakäytännöissä ja tarpeissa samanlainen asiantuntija, kuin asiakas itse. Toisaalta asiakas on harvoin tietojärjestelmäarkkitehtuurin asiantuntija. Näiden kahden osapuolen välisen dialogin ongelmissa lieneekin syy moniin tietojärjestelmissä ilmeneviin tyytymättömyyksiin. Yhteisen kielen löytäminen voi olla vaikeaa, jos kehittäjä ei ymmärrä mitä asiakas haluaa, eikä asiakas ymmärrä, minkälaiset asiat ovat edes mahdollisia.

Kehittäjällä onkin suuri vastuu omaksua asiakkaan toimintatapojen tietämystä ja toimialatuntemusta. Usein tietojärjestelmäarkkitehdit saattavat myös joutua huomaamaan, että hyvä tuote ei olekaan välttämättä se, jonka asiakas haluaa, vaan se, jonka asiakas oikeasti tietämättään tarvitsee.

Asiakas myös odottaa saavansa asiantuntijalta juuri asiantuntijuutta tietojärjestelmätuotannosta. Asiakas on oman toimintansa paras asiantuntija, mutta ei olisi mielekästä vaatia häneltä syvempää ymmärrystä myös tietojärjestelmätuotannosta.

9. Ohjelmistotuotantoparadigmat ja massaräätälöinti

Ohjelmistotuotantoparadigmojen suhde massaräätälöintiin on kaksijakoinen. Toisaalta mikään ohjelmistotuotantoparadigma ei sellaisenaan täysin vastaa teollisuuden massaräätälöintiä, sillä toimintaympäristö on poikkeava. Ohjelmistotuotannossa ei valmisteta aineellista tuotetta raaka-aineista kuten muussa teollisuudessa. Raaka-aineita ei siis kulu, niitä ei tarvitse hankkia eikä niiden hankintaan liittyvää tuotantoketjua tarvitse toteuttaa. Tietojärjestelmätuotannossa sen sijaan on usein aineellisia hyödykkeitä, kuten laitteita, mutta täysin suoraa analogiaa perinteiseen teollisuuteen ei silti voida muodostaa immateriaalisen ohjelmisto-osuuden suuren merkityksen vuoksi.

Toisaalta taas monia yhtymäkohtia ohjelmistotuotantoparadigmojen ja perinteisen teollisuuden välillä voidaan löytää. Modulaarinen ohjelmistotuotanto, ohjelmistokomponenttien uudelleenkäyttö ja tuotantolinja-ajattelu luovat yhtymäkohtia ja samankaltaisuuksia.

9.1. Ketterät ohjelmistotuotantoparadigmat

Etenkin ohjelmistotuotannon puolella 1990-luvun jälkimmäinen puolisko oli nopean muutoksen aikaa. Tammikuusta 1994 helmikuuhun 2000 teknologiapörssi NASDAQ:n indeksi nousi 605% 775,80:stä 4 696,69:ään. Tätä ajanjaksoa on sittemmin alettu kutsua IT kuplaksi tai *dot-com kuplaksi*. [Galbraith and Hale, 2003]

Vaikka kuplan puhjettua koimmekin seesteisemmän ajankauden, ympäristön muutos jatkuu edelleen nopeana. Vastatakseen nopeaan toimintaympäristön muutokseen alalla on omaksuttu uusia, ketteriä tuotantoparadigmoja. Tällaisia uusia ketteriä menetelmiä ovat mm. *Extreme Programming (XP), Dynamic Systems Development Methodology* ja *Scrum* [Highsmith and Cockburn, 2001].

Highsmith ja Cockburn [2001] kertovat tilastosta, jonka mukaan yli 200:sta tutkitusta ohjelmistoprojektista miltei puolesta ei löytynyt alkuperäisiä suunnitelmia. Heidän mukaansa tilanne selittyy sillä, ettei suunnitelmien noudattaminen ole enää keskeisen tärkeää. Sen sijaan projekteissa tähdätään asiakastyytyväisyyteen tuotetta luovutettaessa, ei projektia suunniteltaessa. Monissa seuratuissa projekteissa suuria muutoksia tuli kehittäjistä johtumattomista syistä kesken tuotannon vaatimusmäärittelyihin ja käytettäviin teknologioihin.

Barry Boehm on esittänyt melko kuuluisan teorian ohjelmistoprojektin muutosten hinnasta – *life cycle cost differentials theoryn* – jonka mukaan muutoksen kustannukset nousevat kehitystyön edetessä. Mitä pidemmälle toteutuksessa on ehditty, sitä työläämmiksi (ja siten kalliimmiksi) muutosten tekeminen tulee.

Tämän teorian ajatuksen mukaan monet yrityskulttuurin kypsyydeltään nuoremmat yritykset toimivat sen mukaan, että jos vain yritämme tarpeeksi kovasti, voimme

ennakoida kaikki vaatimukset jo projektin alussa ja välttää näin kalliita muutoksia myöhemmin. Tämänkaltainen ennakoiva ja kankea ajattelutapa on usein pohjana niin kutsutun vesiputousmallin taustalla. Vesiputousmallissa pääpiirteissään jokainen tuotannon vaihe pyritään toteuttamaan alusta loppuun kokonaisuutena ennen seuraavaan vaiheeseen siirtymistä. Esimerkiksi koko tuote suunnitellaan alusta loppuun ennen kuin sen toteuttamista aloitetaan ja sitten koko toteutusvaihe suoritetaan loppuun asti ennen kuin tuotetta aletaan testaamaan mahdollisten virheiden varalta.

Tämä malli on yksinkertainen ja ideaalitilanteessa se saattaisi tuottaa toivotun lopputuloksen. Harvoin kuitenkaan muuttuvassa ympäristössä kaikkia vaatimuksia voitaisiin ennustaa täydellisesti etukäteen ja tämänkaltaisen osioidun ja kankean kehitysmallin kanssa ajaudutaankin helposti tekemään paljon toistavaa työtä, joka puolestaan kasvattaa projektin läpimenoaikaa huomattavasti. Muutosten mahdollisuuden eliminoiminen varhaisessa vaiheessa puolestaan tarkoittaa joustamattomuutta muuttuvaa ympäristöä ja muuttuvia vaatimuksia kohtaan.

Koska kuitenkin ohjelmistotuotannon ala toimii alati muuttuvassa ympäristössä, sen sijaan, että yrityksen yrittäisivät eliminoida muutoksen varhaisessa vaiheessa, tulisikin niiden keskittää huomionsa siihen, miten muutoksesta saataisiin halvempaa ja miten siihen voitaisiin varautua paremmin pysyen joustavana ja mukautuvana [Highsmith and Cockburn, 2001].

Toisena peruslähtökohtana ketterissä tuotantomenetelmissä on variaatioiden hallinta. Perinteisen tietojärjestelmätuotannon paradigmat ohjaavat jatkuvalla mittaamisella ja prosessiohjauksella variaatiot pois tuotteesta ajatellen, että variaatio tarkoittaa samaa kuin virhe [Highsmith and Cockburn, 2001]. Vaikka variaatiot saattavat edelleen aiheutua virheistä, myös toimintaympäristön muutos aiheuttaa variaatioita.

Ketterien menetelmien kolmantena peruslähtökohtana voitaisiin pitää laatua, joka voidaan saavuttaa sekä edellä mainitun nopean prototyyppitestauksen myötä, että ketteriin menetelmiin kuuluvan suunnittelumallien laajan käytön kautta [Bozheva and Gallo, 2005]. Odotukset tietojärjestelmille ovat kasvaneet vuosien varrella.

”*Markkinat odottavat innovatiivisia, korkealaatuisia, kohdennettuja ohjelmistotuotteita – ja pian*.” [Highsmith and Cockburn, 2001]

Ketterät ohjelmistotuotantomenetelmät painottavat laatua suunnittelussa. Koska suunnittelua tehdään samanaikaisesti toteutuksen kanssa, saatetaan ketteriä menetelmiä kutsua virheellisesti *ad hoc* –kehittämiseksi tai villin lännen kehittämiseksi. Villin lännen kehityksessä (engl. *Cowboy coding*) sovelluskehittäjien katsotaan toimivan villeinä vailla suunnitelmia ja ohjausta. Kuitenkin, vaikka ketterissä menetelmissä koko tuotetta ei

olekaan suunniteltu aluksi valmiiksi, on suunnittelu vahvasti mukana toteutuksessa – sitä vain toteutetaan pienemmissä palasissa tarpeen mukaan.

Ketterät menetelmät nojaavat vahvasti ohjelmistokoodin esittelemiseen. Projektin jäsenille ja asiakkaalle voidaan esittää toimivaa ohjelmistokoodia sen sijaan, että vain puhuttaisiin asioista, joita voitaisiin tehdä. Siis sen sijaan, että luvataan jonkin ominaisuuden olevan melko helposti ja nopeasti toteutettavissa, siirrytään käytännön tasolla ja näytetään miten toteutettu ominaisuus todella toimii. Näin asiakaskin saattaa saada paremman käsityksen koko prosessista ja aiemmin vain käsitteiden tasolla kuvatut asiat saavat konkretian. Toimiva ohjelmistokoodi ei valehtele.

Ketterät menetelmät saattavat hyötyä myös laadullisesti dokumentaatioiden puuttumisesta. Kun kehittäjät eivät voi nojata kaikessa valmiisiin suunnitteludokumentaatioihin ilman tarvetta omalle ajattelulle, he oletettavasti alkavat keskustelemaan kollegoidensa kanssa ja saattavat siten löytää uusia, parempia ratkaisuja. Lisäksi näin kehittäjät joutuvat käymään dialogia asiakkaan kanssa, jolloin molemmat puolet saavat paremman kuvan kokonaisuudesta. He voivat muuttaa tärkeysjärjestyksiä ja löytää vaihtoehtoisia toteutustapoja ongelmallisiksi osoittautuneille ominaisuuksille.

On mahdollista, että monissa ohjelmistoprojekteissa asiakas toivoo jotain sellaista ominaisuutta, jonka toteuttaminen on verrattain työläämpää kuin mikä asiakkaan tarve kyseiselle ominaisuudelle. Kun kehittäjä ja asiakas pääsevät keskustelemaan asiasta, saattaa käydä ilmi, että jokin toinen ratkaisumalli tuottaisi asiakkaalle saman arvon, mutta vähentäisi työmäärää.

Näissä esimerkeissä korostuu ketterille menetelmille tyypillisen kehittäjän ja asiakkaan välisen vuoropuhelun hyödyt perinteisempään ohjelmistotuotantoon verrattuna. Asiakaspalavereihin usein osallistuvat työnjohdon edustajat, jotka toisaalta saattavat puhua paremmin asiakkaan kanssa samaa kieltä, mutta toisaalta eivät ole niin syvällisesti mukana käytännön ohjelmistokehityksessä kuin itse ohjelmistokehittäjät.

Ketterät ohjelmistotuotantomenetelmät kohdistuvat paljolti samankaltaisten ongelmien voittamiseen, tai niihin varautumiseen, mutta niiden käytäntö ja toteutus vaihtelevat jonkin verran.

9.1.1. Extreme programming

Extreme programming (lyh. XP), kuten muutkin ketterät menetelmät, tähtää muutoksen hinnan pienentämiseen koko ohjelmistoprojektin läpiviennin ajan. Highsmith ja Cockburn [2001] kuvaavat XP:tä siten, että sitä käyttävät ohjelmistokehittäjät pyrkivät tuottamaan ensimmäisen testiversion viikoissa saadakseen nopeasti palautteen asiakkaalta ja päästäkseen nopeasti testaamaan ratkaisua mahdollisimman aidossa ympäristössä.

Ongelmiin pyritään myös löytämään mahdollisimman yksinkertaisia ratkaisuja vähentääkseen muutoksen tarvetta ja toisaalta helpottamalla mahdollisia muutoksia.

Proseduurit ovat mahdollisimman rajattuja ja yksinkertaistettuja, jolloin jonkin tietyn toiminnallisuuden vaihtamiseen vaadittu työmäärä ja aika pienenee.

Myös suunnittelulaatua pyritään parantamaan jatkuvasti, jolloin seuraavan esitettävän version kehittäminen tulee jatkuvasti hieman halvemmaksi ja nopeammaksi. Lisäksi näitä esitettäviä versioita pyritään testaamaan jatkuvasti, jotta mahdolliset virheet voitaisiin havaita mahdollisimman varhaisessa vaiheessa, jolloin niiden korjaaminen on halvempaa ja nopeampaa.

9.1.2. Dynamic Systems Development Methodology

Dynamic Systems Development Methodology (lyh. DSDM) pyrkii varmistamaan laadun tuottamalla nopeasti sarjan prototyyppiratkaisuja kuhunkin tuntemattomaan alueeseen. Tällaisia alueita voivat olla esimerkiksi uudet teknologiat, uudet liiketoimintaympäristöt ja uudet käyttöliittymät. [Highsmith and Cockburn, 2001]

Toisin kuin Extreme Programming, Dynamic Systems Development Methodology ja Scrum keskittyvät enemmän projektinhallintaan ja yhteistyötahojen yhteistoimintaan, kuin itse ohjelmistotuotannon erityiskysymyksiin.

9.1.3. Scrum

Scrum hakee laatua verrattain nopeista kehitysiteraatioista ja vahvasta kehityksen seurannasta. Scrum -mallissa kehitystiimi järjestää säännöllisesti päivittäin 15 minuutin Scrum –tapaamisia, joissa tilaisuuden johtaja, Scrum -master, kysyy jokaiselta osallistujalta edellisen päivän aikaansaannokset ja mahdolliset esiin nousseet ongelmakohdat. Näiden päivittäisten lyhyiden tapaamisten lisäksi järjestetään laajemmat katselmoinnin jokaisen 30 päivää kestävän iteraation päätteeksi. Jokainen iteraatio on etukäteen määritelty ja koko prosessi on pääpiirteissään suunniteltu. [Highsmith and Cockburn, 2001]

9.2. Ketterä tietojärjestelmätuotanto ja massaräätälöinti

Ketterien menetelmien yhteys massaräätälöintiin löytyy käytännön tasolta. Ketteriin tietojärjestelmätuotantomenetelmiin liittyy oleellisesti esimerkiksi suunnittelumallien laaja hyödyntäminen ja sitä kautta myös modulaarinen, komponenttipohjainen tietojärjestelmätuotanto [Bozheva and Gallo, 2005].

Ketteriä tietojärjestelmätuotantomenetelmiä ei siis sellaisenaan ole suunniteltu juuri massaräätälöinnin tarpeisiin, mutta katsauksessa mahdollisesti tietojärjestelmien massaräätälöintiä tukeviin tekniikoihin ja toimintamalleihin, niiden voidaan katsoa osaltaan toteuttavan tiettyjä massaräätälöinnin vaatimuksia, tai massaräätälöintiä tukevia ominaisuuksia ja toimintatapoja.

Suunnittelumallien lisäksi ketterät tietojärjestelmätuotantomenetelmät hyödyntävät myös ohjelmistokehyksiä ja yleisesti massaräätälöintiin liitettäviä hyviä

ohjelmistosuunnittelutapoja ohjelmistoarkkitehtuureista massaräätälöinnille otollisiin ohjelmistokehitystapoihin [Greenfield and Short, 2003].

9.3. Modulaarinen, komponenttipohjainen tietojärjestelmätuotanto

Tietojärjestelmät ovat luonteeltaan monin osin triviaalisti modulaarisia ja komponenttipohjaisia. Vaikka joskus toteutettavat tietojärjestelmät saattavatkin vaatia uusia käyttölaitteita (enlg. *Human Interface Device, HID*) tai tiedonsiirtoväyliä (esim. työstökoneiden PLC –pohjaiset komentokanavat), niin pääsääntöisesti nämäkin komponentit toteutetaan samalla kertaa uudelleenkäytettäviksi, erillisiksi osiksi. Ohjelmistokehityksen puolella modulaarinen, komponenttipohjainen suunnittelu ei kuitenkaan omaa yhtä pitkiä perinteitä, kuin tietojärjestelmätuotannon lähempänä perinteistä teollista tuotantoa olevat osat.

Komponenttipohjainen ohjelmistotuotanto (engl. *Component-based software engineering, CBSE*) tähtää ohjelmistojen koostamiseen *yhdistä ja käytä* –komponenteista (engl. *plug and play*) [Aoyama, 1998]. Alunperin komponenttipohjaisuuteen siirtymisen moottorina toimi ohjelmistokoodin laajemman uudelleenkäytön saavuttaminen, mutta siitä löytyy paljon samankaltaisuuksia myös massaräätälöinnin ajatuksiin. Ohjelmistotuotteen koostaminen valmiista ohjelmistokomponenteista onkin tietyssä rajatussa mielessä massaräätälöintiä.

Vaikka ohjelmistokomponenttien systemaattinen hyödyntäminen ohjelmistotuotannossa onkin verrattain uusi käytäntö, ei ajatus ohjelmistokomponenteista ole suinkaan uusi. McIlroy [1968] mainitsi jo vuoden 1968 NATO konferenssissa seuraavasti:

> *"A [software] components industry could be immersly succesful."* [McIlroy, 1968]

Lukuun 7.1 viitaten, jos ohjelmistokomponentti ymmärretään liian löyhästi rajatuksi, saatetaan joutua kirjoittamaan ylimääräistä ohjelmistokoodia liittämään komponentit toisiinsa ja toimimaan komponenttien välissä tiedonvälittäjinä. Vaikka ohjelmistokomponenteille onkin olemassa useita erilaisia määritelmiä, käytetään komponenttipohjaisessa ohjelmistotuotannossa *liitä ja käytä* määritelmää, jossa yksittäisten komponenttien tulisi toimia itsenäisinä toimijoina, jotka kommunikoivat toistensa kanssa muodostaakseen toimivan kokonaisuuden. [Szyperski, 1998]

Ohjelmistokomponentteja komponenttipohjaisessa ohjelmistotuotannossa kuvataan viidellä ominaisuudella. Täyttääkseen ohjelmistokomponentin määritelmän komponenttipohjaisessa ohjelmistotuotannossa tulee ohjelmistokomponentin täyttää nämä seuraavat viisi ominaisuutta [Aoyama, 1998]:

1. Liitä ja käytä: ohjelmistokomponentin tulee olla liitettävissä toisiin ohjelmistokomponentteihin ja/tai ohjelmistokehykseen/-kehyksiin ohjelmiston ajon aikana ilman ohjelmakoodin kääntämisen (engl. *compile*) tarvetta
2. Rajapintakeskeisyys: ohjelmistokomponentin tulee eriyttää rajapinta toteutuksestaan ja kätkeä toteutus siten, että komponentteja voidaan yhdistää pelkän rajapinnan avulla tuntematta toteutusta
3. Arkkitehtuurikeskeisyys: ohjelmistokomponenttien tulee olla suunniteltuja ennalta määritellyn rajapinnan mukaisiksi, jolloin niitä voidaan liittää toisiin ohjelmistokomponentteihin ja/tai ohjelmistokehyksiin
4. Vakiointi: ohjelmistokomponenttien rajapinnan tulee olla vakioitu, jolloin se voidaan antaa kolmansien osapuolien kehitettäväksi ja kolmannet osapuolet voivat myös sitä hyödyntää
5. Kilpailtu jakelu: ohjelmistokomponentteja voidaan jaella kilpailluilla kauppapaikoilla ja niitä voidaan hankkia jakelijoilta

Näiden määritelmien mukaan toteutetut ohjelmistokomponentit täyttävät hyvin massaräätälöinnin perusvaatimuksia. Jos ohjelmistokomponentteja voidaan ostaa valmiina ja niitä voidaan liittää ennalta määriteltyjen rajapintojen mukaan, kuten esimerkiksi auton osia, ja koostaa näin halutunlainen tuote, ollaan hyvin lähellä perinteisen massaräätälöinnin ja tuotantolinjatuotannon lähtökohtia.

Vaikka ajatus ohjelmiston automaattisesta koostamisesta valmiista komponenteista kuulostaakin merkittävästi edullisemmalta tavalta valmistaa ohjelmistotuotteita kuin mitä tähänastiset tuotantotavat tarjoavat, ei ohjelmistotuotanto ole kuitenkaan siirtynyt merkittävissä määrin pelkästään ohjelmistokomponenteista koostamiseen. Yksi merkittävä syy tähän lienee se, että ohjelmistotuotteet pyrkivät kuitenkin täyttämään jonkin tietyn, melko rajatun, tarpeen, jolloin vakioituja ohjelmistokomponentteja jokaiseen tarpeeseen tuskin on vielä olemassa. Toisaalta tarpeena on myös kehittää yksilöllisiä tuotteita, eikä tiettyjä innovaatioita haluta jakaa kilpailijoiden kanssa. Jos kaikki ohjelmistoyritykset koostaisivat tuotteensa yhteisistä ohjelmistokomponenteista, olisi kilpailijasta erottuminen käytännössä mahdotonta. Tilanne olisi melko samankaltainen, kuin jos kaikki autovalmistajat koostaisivat autonsa täysin saman osavalikoiman osista ja kultakin valmistajalta voisi hankkia täysin samanlaisen ajoneuvon. Ajoneuvojen osalta tähän suuntaan on tosin jo menty, monien rakenteellisten osien ollessa samanlaisia yli malli- tai merkkirajojen.

9.4. Ohjelmistoarkkitehtuurit ja massaräätälöinti

Ohjelmistoarkkitehtuurit ovat lähempänä metatasoa, kuin esimerkiksi hyvin käytännönläheiset ohjelmistokomponentit. Ohjelmistoarkkitehtuuri on rakenteellinen suunnitelma tai malli siitä, miten ohjelmisto koostuu.

Bosch [2004] määrittelee IEEE 1471 standardiin viitaten ohjelmistoarkkitehtuurin yleisjärjestykseksi järjestelmälle sisältäen sen komponentit, niiden suhteet toisiinsa ja ympäristöön sekä koko järjestelmän pääpiirteet ohjaten sen suunnittelua ja kehitystä.

Ohjelmistoarkkitehtuuri vastaa myös paremmin käsitettä tietojärjestelmästä kuin pelkästä ohjelmistosta, sillä ohjelmistoarkkitehtuuri määrittelee myös ohjelmiston suhteet ympäröivään maailmaan, kuten rajapinnat tiedonkeruulaitteille ja mahdollisille ulkoisille komponenteille.

Ohjelmistoarkkitehtuuri voi siis pitää sisällään ohjelmistokomponentteja, suunnittelumalleja, sekä ohjelmistokehyksiä. Kuitenkin siinä, missä ohjelmistokomponenteista, suunnittelumalleista ja ohjelmistokehyksistä voidaan löytää suoria yhtymäkohtia tietojärjestelmien massaräätälöintiin, voidaan niitä ohjelmistoarkkitehtuureissa löytää vain välillisesti näiden edellisten kautta.

Ohjelmistoarkkitehtuuri voi myös luonnollisesti olla joko hyvin massaräätälöintiin sopiva tai huonosti massaräätälöintiin sopiva. Hyvälle ohjelmistoarkkitehtuurille ei liene muuta määritelmää tai mittaria kuin se, että se sopii hyvin haluttuun projektiin ja on riittävän kattava ja selkeä. Kuitenkin, ohjelmistoarkkitehtuuri on se kohta, jossa tehdään ratkaisut sen suhteen, miten hyvin tai huonosti kehitettävät ohjelmistokomponentit toteuttavat massaräätälöinnin vaatimuksia.

9.5. Suunnittelumallit ja massaräätälöinti

Suunnittelumallit (engl. *design patterns*) ovat suunnittelua ohjaavia kaavoja hieman samalla tavalla kuin räätälien käyttämät kaavat vaatteiden valmistamisessa.

Suunnittelumallit ovat osa ohjelmistoarkkitehtuuria, mutta koskettavat huomattavasti pienempää osakokonaisuutta kuin koko ohjelmistoarkkitehtuuri. Suunnittelumallit koostuvat usein vain yhdestä muutamaan ohjelmistokomponenttiin. Suunnittelumallit jaottuvat käyttötarkoituksensa mukaan kategorioihin, mutta koska suunnittelumalleille ei ole mitään virallista standardia, niiden määrää on mahdotonta määrittää tarkasti. Käytännössä suunnittelumallit elävät tai kuolevat sen mukaan miten ne yleistyvät ja miten kehittäjät niitä omaksuvat.

Ajatus suunnittelumalleista vertautuu hyvin esimerkiksi talon rakentamiseen. Aikojen saatossa jonkin tietyn kohdan rakentamiseen on löydetty hyvä ratkaisumalli, jota on sittemmin alettu käyttämään usein ja se on vakiintunut *de facto* –tavaksi toteuttaa kyseinen kohta, esimerkkinä ikkuna-aukko. Itse asiassa koko termin suunnittelumalli loi arkkitehti Christopher Alexander kirjassaan *A Pattern Language*, jolla hän halusi tarjota tavallisille ihmisille suunnittelumallikielen, jotta he voisivat suunnitella omia talojaan ja yhteisöjään [Mattsson, 1996].

Suunnittelumalli on siis yleisesti järkeväksi havaittu tapa ratkaista jokin tietty ongelma tai toteuttaa jokin tietty toiminnallisuus.

Monet gammakategorian, eli Gamman *et al.* kirjassaan *Design Patterns: Abstraction and Reuse of Object-Oriented Design* [1993] esittelemät suunnittelumallit, kuten *Proxy, Decorator* ja *Facade* tarjoavat olemassa oleviin ohjelmistokomponentteihin liitännäisiä, jotka mahdollistavat niiden yhteentoimivuuden muiden ohjelmistokomponenttien kanssa ilman, että niitä olisi alunperin suunniteltu yhteentoimiviksi [Mattsson, 1996]. Näin myös irrallisista ohjelmistokomponenteista, jotka eivät täytä komponenttipohjaisen ohjelmistosuunnittelun asettamia vaatimuksia, voidaan saada massaräätälöintiin paremmin sopivia liitä ja käytä -komponentteja.

9.6. Automaattinen ohjelmakoodin generoiminen

Forward engineering tarkoittaa korkean tason mallin tai määritelmän toteuttamista fyysiseksi implementaatioksi. Ohjelmistotuotannosta puhuttaessa tosin fyysinen implementaatio ei luonnollisesti ole kovin fyysinen. Kuitenkin, jos korkean tason malli on tarpeeksi tarkka, siitä voidaan automaattisesti gereroida ohjelmistokoodia tarkoitukseen sopivilla työkaluilla.

UML (Unified Modeling Language) on mallinnuskieli, jolla voidaan märitellä esimerkiksi ohjelmiston koodirakenteen luokkia ja metodeja. Monet UML -mallinnustyökalut tarjoavat mahdollisuuden asettaa näille luokille ja metodeille näkyvyysarvoja sekä asettaa metodeille oletettuja parametreja ja paluuarvoja. Näiden tietojen pohjalta monet mallinnustyökalut pystyvät generoimaan ohjelmakoodia joillakin ohjelmointikielillä.

10. Ohjelmistokehykset ja massaräätälöinti

Ohjelmistokehykset tarjoavat erilaisen näkymän massaräätälöintiin. Siinä missä ohjelmistokomponenteista voitiin koostaa liitä ja käytä –tavalla laajempia kokonaisuuksia, tarjoavat ohjelmistokehykset yleensä valmiina monia usein vaadittuja ominaisuuksia antaen näin kehittäjille hyvän alustan rakentaa ohjelmistotuotettaan.

Ohjelmistokehys on uudelleenkäytettävä, ”puolivalmis” sovellus, jota voidaan edelleen kehittää halutunlaiseksi tuotteeksi [Fayad and Schmidt, 1997].

Suuri osa ohjelmistotuotteiden valmistamiseen käytetystä ajasta ja kustannuksista menee samojen ja useille ohjelmistoille yhteisten ratkaisumallien ja ohjelmistokomponenttien uudelleenkeksimiseen ja uudelleentoteuttamiseen. Etenkin alati heterogenisoituvat laitteistomarkkinat, useat käyttöjärjestelmät ja laitteistoalustat vaikeuttavat laadukkaiden, siirrettävien, tehokkaiden ja edullisten ohjelmistotuotteiden kehitystä. [Frayad and Schmidt, 1997]

Ohjelmistokehyksellä, jossa monet usein vaaditut perusominaisuudet, kuten tietokanta-abstraktiot ja käyttöliittymäkirjastot ovat valmiina, voidaan saada nopeampi ja edullisempi liikkeellelähtö ohjelmistoprojektiin. Lisäksi, kun ohjelmistokehyksen uudelleenkäyttö lisääntyy, on myös todennäköistä, että siinä olevat puutteet ja viat havaitaan ja korjataan. Näin saavutetaan hyötyjä laadussa, kustannuksissa ja projektin läpivientiajassa.

Ohjelmistokehysten voidaan katsoa myös toteuttavan tai osaltaan mahdollistavan, massaräätälöintiä. Jos ohjelmistokehys käsitteellistää tietokannan, voidaan tietojärjestelmä asettaa käyttämään useita vaihtoehtoisia tietokantoja ilman tarvetta muuttaa itse ohjelmakoodia.

Ohjelmistokoodin systemaattinen uudelleenkäyttäminen, modulaarinen, komponenttipohjainen rakenne ja suunnittelumallien laaja käyttö ohjelmistokehyksissä avaavatkin ovat tietojärjestelmien massaräätälöinnille [Greenfield and Short, 2003].

Haittapuolena ohjelmistokehysten käyttämisessä on joissain tapauksissa niiden laajuus. Ohjelmistokehykset tarjoavat usein paljon enemmän ominaisuuksia, kuin kuhunkin projektiin todella tarvittaisiin, jolloin ohjelmistokoodin koko kasvaa turhaan.

Ohjelmistokehysten tietoturvan voidaan katsoa olevan sekä haitta että hyöty. Etenkin verkkosovelluksissa ohjelmistojen tulee voida suojautua monilta pahantahtoisilta hyökkäyksiltä. Avoimen lähdekoodin ohjelmistokehyksien toiminnallisuus on kaikkien tarkistettavissa lähdekooditasolla, joten mahdolliset heikkoudet ovat sekä hyökkääjien, että kehittäjien löydettävissä. Koska avoin lähdekoodi avaa heikkoutensa kaikkien nähtäville, on hyvin todennäköistä, että mahdolliset virheet löydetään ja korjataan nopeasti.

10.1. Ohjelmistokehysesimerkki 1: Codeigniter

Codeigniter on PHP –kielelle EllisLab Inc.:n tarjoama ilmainen ohjelmistokehys verkkosovellusten kehittämiseen. Se tarjoaa monia valmiita kirjastoja esimerkiksi tietokantayhteyksien käsitteellistäminen useille eri tietokantamoottoreille, toteuttaa MVC-mallia (Model-View-Controller), sekä tarjoaa automaattisen suojauksen useille verkkohyökkäyksille, kuten SQL- ja HTML-injektiot sekä XSS (Cross Site Scripting) [Blanco and Upton, 2009].

Koska valtaosa verkkosovelluksista tarvitsee hyvin samankaltaisia ominaisuuksia, kuten tiedon tallentaminen ja hakeminen tietokannasta, evästeistä tai istunnoista, URI -polkujen (Uniform Resource Identifier) käsitteleminen ja luominen sekä monet käyttöliittymäelementit, on valmiin ohjelmistokehyksen käyttäminen verkkosovellusten kehittämisessä usein järkevää.

Koska esimerkiksi tiedon tallentaminen ja hakeminen on käsitteellistetty, kehittäjän kannalta järjestelmän käyttämä tietokantamoottori on tietyin rajoituksin yhdentekevä. Tällöin voidaan järjestelmää massaräätälöidä asentamalla se hyvin pienin muutoksin, käytännössä asetustiedostoa muokkaamalla, useille eri alustoille.

10.2. Ohjelmistokehysesimerkki 2: WordPress

WordPress on suosittu PHP –kielellä toteutettu avoimen lähdekoodin verkkojulkaisujärjestelmä, jota käytetään pääasiassa blogien kirjoittamiseen. WordPressiä käyttäen kehitettyjä blogeja tai sivustoja on sen kehittäjien arvion mukaan yli 56 000 000 kappaletta (lokakuu 2012).

WordPress on hyvä esimerkki tietojärjestelmien massaräätälöinnistä. Yksi määritelmä tietojärjestelmän olemiseen massaräätälöity on se, että käyttäjät, tai kehittäjät käyttäjien tukena, voivat muokata tuotteen toimimaan tietyllä tavalla ja täyttämään tietyt tarpeet [Meyer and Webb, 2005].

Meyer ja Webb [2005] esittävät, että onnistunut tietojärjestelmän massaräätälöinti vaatii modulaarisen, kerroksittaisen ohjelmistoarkkitehtuurin. Samanlaiset periaatteet toistuvat myös CodeIgniter –ohjelmistokehyksessä. WordPress on vain rajatumpi tiettyyn tarkoitukseen ja sitä myöten siihen on toteutettu ne tietyt, tärkeiksi katsotut ominaisuudet huomattavasti pidemmälle, kuin yleiskäyttöisempään CodeIgniteriin.

Watson [2007] listaa kolme pääkohtaa, joiden valossa juuri WordPress on niin hyvä esimerkki tietojärjestelmän massaräätälöinnistä käytännössä. Ensinnäkin blogikirjoittamisesta on tullut todella laaja ilmiö. Miljoonat ihmiset perustavat - ja muokkaavat mieleisikseen – blogeja. Yksityishenkilöiden lisäksi myös yritykset ovat lähteneet mukaan blogien maailmaan tarpeinaan niin markkinointi, kuin yhteisöjen rakentaminenkin.

Toisekseen, koska WordPress on julkaistu avoimen lähdekoodin lisenssillä, tällä käyttäjämassalla on mahdollisuudet ja oikeus muokata ohjelmistoa haluamakseen. Todella monet kehittäjät jakavatkin tekemiään laajennuksia ohjelmistoliitännäisinä verkossa ja parhaassa tapauksessa uusi käyttäjä voi räätälöidä WordPress –asennukseensa uusia toiminnallisuuksia muutamalla napinpainalluksella. Näihin ohjelmistoliitännäisiin kuuluu mm. keskustelualueita, verkkokauppasovelluksia ja kuvagallerioita.

Kolmantena kohtana Watson mainitsee vielä WordPressin viralliset räätälöidyt versiot. Perus WordPress –asennuksen lisäksi tarjolla on myös sosiaalisen median sivustojen alustaksi tarkoitettu WPMU (WordPress Multi-User) sekä WordPress.com sivusto, joka tarjoaa samoja toiminnallisuuksia kuin yksittäisasennettu WordPress, mutta toimii todellisuudessa itsekin WordPressin päällä.

10.3. Ohjelmistokehysesimerkki 3: XNA

XNA (XNA's Not Acronymed) on sen kehittäjän Microsoft Corporationin mukaan ohjelmistokehys, vaikka se ominaisuuksiltaan täyttäisikin paremmin väliohjelmiston kuvauksen. Väliohjelmisto eroaa ohjelmistokehyksestä siinä, että väliohjelmisto ei ole niin selkeästi osa lopullista ohjelmistotuotetta, vaikkakin se tarjoaa monia samoja ominaisuuksia kuin ohjelmistokehykset [Middleware Resource Center, 2012].

XNA on oliopohjainen peliohjelmointiin tarkoitettu ohjelmistokehys, jonka avulla voidaan toteuttaa DirectX ohjelmia .Net-yhteensopivilla kielillä, pääasiassa C#:lla [Miller and Johnson, 2010].

XNA tarjoaa hyvät lähtökohdat pelillisten ohjelmistotuotteiden massaräätälöintiin. XNA:lla kehitettyjä ohjelmia voidaan melko helposti toteuttaa kääntämisvaiheessa Windowsin uusille versioille, Xbox 360 pelikonsoleille sekä Windows Phone 7 puhelimille. Näiden lisäksi kolmansien osapuolten MONO kirjaston tarjoaman CLR:n (Common Language Runtime) kautta XNA ohjelmia voidaan käyttää myös Andoidilla, BSD:llä, iOS:llä, Linuxilla, OS X:llä, Solariksella ja Unixilla sekä PlayStation 3 ja Wii konsoleilla. [Miller and Johnson, 2010].

Sen lisäksi, että XNA ohjelmia voidaan jaella useille alustoille, voidaan sen tarjoamia ohjelmistokirjastoja pitää myös massaräätälöinnin osana. Koska XNA tarjoaa valmiina rajapinnan DirectX –ohjelmoinnille, sprite -grafiikalle, syöttölaitteille, kuten näppäimistölle, hiirelle ja Xbox 360 peliohjaimelle, sekä valmiit metodit monille kolmiulotteisen matriisilaskennan kaavoille, voidaan sen katsoa tarjoavan osittaista massaräätälöintiä pelillisille ohjelmistoille systemaattisen ohjelmistokoodin uudelleenkäytön ja modulaarisen ohjelmistorakenteensa kautta.

XNA voidaan yhdistää myös .NET ympäristön käyttämään WinForms (Windows Forms) graafista API:a (Applications Programming Interface). Visual Studio 2012 IDE (Integrated Development Environment) tarjoaa visuaalisen kehitysympäristön WinForms

–kehitykselle. WinFormsiin kuuluvia ohjelmistokomponentteja voidaan raahata ja pudottaa WinForms –ikkunaan hyödyntäen näin komponenttipohjaista ohjelmistotuotantoa ja systemaattista ohjelmistokoodin uudelleenkäyttöä, joka itsessäänkin täyttää ohjelmistotuotteiden massaräätälöinnin määritelmiä. [Microsoft, 2012]

11. Yhteenveto

Tietojärjestelmien massaräätälöinti on tutkimuskohteena verrattain uusi. Massaräätälöintiä puolestaan voidaan katsoa toteutettaneen jo Kiinan ensimmäisen keisarin Qin Shi Huangdin hautaa vartioivan terrakotta-armeijan valmistuksessa vuoden 221 paikkeilla. Sittemmin teollisuuden kehitysaskeleet, kuten Henry Fordin T-mallin liukuhihnatuotanto ovat tuoneet modulaarisuutta massatuotantoon valmistellen tietä ajatukselle massaräätälöinnistä.

Tietojärjestelmätuotanto on lainannut monia ajatuksia, kuten suunnittelumallit ja (ohjelmisto-)komponentit perinteisestä teollisuudesta. Monet vallalla olevat tietojärjestelmätuotannon periaatteet, kuten ketterät menetelmät, suunnittelumallit, ohjelmistokehykset ja tuotantolinja-ajattelu, ovat tuoneet massaräätälöinnin periaatteita lähemmäs.

Joitain massaräätälöinnin, sellaisenaan, kuin se teollisuudessa on toteutettuna, osia voidaan tuskin koskaan täysin liittää tietojärjestelmätuotantoon sen immateriaalisen luonteen vuoksi. Perinteinen teollisuus pyörii kuitenkin niin voimakkaasti raaka-aineiden, varastoinnin ja fyysisen jakelun ympärillä – asioiden, jotka eivät sellaisenaan ole olennainen osa tietojärjestelmätuotantoa.

Tietojärjestelmätuotannossa, kuten varmasti muussakin tuottavassa teollisuudessa, pyritään jatkuvasti kehittämään prosesseja ja löytämään parempia ratkaisuita laadun, kustannustehokkuuden, muutoksiin reagoimisen ja projektien läpivientiajan parantamiseksi. Monet näistä ratkaisuista omilla alueillaan toteuttavat osia massaräätälöinnin periaatteista. Niitä ei kuitenkaan ole toteutettu juuri massaräätälöintiä silmälläpitäen, vaan ratkaisumallit on suunniteltu vastaamaan tiettyihin olemassa oleviin ongelmakohtiin.

Tietojärjestelmien massaräätälöinti lähtee ohjelmistokehityskäytännöistä. Vuosikymmenten saatossa verrattain nuori tietojärjestelmätuotannon ala on hionut käytäntöjään jatkuvasti toimintaansa tehostaen. Suunnittelumallien systemaattinen hyödyntäminen luo modulaarisempia ohjelmistokomponentteja, joita voidaan uudelleenkäyttää ja yhdistellä. Ohjelmistoarkkitehdit voivat suunnitella mukautettavia ohjelmistotuotteita näistä ohjelmistokomponenteista luoden kustannussäästöjä ja parantaen lopputuotteen laatua käyttämällä hyvin testattuja osia.

Ohjelmistokehykset vievät massaräätälöinnin vielä pidemmälle tarjoten yhtä aikaa monia uudelleenkäytettäviä komponentteja, noudattaen modulaarista rakennetta sekä abstraktoiden tiettyjä vaihtoehtoisia toteutuksia vaativia osuuksia. Avoimen lähdekoodin ohjelmistokehysten jakelu verkossa on luonut kehittäjien yhteisöjä, jotka jakavat tekemiään ohjelmistoliitännäisiä toisille ohjelmistokehyksen käyttäjille.

WordPress alustana on luonut todella helpon tavan räätälöidä ohjelmisto itselleen sopivaksi. Ja sen miljoonien käyttäjien massa on tehnyt ilmiöstä todellista yhteistoiminnallista massaräätälöintiä.

Tietojärjestelmien massaräätälöinti ei kuitenkaan ole täysin ongelmatonta. Räätälöintiä suorittavat niin käyttäjät kuin asiantuntijatkin hämärtäen asiakkaan ja asiantuntijan rooleja. On helppoa kuvitella, miten harkitsematon ohjelmistoliitännäisten asentaminen yrityksen verkkosivujen julkaisujärjestelmään voi avata yrityksen palvelinkoneen hakkereiden hyökkäyksille tai tietovarkauksille.

Myös räätälöityjen tuotevarianttien astronomisen suuret määrät ovat omiaan luomaan massahämmennystä ja haittaamaan asiakkaan päätöksentekokykyä. Tietyn ominaisuuden mukaan ottaminen tai poisjättäminen kasvattaa varianttien määrää eksponentiaalisesti. Herää kysymys siitä, montaako tietojärjestelmään toteutettua ominaisuutta tullaan todella hyödyntämään.

Yksi asia lienee kuitenkin varma. Nopeasti muuttuvassa ohjelmistotuotannon maailmassa tietojärjestelmien massaräätälöinti ei ole vielä saavuttanut lopullista muotoansa. Uusia toimintatapoja ja käytäntöjä tullaan varmasti kehittämään jatkossakin ja monet nyt vallalla olevat parhaat käytännöt saattavat jäädä vain atk-historian kuriositeeteiksi. Kustannustehokkuus, laadun parantaminen ja muutoksiin reagoiminen ovat kuitenkin tärkeitä lääkkeitä tämän päivän nopeasti muuttuvaan pelikenttään, joten tietojärjestelmien massaräätälöinti tulee lisääntymään ja toivottavasti vielä löytämään tarkan muodon.

Viiteluettelo

[Ahoniemi *et al.*, 2007] Lea Ahoniemi, Markus Mertanen, Marko Mäkipää, Matti Sievänen, Petri Suomala ja Mikko Ruohonen, *Massaräätälöinnillä kilpailukykyä*. Teknologiateollisuus Ry, 2007

[Aoyama, 1998] Mikio Aoyama, New age of software development: how component-based software engineering changes the way of software development? *Int. workshop on CBSE, 1998*

[Apple, 2012] Apple Inc.:n verkkosovellus, 2012. Saatavilla osoitteessa *http://store.apple.com/us/configure/MD771LL/A?*, viitattu 12.06.2012

[Arjen tietoyhteiskunnan neuvottelukunta, 2010] Arjen tietoyhteiskunnan neuvottelukunta, *Tuottava ja uudistuva Suomi. Digitaalinen agenda vuosille 2011 – 2020*. Liikenne- ja viestintäministeriö, 2010

[Armour, 2007] Philip G. Armour, Twenty percent - planning to fail on software projects. *Comm. of the ACM,* **Volume 55,** *2007*

[Blanco and Upton, 2009] Jose Argudo Blanco and David Upton, *CodeIgniter 1.7*. Packt Publishing, 2009

[Blender Foundation, 2012] Blender Foundationin verkkosivut, 2012. Saatavilla osoitteessa http://wiki.blender.org, viitattu 2.11.2012

[Bosch, 2004] Jan Bosch, Software architecture: the next step. *Software Architecture, First European Workshop (EWSA),* **Volume 3047** *of LNCS, 2004*

[Bozheva and Gallo, 2005] Teodora Bozheva and Maria Elisa Gallo, Framework of agile patterns. *EuroSPI 2005,* **Volume 3792** *of LNCS, 2005*

[Business Dictionary, 2012] WebFinance Inc.:n verkkosivut, 2012. Saatavilla osoitteessa http://www.businessdictionary.com/definition/information-system.html, viitattu 5.11.2012

[Dell, 2012] Dell Inc.N verkkosovellus, 2012. Saatavilla osoitteessa http://configure.us.dell.com/dellstore/config.aspx?oc=bpcts7&model_id=optiplex-990&c=us&l=en&s=bsd&cs=04, viitattu 20.10.2012

[Da Silveira *et al.*, 2001] Giovani Jose Caetano da Silveira, Flavio S. Fogliatto, Denis Borenstein, Mass customization: Literature review and research directions. *Int. Journal of Production Economics,* **Volume 72**, *2001*

[Eveleens and Verhoef, 2008] J.L. Eveleens and C. Verhoef, The rise and fall of the Chaos report figures. *VU University Amsterdam, Department of Computer Science, December 2008*. Available as http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.143.7918&rep=rep1&type=pdf

[Fayad and Schmidt, 1997] Mohamed E. Fayad and Douglas C. Schmidt, Object-oriented application frameworks. *Comm. of the ACM,* **Volume 40,** *1997*

[Franke and Piller, 2003] Nikolaus Franke and Frank Piller, Key research issues in user interaction with configuration toolkits in a mass customization system. *Int. Journal of Technology Management,* **Volume 26,** *2003*

[Forza and Salvador, 2007] C. Forza and F. Salvador, *Product Information Management for Mass Customization: connecting customer, front-office, and back-office for fast and efficient customization*. Palgrave Macmillan, 2007

[Galbraith and Hale, 2003] James K. Galbraith and Travis Hale, Income distribution and the information technology bubble. *Assoc. of Public Policy Analysis and Management Fall Conference, 2003*

[Gamma *et al.*, 1993] Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides, *Design Patterns: Abstraction and Reuse of Object-Oriented Design*. Lecture Notes in Computer Science, **Volume 707,** 1993

[Gilmore and Pine, 1997] James Gilmore and B. Joseph Pine II, The four faces of mass customization. *Harvard Business Review, 1997*

[Greenfield and Short, 2003] Jack Greenfield and Keith Short, Software factories: assembling applications with patterns, models, frameworks and tools. *Proceedings of OOPSLA, 2003*

[Hernesniemi, 2010] Hannu Hernesniemi (toim.), *Digitaalinen Suomi 2020. Älykäs tie menestykseen.* Teknologiateollisuus Ry, 2010

[Highsmith and Cockburn, 2001] Jim Highsmith and Alistair Cockburn, Agile software development: the business of innovation. *Computer, 2001*

[Huffman and Kahn, 1998] C. Huffman and B. Kahn, Variety for sale: Mass customization or mass confusion? *Journal of Retailing,* **Volume 74**, *1998*

[Järvinen ja Järvinen, 2011] Pertti Järvinen ja Annikki Järvinen, *Tutkimustyön metodeista*. Opinpaja Oy, 2011

[Koskimies ja Mikkonen, 2005] Kai Koskimies ja Tommi Mikkonen, *Ohjelmistoarkkitehtuurit*. Talentum, 2005

[Kotisivukone, 2012] Ideakone Oy:n verkkosivut. Saatavilla osoitteessa http://www.kotisivukone.fi. Viitattu 2.11.2012

[Krueger, 2006a] Charles W. Krueger, New methods in software product line development. *Software Product Line Conference, 2006*

[Krueger, 2006b] Charles W. Krueger, New methods in software product line practice – examining the benefits of next-generation SPL methods. *Comm. of the ACM,* **Volume 49**, *2006*

[Mattsson, 1996] Michael Mattsson, Object-oriented frameworks – a survey of methodological issues, Licentiate thesis, Dept. of Computer Science, Lund University, 1996

[McIlroy, 1968] M. D. McIlroy, Mass-produced software components. *Software Engineering Concepts and Techniques, 1968 NATO Conference on Software Engineering, 1968*

[Meyer and Webb, 2005] Marc H. Meyer and Peter H. Webb, Modular, layered architecture: the necessary foundation for effective mass customization in software. *Int. Journal of Mass Customisation,* **Volume 1,** *2005*

[Middleware Resource Center, 2012] Middleware Resouce Centerin verkkosivut. Saatavilla osoitteessa http://www.middleware.org/whatis.html, viitattu 1.10.2012

[Microsoft, 2012] Microsoft Corporationin verkkosivut. Saatavilla osoitteessa http://msdn.microsoft.com/en-us/library/vstudio/bb386063.aspx, viitattu 2.11.2012

[Miller and Johnson, 2010] Tom Miller and Dean Johnson, *XNA Game Studio 4.0 Programming: Developing for Windows Phone 7 and Xbox 360*. Addison-Wesley, 2010

[Mäkipää *et al.*, 2012] Marko Mäkipää, Pasi Paunu ja Timo Ingalsuo, Design configurator: a tool for order engineering, *5th Int. Conference on Mass Customization and Personalization in Central Europe, 2012*

[Pine, 1993] B. Joseph Pine, *Mass Customization: The New Frontier in Business Competition.* Harvard Business School Press, 1993

[Planeetta Internet Oy, 2012] Planeetta Internet Oy: verkkosovellus, 2012. Saatavilla osoitteessa *https://www.planeetta.net/palvelin/tilauslomake.html*, viitattu 12.06.2012

[Reel, 1999] John S. Reel, Critical success factors in software projects. *IEEE Software, May/June 1999*

[Ruohonen *et al.*, 2006] Ruohonen, M., Riihimaa, J., and Mäkipää, M., Knowledge based mass customisation strategies: cases from Finnish metal and electronics industries. *Int. Journal of Mass Customization, Vol. 1., Nos 2/3, pp. 340-359, 2006*

[Standish Group, 2003] The Standish Group: Chaos Chronicles Version 3.0., 2003

[Szyperski, 1998] Clemens Szyperski, *Components Software: Beyond Object-Oriented Programming*. AMC Press/Addison-Wesley, 1998

[Teresko, 1994] John Teresko, Mass customization or mass confusion. *Industry Week, June 20, 1994*

[Virtanen, 2011] Taneli Virtanen, Tietotekniikan merkitys organisaatiolle – case: TYKS Vakka-Suomen sairaala, Opinnäytetyö (AMK), Turun ammattikorkeakoulu, 2011

[Volvo Car Corp., 2012] Volvo Car Corporationin verkkosovellus, 2012. Saatavilla osoitteessa *http://www.volvocars.com/fi/footer/Pages/VDS.aspx*, viitattu 12.06.2012

[Watson, 2007] Andrew Watson, WordPress: Blogging software as open, modular, multi-level platform. *Proceedings of the MCPC, 2007*