Erkki Mäkinen (toim.)

Pieniä tietojenkäsittely-tieteellisiä tutkimuksia

Syksy 2006

TIETOJENKÄSITTELYTIETEIDEN LAITOS
TAMPEREEN YLIOPISTO

D-2007-1

TAMPERE 2007

TAMPEREEN YLIOPISTO
TIETOJENKÄSITTELYTIETEIDEN LAITOS
JULKAISUSARJA D – VERKKOJULKAISUT
D-2007-1, TAMMIKUU 2007

Erkki Mäkinen (toim.)

Pieniä tietojenkäsittely-tieteellisiä tutkimuksia

Syksy 2006

TIETOJENKÄSITTELYTIETEIDEN LAITOS
33014 TAMPEREEN YLIOPISTO

ISBN 978-951-44-6852-0
ISSN 1795-4274

Sisällysluettelo

Pokeri ja tekoäly

Sauli Alho

Tiivistelmä.
Tässä tutkielmassa tarkastellaan internetissä pelattavaa pokerin muotoa nimeltä Texas Hold'em. Tarkemmin rajattuna Limit-versiota, johon pystytään kehittelemään tekoälyä hyväksi käyttäviä pokeriohjelmia. Tutkielmassa käydään läpi näiden ohjelmien toimintaa ja pelistrategiaa.

Avainsanat ja –sanonnat: Pokeri, tekoäly
CR-luokat: I.2, G.3

1. Johdanto

Tekoälytutkimus on jo niittänyt mainetta pelien parissa muunmuassa shakin ansiosta. Shakissa, kuten monessa muussakin niin sanotussa täydellisen informaation pelissä, on jo todistettu, että tietokone pystyy voittamaan maailman parhaat pelaajat. Tämä johtuu suurelta osin tietokoneiden hyvin suuresta laskentakapasiteetista. Nykypäivänä onkin jo siirrytty uusien haasteiden pariin. Jonkin aikaa on jo tutkittu sitä, miten tekoäly pärjää sellaisissa peleissä, joissa kaikki informaatio ei olekaan saatavilla. Suurin kiinnostuksen aihe on ollut pokeri, jossa tekoälyltä vaaditaan paljon enemmän kuin aiemmin. Pelkkä suuri laskentavoima ei siis enää pelkästään riitä, vaan tekoälyltä vaaditaan uusia ominaisuuksia parhaiden pokerinpelaajien kukistamisessa. Tällä hetkellä pokeriohjelmilla on saatu aikaa hyviäkin tuloksia ihmisiä vastaan, varsinkin kaksinpeleissä, mutta parhaiden kukistamiseen ne eivät kuitenkaan vielä pysty. Pokeri tarjoaakin suuren haasteen erilaisille tekoälysovellutuksille. Tässä kirjoituksessa selvitetään, miten tekoäly toimii pokeriohjelmassa, mitä siltä vaaditaan hyvän pelaajan voittamiseen ja miksi se ei vielä siihen pysty.

2. Limit Texas Hold'em

Käsittelen kirjoituksessani tämän hetken suosituinta muotoa pokeripeleistä eli Texas Hold'emia. Sillä ratkaistaan vuosittain pokerin maailmanmestaruus ja sitä

pelataan ympäri maailmaa kasinoilla sekä internetissä. Hold'emia pidetään monimutkaisimpana pokerin muotona, mutta samalla se on säännöiltään hyvinkin yksinkertainen.

Limit Hold'em eroaa No Limit Hold'emista siinä, että panosten suuruus on ennaltamäärätty. Esimerkiksi 1/2-limitissä ensimmäisellä kahdella kierroksella on panos 1 ja seuraavilla 2. Tästä rakenteesta johtuen Limit Hold'demiin on mahdollista ohjelmoida käsittelemämme tekoälyllä varustettu pokerirobotti. Pokerirobottien ei tiedetä menestyneen No Limit Hold'demissa, koska siinä saa panostaa kuinka paljon vain ja koska tahansa. Tästä syystä esimerkiksi pelkkä matemaattinen lähestymistapa ei ole mahdollista. Lisäksi myös muita muuttujia on enemmän kuin limiittipeleissä.

2.1 Säännöt

Pelin alkaessa kaksi pelaajaa jakajasta vasemmalle laittavat etukäteen pöytään blindit. Blindejä on kaksi, pieni (small blind) ja iso (big blind). Pikkublindin maksaa aina jakajasta seuraava ja hänestä seuraava maksaa ison blindin. Tämän jälkeen kaikille pelaajille jaetaan kaksi omaa ns. pimeää korttia. Tämän jälkeen on ensimmäinen panostuskierros. Panostuksen aloittaa ison blindin vasemmalla puolella oleva pelaaja. Pelaaja voi joko luovuttaa (fold), maksaa (call) tai korottaa (raise). Panostuskierroksen jälkeen jaetaan pöydälle kolme kaikille yhteistä korttia, kuvapuoli ylöspäin (flop). Tämän jälkeen alkaa toinen panostuskierros. Kierroksen aloittaa nyt ja myös seuraavilla kierroksilla aina pelaaja, joka on jakajasta seuraava. Jos hän ei lähtenyt mukaan, niin hänen vasemmalla puolellaan seuraava mukana oleva pelaaja aloittaa.

Seuraavaksi jaetaan pöydälle neljäs yhteinen kortti (turn). Tässä vaiheessa, jos pelataan limit hold'emia, sovitut panokset tuplaantuvat. Panostuskierroksen jälkeen jaetaan vielä kaikille yhteinen viides kortti (river), jonka jälkeen on viimeinen panostuskierros. Viimeisten panostusten jälkeen vielä mukana olevat pelaajat näyttävät korttinsa. Paras viiden kortin pokerikäsi kahdesta omasta ja viidestä yhteisestä kortista muodostettuna voittaa potin. Jos kahdella tai useammalla pelaajalla on samanarvoinen pokerikäsi, korkeimman hain (kicker) omaava voittaa. Mikäli hai ei mahdu viiteen korttiin tai kortit ovat kummallakin pelaajalla samat, potti jaetaan.

Käsien arvojärjestys: 1. Värisuora, 2. Neloset, 3. Täyskäsi, 4. Väri, 5. Suora, 6. Kolmoset, 7. Kaksi paria, 8. Pari, 9. Korkein yksittäinen kortti.

2.2 Strategia limit hold'emissa

Hold'em on monimutkainen peli, jossa eteen tulee koko ajan uusia tilanteita. Näihin tilanteisiin ei voi antaa yksiselitteistä vastausta - tee niin tai näin - vaan yleensä oikea menettely riippuu monesta tekijästä. Selvittääksen millaisia ratkaisuvaihtoehtoja hyvä pokerinpelaaja käy läpi yhden käden aikana, esittelen esimerkkikäden (mukaillen [Billings et al., 2002]). Korttien arvot ovat seuraavat: K = kuningas (king), Q = rouva (Queen), J = jätkä (jack), T = kymppi (ten). Eri maista on käytetty seuraavia merkintöjä: h = hertta (hearts), d = ruutu diamonds), c = risti (club) ja s = pata(spades). Samaa maata olevia kortteja tarkoittaa s = suited ja eri maata olevia kortteja o = offsuited.

Peli on 1€-2€ limit hold'emia, jossa on mukana 10 pelaajaa. Meillä on button eli toimimme viimeisenä. Ensimmäiset kaksi vasemmalta laittavat blindit ja kortit jaetaan. Toiminnan aloittaa blindeistä seuraava. Tämä aikaisesta positiosta pelaava pelaaja maksaa 1€ pottiin. Kolme seuraavaa heittää korttinsa pois, keskipositiossa oleva pelaaja maksaa 1€ ja kaksi seuraavaa heittää korttinsa pois. Meillä on kädessä 8c-7c, joka on suhteellisen hyvä vetokäsi. Sitä ei voi kuitenkaan pelata, mikäli kohtaa korotuksen edeltä olevilta pelaajilta tai joudut pelaamaan vain yhtä pelaajaa vastaan. Vetokädet haluavat siis monta pelaajaa pottiin, sillä osuessaan harvemmin saavat ne näin myös rahaa enemmän pottiin. Aikaisemmista jaoista olemme panneet merkille, että aikaisesta positiosta pelaava on suhteellisen tiukka pelaaja ja voimme näin ollen olettaa, että hänellä on kaksi isoa korttia. Ison parin suljemme pois, sillä hän ei korottanut. Keskipositiosta pelannut pelaaja on taas löysä pelaaja, joka pelaa miltei kaikilla korteilla. Hänellä voi siis olla lähes mitä vain. Maksamme 1€ pottiin, jotta saamme myös blindit mukaan peliin. Pikkublindi, jonka tiedämme suhteellisen tiukaksi pelaajaksi, jaksaa, ja iso blindi chekkaa (check). Floppiin tulee Qd-8s-5c. Meillä on siis keskipari. Jos käsi tällä hetkellä ei ole paras, niin sillä on viisi suoraa korttia pakassa (outteja), joilla siitä mahdollisesti tulee paras. Nämä kortit ovat: 8d, 8h, 7s, 7d, 7h. Meillä on myös mahdollinen kahden peräkkäisen kortin tuoma värin tai suoran mahdollisuus. Flopin ollessa ns. sateenkaari (rainbow-flop), sillä ei ole suoraa värinvetomahdollisuutta, eikä myöskään tässä tapauksessa kovin hyviä suoranvetomahdollisuuksia. Näin ollen voidaan olettaa, että pelaajat, jotka jatkavat peliä, ovat jo saaneet osuman omiin kortteihinsa, vetokäsien sijaan. Peli jatkuu molempien blindien chekatessa ja aikaisessa positiossa oleva pelaaja lyö pottiin 1€. Keskipositiossa oleva pelaaja luovuttaa. Tiedämme entuudestaan, että aikaisemman position pelaaja harvoin bluffaa (bluff) ja ottaen huomioon hänen mak-

sunsa positiosta, niin suurella todennäköisyyydellä voimme olettaa, että hänellä on rouva kädessä. Potissa on siis 6€ ja jos maksamme, niin saamme pottikertoimeksi 6:1. Kun oletamme, että blindit todennäköisesti luovuttavat, niin maksu on kertoimien mukaan niukasti kannattava. Voitamme jaon noin kerran neljästä, jos vastassa on rouva ja toinen iso kortti. Meidän pitää myös harkita korottamista, jolla voisimme mahdollisesti ajaa blindit pois osumasta samoihin kortteihin kuin me. Toinen syy korottaa näissä tilanteissa on hankkia ns. ilmainen kortti (free-card) turnilla. Korottaessamme aikaisemman position pelaaja maksaa ja sen jälkeen chekkaa potin meille, jonka mekin chekkaamme. Näin näemme turnin ilmaiseksi. Vastassamme oleva pelaaja voi kuitenkin heti korottaa meidän korotukseemme, joka olisi meidän kannaltamme huono, sillä menetämme kaksi ylimääräistä panosta. Hän voisi myös vain maksaa korotuksen, mutta turnilla kuitenkin heti panostaa uudelleen, joka maksaisi meille yhden ylimääräisen panoksen. Päättämme kuitenkin vain maksaa 1€. Blindit luovuttavat, joten olemme kahdestaan enää jäljellä. Seuraava kortti on 6s, joka antaa meille suoranvedon kumpaankin päähän. Vastustajamme panostaa 2€. Saamme siis mahdollisuuden voittaa 2€ panoksella 9€. Olemme vain noin 2:1 altavastaajia, joten maksu on hyvin kannattava. Tässä tapauksessa harkitsemme myös taas korottamista. Korottamalla vastustaja joutuu miettimään, kuinka vahva käsi meillä on. Pöydässä on 8,5 ja 6 joten meillä voisi olla jo suora, tai ainakin kaksi paria, koska monet pelaajat pelaavat esim. 5-6 tyyppisiä käsiä. Olisimme myös voineet flopata kolmoset ja odottaa niillä korotusta turnille, joka on hyvin yleinen tapa, joskaan ei välttämättä oikea. Mikäli vastustaja maksaa, hän todennäköisesti sököttää turnilla, jolloin näemme kortit samalla hinnalla kuin vain maksaisimme turnin ja riverin, mutta ansaitsisimme yhden ison panoksen, jos vetomme onnistuu. Päätämme korottaa ja vastustaja maksaa. Riverillä tulee pöytään 6h. Vastustajamme sököttää. Nyt voisimme taas harkita bluffaamista toiseksi parhaalla kädellä, kuten kuvittelemme sen olevan. Tiedämme kuitenkin entuudestaan vastustajan yleensä katsovan loppuun asti, joten bluffaaminen tuskin on kannattavaa. Me myös sökötämme ja vastustaja näyttää Qc-Tc, jolla hän voittaa.

3. Hyvän pelaajan vaatimuksia

Pelaajalta vaaditaan monia erilaisia taitoja, jotta pystyy pelaamaan huipputasolla ja huippupelaajia vastaan. Tietenkin myös pokeriohjelmasta pitää löytyä nämä samat ominaisuudet.

Billings ja muut [1998] listaavat näistä muutamia tärkeimpiä, joita ohjelma käyttää hyväkseen pelatessaan pokeria: käden vahvuus, käden mahdollisuudet, bluffaaminen, pelaajan arvaamattomuus, panostusstrategia ja vastustajan luokittelu.

Käden vahvuudella määritellään oman käden paremmuus muihin käsiin verrattuna. Yksinkertaisimmillaan se laskee omien korttien ja pöydässä sillä hetkellä olevien korttien vahvuutta. Parempi arviointi ottaa myös mukaan pelaajien määrän pelissä, pelaajan paikan pelissä ja myös todennäköisyydet vastustajien mahdollisille käsille.

Käden mahdollisuuksia laskettaessa kone laskee todennäköisyyden sille, että käsi voittaa tai sitten johdossa oleva käsi häviää, kun kortteja tulee pöytään. Oletetaan esimerkiksi, että kädessä on neljä samaa maata olevaa korttia. Silloin kädellä on sillä hetkellä huono arvo, mutta on hyvä mahdollisuus voittaa värillä, kun kortteja tulee vielä pöytään. Toisaalta taas korkean parin voiton todennäköisyys pienenee, kun pöytään tulee monia erilaisia vetokäsiä suosivia kortteja.

Panostusstrategia määrittää tilanteet, kun pitää luovuttaa, maksaa tai korottaa. Tähän vaikuttavat monet eri asiat: käden vahvuus, pottikertoimet, vastustajat, oma paikka pöydässä, pöytäimago, oma tyyli pelata ym.

Bluffaamisella tarkoitetaan jaon voittamista heikolla kädellä. Bluffaaminen on tärkeä tekijä voitollisessa pelaamisessa. Se hämää vastustajaa ja antaa mahdollisuuden voittaa enemmän myös hyvillä käsillä. Tekoäly pystytään ohjelmoimaan niin, että se bluffaa teoreettisesti oikeissa kohdissa ja tietyin väliajoin. Käytännössä pitää ottaa myös huomioon muut tekijät, kuten käden mahdollisuudet parantua ja vastustajan todennäköisyys luovuttaa.

Ennalta-arvaamattomuus tekee vastustajalle vaikeaksi mallintaa pelaajan strategiaa. Pelin sekoittaminen, eli toimiminen eri tavalla samantyylisissä tilanteissa, aiheuttaa vastustajalle vaikeuksia lukea peliä ja mahdollistaa täten hänet tekemään virheitä perustuen virheellisiin olettamuksiin käden vahvuudesta.

Vastustajan luokittelulla tarkoitetaan yksinkertaisimmillaan vastustajien luokittelemista erilaisiin kategorioihin, kuten esimerkiksi löysä-passiivinen tai

tiukka-agressiivinen. Se auttaa pelaamaan tiettyä pelaajaa vastaan tietyllä strategialla. [Billings et al., 1998]

4. Pokeriohjelman toiminta

Samalla tavalla kuin ihmisellä on kaksi eri strategiaa pelata pokeria, niin on myös pokeriohjelmilla. Hold'emin pelaaminen voidaan jakaa kahteen eri alueseen: Pre-Flop -peliin eli kun nähdään omat pohjakortit ja Post-Flop -peliin, jolloin nähdään yhteiset kortit. Jaottelu johtuu osin siitä, että ennen kuin pelaaja näkee flopin, hänellä on vähän tietoa päätöksentekoon. Flopin jälkeen on huomattavasti enemmän muuttujia, jotka vaikuttavat pelaajan päätökseen luovuttaa, maksaa tai korottaa. Tarkastelussa on suurelta osin Albertan yliopistossa kehitetty Poki-ohjelma. Se edustaa kehityksen kärkeä pokerin tekoälysovellutuksissa maailmassa.

4.1 Strategia ennen floppia

Eräs osa-alue Hold'emissa, jossa voidaan antaa selkeitä toimintaohjeita, on kahden ensimmäisen kortin pelaaminen. Tämä johtuu siitä, että monet korttiyhdistelmät eivät ole kovin hyviä. Sklansky ja Malmuth [2004] ovat järjestäneet parhaat lähtökädet kahdeksaan eri ryhmään vahvuusjärjestyksessä, jota lähes orjallisesti seuraamalla aloittelevakin pokerinpelaaja saa hyvät lähtökohdat voitolliseen pelaamiseen.

Poki-ohjelman tekijät ovat käyttäneet hyvin yksinkertaista tekniikkaa selvittääkseen aloituskäsien paremmuuden. He ovat pelanneet useita miljoonia käsiä, joissa kaikki pelaajat maksavat ensimmäisen panoksen ja sen jälkeen kaikki jäljellä olevat kortit jaetaan auki ilman lisäpanostuksia. Näin he ovat saaneet jokaiselle aloituskädelle eräänlaisen tuloarvon. Taulukossa 1 on esitetty nämä tuloarvot Sklanskyn ryhmissä olevien käsien edessä. Mitä suurempi arvo kädellä on, sitä enemmän se on voittanut jakoja muita käsiä vastaan. Isojen parien arvo todellisuudessa olisi vieläkin suurempi kuin tulokset antavat näyttää. Tämä johtuu siitä, että monesti esim. 10 hengen peleissä yleensä enemmän kuin puolet pelaajista heittävät kortit pois ennen floppia, jolloin isot parit voittavat useammin pienempää määrää pelaajia vastaan. Kuten Billings ja muut [2002] toteavatkin, niin tämä tekniikka ei anna välttämättä oikeaa arviota käden odotusarvosta (expected value). Pokin tekijät ovat myös verranneet omia testituloksiaan näihin

kahdeksaan ryhmään ja huomanneet vahvan korrelaation niiden välillä [Billings et al., 2002].

Group 1	Group 2	Group 3	Group 4
+2112 AA	+714 TT	+553 99	+481 T9s
+1615 KK	+915 AQs	+657 JTs	+515 KQo
+1224 QQ	+813 AJs	+720 QJs	+450 88
+935 JJ	+858 KQs	+767 KJs	+655 QTs
+1071 AKs	+718 AKo	+736 ATs	+338 98s
		+555 AQo	+449 J9s
			+430 AJo
			+694 KTs

Group 5	Group 6	Group7	Group 8
+364 77	+304 66*	+214 44	-75 87o
+270 87s	+335 ATo	+92 J9o	+87 53s
+452 Q9s	+238 55	+41 43s	+119 A9o
+353 T8s	+185 86s	+141 75s	+65 Q9o
+391 KJo	+306 KTo	+127 T9o	-129 76o
+359 QJo	+287 QTo	+199 33	-42 42s
+305 JTo	+167 54s	-15 98o	-83 32s
+222 76s	+485 K9s	+106 64s	+144 96s
+245 97s	+327 J8s	+196 22	+85 85s
+538 A9s		+356 K8s	-51 J8o
+469 A8s		+309 K6s	-158 65o
+386 A6s		+245 K5s	-181 54o
+448 A5s		+227 K4s	+41 74s
+422 A4s		+211 K3s	+85 K9o
+392 A3s		+192 K2s	-10 T8o
+356 A2s		+317 Q8s	
+191 65s			

Taulukko 1. Tuloarvojen ja Sklanskyn ryhmien erot [Billings et al., 2002].

Billings ja muut [2002] kritisoivat Sklanskyn ryhmäjakoa. Heidän mielestään siinä on pieniä loogisia virheitä. Esimerkiksi 43s on ryhmässä 7, edellä 53s, joka on ryhmässä 8. Artikkelin kirjoittajien mielestä voidaan osoittaa, että 53s dominoi 43s:a isomman korttinsa ansiosta, koska molemmilla käsillä on sama värin ja suoran mahdollisuudet. Samalla tavalla 52s dominoi 42s:a ja 32s:a, mutta silti 52s ei ole rankattu mihinkään ryhmään, vaikka kaksi viimeistä ovat ryhmässä 8.

On tosiaan totta, että 53s voittaa 43% jaoista ja 43s voittaa 26% jaoista, joissa ne ovat kaksin vastassa. Tästä huolimatta kolmasosa jaosta menisi tasan, joten mistään suuresta dominoinnista ei ole kyse. Verrattuna esim. KJs ja QJs,jossa ero on KJs:n hyväksi 70%-28%. Erot johtuvat siitä, että isommat kädet ovat vastakkain hyvin paljon useammin showdownissa kuin pienet. Hyvin harvoin edes tapahtuu sellaista tilannetta, joissa nämä pienet kädet olisivat kaksin showdownissa vastakkain.

Toisessa tilanteessa, jossa 52s dominoi, 42s ja 32s on sama tilanne. Taulukosta 1 nähdään, että molemmat jälkimmäiset kädet ovat tuloarvoltaan miinuksella, joten vaikka 52s dominoi näitä käsiä, niin silti voidaan kyseenalaistaa sen arvokkuus tietokoneen laskelmissa, saati sitten yleensä sen käden pelaamisessa. Itse en näistä käsistä pelaa käytännössä mitään, poikkeuksena iso blindi ja korottamaton potti. Silloin ei joudu myöskään pohtimaan, onko jotain toista kättä edellä, koska yleensä aina on ennen floppia jäljessä.

4.2 Strategia flopin jälkeen

Aloituskorttien pelaaminen on vielä suhteellisen mekaanista toimintaa. Flopin jälkeen ei sitten enää olekaan niin helppoa määrittää sitä, mikä käden arvo on suhteessa muihin. Ässäpari on saattanut muuttua ylivoimaisesta suosikista, kovasti tuuria vaativaksi altavastaajaksi. Hyvinkin kyseenalainen aloituskäsi taas saattaa muuttua flopissa suureksi suosikiksi voittamaan potin.

Pyysingin ja Erolan [2005] mukaan tärkeimpiä huomioon otettavia asioita flopin tullessa pöytään ovat:

- Oman käden vahvuus: Tuliko saletit (nuts), vetokortit vai ei osunut ollenkaan?
- Flopin tyyppi: Floppi J42 antaa AJ:lle kärkiparin (top pair), joka on yleensä johdossa. Mikäli floppiin tulee JhTh8h niin saatat olla pulassa, koska pöydässä on värin ja suoran vetäjiä tai jo valmiita sellaisia.

- Vastustajien lukumäärä: Vastustajien lukumäärä vaikuttaa merkittävästi päätöksiin. Jos on korottanut ennen floppia ja floppi ei osu, voi yhden pelaajan saada luopumaan potista agressiivisesti pelaamalla, mutta neljää pelaajaa vastaan se on käytännössä mahdotonta.
- Vastustajien kädet: Aina pitää miettiä, mitä vastustajilla on kädessään.
- Potin ja panostamisen suhde: Pottikertoimien (pot odds) arviointi on tärkeää oikeiden ratkaisujen tekemisessä.

Poki-ohjelma käyttää flopin jälkeisen strategian määrittämiseen kolmea eri askelta, joissa nämä edellä mainitut asiat tulevat esiin. Ensin lasketaan käden arvon suhteessa muiden pelaajien kortteihin. Tämän arvon mukaan tehdään valinta luovutuksen, maksun tai korotuksen suhteen. Kolmantena vaikuttavana asiana ohjelma päivittää jokaisen panostuskierroksen jälkeen vastustajan mahdollisten käsien todennäköisyyksiä ja tekee myös sen mukaan päätöksiä omasta pelaamisestaan. Esimerkiksi AK käden olemassaolo on todennäköisempää kuin 72, joka yleensä kipataan ennen floppia. [Billings et al., 2002]

Käden vahvuuden laskemiseen on suhteellisen yksinkertainen algoritmi, jolla voi laskea oman käden vahvuuden toista satunnaista kättä vastaan, kun floppi on pöydässä. Algoritmi 1 on Pokin tekijöiden versio käden vahvuuden määrittelemisestä.. Netistä on saatavana myös monia erilaisia ilmaisia versioita, joita voi käyttää apuna samalla kun pelaa, mikäli vain aikaa riittää.

```
HandStrength(ourcards,boardcards)
{
ahead = tied = behind = 0
ourrank = Rank(ourcards,boardcards)
for each case(oppcards)
{
opprank = Rank(oppcards,boardcards)
if(ourrank > opprank) ahead += 1
else if(ourrank == opprank) tied += 1
else  behind += 1
}
handstrength = (ahead + tied/2) / (ahead + tied + behind)
return(handstrength)
}
```

Algoritmi 1. Käden vahvuuden laskeminen [Billings et al., 2002].

Algoritmi 1 kertoo siis, mikä on todennäköisyys sille, että käsi on sillä hetkellä toista satunnaista kättä parempi. Se ei siis vielä kerro, mikä on todennäköisyys sille, että käsi myös voittaa jaon.

Ei siis riitä, että tiedetään, mikä käden tämän hetkinen arvo on, vaan pitää myös saada arvio siitä, mikä se, on kun kaikki kortit ovat tulleet pöytään. Poki laskee positiivisen arvon kädelle eli todennäköisyyden sille, että käsi joka ei vielä paras, tulee voittamaan. Vastaavasti se laskee myös negatiivisen arvon eli todennäköisyyden sille, että käsi, joka nyt on johdossa, ei ole sitä enää lopussa. Algoritmissa 2 on esitelty Pokin käden mahdollisuuksien lasku. [Billings et al., 2002]

```
HandPotential(ourcards,boardcards)
{ /* Hand potential array, each index represents ahead, tied,
     and behind. */
integer array HP[3][3] /* initialize to 0 */
integer array HPTotal[3] /* initialize to 0 */
ourrank = Rank(ourcards,boardcards)
/* Consider all two card combinations of the remaining cards for the opponent.*/
for each case(oppcards)
{
opprank = Rank(oppcards,boardcards)
if(ourrank > opprank) index = ahead
else if(ourrank = opprank) index = tied
else /* < */ index = behind
HPTotal[index] += 1
/* All possible board cards to come. */
for each case(turn)
{
for each case(river)
{ /* Final 5-card board */
board = [boardcards,turn,river]
ourbest = Rank(ourcards,board)
oppbest = Rank(oppcards,board)
if(ourbest > oppbest) HP[index][ahead ] += 1
else if(ourbest == oppbest) HP[index][tied ] += 1
else /* < */ HP[index][behind] += 1 } } }
/* PPot: were behind but moved ahead. */
PPot = (HP[behind][ahead] + HP[behind][tied]/2 + HP[tied][ahead]/2)
/ (HPTotal[behind] + HPTotal[tied]/2)
/* NPot: were ahead but fell behind. */
NPot = (HP[ahead][behind] + HP[tied][behind]/2 + HP[ahead][tied]/2)
/ (HPTotal[ahead] + HPTotal[tied]/2)
return(PPot,NPot)
 }
```

Algoritmi 2. Käden mahdollisuuksien laskenta [Billings et al., 2002].

Lausekkeesta 1 voidaan laskea käden todellinen arvo, joka tarkoittaa todennäköisyyttä, jolla käsi voittaa jaon myös showdownissa. Oletetaan, että kädessä on Ad-Qc ja floppi on Jh-4c-3h. Tällöin Algoritmi 1 mukaan Ad-Q käsi on johdossa flopilla satunnaista kättä vastaan 58,5% todennäköisyydellä. [Billings et al., 2002] Ad-Qc voittaa kyseisen jaon yhtä pelaajaa vastaan 50.55% todennäköisyydellä joka kuvastaa siis käden todellista arvoa.

Pr(voittaa jaon) = Pr(edellä)×Pr(vastustaja ei paranna)
+ Pr(itse perässä)×Pr(me parannamme)
= HS ×(1 − NPot) +(1 − HS)×PPot.

Lauseke 1. Käden todellisen arvon laskeminen [Billings et al., 2002].

Käden vahvuus, mahdollisuudet ja todellinen arvo ovat yksinkertaisia algoritmeja, joilla saadaan kerättyä tietoa todennäköisyyksistä, jotka toimivat perustana oikeille pelillisille päätöksille [Billings et al., 2002]. Näitä matemaattisia todennäköisyyksiä hyväksi käyttäen Poki tekee päätöksiä siitä, kannattaako kädellä lähteä maksamaan, korottamaan vai luovuttamaan.

Mikään strategia ei ole täydellinen ilman vastustajan mallintamista ja jokaisen hyvän pelaajan pitää kehittää tietty malli jokaisesta kanssapelaajastaan, jotta voi löytää heidän heikkoutensa. [Billings et al., 2002]. Internetin pelihuoneissa on mahdollista kirjoittaa pelaajien kohdalle muistiinpanoja siitä, miten he ovat pelanneet tai minkä tyyppisiä pelaajia he ovat. Tyypillisiä muistiinpanoja ovat esim. löysä pelaaja, tiukka pelaaja, agressiivinen, passiivinen, korottaa vedolla ym. Näillä muistiinpanoilla pystyy tekemään jonkinlaista analyysiä pelaajista. Jotkut tuntemani pelaajat sanovat, ettei heidän tarvitse tehdä muistiinpanoja, vaan pystyvät erottamaan hetkessä tietyntyyppiset pelaajat. Taitavien pelaajien on helppo havaita useimpien pelaajien pelityyli. Tästä johtuen hyvät pelaajat myös vaihtavat välillä omaa pelityyliään, jottei se olisi koko ajan helposti luettavissa. Ihmisten on helpompi mukautua, tehdä muutoksia ja muuttaa analyysiään pelaajasta pelin aikana. Pokeriohjelmalle se on vaikeampaa.

Vastustajan mallintaminen on pokerissa vaikein tehtävä ohjelmalle. Sitä voidaan yrittää tehdä päättelemällä käden vahvuutta vastustajan sen astisen toi-

minnan perusteella ja olettaa että, vastustaja pelaa suhteellisen oikein. Toinen tapa on olettaa, että vastustaja toimii kuten ennenkin vastaavassa tilanteessa. [Billings et al., 2002] Tietokone voi kerätä periaatteessa vaikka kuinka paljon tietoa jokaisesta pelaajasta, mutta jos tämä pelaaja pelaa seuraavana päivänä täysin erilaisesti, kerätystä informaatiosta ja kyseessä olevan pelaajan mallinnuksesta ei ole mitään hyötyä. Ohjelman on sitä vaikeampi mukautua, mitä enemmän tietoa yhdestä pelaajasta tulee, ja sen on vaikeampaa pysyä sellaisen hyvä pelaajan pelissä, joka tietyin väliajoin muuttaa pelityyliään. [Billings et al., 2002]

5. Ihmisen ja koneen väliset erot pokerin pelaamisessa

Matematiikka on vain yksi osa-alue pokerista ja vaikkakin se on tärkeä, niin huomattavasti tärkeämpää on ymmärtää ja osata käyttää hyväkseen pokerissa valitsevia lainalaisuuksia. [Sklansky, 2005].

Tässä lauseessa kiteytyy mielestäni parhaiten tämän hetkinen ero ihmisen hyväksi tekoälyä vastaan. Käytännössä jokainen peruskoulusta päässyt osaa tehdä matemaattisesti oikeat ratkaisut pelin aikana. Paljon vaikeampaa on muuttaa peliään vastustajien mukaan ja tehdä psykologisesti oikeita ratkaisuja. Tietokone pystyy mallintamaan vastustajan peliä perustuen pelaajan aiemmin tekemiin ratkaisuihin. Tämä vaatii kuitenkin paljon erilaista dataa ja paljon aikaa. Pokeriohjelma pystyy kaivamaan vaikka 10 000 kättä sitten pelatun jaon ja näkemään siitä, miten tietty pelaaja on pelannut kyseisen jaon. Tämä on tietysti suuri etu, mutta mielestäni ihminen pystyy silti mallintamaan paremmin toisen pelaajan käyttämän pelityylin. Ihmisen on huomattavasti helpompi ja nopeampi muuttaa pelityyliään sen mukaan, miten tietty vastustaja pelaa. Hyvä pelaaja saattaa jo yhden käden perusteella pystyä tekemään vastustajastaan riittävän analyysin. Kohtuullisen hyvän pelaajankin on esim. helppo huomata, jos joku pelaaja syystä tai toisesta alkaa pelaamaan huonosti. Pokeriohjelma taas ei pysty millään mukautumaan siihen ajoissa, koska sillä on toisenlaista tietoa pelaajan historiasta ja se tekee edelleen päätöksiä sen mukaan.

Psykologia on tärkeä osa-alue pokerissa [Sklansky, 2005]. Vaikkakin sitä pystyy hyödyntämään vähemmän Limit Hold'emissa kuin No Limit Hold'emissa, niin silti sillä on suuri vaikutus voitollisessa pelaamisessa. Keinoja ovat esimerkiksi oman pelin muuttaminen ja vastustajan hämääminen pelaamalla toisinaan

tahallaan väärin tai eri tavoilla. Nämä kaikki vaikuttavat siihen, miten vastustaja ajattelee sinun pelaavan tulevaisuudessa samat tilanteet. Saat esimerkiksi kolmoset floppiin taskupareillasi ja slouvaat (slowplay) sen chekkaamalla flopissa. Saatat myös check-raiseta turnissa myöhemmin. Jos vastustajasi maksaa loppuun asti, hän näkee miten olet pelannut kolmoset. Seuraavan kerran kun saat kolmoset uudestaan flopista, niin tällä kertaa lyömällä pottiin vastustajasi ei välttämättä osaa lukea sinua kolmosiin. Pitää siis myös ajatella pidemmälle kuin vain sen hetkisen käden pelaamiseen.

Pokerissa pitää pystyä myös ajattelemaan sitä mitä vastustajalla on ja mitä hän luulee sinulla olevan, eikä keskittyä vain omaan käteensä. Mitä korkeamalle tasolle pokerissa nouset ja mitä parempia vastustajia kohtaat, sitä hankalammaksi heidän käsiensä lukeminen ja heidän pelityylinsä mallintaminen tietysti käy.

Tekoälyllä varustettu pokeriohjelma on ihmistä parempi melkein kaikilla muilla osa-alueilla. Se tekee aina matemaattisesti oikeat ratkaisut lyödessään, korottaessaan tai luovuttaessaan. Se on myös kärsivällinen, mikä on mielestäni tietokoneen suurin vahvuus ihmiseen nähden. Tietokone jaksaa odottaa hyviä käsiä ja hyviä tilanteita koko ajan. Tavallisen ihmisen odotettua pari kierrosta ilman mitään hyviä aloituskortteja, alkaa K-3 näyttämään yllättävän hyvältä.

Toinen tärkeä asia on tunteet. Hyvänkin pelaajan päässä saattaa alkaa kuohumaan, kun joku vastustajista voittaa täysin uskomattomasti, kaikkia todennäköisyyksiä vastaan ja vielä huonosti pelattuna. He saattavat alkaa pelaamaan löysemmin ja alkavat tekemään huonoja päätöksiä - tällaista olotilaa nimitetään tiltiksi. Tietokone taas pelaa koko ajan tasaisesti ja hyvin huolimatta mahdollisista takaiskuista.

6. Yhteenveto

Monissa tietokonepeleissä käytetään nykyään jollain tasolla toimivaa tekoälyä. Joissakin se toimii hyvin ja joissakin taas huomattavasti huonommin. Tekoäly kuitenkin tulee kehittymään koko ajan. Tämä tapahtuu myöskin vääjämättä pokeriohjelmissa, jotka tarjoavatkin mielenkiintoisen alustan erilaisille tekoälysovellutuksille. Tekoäly pystyy luomaan pokeriohjelmalle hyvän matemaattisen strategian pelata vahvaa pokeria heikompia vastustajia vastaan. Sen etuna on myös kärsivällisyys ja tasainen suorittaminen koko ajan, johon ihminen ei pysty.

Kuitenkin mitä korkeammalle tasolle noustaan pokerissa, sitä tärkeämmäksi nousee vastustajan tunteminen. Vaikka tekoäly pystyy mallintamaan jollain tavalla vastustajaa tämän aikaisempien tekojen perusteella, ei se pysy sellaisten hyvien pelaajien perässä jotka osaavat aika ajoin vaihtaa tyyliään. Tämä on suurin syy siihen, miksi vahvinkaan pokeriohjelma ei vielä päihitä parhaita pelaajia. Itse toivon ja uskon, että pelaajien luovuus päihittää vielä kauan pokeriohjelmat.

Viiteluettelo

[Billings et al., 1998] Darse Billings, Dennis Papp, Jonathan Schaeffer and Duane Szafron, Poker as a Testbed for AI Research. *Advances in Artificial Intelligence*, Springer-Verlag, 1998, 228 – 238.

[Billings et al., 2002] Darse Billings, Aaron Davidson, Jonathan Schaeffer and Duane Szafron, The challenge of poker. *Artificial Intelligence* **134** (1-2), (Jan. 2002), 201-240.

[Pyysing&Erola, 2005] Aki Pyysing & Marko Erola, *Pokerin käsikirja*, Like, 2005.

[Slansky and Malmuth, 2004] David Sklansky and Mason Malmuth, *Hold`em Poker for Advanced Players,* third edition, Two Plus Two Publishing, 2004.

[Sklansky, 2005] David Sklansky, *The Theory of Poker*, fourth edition Two Plus Two Publishing, 2005.

Liite Pokerisanasto

Aggressiivinen pelaaja Pelaajaa sanotaan aggressiiviseksi, jos hän korottaa paljon.

Blind Alkupanos. Texas Hold'emissa on kaksi blindiä, iso ja pieni.

Bluffata Hämätä vastustajaa uskottelemalla vahvaa kättä ja saada tämä luovuttamaan.

Button Osoittaa jakajan paikan.

Check Tarkoittaa samaa kuin sököttäminen eli ei panosta omalla vuorolla.

Check-raise Pelaaja ensin sököttää ja korottaa sen jälkeen mikäli joku pelaaja on panostanut.

Flop Kolme ensimmäistä yhteistä korttia.

Foldata Luovuttaa.

Free-card Ilmainen kortti, jonka maksamisesta ei tarvi maksaa mitään. Käytännössä, jos korottaa flopilla ilmaisen kortin toivossa turnilla, maksaa puolikkaan ison panoksen.
Kicker Kun kahdella pelaajalla on sama pari niin paremmuuden ratkaisee korkein pariton kortti.
Kipata foldata, luovuttaa.
Limit Tietyt ennaltamäärätyt rajat panostamiselle.
Löysä pelaaja Pelaaja, joka pelaa useimmat alkukorttinsa.
No-Limit Pelitapa, jossa voi korottaa ja panostaa kuinka paljon haluaa, missä kohtaan peliä tahansa
Nuts Saletit, sen hetkinen paras käsi.
Offsuited Eri maata olevat kortit.
Passiivinen pelaaja Korottaa harvoin.
Pottikerroin Panoksen suhde potissa olevaan rahaan nähden. Jos todennäköisyys saada voittokäsi on suurempi kuin panoksesi suhde pottiin,sinun kannattaa pelata.
River Viides ja viimeinen yhteinen kortti. Kutsutaan myös nimellä fifth street.
Showdown Korttien paljastus lopussa.
Slowplay Slouvata eli pelata hyvät kortit esittäen heikkoa kättä.
Suited Samaa maata olevat kortit.
Sököttää Chekata, ei panosta omalla vuorolla.
Tilt Pelaajan sanotaan olevan tiltissä, kun hän hermonsa menettäneenä alkaa pelata paljon aikaisempaa huonommin.
Tiukka pelaaja Pelaaja joka valikoi tarkasti aloituskorttinsa.
Top-pair Kärkipari, jonka muodostaa pari omasta kortista ja flopin korkeimmasta kortista.
Turn Neljäs yhteinen kortti. Kutsutaan myös nimellä fourth street.
Vetopeli Pelaaja tarvitsee tulevia kortteja saadakseen pelinsä täyteen.

Troijan hevoset

Heini Anttila

Tiivistelmä

Tämä tutkielma tarkastelee Troijan hevosten, eli troijalaisten tietokoneohjelmien, toimintaa, vaikutuksia ja leviämistä. Työssä mietitään myös, miten havaita troijalaisia ja miten suojautua niitä vastaan.

Avainsanat ja -sanonnat: Troijan hevonen, troijalainen, takaovi, matkapuhelin, kyberterrorismi.
CR-luokat: K.6.5, D.4.6

1. Johdanto

Troijalainen, toiselta nimeltään Troijan hevonen, on saanut nimensä kuuluisasta antiikin puuhevosesta, jonka mahassa kreikkalaiset pääsivät piirittämäänsä kaupunkiin. Tietokoneessa troijalainen toimii samalla periaatteella. Ulospäin se näyttää hyödylliseltä tai viattomalta ohjelmalta, esimerkiksi peliltä, mutta alkaa koneeseen päästessä tehdä jotain ylimääräistä. [Toivanen, 2004]

Troijalainen voi seurata koneen näppäimistöä ja kaapata siltä kirjoitetut salasanat, tuhota ja muokata tiedostoja, tai antaa hyökkääjälle etähallintamahdollisuus kohdejärjestelmään. Troijalaiset eivät pysty levittämään itseään, vaan tulevat aina osana toista ohjelmaa. Yleensä troijalainen asentuu järjestelmään mahdollisimman näkymättömästi. [Boström, 2003; Paananen, 2003; Paavilainen, 1998; Toivanen, 2004]

2. Troijalaisen alatyypit

Tässä luvussa tarkastellaan lähemmin erilaisia troijalaisia, niiden toimintaa, sekä leviämistapoja. Troijalaiset voidaan jaotella eri tyyppeihin niiden toiminnallisuuden perusteella. Troijalainen ohjelma voi myös olla näiden eri tyyppien yhdistelmä, sillä toisinaan troijalaisen tekijä haluaa mieluummin kaikki toiminnallisuudet samassa ohjelmassa, sen sijaan että olisi monta eri ohjelmaa hallittavana. [Kaspersky, 2006] Mitä syvemmin troijalaisia tarkastellaan, sitä useampiin kategorioihin ne voisi jakaa. Tässä on kuitenkin päädytty luettelemaan vain joitakin yleisimpiä ja tunnetuimpia alatyyppejä.

2.1. Takaovi (Backdoor)

Takaoven avulla saadaan otettua yhteys etäjärjestelmään. Toisin kuin lailliset etäyhteysohjelmat, nämä troijalaiset asentuvat, käynnistyvät ja toimivat näkymättömissä, ilman että koneen käyttäjä tietää asiasta mitään. [Emm, 2006a]

Tunkeutujat käyttävät erilaisia temppuja tartuttaakseen takaoven etäkoneeseen - he lähettävät tiedostot esimerkiksi toisen troijalaisen mukana, tai lähettävät sähköpostilla mielenkiintoiselta vaikuttavan tiedoston, jonka sitten pahaaavistamaton ihminen lataa koneelleen. Myös jotkut virukset ja madot voivat pudottaa takaoven tartuttamaansa koneeseen. [F-Secure-Vdescs, 2006]

Kun takaovi on asennettu, ohjelma itse tai sen kautta järjestelmään päässyt tunkeutuja yleensä tukkii mahdollisuuden asentaa uusia takaovia. Syy tähän on se, että yhden takaoven koneelle saanut hyökkääjä tuskin haluaa, että jonkun toisen hallinnoima, vastaavanlainen ohjelma käy kisaamaan koneen määräysvallasta. [Boström, 2003]

Takaovi on vaarallisin ja levinnein troijalaisen alatyyppi. Kaikista monimutkaisimmat takaovet antavat tunkeutujan tehdä etäkoneella mitä vain. Asennuksen jälkeen troijalainen voidaan ohjeistaa lähettämään, vastaanottamaan, käynnistämään ja tuhoamaan tiedostoja, valvomaan koneen käyttäjän tekemisiä ja selailemaan hänen tiedostojaan, kommunikoimaan käyttäjän kanssa ja paljon muuta. Jotkin takaovet jopa antavat hyökkääjän kuunnella ja nähdä mitä etäkoneen äärellä tapahtuu, mikäli koneeseen vain on liitetty mikrofoni tai nettikamera. [Emm, 2006a; F-Secure-Vdescs, 2006]

2.2. Latausohjelma (Downloader)

Troijalainen latausohjelma yrittää avata yhteyden Internet-sivustolle ladatakseen koneelle uusia tiedostoja ja käynnistää ne ilman koneen käyttäjän hyväksyntää. Yleensä latausohjelma lataa erityyppisiä troijalaisia, mutta myös muita haitallisia ohjelmia. Onnistuessaan tehtävässään, latausohjelma saattaa poistaa itsensä. [F-Secure-Vdescs, 2006; Kaspersky, 2006]

Koska latausohjelmat ovat pieniä kooltaan, ne voidaan piilottaa muiden yhtä pienien, mutta vaarattomien, tiedostojen sekaan. Virustorjuntaohjelmistoilla saattaa olla vaikeuksia tunnistaa näitä troijalaisia siksi, että todennäköisesti ei ole kahta tunkeutujaa, joka antaisi ohjelmalle haettavaksi saman verkko-osoitteen ja tiedostot. Myös latausohjelman koko saattaa vaihdella jokaisen asennuskerran jälkeen. [Duke, 2002]

2.3. Pudottaja (Dropper)

Troijalainen pudottaja on ohjelma, joka pudottaa koneelle erityyppisiä haitallisia ohjelmia, kuten muita troijalaisia ja matoja. Monesti Internetissä toimivat mainosyhtiöt käyttävät pudottajia pudottaakseen käyttäjän koneelle mainos-, ja vakoiluohjelmia sekä troijalaisia latausohjelmia.

Tyypillinen pudottaja on tiedosto, jonka sisälle on pakattu muita tiedostoja. Kun pudottaja käynnistetään, se irrottaa kaikki tiedostot sisältään johonkin kansioon, yleensä väliaikaiseen sellaiseen, ja käynnistää tiedostot samanaikai-

sesti. [F-Secure-Vdescs, 2006] Samassa paketissa voi tulla monenlaisia haitallisia ohjelmia, joilla ei ole mitään tekemistä toistensa kanssa. Ne saattavat toimia erilailla toisiinsa nähden tai ovat jopa eri ohjelmoijien kirjoittamia. [Emm, 2006a] Harhauttaakseen käyttäjää pudottaja sisältää usein myös vaarattomia tiedostoja.

Pudottajat luodaan yleensä erityisillä ohjelmilla, jotka antavat päättää, minkälaisia toimintoja pudottaja sisältää sekä mitä tiedostoja pakettiin laitetaan. [F-Secure-Vdescs, 2006] Pudottajia käytetään usein tunnettujen troijalaisten kuljettamiseen, sillä on huomattavasti helpompaa kirjoittaa pudottaja kuin täysin uusi troijalainen, jota virustorjuntaohjelma ei huomaisi. [Emm, 2006a]

2.4. Salasanavaras, tietovaras ja vakoilija (Password/Data Stealer, Spy)

Troijalainen salasanavaras on suunniteltu varastamaan salasanoja käyttäjän koneelta. Monesti troijalainen tallentaa myös tietoja käyttäjän tietokoneesta, kuten esimerkiksi IP-osoitteen. [Emm, 2006a; F-Secure-Vdescs, 2006]

Ohjelma aloittaa näppäinpainallusten tallentamisen silloin, kun käyttäjää pyydetään kirjoittamaan käyttäjätunnus ja salasana. Troijalainen säilöö näppäinpainallukset ja lähettää nämä tiedot tunkeutujan sähköpostiosoitteeseen. Varastettujen käyttäjätunnuksien ja salasanojen avulla tunkeutuja voi esimerkiksi lukea käyttäjän sähköpostiviestejä.

Troijalainen tietovaras on ohjelma, joka etsii tiettyjä tiedostoja tai tietoja tartutetulta koneelta ja lähettää nämä tiedot tunkeutujalle. Jotkin tietovarkaat yrittävät esimerkiksi varastaa koneeseen asennetun ohjelmiston sarjanumeron.

Troijalainen vakoilija on ohjelma, joka tallentaa tiettyjä tapahtumia tartuttamallaan koneella. Tällainen troijalainen voi esimerkiksi tallentaa näppäinpainalluksia, pitää listaa ohjelmista, joita käyttäjä on käynnistänyt, sekä säilöä verkko-osoitteita, joilla käyttäjä on käynyt. Troijalainen lähettää nämä tiedot tunkeutujalle tietyin väliajoin. Joissain tapauksissa vakoilija toimii vain tiettyyn päivämäärään asti ja poistaa sitten itsensä järjestelmästä. [F-Secure-Vdescs, 2006]

2.5. Muita troijalaisia

Troijalainen napsauttelija (Clicker) yrittää jatkuvasti yhdistää tietyille Internetsivuille. Näin saadaan esimerkiksi kohdesivun käyntimäärää nostettua, jolloin sivustolla näytettävistä mainoksista saadaan enemmän rahaa, tai ohjataan käyttäjä sivustolle, josta löytyy muunlaisia haitallisia ohjelmia, kuten toinen troijalainen. Troijalaista voidaan käyttää myös palvelunestohyökkäykseen, jossa kohdesivuston käyttö saattaa estyä kokonaan ja aiheuttaa sivuston omistavalle yritykselle tappioita. [Emm, 2006a; F-Secure-Vdescs, 2006]

Troijalainen välipalvelin (Proxy) antaa tunkeutujan käyttää Internetiä tartutetun koneen kautta. Tällöin yhteys pystytään jäljittämään vain tartutettuun tietokoneeseen ja tunkeutujan oikea osoite jää tuntemattomaksi. Näitä troijalaisia käytetään usein roskapostin massalevitykseen. [Emm, 2006a; F-Secure-Vdescs, 2006]

Troijalaisen ilmoittajan (Notifier) tarkoituksena on kertoa tunkeutujalle tapahtumista käyttäjän tietokoneella. Se voi ilmoittaa esimerkiksi, kun jokin haitallinen ohjelma, kuten toinen troijalainen, on asennettu käyttäjän tietokoneelle. Ilmoitus tapahtuu joko lähettämällä sähköpostia, pikaviestejä tai ottamalla yhteys tietyille sivustoille. Nämä troijalaiset tulevat usein samassa paketissa muiden haitallisten ohjelmien kanssa. [Emm, 2006a; F-Secure-Vdescs, 2006]

Looginen pommi (Logic bomb) on vahinkoa tekevä troijalainen ohjelma, joka käynnistyy tiettyjen ehtojen täyttyessä. Ehtona voidaan käyttää esimerkiksi päivämäärää tai kellonaikaa. [Balding et al., 2002]

Voidaan olla montaa mieltä siitä, onko pilaohjelma troijalainen. Pilaohjelma ei aiheuta varsinaista vahinkoa, kuten troijalaiset yleensä, vaan ainoastaan ärsyttää tai hauskuuttaa käyttäjää. Mutta koska pilaohjelmat usein yhdistetään troijalaisiin, on näistä myös hyvä mainita. Pilaohjelma on yksi ensimmäisistä troijalaisen muodoista. Eräs vanha esimerkki PDP Cookie –ohjelma ilmestyy näytölle ja pyytää uhrilta keksiä. Toinen laajalle levinnyt esimerkki CokeGift-ohjelma puolestaan tarjoaa uhrille koneen CD-aseman luukkua juomatelineeksi. [Balding et al., 2002]

3. Leviämistavat

Troijalaiset eivät voi levitä itsestään, toisin kuin virukset. Kaikki troijalaiset tulevat aina jonkin toisen ohjelman mukana, joko osana itse ohjelmaa tai erillisenä, asennusvaiheessa järjestelmään ujutettavana kylkiäisenä. [Boström, 2003]

Yleinen troijalaisten lähde on Internetissä sijaitsevat uutisryhmät, joissa lähetellään binääritiedostoja, sekä laittomia ohjelmia levittävät vertaisverkot ja pornografista materiaalia sisältävät sivustot. [Balding et al., 2002]

Ohjelmat, joissa itsessään sisällä on troijalainen, voivat olla esimerkiksi pelejä tai muita harmittomilta vaikuttavia ohjelmia. Näiden levitys onnistuu vaikka Internetin keskustelupalstoilla. Riittää, että mainostaa uutta hienoa ohjelmaa, ja odottaa, että joku kiinnostunut lataa ohjelman ja käynnistää sen koneellaan.

Kylkiäisenä tulevat troijalaiset voivat tulla suosittujenkin ohjelmien mukana. Näiden ujuttaminen mukaan onnistuu esimerkiksi murtamalla ohjelman kotisivupalvelin tai levityspaikka, ja korvaamalla alkuperäinen ohjelman asennuspaketti toisella paketilla, joka sisältää troijalaisen. Jos kyseessä on avoimen lähdekoodin ohjelma, hyökkääjä voi ujuttaa troijalaisen melkein minne tahansa.

Suljetun lähdekoodin ohjelmiin on huomattavasti vaikeampi ujuttaa ulkopuolista koodia. [Boström, 2003]

4. Havaitseminen

Troijalaisen havaitseminen ei aina onnistu virustutkilta. Troijalaisen koodi nimittäin ei ole ihmisen luettavissa tai edes konekielistä. Ainoat binääritiedostoista tunnistettavat merkkijonot ovat tekijänoikeusviestejä, virheilmoituksia tai muuta ylimääräistä dataa. Kuitenkin jos tiedetään troijalaisen koodista edes jotain, voidaan kyseistä merkkijonoa käyttää troijalaisen tunnistamiseen. [Balding et al., 2002]

Ihmisen näkökulmasta troijalainen yleensä toimii täysin näkymättömissä, mutta on myös tapauksia, joissa ohjelma antaa tarkoituksella virheilmoituksen, että ohjelma ei toimikaan. Käyttäjä huokaisee helpotuksesta ja antaa asian olla. Ilmoituksesta huolimatta troijalainen on kuitenkin jo päässyt järjestelmään. [Boström, 2003]

Monet troijalaiset saattavat esiintyä asiallisina, tunnettuina ohjelmina, joiden toiminta järjestelmässä vaikuttaa normaalilta. Troijalaisia ei siis välttämättä voida havaita käynnissä olevien prosessien listasta. Eräs tapa havaita troijalaisia on etsiä epäilyttäviä tiedostoja hakemistorakenteista. Jos huomataan, että jonkun itse tekemän tiedoston muokkauspäivämäärä, luontipäivämäärä tai koko on vaihtunut, ja tietää ettei siihen ole koskenut, voi tämä olla merkki pahantahtoisesta toiminnasta. Tosin tämä ei ole kovinkaan hyvä puolustus, muuten kuin alkeellisia hyökkäyksiä vastaan. Tiedoston päivämäärä ja koko voidaan helposti väärentää.

Tavalliselle käyttäjälle, joka ei tunne tietokonetta tarpeeksi hyvin, troijalaisen tunnistaminen voi olla hankalaa. Se ei ole helppoa edes kokeneelle ohjelmoijalle, jolla on mahdollisuus tutkia ohjelman lähdekoodia. Ohjelmakoodin kääntäminen ihmiselle ymmärrettävään muotoon vaatii paljon kärsivällisyyttä.

Kun troijalainen löydetään ja poistetaan, saattavat sen jalanjäljet, piilotetut sisäänpääsyaukot, säilyä hakemistojärjestelmän tai Windowsin rekisterin hämärissä nurkissa. Tämä on erityinen huolenaihe yrityksissä, joissa samaa järjestelmää käyttää monta työntekijää. [Balding et al., 2002]

5. Suojautuminen

5.1. Kotona

Paras tapa suojautua troijalaisilta kotona on olla vahingossakaan ajamatta mitään tuntemattomia ohjelmia. Tutulta näyttävän ohjelman lataaminen joltain epämääräiseltä sivulta ei kannata, sillä sen mukana saattaa tulla myös troijalai-

nen. Jokainen uusi asennettu ohjelma lisää riskiä epäluotettavan koodin tunkeutumisesta järjestelmään. Lisäksi virustutka on hyvä pitää ajan tasalla. Virustutkat tunnistavat ainakin osan tunnetuimmista troijalaisista. Myös palomuuri on syytä olla, sillä se osaa tarkkailla sitä, mitkä ohjelmat pyrkivät ottamaan yhteyden verkkoon. Ellei troijalainen ole osannut poistaa palomuuria käytöstä, pitäisi palomuurin estää yhteydenotto eri koneiden välillä. [Balding et al., 2002; Boström, 2003]

5.2. Työpaikalla

Työpaikoilla hyvä suojaus troijalaisia vastaan on palomuuri, virustorjunta, sekä varmuuskopion ottaminen koko koneen sisällöstä lyhyin väliajoin. Esimerkiksi erotettu työntekijä saattaa vihastuksissaan jättää jälkeensä loogisen pommin. Tämän vuoksi kannattaa tarkistaa kaikki yleiset tiedostot, joihin työntekijä pääsi käsiksi, sekä siirtää hänen tiedostot ja ohjelmat varmuuskopiolle ja poistaa itse koneelta kaikki mitä ei enää tarvita. Jos katastrofi kuitenkin tapahtuu, saa koneen sisällön palautettua vahingoittumattomana. [Ogletree, 2001]

Toinen tärkeä asia on työntekijöiden koulutus, jotta he pystyisivät omalla toiminnallaan vaikuttamaan siihen, että haitallisia ohjelmia ei asennu heidän työasemille. [Allen, 2002] Troijalaiset naamioituvat usein peleiksi, pilailuohjelmiksi, näytönsäästäjiksi ja muiksi ohjelmiksi, joita usein vaihdetaan sähköpostitse, varsinkin silloin, kun tiukkoja järjestelmä- tai tietoturvasääntöjä ei ole annettu tai niiden noudattamista ei valvota. Jos ohjelma sisältää yksityisyyttä loukkaavan troijalaisen tai tuhoa tekevän troijalaisen, jonka toimintaa on viivästetty loogisella pommilla, ohjelman vahingollisuudesta tietämätön uhri saattaa jakaa ohjelmaa eteenpäin. [Balding et al., 2002]

6. Kustannukset eri laiteympäristöissä

Troijalaiset harvoin aiheuttavat tietokoneen omistajalle suoria rahallisia menetyksiä. Joskin tämäntyyppiset hyökkäykset ovat yleistymässä. Järjestelmään tunkeutunut hyökkääjä saattaa laittaa tärkeitä tiedostoja salasanan taakse. Hyökkääjä sitten lähettää tietokoneen omistajalle viestin, jossa käsketään maksaa tietty rahasumma Internetissä toimivalle pankkitilille. Jos käyttäjä ei tottele, saisi hän sanoa tiedostoille hyvästi. [Emm, 2006a]

Matkapuhelimessa troijalainen voi sen sijaan aiheuttaa mittavat kustannukset puhelimen omistajalle. Puhelin saattaa lähettää itsestään jopa satoja tekstiviestejä päivässä käyttäjän luettelosta löytyviin numeroihin tai numeroihin, jotka on koodattu troijalaisen sisään, ilman että puhelimen omistaja huomaa mitään. Omistaja luultavasti saa tietää hyökkäyksestä vasta saatuaan suuren laskun kuukauden päästä. Troijalaisen avulla pystytään myös varastamaan mat-

kapuhelimeen säilöttyjä henkilökohtaisia tietoja. [Emm, 2006b; Yap and Ewe, 2005]

7. Yhteenveto

Troijalaisten tuotannossa vain mielikuvitus on rajana. Ne voidaan ohjelmoida tekemään melkein mitä vain ja niitä voidaan jakaa eteenpäin monella eri tavalla. Ne voivat tarttua mihin tahansa käyttöjärjestelmään, ja laiteympäristöistä ainakin tietokoneisiin ja matkapuhelimiin. Viime aikoina troijalaisista on tullut yhä etenevässä määrin kyberterrorismin välineitä, joilla kerätä ihmisiltä rahaa laittomasti. Tällainen suuntaus jatkuu varmasti hamaan tulevaisuuteen.

Viiteluettelo

[Allen, 2002] Julia H. Allen, *CERT Verkkotietoturvan hallinta*. IT Press, 2002.

[Balding et al., 2002] Craig Balding, Billy Barron, Robert Blader, Chad Cook, Jonathan Feldman, David Harley, Joe Jenkins, L. J. Locher, Toby Miller, Brooke Paul, Nicholas Raba, Greg Shipley and Gregory B. White, *Hakkerin käsikirja*. IT Press, 2002.

[Boström, 2003] Mika Boström, *Kotimikron tietoturva*. Talentum, 2003.

[Duke, 2002] David Duke, Downloader Trojans. *Network Security* 11/2002 (November 2002), 4-5.

[Emm, 2006a] David Emm, Focus on trojans – holding data to ransom. *Network Security* 6/2006 (June 2006), 4-7.

[Emm, 2006b] David Emm, Mobile malware – new avenues. *Network Security* 11/2006 (November 2006), 4-6.

[F-Secure-Vdescs, 2006] F-Secure, Virus Description Database. http://www.f-secure.com/v-descs/. Checked 17.12.2006.

[Kaspersky, 2006] Kaspersky Lab, Viruslist. http://www.viruslist.com/. Checked 4.11.2006.

[Ogletree, 2001] Terry Ogletree, *Verkot*. IT Press, 2001.

[Paananen, 2003] Juha Paananen, *Tietotekniikan peruskirja*. Docendo, 2003.

[Paavilainen, 1998] Juhani Paavilainen, *Tietoturva*. Suomen Atk-kustannus, 1998.

[Toivanen, 2004] Aarni Toivanen, *Internet & seniori*. Docendo, 2004.

[Yap and Ewe, 2005] Teck Sung Yap and Hong Tat Ewe, A mobile phone malicious software detection model with behavior checker. In: *Proc. of Web and Communication Technologies and Internet-Related Social Issues, 3rd International Conference on Human.Society@Internet HSI 2005, Lecture Notes in Computer Science* **3597** (2005), Springer, 57-65.

Tietokonepohjainen tunneilmaisuiden tunnistaminen kasvoista ja sen tekniikka

Jouni Erola

Tiivistelmä

Tämä tutkielma on kirjallisuuskatsaus tietokonepohjaiseen tunneilmaisuiden tunnistamiseen kasvokuvista. Tutkielma sisältää perustietoa tekniikoista ja menetelmistä, joita nykyisissä toteutuksissa on käytetty. Tutkielma pohtii myös uusien menetelmien suunnittelulähtökohtia ja parannuksia jo toteutettuihin menetelmiin. Tutkielmassa käsitellään myös lyhyesti tämän tietokoneen ja ihmisen välisen modaliteetin käyttömahdollisuuksia.

Tunneilmaisut ovat sanattoman viestinnän kanava, jonka avulla ihmiset kommunikoivat. Tällä hetkellä perinteisessä tietokone-ihminen-kommunikaatiossa välineinä ovat lähinnä näppäimistö ja hiiri. Puhekäyttöliittymätkin ovat tehneet tuloaan, mutta eivät ole saavuttaneet vielä laajaa käyttöä. Tietokoneen ja ihmisen välinen kommunikaatio on siis melko rajallista. Visiot tulevaisuudesta maalailevat kuvaa, jossa näppäimistöä ja hiirtä ei tarvittaisi lainkaan, vaan kommunikaatio koneen kanssa tapahtuisi puheiden ja eleiden avulla niin kuin ihmisten välinen kommunikaatio. Kone voisi olla tällöin nöyrä palvelija James, joka osaisi ottaa huomioon käyttäjänsä mielialan. James näkisi siis käyttäjänsä kuten toinen ihminen näkee ja mallintaisi muistiinsa ihmisen elehdintää ja käyttäisi tätä tietoa hyväkseen vuoro-vaikutuksessa. Tunneilmaisut ovat tällaisessa visiossa tärkeässä osassa, kuten monet muutkin multimodaalisen käyttöliittymän kommunikointiväylät.

Yksi mielenkiintoinen, tämän tutkimuksen ulkopuolinen, idea olisi käyttäjän varmentaminen. Esimerkiksi neuroverkkoihin perustuvassa tunnistusmallissa voitaisiin käyttäjän identiteetti varmistaa samalla rutiinilla. Kun käyttäjän eleistä on tehty tarpeeksi vahva neuroverkko, se kykenee tunnistamaan käyttäjänsä vertaamalla eleitä olemassa olevien käyttäjien muodostamiin neuroverkkoihin. Yhdessä muiden (esimerkiksi ääni, sormen-jälki) tunnistusmekanismien kanssa, voitaisiin käyttäjä vahvistaa oikeaksi, eikä konetta pääsisi käyttämään kukaan ulkopuolinen.

Avainsanat ja -sanonnat: Hahmontunnistus, kuvantunnistus, tunneilmaisut, kasvot
CR-luokat: H.1.2, H.5.2, I.5.2

1. Johdanto

Tässä tutkielmassa tutustutaan tekniikoihin, joilla kasvojen tunneilmaisuja tunnistetaan digitaalisesta kasvokuvasta. Tunnistusprosessissa käytetään joko pysähtynyttä digitoitua valokuvaa tai liikkuvaa videokuvaa, joista ensimmäistä näyttää olevan tutkittu enemmän kuin jälkimmäistä. Kuvasta pyritään löytämään kasvojen eri osat erilaisten tietokonealgoritmien avulla. Näitä osia ja niiden suhteita analysoimalla pyritään luokittelemaan kasvoilla oleva ele tai ilme ja lopulta tunnistamaan varsinainen tunneilmaisu.

Tunneilmaisuiden tunnistamiseen liittyviä ongelmia ovat mm. hahmontunnistus ja erilaiset inhimilliset piirteet. Esimerkiksi kaikki ihmiset eivät elehdi ja tunneilmaisut eivät ole globaaleja. Persoona sekä siihen liittyvä konteksti (esimerkiksi taustainformaatio, missä tilanteessa elehditään) vaikuttavat lopputulokseen ihmisen tulkitessa tunneilmaisua. Tietokonealgoritmeilla ei välttämättä tätä tietoa ole ja tämä osaltaan heikentää testaustuloksia.

Tunneilmaisuiden tulkinta pitäisi mielestäni olla kaikissa tilanteissa ja kaikissa kulttuureissa toimivaa täysin automaattista ja luotettavaa toimintaa. Koneen pitäisi pystyä itsenäisesti tunnistamaan minkä tahansa etnisen ryhmän edustajan tunneilmaisut ilman että konetta autetaan.

Näyttää kuitenkin valitettavasti siltä, että suurin osa toteutetuista tunneilmaisuiden tunnistamista varten tehdyistä ohjelmista ei kykene tähän. Osa näistä vaatii jopa käyttäjän toimia, jotta tunnistus voidaan ylipäätään tehdä, ja suurin osa menetelmistä ei salli esimerkiksi silmälasien käyttöä eikä tunnistettavalla saa olla partaa. Kuitenkin Taloustutkimuksen (2004) julkaiseman Omnibus-tutkimuksen mukaan yli puolet kyselyyn osallistuneista käyttää silmälaseja (51%). Tämä tarkoittaa sitä, että yli puolella väestöstä eivät nyt toteutetut järjestelmät toimi.

Tehdyt tutkimukset ovat perustuneet yksittäisiin kasvokuviin, jotka on otettu hyvissä valo-olosuhteissa. Tutkimusta olisikin siirrettävä nykyaikaisempaan tekniikkaan, jossa on mahdollista tehdä tunnistaminen videokuvasta.

2. Tunneilmaisuiden tunnistaminen

Tässä luvussa käsitellään tunneilmaisuiden tunnistamisen prosessia ja sen vaiheita. Tunneilmaisuiden tunnistaminen on jaettu kolmeen osaan: kuvankäsittelyyn, eri kasvojen osien tiedon keräämiseen kuvasta sekä eleen tunnistamiseen.

2.1. Kuvankäsittely

Aluksi sisään syötettävä kuva tai video käsitellään. Esimerkiksi kontrastierot tasoitellaan kuvan värihistogrammin analyysillä. Kuvasta voidaan poistaa kohina eli ylimääräiset pikselit, jotka häiritsevät itse tunnistusta. Jos kyseessä on videokuva, voidaan kuvalle tehdä staattisen informaation poisto, eli esimerkiksi kuvan taustalla olevat liikkumattomat esineet yms. voidaan poistaa tekemällä vertailu kahden tai useamman kuvan vä-

lillä. Kun kyseessä on yksi valokuva, ei ole vertailukohtaa ja taustan poisto ei tällöin onnistu.

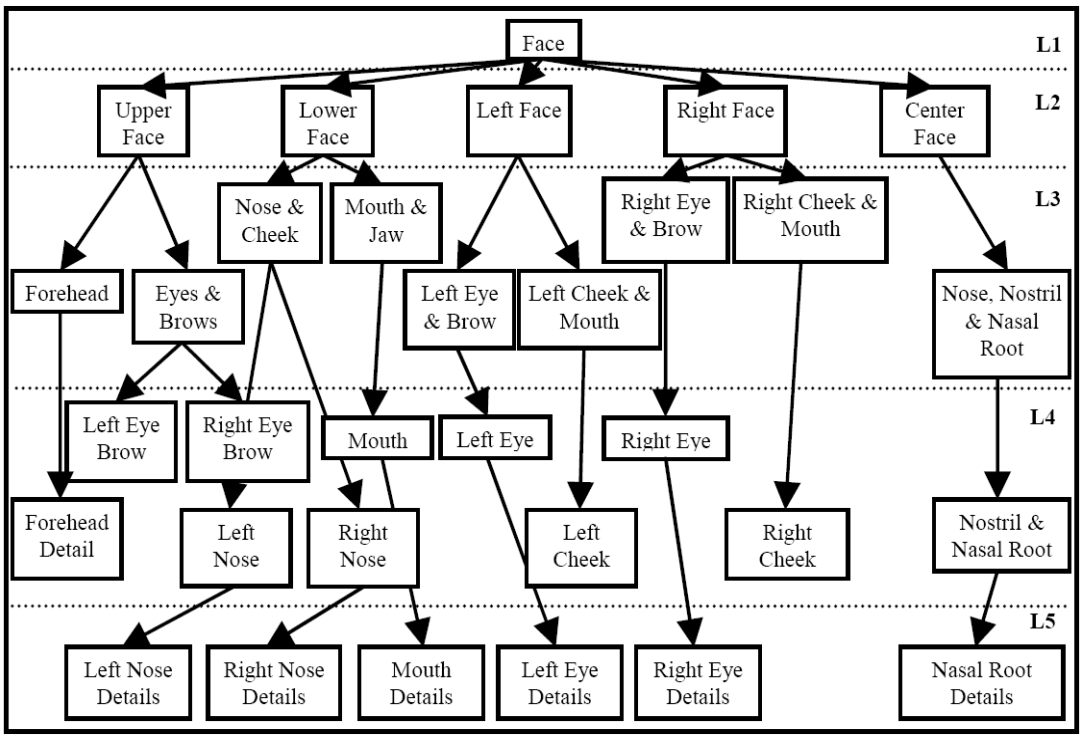
2.2. Kasvojen osien tiedon kerääminen

Esikäsitellystä kuvasta on seuraavaksi löydettävä eri kasvojen alueet: silmät, kulmakarvat, suu. Osassa tutkituissa menetelmistä kuvasta etsittiin myös leuan kaaret ja korvat, kuten Athanasios Nikolaidis ja Ioannis Pitas (2000) ovat tehneet.

Yksi tapa tunnistaa kasvojen osien rajat on kontrastinmuutos. Kuvasta etsitään siis suuria kontrastinmuutoksia ja näiden tapahtuessa havaitaan eri kasvojen osien esiintyminen. Kontrastien vähyys tai puute aiheuttaa suuria ongelmia tunnistusmenetelmälle.

Muita tehokkaampia tekniikoita ovat optisen virtauksen etsiminen ja yksi tehokkaimmista menetelmistä ”Active Contour Models”, jonka ovat kehittäneet Kass ja muut (1987).

Kasvojen osat voidaan tallentaa tietokoneen muistiin monella eri tavalla, esimerkiksi alla olevan kuvan mukaisesti. Kuvassa 1 on esitetty Wongin ja Chon (2006) menetelmä. Siinä eri kasvojen osat on tallennettu puurakenteeseen, jossa kasvot on ensin jaettu ylä-, ala-, vasen-, oikea-, ja keskijaottelulla. Näiden sisältä löytyvät tarkemmin kasvojen alueet, esimerkiksi silmät ovat eri tasoilla, koska ne kuuluvat oikeaan ja vasempaan puoleen. Lisäksi silmät kuuluvat yläosaan kasvoista, joten ne löytyvät useammasta eri puurakenteen haarasta. Tämän rakenteen mukainen tieto syötetään neuroverkolle, joka opettamisvaiheen jälkeen antaa tuloksena eri Facial Action Coding Systemin Action Unit –arvoja. Action Unit –arvo kertoo, mikä osa kasvoista on aktiivisena. Esimerkiksi Action Unit 10 tarkoittaa sitä, että ylempi huuli on noussut ylös. Pelkästään Action Unit –arvojen pohjalta ei vielä voi tietää, mikä tunneilmaisu kasvoilla on, vaan niiden pohjalta on tehtävä tulkinta käyttäen psykologian määritelmiä tunneilmaisuille.



Kuva 1. Kasvot voidaan tallentaa muistiin esimerkiksi Wong ja Chon (2006) rakenteen mukaisesti

Tarkemmin kasvotietojen tallennuksessa voidaan käyttää esimerkiksi eri osien etäisyyksiä toisistaan ja näiden suhteita. Jos henkilö, jonka ilmettä tutkitaan, hämmästyy, kulmakarvat nousevat ylöspäin samalla kasvattaen etäisyyksiä silmiin. Toki kasvoissa tapahtuu muitakin muutoksia, mutta tämä esimerkkinä siitä, kuinka eri kasvojen osien etäisyys kertoo kyseisessä tunneilmaisussa.

Essa ja Pentland (2006) lähestyivät kasvoalueiden mallintamista fysikaalisesta näkökulmasta. Heidän menetelmänsä etsii silmät, nenän ja suun, ja tämän tiedon avulla mallintavat kuvan kasvot kolmiulotteisen lihas-mallin päälle. Tämän kolmiulotteisen mallin avulla menetelmä pystyy tunnistamaan eri Action Unit –osiot ja niiden aktiivisuudet.

Facial Action Coding System (FACS) kuvaa ainoastaan Action Unit –muuttujat eikä lainkaan ilmeiden psykologista merkitystä (ilo, suru, pelko jne.), tätä varten voidaan käyttää esimeriksi FACSAID–tietokantaa, joka kertoo tunneilmaisun, kun sille annetaan kaikki Action Unit –muuttujat, jotka ovat aktiivisia sen hetkisessä ilmeessä.

Tunneilmaisuiden luokitteluun liittyy kuitenkin paljon ongelmia: kulttuurierot, joidenkin kulttuurien tapa olla näyttämättä tiettyjä tunteita kasvoillaan, sekä joidenkin tunneilmaisuiden sekoittuminen keskenään. Ihmisellä on yleensä käytössään paljon taustainformaatiota, kun taas koneen tehdessä tulkintaa konetta ei voida olettaa tietävän tunnistettavan persoonallisuudesta tai tavasta elehtiä mitään.

2.3. Ideaalijärjestelmä

Omasta mielestäni ideaalinen tunneilmaisuiden tunnistava tietokone olisi täysin automaattinen sekä mukautuva erilaisiin käyttöympäristöihin. Ei voida vaatia että käyttäjä ei käytä silmälaseja tai käytä partaa. Ei myöskään voida vaatia optimaalisia valo-olosuhteita eikä useampaa kameraa, puhumattakaan kasvoihin teipattavista pienistä pyöreistä tarroista.

Optimijärjestelmä toimisi nykyisillä web-kameroilla käyttöjärjestelmän yhteydessä niin, ettei käyttäjän tarvitse kiinnittää siihen mitään huomiota. Järjestelmä tarjoaisi käyttöjärjestelmän kautta yleisen rajapinnan ohjelmille jonka avulla ne voisivat hakea tietoa käyttäjän tunnetilasta, jos tämä tieto on saatavilla.

Ideaalijärjestelmän ei tarvitse tunnistaa elettä samalla sekunnilla kun se kytketään päälle ensimmäistä kertaa. Tällöin tunnistusrutiinilla olisi mahdollisuus tutustua käyttäjäänsä ja kalibroida esimerkiksi neuroverkkonsa sopivaksi, jotta tunnistus on myöhemmin mahdollista mahdollisimman luotettavasti.

Elävästä videokuvasta on mahdollista tehdä tunnistus useamman ”kuvan” perusteella, eikä tunnistus rajoitu nykyisten tutkimusten yhden mallivalokuvan periaatteeseen. Uskoisin, että on helpompaa tunnistaa ja havaita ele, jos esimerkiksi nähdään kuinka ele kehittyy ja missä ajassa.

Yksi järjestelmän ominaisuuksista tulisi olla se, että järjestelmä selviäisi tilanteesta jossa jotain kasvon osaa ei löydetä tai tunnisteta. Toisin sanoen, optimaalijärjestelmä ei saisi perustua vaatimukseen että kuva on täydellinen – tunnistus on tapahduttava tästä huolimatta, siinä määrin kuin se on mahdollista. Tähän lopputulokseen on tullut myös Pantic ja Rothkrantz (2000) omassa tutkimuksessaan.

Järjestelmän pitäisi olla oppiva ja sopeutuva. Koska eleiden voimakkuus vaihtelee henkilöittäin, yleinen kiinteä parametrointi ja säännöstö ei toimi universaalilla tasolla.

3. Menetelmistä

Tässä luvussa esitellään menetelmiä, jotka liittyvät hahmontunnistukseen ja tunnistettujen hahmojen tulkintaan kohti tunneilmaisuita. Luvun tarkoitus on antaa yleinen käsitys siitä minkälaisin menetelmin eleitä voidaan tunnistaa.

3.1. Hahmontunnistus

Ihmisen kasvoista otettu kuva sisältää värisävyjen liukuja ja jyrkkiä kontrastimuutoksia. Hahmontunnistusalgoritmit käyttävät kuvissa hyväkseen näitä ominaisuuksia löytääkseen kuvasta reunoja ja alueita, joita pidetään kasvojen eri osina. Esimerkiksi kulmakarvat ovat yleisesti tummat muihin kasvoihin verrattuna, ja näin niiden reunat on helppo löytää. Esimerkiksi kuva voidaan ensin muuttaa harmaasävyiksi ja tämän jälkeen kuvaa pehmennetään jotta siitä poistuu tarpeettomat pienet detaljit, joita ei tarvita hahmontunnistuksessa. (Guizatdinova & Surakka 2006)

Mikäli kuvassa on huono valaistus, hiukset ovat kasvojen edessä tai henkilöllä on partaa, joutuvat hahmontunnistusrutiinit ongelmatilanteisiin.

3.2. Facial Action Coding System

Todennäköisesti käytetyin (Pantic & Rothkrantz 2000) luokittelumalli kasvojen tunneilmaisuille on FACS (Facial Action Coding System), jonka kehittivät Ekman ja Friesen 1970-luvulla. Luokittelumalliin kuuluu 44 ns. toimintayksikköä (Action Unit) joiden avulla järjestelmää käyttämään koulutettu ihminen pystyy selvittämään eleen kasvokuvasta.

FACS -järjestelmässä on 46 Action Unit (AU) -muuttujaa jotka kuvaavat muutoksia kasvojen eleessä. Näiden yhdistelmä tuottaa suuren joukon mahdollisia kasvojen eleitä. Esimerkiksi ilo on usean muutoksen yhdistelmä. Tällöin ylähuulen kulmat nousevat ylös (AU 12+13), ja/tai suu aukenee (AU 25+27). Samalla on havaittavissa huulen kohoamista (AU) 10) ja poskiuurteiden syvenemistä (AU 11). Tämä on kuitenkin vain yhdenlainen hymy, hymystä voi olla useita variaatioita. (Essa & Pentland, 1995)

FACS ei kuitenkaan ole mutkaton tapa tunnistaa eleitä koska FACS muodostaa suuren määrän erilaisia yhdistelmiä, kuten Essa ja Pentland (1995) ovat havainneet.

3.3. Malliin perustuva tunnistus

Malliin perustuvassa (template based) tunnistuksessa tutkittavasta kuvasta muodostetaan malli, jota verrataan muihin ennalta määriteltyihin malleihin. Ennalta määritellyt mallit voivat olla esimerkiksi valokuvia tai tietorakenteita. Yksi tällainen mallinnustapa on käyttää eigenfaces–ratkaisua jossa käytetään eigenvector–muotoa kuvaamaan kasvot. Kun kuvasta on tehty eigenfaces-rakenne, siitä voidaan piirtää harmaasävykuva, josta voi tunnistaa kasvojen rakenteen.

Mallintaminen voidaan tehdä esimerkiksi silmien, nenän ja suun etäisyydestä toisistaan. Tällöin malli koostuisi neljästä oliosta ja niiden suhteita kuvaavasta tiedosta.

Mallin vertailu voi perustua rinnakkaisvertailuun tutkittavan kuvan ja esimerkkikuvien välillä. Muistiin on tallennettu esimerkiksi 100 erilaista kuvaa ja niissä esiintyvä tunneilmaisu. Vertailukuvista on myös muodostettu malli samalla menetelmällä kuin analysoitavasta kuvasta. Vertailussa verrataan muistissa oleviin kuviin tutkittavan kuvan mallinnettua elettä. Näistä valitaan lähimmäksi todettu vaihtoehto ja rinnakkaisvertailtavan kuvan ele todetaan myös tutkittavan kuvan eleeksi.

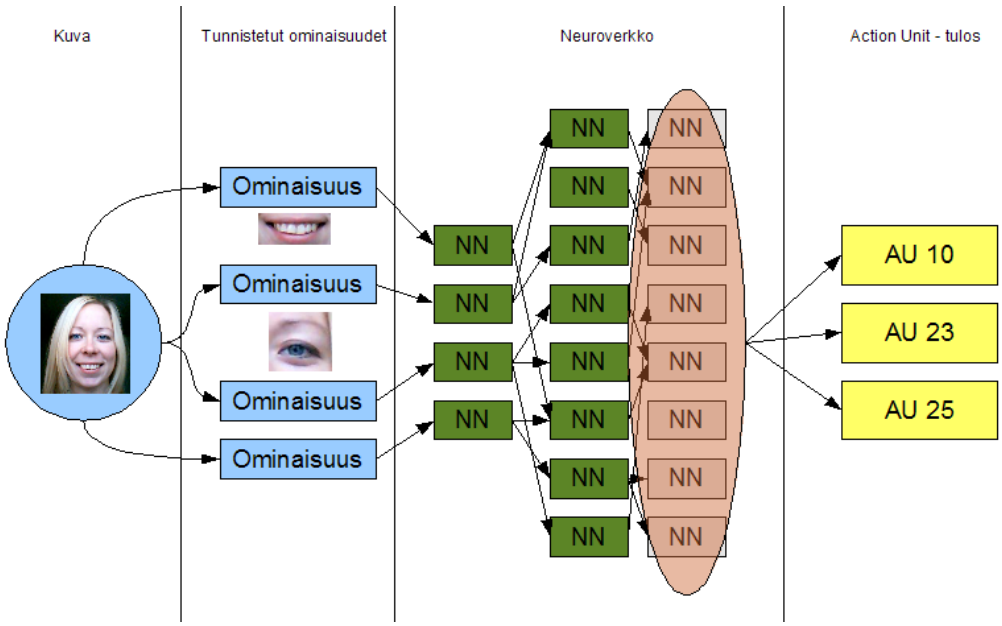
Malliin perustuva tunnistus voidaan toteuttaa myös mahdollisuuksien verkko (potential net) -tekniikalla jossa mahdollisista rakenteista muodostetaan verkko. Esimerkiksi silmien, nenän ja suun välille rakennetussa verkossa verkon eri osilla on erilainen painotus. Esimerkiksi suu voidaan painottaa vähemmän merkitseväksi koska sen tunnistuksessa tapahtuu eniten virheitä (Pantic, Rothkrantz 2000).

3.4. Neuroverkkoihin perustuva tunnistus

Neuroverkot ovat informaation käsittelyn matematiikan tai laskennan malleja, jotka perustuvat yhdistävään laskentaan. Niiden toiminta perustuu fyysisten aivojen neuronien verkottuneen rakenteen jäljittelyyn, eli neuroverkot pyrkivät tekemään tietojen käsittelyä kuten aivojen neuronit.

”Siinä kun tavallisissa asiantuntijajärjestelmissä käytetään "jos-niin"-sääntöpareja (jos raidat, niin seepra; jos pitkät korvat, niin aasi), neuroverkkoa opetetetaan esimerkkien avulla (nämä ovat eri-ikäisiä seeproja, nämä aaseja). Pyritään siihen että neuroverkko oppii muuttujien epälineaariset riippuvuussuhteet suoraan havaintoaineistosta (kavioeläinesimerkissämme oppii tarkastelemaan korvia ja värin kuvioita, ei esim. jalkojen pituutta).” (Wikipedia, Neuroverkot)

Neuroverkkoihin perustuvassa tunnistamisessa kasvoista muodostetaan neuroverkkoon sopiva matriisi. Tämä syötetään neuroverkolle ja neuroverkolta saadaan esimerkiksi Ekmanin FACS–mallin mukainen ristikko eri toimintayksiköistä (AU), jonka avulla tunne voidaan tulkita.



Kuva 3. Kuvasta tunnistettujen ominaisuuksien siirtyminen neuroverkon läpi. Neuroverkkojen tasoja voi olla useampia, kuvassa tasoja on vähennetty.

4. Teknologioiden esittely

Tässä luvussa esitellään muutamia algoritmeja ja tekniikoita, joita on käytetty toteutetuissa hahmontunnistuksissa. Tarkoituksena ei ole luetella kaikkia mahdollisuuksia, vaan antaa kuva siitä, minkälaisin teknologisin menetelmin tunnistus on mahdollista toteuttaa.

4.1. Histogram equalization

Histogrammin tasaus (Histrogram equalization) on yleinen menetelmä, jota käytetään ennen kuvatunnistuksen aloittamista. Sen tehtävänä on tasata värien voimakkuuksia kuvassa. Menetelmän käyttö parantaa kuvan kontrasteja ja mahdollistaa näin tarkemman tunnistuksen. Jos esimerkiksi hahmontunnistus perustuu kontrastimuutoksiin, niin tätä menetelmää käyttämällä jokaisesta kuvasta tulee yhtä vahva värien kontrastien osalta ja tasavertainen muiden tunnistettavien kuvien kanssa. (Wikipedia, Histogram_equalization)

4.2. Active Contour Models

Tätä menetelmää kutsutaan toisella nimellä ”snakes”. ”Active Contour Models” - menetelmän ovat kehittäneet Kass ja muut vuonna 1987. Menetelmä perustuu Newtonin elastisen teorian, mekaniikan lakeihin ja muotojen realismiin. Tällä tarkoitetaan siis sitä, että esimerkiksi kasvojen piirteet noudattavat tiettyjä luonnollisia muotoja, joita voidaan mallintaa näitä sääntöjä noudattavilla ääriviivoilla. Menetelmä pystyy havaitsemaan hyvin vähäkontrastisia kuvasta ääriviivoja koska se voi olettaa muodon jatkuvan Newtonin lakien mukaisesti, kun taas perinteinen reunantunnistus etsii täysin sokkona kontrastinmuutoksia.

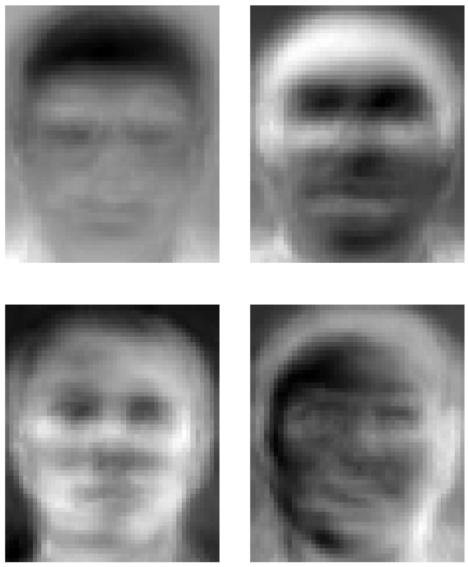
4.3. Canny Edge Detector

Canny Edge Detector –menetelmä perustuu kontrastimuutokseen. Sen tuloksena on kontrastimuutokset, eli todennäköiset reunat eri kasvoalueiden välillä. Menetelmä on monen mielestä paras reunantunnistusalgoritmi. Menetelmä käyttää muuntautuvaa kynnystä kontrastimuutokselle, ja ehkä tämä tekee siitä tehokkaamman kuin muista kontrastiin perustuvista menetelmistä. (Neoh & Hazanchuk, 2005)

4.4. Eigenfaces

Maallikon termein kuvattuna eigenfaces on sarja standardoituja kasvojen ”sisältöjä” jotka on muodostettu statistiikan analyysillä. Sarja muodostetaan monen kuvan perusteella. Jokaisen ihmisen kasvot ovat yhdistelmä näitä standardoituja kasvoja. Yhden henkilön kasvot voi olla muodostuneet niin että niissä on 10% kasvoista 1, 24% kasvoista 2 ja niin edelleen. Tämä tarkoittaa sitä, että jos tahdotaan tallentaa henkilön kasvot kasvontunnistusta varten, ohjelmat vaativat vähemmän muistia kuin digitoidut valokuvat.
(Wikipedia, Eigenfaces)

Eigenfaces on siis tekniikka jolla kasvot pystytään mallintamaan, niin että niiden rakenne ja muoto tallentuu hyvin. Eigenfaces-rakenteesta voidaan piirtää kasvokuva, josta henkilö on melko hyvin jopa tunnistettavissa, mutta se ei kuitenkaan vastaa alkuperäistä kuvaa.



Kuva 3. Eigenfaces-rakenne piirrettynä takaisin kuvaksi (Wikipedia, Eigenfaces)

4.5. Dynaamisen mallin ja liike-energian yhdistelmä

Essa ja Pentland (1995) ovat lähestyneet tunnistusta muista poikkeavalla tavalla. He tutkivat kuvasta liike-energiaa ja muodostavat tämän avulla fyysisen mallin kasvolihaksista. He myöskin käyttävät kahta erilaista menetelmää rinnakkain: ensimmäinen menetelmä perustuu fyysiseen kasvomalliin, joka perustuu lihasten arvioituun liikkeeseen. Toinen menetelmä muodostaa mallin, joka muodostaa fyysisen mallin pohjalta toisen mallin pohjautuen liike-energiaan. Menetelmien lopputulos yhdistetään ja yhdistämisen avulla saadaan eri kasvojen osien tiedot tarkemmin selville.

Essa ja Pentland (1995) ovat kummallakin menetelmällään päässeet 98 prosenttiin tunnistusvarmuudessa materiaalilla jota he kutsuvat ”riippumattomaksi” (”independet test database”). Epäselväksi jää, kuinka hyvin tämä malli toimisi tuntemattomassa materiaalissa. Tutkimuksessaan he kuitenkin ilmoittavat, että liike-energiaan perustuva tunnistus kuitenkin häiriintyy jos koehenkilö liikuttaa kasvojaan nopeasti.

4.6. Optisen virtauksen menetelmä

Hahmontunnistus-osiossa useissa toteutetussa kasvojen tunnistusmenetelmissä on käytetty optisen virtauksen menetelmää (optical flow) jossa kuvasta etsitään alueita, joissa kuvan valoisuus muuttuu pehmeästi. Kuvasta saadaan muodostettua vektoreita, joilla on pituus ja suunta. Näistä vektoreista voidaan käyttää esimerkiksi vain pystysuuntaista tietoa, jonka perusteella etsitään muutos neutraalista ilmeestä eleeseen. (Berthold & Horn 1980)

5. Ongelmat

5.1. Menetelmien testausten heikkoudet

Tämä tutkimus ei kata kaikkia maailmalla tutkittuja toteutuksia, mutta niissä tutkimuksissa, joihin tässä tutkimuksessa tutustuttiin, oli yksi toistuva heikkous. Suurimmassa osassa tutkimuksista rajoitteena tunnistukselle on käyttäjän silmälasit sekä parta. Koska tämä aiheuttaa todellisissa toteutuksissa kriittisen ongelman, ovat tutkimusten lähtökohdat vajaita. Tämän tutkielman alussa todettiin yli puolen väestöstä käyttävän silmälaseja. Tämä tarkoittaisi sitä että menetelmät eivät toimisi kuin pienemmässä osassa väestöstä.

Osassa tutkimuksista jopa kasvojen asento sekä liikehdintä haittasivat tunnistusta. Yleinen tutkimusten kliinisyyden vaatimus on hämmästyttävää. Lähtökohta uusille tutkimuksille ei voi olla se, että tehdään tunnistusmenetelmä, joka tunnistaa kasvot laboratorio-olosuhteissa otetuista valokuvista tai videosta, vaan tunnistusmenetelmä pitäisi suunnitella oikeaan ympäristöön. Nykyisiä tunnistusmenetelmiä, etenkin niitä, jotka ovat herkimpiä silmälaseille tai pään liikkeille, ei voida järkevästi muuttaa toimimaan ns. todellisissa ympäristöissä. Tässä vaiheessa, kun tietokonepohjainen tunnistus olisi mahdollista ja jopa arkipäivää, suunnittelulähtökohtien ja päämäärän pitäisi ohjata varsinaista menetelmätutkimusta eikä toisinpäin.

Tutkimuksissa on käytetty usein samaa materiaalia lopullisessa testauksessa kuin itse kehittämisessä tai neuroverkon opettamisessa. Mielestäni tällaiset tutkimukset eivät ole luotettavia, eivätkä anna realistista kuvaa menetelmien toimivuudesta. Esimerkiksi Yoneyama ja muut (1997) toteuttivat eleiden tunnistamisen neuroverkkojen avulla ja saavuttivat melko korkean tunnistusprosentin – 92 prosenttia eleistä pystyttiin tunnistamaan. Tutkimuksessa käytettiin samaa kuvasarjaa neuroverkon opettamisessa ja testaamisessa. Mielestäni etenkin neuroverkkojen kohdalla tämä on väärä menetelmä, koska neuroverkot ovat opetuksen takia sopeutuneet juuri näihin kuviin. Lopullisessa testauksessa tulisi käyttää kuvia tai henkilöitä, jotka eivät ole osallistuneet järjestelmän kehittämiseen.

5.2. Humaanit ongelmat

Panticin ja Rothkrantzin (2000) mukaan ei-verbaalinen viestintä on tilannekohtaista. Myös konteksti, jossa ele tapahtuu, on hankalaa tunnistaa automaattisesti. Nämä molemmat seikat vaikuttavat eleen tulkintaan.

Psykologian tutkijat ovat tulleet siihen tulokseen, että kasvojen eleet eivät ole universaaleja (Garret & McKenzie 1993). Lisäksi joissain kulttuureissa tiettyjä eleitä ei edes näytetä tai eleet ovat hyvin pieniä. Uusissa toteutettavissa menetelmissä olisi myös otettava huomioon etninen tausta sekä sukupuoli – uusia menetelmiä olisi testattava testisarjalla, johon kuuluu kaikkien etnisten ryhmien ja eri sukupuolten edustajia.

Kahden ihmisen kasvokkain tapahtuvassa kommunikaatiossa virhe korjataan käyttämällä tietoa toisesta viestimestä. Esimerkiksi jos havaittavan kasvoja ei nähdä, voidaan

silti eleitä lukea vartalon eleistä. Jos osa kasvoista on peitetty esimerkiksi kädellä, ihmisen aivot osaavat täyttää puuttuvan informaation. (Pantic & Rothkrantz 2000) Mikä merkitys on sillä että tuntee henkilön erittäin hyvin? Tarvitseeko silloin edes nähdä kasvoja tai muita vartalon eleitä tietääkseen miten toinen henkilö reagoi omaan viestiin?

Ihminen myös elehtii harvoin ”puhtaasti”. Suurin osa ihmisen eleistä on sekoituksia useasta eleestä. Tämän takia järjestelmän ei ole järkevää kertoa yhtä tunneilmaisua vaan useampi tunneilmaisu joihin kyseinen ilmaisu sopii. (Pantic & Rothkrantz 2000)

Ongelmaksi voi myös muodostua taustainformaation puute. Tätä ovat pohtineet Carroll ja Russell (1996). Esimerkki: henkilö antaa lahjan henkilölle jonka eleitä tarkkaillaan. Henkilö järkyttyy lahjan saamisesta. Miksi? Eikö lahjan saaminen ole aina mukavaa? Jos kuvitellaan sama tilanne tietokone-ihminen kommunikaatiossa: tietokone päättää piristää käyttäjäänsä ja antaa tälle virtuaalisen lahjan. Käyttäjä järkyttyy. Käyttäjän oletettua vastakkaiset reaktiot hämmentävät taatusti tietokoneen, joka oli valmis mittaamaan onnellisuutta käyttäjän kasvoilta. Puuttuu siis oleellinen tieto: taustainformaatio. Ensimmäisessä esimerkissä tarkkailtava henkilö on aviomies, joka ei tiedä olevansa jo isä. Lahjan antaja vaimo joka ojentaa ”Vauvan käyttöohjeet” -kirjan. Nyt järkytys on helpommin ymmärrettävä reaktio.

6. Yhteenveto

Tietokonepohjainen tunneilmaisuiden tunnistaminen kasvokuvista on monipuolinen tutkimusalue, sillä täysin varmasti tunnistavaa menetelmää ei ole vielä löydetty. Tekniset lähestymistavat vaihtelevat laajasti ja erilaisten hahmontunnistusalgoritmien skaala on laaja.

Testejä vaivaa kuitenkin kliininen lähestyminen: parran, silmälasien olemassaolo sekä kasvojen nopea liikehdintä häiritsee tunnistusalgoritmeja. Menetelmiä kehittäessä tulisi käyttää eri valokuva- tai videomateriaalia kuin varsinaisissa testeissä, etenkin neuroverkkoihin pohjautuvien menetelmien kohdalla.

Tunnistaminen saattaa olla vaikeaa staattisista valokuvista, sillä kasvot saattavat olla epäsuotuisasti asemoitu valaistusolosuhteiltaan. Videokuvan käyttö yhden valokuvan sijaan voisi tuoda ratkaisun tähän ongelmaan. Lisäksi hahmontunnistus sekä neuroverkko, joka ei häiriintyisi esimerkiksi peitetyistä kasvojen osista, voisi saavuttaa hyvän onnistumisprosentin.

Uskon että parhaan tuloksen saa käyttämällä esimerkiksi Active Contour Models –algoritmia ja tähän mahdollisimman hyvin soveltuvaa neuroverkkoa.

Viiteluettelo

James M. Carroll, James A. Russell (1996). Do facial expressions signal specific emotions? Judging emotion from the face in context. *Journal of Personality and Social Psychology,* 70, 2, 205-218.

Irfan A. Essa, Alex Pentland (1995). Facial Expression Recognition Using a Dynamic Model and Motion Energy, *In: Proceedings of the Fifth International Conference on Computer Vision,* 360-367.

Yulia Gizatdinova, Veikko Surakka (2006), Feature-Based Detection of Facial Landmarks from Neutral and Expressive Facial Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28, 1, 135-139.

Berthold Horn (1980). Determining optical flow. Massachusetts Institute of Technology, Artificial Intelligence Laboratory.

M. Kass, A. Witkin, D. Terzopoulos (1987). Snakes: Active contour models, *In: Proc. Int'l Conf. Computer Vision*, ICCV87,London.

Garrett D. Kearney, Sati McKenzie (1993). Machine interpretation of emotion: Design of a memory-based expert system for interpreting facial expressions in terms of signaled emotions. *Cognitive Science: A Multidisciplinary Journal,* 17, 4, 589-622.

Hong Shan Neoh, Asher Hazanchuk (2005). Adaptive Edge Detection for Real-Time Video Processing using FPGAs. Altera Corporation.

Athanasios Nikolaidis, Ioannis Pitas (1998). Facial feature extraction and determination of pose. *Pattern Recognition,* 33, 1783-1791.

Maja Pantic, Leon J. M. Rothkrantz (2000). Automatic Analysis of Facial Expressions: The State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 1424-1445.

Taloustutkimus Oy (2004). Omnibus – Silmälasien käyttö. Optisen Alan Tiedotuskeskus.

Jia-Jun Wong, Siu-Yeung Cho (2006). Facial Emotion Recognition by Adaptive Processing of Tree Structures. Nanyang Technological University.

Wikipedia, 2006. Wikipedia – Vapaa tietosanakirja, eigenfaces – määrittely (15.12.2006).

M. Yoneyama, Y. Iwano, A. Ohtake, K. Shirai (1997), "Facial Expressions Recognition Using Discrete Hopfield Neural Network," icip, 117, 1997 *International Conference on Image Processing (ICIP'97)* , 1.

HDR-kuvan luominen valokuvia yhdistämällä

Eero Hietaranta

Tiivistelmä

Tässä tutkimuksessa on tutkittu usean samasta kohteesta eri valotusarvoilla otetun digitaalisen valokuvan yhdistämistä yhdeksi HDR-kuvaksi. Lyhenne HDR tulee sanoista *High Dynamic Range*. Kun useimmissa digitaalisissa kuvissa tallennetaan kuvapisteiden kirkkauksia mustasta valkoiseen, niin HDR-kuvaan tallennetaan mustaa tummempia ja valkoista kirkkaampia sävyjä. Mittavamman kuvainformaation ansiosta HDR-kuvan kirkkauttaa voidaan jälkikäteen muuttaa enemmän kuin tavallisessa valokuvassa.

Avainsanat: HDR-kuva, digitaalinen, valokuva, yhdistäminen
CR-luokat: I.3

1 Johdanto

Kun tavallisella digitaalisella kameralla otetaan valokuva, ei kuvaan tallennu paljoakaan enempää kuin se, mitä silmä voi lopullisesta kuvasta havaita. Tämä koskee erityisesti kuvan kirkkautta, eikä sitä voi jälkikäteen muuttaa paljoakaan kontrastin kärsimättä. Tämän voi havaita säätämällä kuvan kirkkautta tavanomaisissa kuvankäsittelyohjelmissa. Asia tulee toisinaan esille myös digitaalisia kuvia käsittelevissä julkaisuissa. Joihinkin tarkoituksiin tavallisen kuvan arvoalue ei ole riittävä. Ongelmaa voidaan kiertää käyttämällä HDR-kuvaa. HDR-kuva on kuva, johon tallentuu merkittävästi enemmän kirkkausasteita kuin tavallisiin digitaalisiin valokuviin. Eri lähteet esittävät asian hieman eri sanoin. Tämä määritelmä mukailee Wikipediassa [6] ja openexr.com:issa [3] esitettyjä.

Samainen Wikipedian artikkeli toteaa HDR-kuvia käytetyn alunperin tie-

tokoneella luodussa grafiikassa, josta niiden käyttö sittemmin levinnyt valokuviinkin [6]. Siitä, miten HDR-kuvia hankitaan ammattikäytössä, kuten elokuvateollisuudessa, on hieman vaikea saada tarkkaa tietoa. Joillain verkkosivustoilla, kuten esimerkiksi openexr.com:issa [3] annetaan ymmärtää, että negatiivifilmiä ainakin käytetään. Myös tähän tarkoitukseen tehdyistä laitteista on joitain mainintoja.

Tavallinen digikameran omistava harrastelija voi luoda tällaisia kuvia itse yhdistämällä eri valotuksilla otettuja tavallisia valokuvia. Tämä menettely asettaa kuitenkin tiettyjä rajoitteita. Kun generoidaan HDR-kuva yhdistämällä useita valokuvia, on tärkeää, että kuvien rajaus on sama. Kuvissa, joiden resoluutio on pieni, jo muutaman pikselin ero rajauksessa voi aiheuttaa näkyvää heikkenemistä kuvan terävyydessä. Kuvan luominen useasta kuvasta sulkee myös lähes väistämättä liikkuvat kohteet mahdollisten kuvauskohteiden ulkopuolelle.

Toisaalta useista valokuvista koottu HDR-kuva sallii erittäin suuren dynaamisen arvoalueen. Valokuviahan voi ottaa liikkumattomasta kohteesta vaikka kuinka monta. Lähinnä käytössä oleva kamera asettaa rajat, joskin on tietysti mahdollista, että myös kuvaformaatin rajat tulevat vastaan.

Tämä tutkimus tarkastelee, kuinka useita valokuvia voidaan yhdistää yhdeksi HDR-kuvaksi. Tutkimus painottuu vahvasti käytännön kokeiluihin ohjelmoimalla.

2 Taustatietoja

HDR ja HRDI (*High Dynamic Range Imaging*) ovat hieman hankalia lyhenteitä suomentaa. En ole kuullut minkäänlaisesta suomennoksesta, saati sitten vakiintuneesta sellaisesta, näille lyhenteille. Tyydyn tässä yhteydessä käyttämään suomennosta *suuri dynaaminen arvoalue*.

Mainitessani tässä tutkielmassa digitaalisen kuvan 8- tai 16-bittisyydestä tarkoitan tällä värikanavan bittisyvyyttä. Käytäntö on käsittääkseni melko yleinen.

Lisäksi käytän tässä tekstissä joitain käsitteitä, jotka eivät ehkä ole kovin laajasti tunnettuja. Allaolevien käsitteiden tarkemmat määritelmät voi löytää kirjallisuudesta ja monilta verkkosivuilta.

2.1 Käsitteet

- **Aukko**
  Aukolla, englanniksi *f-stop*, voitaneen katsoa tarkoitettavan valokuvauksessa tietyissä yhteyksissä valon voimakkuuden kaksinkertaista suhdetta.

- **Valotusarvo**
  Valotusarvolla voitaneen katsoa tarkoitettavan jotain määrättyä altistusta valolle. Englanninkielisessä kirjallisuudessa käytetään termiä *Exposure Value*.

- **Värikanava**
  Värillisessä digitaalisessa kuvassa pikseli, eli kuvapiste, koostuu tyypillisesti kolmesta väristä, usein punaisesta, vihreästä ja sinisestä. Jokainen esitetään lukuarvona, jota kutsutaan värikanavaksi.

3 Tutkimusvaiheet

Kuvien yhdistämisestä näyttäisi olevan saatavilla niukasti kirjallista aineistoa. Aihepiirin keskeiset käsitteet on kuitenkin kuvattu kattavasti verkossa saatavilla olevassa materiaalissa.

Tavallisten valokuvaformaattien rakenteesta löytyy verkosta materiaalia runsaasti. Myöskin HDR-kuvista, ja niissä käytetyistä tavoista tallentaa dataa löytyy verkossa julkaistua materiaalia. Kattava lähde pienille tiedonmurusille tämän tutkimuksen aiheen osalta on Wikipedia. Muu löytynyt kirjallinen aineisto käsittelee erinäisiä aiheita sivuten paikoin samoja yksityiskohtia kuin tämä tutkimus.

3.1 Lähdekuvien hankinta

Tutkimuksessa käytetyt lähdekuvat valokuvattiin digitaalikameralla tätä tarkoitusta varten. Kuvat otettiin niin sanottuun raw-muotoon, josta ne muunnettiin 16-bittisiksi lineaarisiksi ppm-kuviksi Dcraw-ohjelmaa käyttämällä. Nämä muunnetiin edelleen 8-bittisiksi ppm-kuviksi Image Magic ja GIMP-ohjelmilla.

Digitaalisten kameroiden tuottamat jpeg-kuvat ovat gamma-korjattuja mitä luultavimmin arvolla $2,2$. Gamma-arvo $2,2$ on sRGB-standardi [4]. Tietääkseni useimmat digitaalikamerat käyttävät oletusarvoisesti sRGB-väriavaruutta. Tällainen kuva voidaan muuntaa 8-bittiseksi lineaariseksi ppm-kuvaksi Image Magic -ohjelmalla soveltamalla ensin gamma-arvoa $0,45$, mikä on likimäärin sama kuin $1/2,2$, ja tallentamalla kuva ppm-muotoon.

3.2 Ohjelmointi

Käytännön kokeiluna tutkimuksessa koodattiin pienimuotoinen ohjelma, joka pystyy yhdistelemään valokuvia yhdeksi HDR-kuvaksi.

Ohjelmointikieleksi oli valittiin C. Kirjastoina käytettiin C:n standardikirjastoa sekä SDL:ää. Ohjelma on koodattiin GNU/Linux-ympäristössä. Kielen valinnan ensisijaisena perusteena oli tehokkuus.

4 Ohjelman toiminta

4.1 Tallennusformaatti

Digitaalisessa kuvassa pikselin, tai sen värikavavan, kirkkaus ilmaistaan lukuarvona. Monissa tavanomaisissa kuvaformaateissa käytetään 8-bittisiä kavavia, mikä sallii 256-porrasta välillä 0–255. Tyypillisesti arvo 0 tarkoittaa mustaa ja 255 valkoista, mikä näyttäsi olevan ainakin kuvankäsittelyohjelmissa yleinen esitystapa. Kanavan bittimäärästä ja maksimiarvosta riippumatta nollan ja maksimin väliin jäävät arvot on mahdollista tulkita usealla eri tavalla, joista seuraavassa esitetyt tulevat usein esille alan kirjallisuudessa.

Kenties yksinkertaisin tapa tulkita arvot on lineaarinen tulkinta, missä lukuarvo on suoraan verrannollinen värikanavan kirkkauteen. Tämä siis tarkoittaa, että kavava, jonka arvo on 150, on kaksi kertaa niin kirkas kuin kanava, jonka arvo on 75. Tästä seuraa, että lukuarvojen ollessa pieniä, kaksinkertaisen kirkkauseron ilmaisemiseen käytettävät luvut ovat absoluuttisesti lähellä toisiaan. Jos kirkkausarvot ilmaistaan kokonaislukuina, jää yhden aukon välille vain muutama porras väliin jäävien arvojen ilmaisemiseen.

Värikanavien kirkkaudet voidaan myös tallentaa logaritmisesti, jolloin tietty absoluuttinen muutos kanavan arvossa tarkoittaa kirkkauden muuttumista jollain tietyllä kertoimella. Gregory Ward Larson [2] painottaa logaritmisen tallennuksen hyötyjä ja esittää myös kaavoja kuvadatan muuntamiseen logaritmiseen muotoon ja takaisin käyttäen 2-kantaista logaritmia. Jos lineaarinen kuvadata on tallennettu kokonaislukuina, on värikanavan maksimiarvo luvun 2 potenssi, joten 2-kantaisen logaritmin pitäisi soveltua hyvin kuvadatan muuntamiseen logaritmiselle asteikolle.

Kun kuvia esitetään ruudulla, on kuvadata usein gamma-korjatulla asteikolla. Kuvassa itsessään saattaa olla esimerkiksi sRGB-väriavaruuden gamma-arvo $2,2$ [4, 5]. Gamma-korjaus voi olla myös laitteistotasolla [1]. Näyttölaitteen gamma noudattaa funktiota $I = K * v^g$ [4]. Gamma-korjaus voidaan vastaavasti suorittaan funktiolla $I = K * v^{1/g}$. Gamma-korjauksen funktio on esitetty tämänkaltaisessa muodossa CGSD:n verkkosivuilla [1]. Näissä funktioissa I on gamma-muunnettu arvo, v pikselin tai värikanavan arvo ja g on gamma. K on vakio. Wikipedia [5] esittää tämän kaavan ilman vakiota K, mikä on ehkä järkevää. Ward [4] toteaa, että K on I:n maksimiarvo, jos arvot on normalisoitu välille 0–1. Kumpi tahansa muoto kaavoista on yksistäänkin riittävä, sillä gamma-arvosta voidaan aina laskea käänteisluku. Esimerkiksi $1/2,2 \approx 0,45$. Useimmissa lähteissäkään jälkimmäistä muotoa ei tuoda erikseen esille.

Tutkimuksen yhteydessä tehty ohjelma käyttää värikananvien tallentamiseen 16-bittisiä kokonaislukuja. Valinta perustuu olettamukseen, että 8 bittiä ei ole riittävä tarkkuus värikanavan tallentamiseen, jos dynaaminen arvoa-

lue on suuri. Kahdeksanbittisten kuvien rajoittuneisuus tuodaan usein esille digitaalisia kuvia käsittelevässä kirjallisuudessa. Päätös käyttää kokonaislukuja perustuu siihen tosiasiaan, että valittu ohjelmointiympäristö ei tarjoa muuta 16-bittistä perustietotyyppiä.

Kokonaisluvun käytön johdosta data on järkevää tallentaa logaritmisesti. Ymmärtääkseni 16-bittisiä värikanavia pidetään yleisesti riittävinä tavallisen valokuvan tallentamiseen lineaarisesti, mutta lienee epätodennäköistä että lineaarinen esitys riittäisi kovin suurten kirkkauserojen tallentamiseen. Ohjelma käyttää 2-kantaista logaritmia koska se soveltuu hyvin tarkoitukseen ja tutkimuksessa käytetty ohjelmointiympäristö tarjoaa sen kirjastofunktiona.

4.2 Kuvien yhdistäminen

Kun samasta kohteesta otetaan useita valokuvia eri valotusarvoilla, pikselit toistuvat kuvissa eri kirkkauksilla. Jotta kuvat voitaisiin yhdistää, on kuvien data sovitettava samalle asteikolle.

Tässä tutkimuksessa koodattu ohjelma yhdistää usean samaa kohdetta esittävän, yhden aukon välein valotetun ja lineaarisesta, 8-bittisestä kuvadatasta koostuvan ppm-kuvan yhdeksi HDR-kuvaksi. Lähdekuvien formaatin valinta perustuu helppoon käsiteltävyyteen. Ppm-formaatti on todella yksinkertainen [7]. Näin ollen tiedostojen lukeminen on helppo koodata itse, minkä johdosta ohjelmaan ei tarvitse linkittää erillistä kirjastoa tätä varten. Käytettäessä 8-bittisiä värikanavia ei ole tarpeen huolehtia tavujärjestyksestä lähdekuvien tiedostoissa.

Data muunnetaan logaritmiselle asteikolle käyttämällä kaavaa

$$V' = 256 * \log_2(V * 2^i), \tag{1}$$

missä V on alkuperäinen 8-bittisen kanavan arvo. i on nollasta alkava iteraatiomuuttuja, kun kuvia käydään läpi kirkkaimmasta himmeimpään. Kaavan on pohjimmiltaan samanlainen kuin Larsonin esittämät [2], mutta yksinkertaisempi. Ohjelman käyttämään kaavaan on lisätty arvon kertominen 2^i:llä, jotta eri kuvista luettu data saadaan muunnettua samalle asteikolle,

mutta se ei suoranaisesti liity datan muuntamiseen logaritmiseksi. Varsinainen muunnos voitaisiin esittää muodossa $V' = 256 * \log_2 V$. Käytännössä kaavaa (1) ei ole kirjoitettu ohjelmaan sellaisenaan, vaan kahdessa osassa. Tämä on esitetty pseudokoodissa 1.

Kuvista huomioidaan värikanavien arvot, jotka ovat välillä 55–220, lukuunottamatta kirkkainta ja himmeintä kuvaa, joista huomioidaan enemmän. Arvojen 55 ja 220 kirkkausero on lineaarisessa esitystavassa kaksi aukkoa, ja kun ohjelma edellyyttää kuvia, joiden valotusten ero on yksi aukko, seuraa, että kahdesta peräkkäin luetusta kuvasta menee yhden aukon verran arvoja päällekkäin. Käytäessä kuvia läpi kirkkaimmasta himmeimpään, järjestyksessä n olevan kuvan arvoalue 110–220 vastaa kuvan $n + 1$ arvoaluetta 55–110. Kaavassa (1) eri kuvien arvot skaalataan samaan suuruusluokkaan kertomalla arvo kuvan järjestynumeron perusteella $V * 2^i$. Yhdistettäessa kuvia näin useimpien kuvapisteiden kohdalla lopulliseen kuvaan tallennettava arvo pohjautuu kahteen arvoon. Näissä tilanteissa lasketaan keskiarvo. Kirkkaimmasta kuvasta huomioidaan arvot väliltä 0–220 ja himmeimmästä kuvasta väliltä 55–255.

Kirkkaimman kuvan yhteydessä saattaa tulla vastaan tilanne, jossa logaritmifunktiolle syötetään arvo nolla. Tunnetusti logaritmifunktioita ei ole määritelty kuin nollaa suuremmille reaaliluvuille. Käytetyssä ohjelmointiympäristössä tämä tilanne aiheuttaa virheen, mutta sitä ei ollut tutkimuksen yhteydessä tarpeen huomioida. Kirjastofunktion palauttama arvo, negatiivinen ääretön, muuntuu nollaksi, kun suoritetaan tyyppimuunnos liukuluvusta etumerkittömään kokonaislukuun.

Pseudokoodi 1 esittää kuvien yhdistämisen ja datan muuntamisen logaritmiasteikolle. Taulukko *tmpdata* sisältää kuvien yhdistämisessä syntyvän kuvainformaation, taulukkoon *data* tallennetaan logaritminen kuvainformaatio ja taulukko *nvalues* sisältää tiedon siitä, kuinka monta arvoa vastaavaan *tmpdata*n alkioon on summattu.

Pseudokoodi 1 tmpdata[ ] data[ ] $nvalues$[ ]

```
1: for i ← 0; i < n; i ← i + 1 do
2:   for j ← 0; j < s; j ← j + 1 do
3:     value ← getValue()
4:     if i = 0 and value < 220 then
5:       tmpdata[j] ← tmpdata[j] + value ∗ 2^i
6:       nvalues[j] ← nvalues[j] + 1;
7:     end if
8:     if i > 0 and i < n − 1 and value > 55 and value < 220 then
9:       tmpdata[j] ← tmpdata[j] + value ∗ 2^i
10:      nvalues[j] ← nvalues[j] + 1;
11:    end if
12:    if i = n − 1 and value > 55 then
13:      tmpdata[j] ← tmpdata[j] + value ∗ 2^i
14:      nvalues[j] ← nvalues[j] + 1;
15:    end if
16:  end for
17: end for
18: for i ← 0; i < s; i ← i + 1 do
19:   if nvalues[i] > 0 then
20:     tmpdata[i] ← tmpdata[i]/nvalues[i]
21:   end if
22: end for
23: for i ← 0, j ← 0; i < s; i ← i + 1 do
24:   data[i] = 256 ∗ log_2(tmpdata[j])
25:   j ← j + 1
26: end for
```

4.3 Kuvan tallentaminen

Ohjelma ei käytä mitään valmista formaattia HDR-kuvan tallentamiseen. Kuvasta tallennetaan 8 tavun mittainen otsikkodata, joka sisältää olennaiset tiedot kuvan lukemiseen, jonka lisäksi varsinainen kuvainformaatio tallennetaan raakadatana.

Kuvainformaatio muunnetaan 2-kantaiselle logaritmiasteikolle. 8-bittisen valokuvan värikanavan arvoalue väliltä 0–255 muuntuu näin ollen välille $0 \leq V < 8$ ($\log_2 256 = 8$), olettaen että $log_2(0)$ muunnetaan nollaksi. Ohjelmassa lukuarvot on käsiteltävä tässä vaiheessa liukulukuina, koska kokonaislukuina näin pienen arvoalueen tarkkuus ei riitä tarkoitukseen. Arvot kerrotaan vakiolla 256, jolloin ne muuntuvat välille 0-2048. Useita kuvia käsiteltäessä lineaarisen arvoalueen maksimiarvo (2^{n-1}) määräytyy kuvien lukumäärästä n. Tämä siis olettaen, että kuvat on valotettu yhden aukon välein toisistaan. Kun tästä otetaan 2-kantainen logaritmi, kasvattaa jokainen kuva logaritmista arvoaluetta yhdellä, ($\log_2 512 = 9$),($\log_2 1024 = 10$) jne. Kun tämä asteikko kerrotaan 256:lla on helppoa havaita, kuinka yhtä aukkoa kohti tallentuu 256 porrasta arvoja.

4.4 Kuvan esittäminen

Ohjelma muuntaa tallennetun kuvan esitettävään muotoon prosessilla, joka on pitkälti käänteinen tallennukseen nähden. Lisäksi esitettävä kuvainformaatio. gamma-korjataan esittämistä varten.

Muunnettaessa kuvaa tallennetusta muodosta ruudulla esitettavään muotoon, jaetaan luetut arvot ensiksi vakiolla 256, minkä jälkeen luku 2 korotetaan tähän lukemaan kaavan

$$V = 2^{V'/256} \tag{2}$$

mukaisesti. Tämän jälkeen arvolle suoritetaan gamma-korjaus kaavalla

$$I = 255 * (V/255)^g. \tag{3}$$

Ohjelman gammakorjaukseen käyttämä kaava on pienehkö muunnos jo aiemmin tässä tekstissä esille tuodusta, viitekirjallisuudessa [4, 5]. esitetystä kaavasta. Siihen on lisätty normalisointi asteikolta 0–255 asteikolle 0–1. Muunnoksessa takaisin asteikolle 0–255 kerroin 255 on Wardin [4] esittämän kaavan vakio K, joka tässä yhteydessä on V:n maksimiarvo.

Pseudokoodi 2 esittää, kuinka muunnos logaritmiasteikolta ruudulla esitettävään muotoon tapahtuu ohjelmassa. Muuttuja zl on nollataso, joka tulkitaan mustaksi. Valkoiseksi tulkitaan arvo zl+8. Arvoalueesta zl–(zl+8) tehdään muunnos välille 0–8, minkä jälkeen sovelletaan kaavoja (2) ja (3).

Pseudokoodi 2 value zl

1: $fv \leftarrow value/256$
2: **if** $fv < zl$ **then**
3: **return** 0
4: **else if** $fv > zl + 8)$ **then**
5: **return** 255
6: **end if**
7: $v \leftarrow 2^{fv-zl}$
8: **return** $255.0 * (v/255.0)^g$

---

5 Yhteenveto

Tutkimuksen mittakaava on pieni, ja ohjelman toteutuksessa monet ratkaisut on tehty jonkinasteisen pakon sanelemina, tai niissä on pyritty mahdollisimman yksinkertaiseen ja helposti ymmärrettävään ratkaisuun. On hyvinkin todennäköistä, että yhdistettävien kuvien sovittaminen samalle asteikolle voitaisiin tehdä muillakin tavoin kuin tässä tutkimuksessa. Varmaa on, että

kuvadatan tallennukseen voisi käyttää hyvinkin erilaisia menettelyjä. Tämä selviää jo lukemalla eri HDR-kuvaformaattien dokumentaatioita.

Mittavampi kirjallinen aineisto olisi ehkä auttanut joidenkin ratkaisujen kanssa. Kuvien yhdistämisestä saattaa hyvinkin olla enemmän kirjallisuutta kuin mitä tämän tutkimuksen tarpeisiin löytynyt määrä antaa olettaa.

Ohjelmassa käytetyistä ratkaisuista esimerkiksi 8-bittisistä kuvista luettu arvoalue 55–220 on arvaus, enemmän tai vähemmän valistunut. Jokin toinen arvoalue saattaisi tuottaa parempia tuloksia. Toisaalta käytännön havaintojen pohjalta voin todeta, että pienikin ohjelmointivirhe saattaa vaikuttaa lopputulokseen enemmän kuin teoreettisen tason yksityiskohdat.

Päätös käyttää lineaarisesti tallennettuja kuvia lähdekuvina saattoi olla liioitellun varovaista. Kun käytetään digitaalista kameraa ja kamerassa sRGB-väriavaruutta, voitaneen melko huoletta lähteä oletuksesta, että kuvien gamma-arvo on $2,2$, ja sen korjaaminen tuottaa lineaarista kuvadataa. Näin ollen ohjelma voisi lukea suoraan gamma-korjattuja kuvia, eikä niitä tarvitsisi juurikaan muokata. Tutkimuksen yhteydessä ehdin kokeilla digitaalikameralla otetun jpeg-kuvan korjaamista Image Magic -ohjelmassa gamma-arvolla $0,45$. Silmämääräinen vertailu muunnetun jpeg-kuvan ja raw-kuvasta kehitetyn lineaarisen luvan välillä ei paljastanut eroja.

HDR-kuvan luominen useita kuvia yhdistelemällä ei ole kovin vaikeaa. Kaupallisia ohjelmiakin tätä tarkoitusta varten on olemassa. Toisaalta tämä menetelmä asettaa merkittäviä rajoituksiakin. Nämä rajoitukset osaltaan vähentävät mahdollisuuksia mittavaan käyttöön.

Viiteluettelo

[1] Computer Graphics System Development Corporation (CGSD).
Gamma Correction Explained.
http:/www.cgsd.com/papers/gamma_intro.html Checked 23.12.2006

[2] Larson G. W. Overcoming Gamut and Dynamic Range Limitations in Digital Images, In *Proc. of the Sixth Color Imaging Conference,* Nov. 1998. http://www.anyhere.com/gward/papers/cic98.pdf Checked 10.10.2006

[3] Industrial Light and Magic. About OpenEXR. http://www.openexr.com/about.html Checked 8.1.2007

[4] Ward G. High Dynamic Range Image Encodings. http://www.anyhere.com/gward/hdrenc/Encodings.pdf Checked 9.10.2006

[5] Wikipedia, Gamma correction. http://en.wikipedia.org/wiki/Gamma_correction Checked 2.10.2006

[6] Wikipedia, High dynamic range imaging. http://en.wikipedia.org/wiki/High_dynamic_range_imaging Checked 9.10.2006

[7] Wikipedia, Portable Pixmap. http://en.wikipedia.org/wiki/Portable_pixmap Checked 5.1.2007

Ketterät menetelmät hallinnollisesta näkökulmasta

Jari Kautiala

Tiivistelmä.
Uusimpia suuntauksia ohjelmistokehityksen menetelmätieteen alueella ovat ketterät menetelmät (*agile methods*). Ketterien menetelmien arvomaailman lähtökohtana on, että yksilö ja kommunikaatio ovat tärkeämpää kuin prosessit ja työkalut, toimiva ohjelmisto on tärkeämpää kuin dokumentaatio, yhteistyön korostamisen tärkeys suhteessa sopimuspykäliin sekä muutokseen mukautuminen ennemmin kuin suunnitelman noudattaminen. Vastuu ja valta ovat yksilöillä ja tiimeillä, jotka raportoivat projektin tilanteen suullisesti projektitapaamisissa. Päätösvalta on jaettu organisaation alimmille tasoille. Ketterissä menetelmissä, niiden luontaisen mukautumisen ansiosta, on usein sisäänrakennettu ja varsin toimiva riskienhallinnan menettely. Ketterät menetelmät ovat varsin. Ne tulevat muuttumaan ja sulautumaan toisiinsa muodostaen entistä tehokkaampia ohjelmistotuotannon menetelmiä. Näyttäisi myös, että kaivattua parempaa tukea projektinhallintaan tulee lisääntymään tulevaisuuden ketterissä menetelmissä. Ketterille menetelmille on selvä tarve ja niiden käyttö näyttäisi lisääntyvän lähitulevaisuudessa.

Avainsanat ja -sanonnat: Ohjelmistotuotanto, projektinhallinta, ketterät menetelmät, agile methods.

CR-luokat: K.6.1, K.6.3, D.2.9

1. Johdanto

Ohjelmistotuotanto on kuulunut suomenkielen käsitteistöön 1960-luvulta lähtien. Sanan alkuperäinen englanninkielinen termi *software engineering* viittasi tekniikkaan ja termi ohjelmistotekniikka herättikin aikanaan keskustelua erillisenä tekniikan alueena. [Haikala ja Märijärvi, 2003] Vapaasti suomennettuna IEEE [610.12] standardi määrittää, että ”ohjelmistotuotanto on systemaattista, kontrolloitua ja mitattavissa olevaa toimintaa, joka tähtää ohjelmistotuotteen kehittämiseen, käyttöönottoon ja ylläpitoon”. Määritelmä kattaa ohjelmistontuotantoprosessin kaikki eri osa-alueet, kuten laatujärjestelmän ja laadun varmistuksen, projektinhallinnan, tuotteenhallinnan, testauksen, määrittelyn, suunnittelun, toteutuksen, dokumentoinnin sekä käyttöönoton ja ylläpidon.

Ohjelmistot on perinteisesti toteutettu projektimuotoisesti. Suomen Standardisoimisliiton projektitoimintasanasto [SFS 1981] määrittää projektin olevan

”varta vasten muodostetun organisaation määrätarkoitusta varten toteuttama ainutkertainen hanke, jonka laajuus- ja laatutavoitteet sekä aika- ja kustannuspanokset on ennalta määritetty”. Nämä kriteerit täyttyvät yleensä myös ohjelmistoprojekteissa. Ohjelmistojen rakentaminen on ainutkertaista. Vaikka samoista lähtökohdista muodostettaisiin toinen rinnakkainen projekti, on todennäköistä, ettei täysin samaan lopputulokseen päästä. Tämä johtuu esimerkiksi inhimillisistä tekijöistä, erilaisesta ympäristöstä ja muuttuvista olosuhteista sekä muista muuttujista. Näiden muuttuvien tekijöiden vallitessa ohjelmistoprojekti toimii hetkellisen elinaikansa tuottaakseen toimintaan sijoitetuilla määrällisillä resursseilla lopputuloksena asiakkaan haluaman tuotteen tai palvelun ennalta määritellyin laatutavoittein.

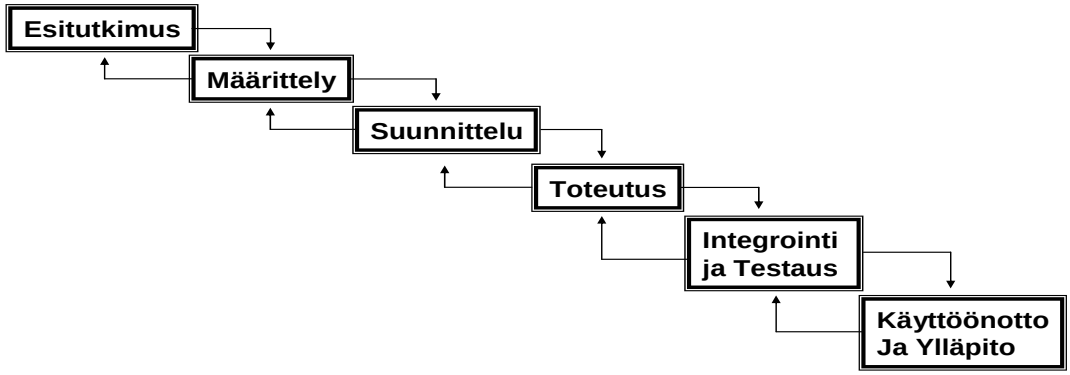
Projektityössä, kuten myös ohjelmistoprojekteissa, on määrättyjä tapoja pyrkiä haluttuun lopputulokseen. Työkokonaisuus pyritään usein jaottelemaan pienempiin osiin, jotta esimerkiksi tarkempi sisällön suunnittelu onnistuu ja toteutukseen kuluvan tuntimäärien määrittäminen helpottuu. Tähän jaotteluun ja kokonaisuuden määrittämiseen on ohjelmistoprojekteissa usein käytetty vesiputousmallia. Vesiputousmallia on käytetty myös muissa insinööritieteissä, kuten esimerkiksi rakennustekniikassa rakennusten ja siltojen toteuttamisessa. On olemassa useita muitakin tapoja ohjelmistoprojektin toteuttamiseen. Valittavaan tapaan vaikuttaa useita muuttujia kuten esimerkiksi lopputuotteen ja asiakkaan vaatimukset, organisaatio sekä liiketoimintaympäristö. Esimerkiksi vaatimukset nopeampaan tuotteen tai palvelun valmistumiseen ja markkinapaineiden kasvaminen ovat johtaneet perinteisten mallien analyysiin ja kritiikkiin. Toisaalta myös ohjelmistoalan työntekijöiden halu vähentää byrokratiaa ja pyrkiä laadukkaampaan lopputulokseen yhteistyössä loppukäyttäjän kanssa ovat osaltansa johtaneet vaihtoehtoisten mallien syntymiseen. Vaihtoehdoiksi ovat nousseet ketterät menetelmät (*agile methods*). [Haikala ja Märijärvi, 2003; Agile manifesto, 2001]

Tämän tutkimuksen tarkoituksena on kuvata perinteisiä ohjelmistotuotannon menetelmiä sekä ketteriä menetelmiä ja verrata näitä keskenään hallinnollisesta näkökulmasta. Ketteristä menetelmistä tässä tutkimuksessa painotetaan Scrum- ja RUP-menetelmiä. Nämä menetelmät poikkeavat toisistaan huomattavasti ja ovat laajalti käytettyjä ohjelmistoteollisuudessa. [Abrahamsson et al., 2002] Tutkimuksessa sovelletaan käsitteellis-teoreettista tutkimusotetta [Järvinen ja Järvinen, 2000].

2. Perinteinen ohjelmistoprojekti ja sen kritiikki

Ohjelmistoprojekteissa ohjelmiston elinkaari on jaettu eri vaiheiksi. Elinkaari alkaa ohjelmiston kehittämisen aloittamisesta ja päättyy tuen loppumiseen ja

ohjelmiston poistamiseen käytöstä. Tällaisia ohjelmiston elinkaaren vaihejakomalleja on useita, mutta perinteisin näistä on vesiputousmalli (*waterfall*). Tässä prosessi alkaa esitutkimuksella ja määrittelyllä, edeten rakennusvaiheen suunnittelun, toteutuksen ja integroinnin kautta käyttöönottoon ja ylläpitoon. Työ etenee vaiheesta toiseen siten, että mikäli kyseisen vaiheen vaihetuotteet ovat toteutettu laatukriteerit täyttäen, voidaan edetä seuraavaan vaiheeseen. Vesiputousmalli kuvataan usein siten, että ylimpänä vasemmalla on esitutkimusvaihe ja alimpana oikealla käyttöönotto muodostaen mielikuvan vesiputouksesta sivulta katsottuna (ks. Kuva 1). [Haikala ja Märijärvi, 2003]



Kuva 1: Vesiputous-malli [Haikala ja Märijärvi, 2003]

Ohjelmisto- kuten muissakin projekteissa on tärkeätä valita sopivin toimintatapa. Pitkään ohjelmistoalan konsulttina toiminut Gerald Weinberg [1982] on kiteyttänyt tämän lauseeksi ”mikään ratkaisu ei sovellu kaikkiin tehtäviin ja johonkin tilanteeseen parhaiten soveltuva lähestymistapa voi olla toisessa tilanteessa kaikista huonoin”. Vesiputousmalli on formaali prosessi, joka määritettyjen vaatimusten perusteella etenee tarkasti suunnitteluun tiedossa olevan ongelman ratkaisemiseksi. Teoreettisesti tarkasteltuna vaiheittain etenevä vesiputousmalli vaikuttaa tehokkaalta eikä sisällä tarpeettomia vaiheita tai tietyn tekemisen toistoa.

Käytännössä on kuitenkin havaittu vesiputousmallin puutteet. Se soveltuu isoihin projekteihin, joissa on stabiili toimintaympäristö, tarkasti tiedossa olevat

vaatimukset ja toimiva valmis arkkitehtuuri. Näin kuitenkin harvoin käytännössä on. Nykyaikainen hektinen maailma vaatii myös ohjelmistokehitysmenetelmiltä enemmän, nopeammin ja parempaa. Ohjelmistojen elinkaari on lyhentynyt. Asiakas ei halua odottaa yli vuotta saadakseen tärkeän ominaisuuden käyttöönsä. Entä jos vaatimuksia halutaankin muuttaa kesken käynnissä olevan projektin johtuen esimerkiksi muuttuneesta tekniikasta, lainsäädännöstä tai asiakkaan muuttuneesta tarpeesta. Vesiputousmallissa ei ole mahdollista vähäistä suurempaan muutoksenhallintaan projektin ollessa käynnissä. Mikäli vesiputousmallin mukaisessa projektissa tehdään muutoksia, on se radikaalia ja todennäköisesti taloudellisesti raskasta tai vähintään muuttaa projektin laajuutta tai kestoa. [Scrum, 2006]

3. Ketterät menetelmät

Uusimpia suuntauksia ohjelmistokehityksen menetelmätieteen alueella ovat *ketterät menetelmät* (agile methods). Ketterien menetelmien arvomaailman lähtökohtana on, että yksilö ja kommunikaatio ovat tärkeämpää kuin prosessit ja työkalut, toimiva ohjelmisto on tärkeämpiä kuin dokumentaatio, yhteistyön korostamisen tärkeys suhteessa sopimuspykäliin sekä muutokseen mukautuminen ennemmin kuin suunnitelman noudattaminen. Edellä mainitut periaatteet korostavat ihmisten merkitystä ohjelmistokehitysprosessissa ja lähtökohdat poikkeavat huomattavasti vesiputousmallin ajatusmaailmasta. Tärkeintä ketterissä menetelmissä on luoda yhteistyöhön kykenevä tiimi, sillä edes toimiva prosessi ei pelasta projektia epäonnistumiselta, jos projektin resurssit eivät toimi tehokkaasti. [Agile manifesto, 2001]

Myös dokumentointiin on kiinnitetty huomiota, sillä sen tuottamiseen ja ajan tasalla pitämiseen kuluu huomattavasti aikaa ja panostusta. Ketterien periaatteiden mukaan dokumentteja tulisikin tuottaa vain, jos niiden tarve on akuutti ja tärkeys merkittävä. [Agile manifesto, 2001]

Asiakkaan osallistumista projektiin korostetaan. Onnistuneen projektin perusta on projektin toteuttaminen kiinteässä yhteistyössä asiakkaan kanssa. Keskeistä on asiakkaalta saatu palaute, jonka tulee olla säännöllistä ja aktiivista. Tähän yhteistyöhön liittyy myös toimivan ohjelmiston korostaminen ohi sopimustekstin. [Agile manifesto, 2001]

Ketterät menetelmät tukevat muutosta. Toteutus ja myös muut projektin vaiheet tehdään iteraatioina, jolloin jokaisen iteraation jälkeen on syntynyt jotain toimivaa toiminnallisuutta asiakkaan käyttöön. Suunnitelmien tulee olla joustavia ja valmiita mukautumaan markkinoiden, teknologioiden tai asiakasvaatimusten muutoksiin. [Agile manifesto, 2001]

Ketterät menetelmät tulevat syrjäyttämään perinteisiä menetelmiä, mutta eivät kokonaan korvaamaan niitä. Erittäin vaativissa projekteissa, kuten sairaalajärjestelmissä, avaruussukkuloissa ja reaaliaikavaatimuksia sisältävissä projekteissa, perinteiset menetelmät tulevat todennäköisesti säilymään. Näissä esimerkiksi ei välttämättä voida soveltaa inkrementaalista kehitystä tai kustannukset ja tarkat suunnitelmat edellytetään olevan jo projektin alussa.

3.1. Menetelmät ja niiden käyttö

Ketterät menetelmät toimii yhdistävänä terminä usealle eri ohjelmistokehitysmenetelmälle. Näitä kaikkia kuitenkin yhdistää useampi yhteinen ominaisuus. Yhteisenä tekijänä on toimivan ohjelmiston pitäminen tärkeimpänä ohjelmistoprojektin etenemän mittarina. Menetelmiä on useita. Mielestäni tärkeimpiä tai mielenkiintoisimpia näistä ovat eXtreme Programming (XP), Rational Unified Process (RUP), Scrum, Mobile-D, Crystal-metodologiaperhe, Dynamic Systems Development ja Feature Driven Development. Näistä tarkemmin jäljempänä esittelen menetelmät Rational Unified Process ja Scrum. Siitä huolimatta, että osa menetelmistä sisältää prosessinomaisia piirteitä, käytän jatkuvuuden vuoksi kuitenkin näistä termiä menetelmä.

3.2. Ketteryys avoimen lähdekoodin kehityksessä

Avoimen lähdekoodin kehityksen edellytyksiä ovat olleet ketteryys, vapaus, luovuus ja yhteisöllisyys. Esimerkkinä massiivisesta avoimen lähdekoodin kehityksestä mainittakoon Linux-käyttöjärjestelmäkehitys. Eric Raymond [1997] kuvasi tämän kehitysideologian tunnetussa artikkelissaan *The Cathedral and the Bazaar*. Artikkelissa Raymond jakaa ohjelmistokehityksen kahteen eri tapaan: katedraali ja basaari. Katedraalikehityksellä Raymond viittaa perinteiseen ohjelmistotuotannon menetelmään, jossa pieni joukko yrityksen ohjelmistokehittäjiä toimii tuottaakseen ennalta määrättyyn ongelmaan ratkaisun. Asiantuntijat toteuttavat ohjelmistoa, ikään kuin muusta maailmasta eristettynä, vastaten projektin lopussa syntyneestä lopputuotteesta sekä sen ylläpidosta ja muutostenhallinnasta. Loppukäyttäjälle toimivan ohjelmiston saamiseen menee useita kuukausia. Basaari-menetelmässä toimitaan täysin päinvastoin. Siinä useita ihmisiä globaalissa maailmassa osallistuu projektiin. Loppukäyttäjä saa käyttöönsä nopeasti toimivan ohjelmiston, joka sisältää ainakin osan vaatimuksista. Ideologiana on julkaista ajoissa ja usein. Ohjelmistokoodi on yhteisomistuksessa, kun virheitä tai parannuskohteita löytyy, voi kuka tahansa muuttaa ja toteuttaa näitä yleisesti jaossa olevaan ohjelmakoodiin.

Ketteryyden taso vaihtelee eri menetelmien välillä huomattavasti. Ääripäiden välillä liikutaan todella vapaasta avoimen lähdekoodin kehityksen mallista tarkkaan ja osittain väärin ymmärrettyynkin RUP-malliin. Kuten Weinberg

[1982] totesi, tärkeintä on valita oikea menetelmä kulloiseenkin tapaukseen sopivaksi.

3.3. RUP-menetelmä

RUP eli *Rational Unified Process* perustuu Ivar Jacobsonin Rational-yhtiössä kehittämään prosessiin. RUP pohjautuu Jacobsonin aiempiin tutkimuksiin ja Unified menetelmään. Rational on nykyään IBM-konsernin omistama yritys, jonka tuotteet ja palvelut ovat pitkälti ohjelmistoprosessia tukevaa työkalustoa. [IBM, 2005]

RUP on iteratiivinen ja jokainen iteraatio muodostaa oman pienen vesiputousmallin tai miniprojektin sisältäen vaiheet alun vaatimusten keruusta testaukseen tuottaen lopputuotteena esitettävää toiminnallisuutta loppukäyttäjälle [Haikala ja Märijärvi, 2003]. Eri vaiheissa käytetään ohjelmistotuotannossa pitkään käytössä olleita tapoja. RUP tukeutuu voimakkaasti standardiin Unified Modeling Language -mallinnukseen [2004] (myöhemmin UML-standardi) sekä komponenttiarkkitehtuuriin vaikka perinteisempiä ohjelmiston rakennustapoja ei pois suljetakkaan. Vaikka RUP:n omistajuus on kaupallisella yrityksellä, ei RUP suoranaisesti edellytä käyttämään kaupallisia työkaluja toisin kuin Lindberg [2003] tutkielmassaan kirjoitti. [IBM, 2005]

RUP on myös prosessikehys muodostaen mahdollisuuden mukauttaa prosessia yrityksen formaaliksi ohjelmistotuotantoprosessiksi. RUP-menetelmän räätälöinti voidaan tehdä kahdella tasolla: projekti- tai organisaatiotasolla. Organisaatioperustaisessa räätälöinnissä voidaan ottaa tarkemmin huomioon organisaation koko ja liiketoiminta alueen erityisvaatimukset sekä komponenttien uudelleenkäytettävyys näkökulmat esimerkiksi eri projektien välillä. [IBM, 2005]

3.3.1. RUP-menetelmän ideologia

RUP pohjaa kuuteen ohjelmistoteollisuudessa parhaaksi toimintatavaksi (*best practises*) havaittuun periaatteeseen. Nämä ovat kategorisoitu muistisäännöksi aakkosten mukaan A:sta F-kirjaimeen alkuperäisessä englanninkielisessä kirjallisuudessa. RUP-menetelmässä käytetyt parhaat toimintatavat ovat ainakin osittain samoja kuin ketterissä menetelmissä yleisesti painotetut periaatteet [Agile manifesto, 2001; IBM, 2005].

Adapt the Process eli prosessin pitää olla muokattavissa kohdeympäristöön sopivaksi. Prosessin mukauttamiseen vaikuttavia tekijöitä ovat esimerkiksi projektin koko, käytetyn teknologian vaatimukset, projektiryhmän mahdollisesta maantieteellisestä hajautumisesta johtuvat vaatimukset, sidosryhmien lukumäärä ja joustavuus vaatimukset. Tarvittaessa prosessia voidaan räätälöidä edellä kuvatusti.

Balance Competing Stakeholder Priorities eli eri vaatimusten tärkeyden erottaminen eri asiayhteydessä. Esimerkiksi liiketoimintaympäristö yleensä asettaa vaatimuksia toteutettavalle ohjelmistolle. Nämä vaatimukset ovat usein hyvin erilaisia verrattaessa loppukäyttäjän asettamiin vaatimuksiin. RUP:ssa vaatimukset määritetään käyttäjänäkökulmasta käyttäen UML-standardissa kuvattua Use Case -tekniikkaa käyttötapausten kuvaamiseen. [UML, 2004]

Collaborate Across Teams -periaate painottaa projektin ryhmien välistä yhteistyötä. Yhteistyöllä tässä yhteydessä tarkoitetaan eri sidosryhmien ja toiminnallisten ryhmien välistä toimintaa esimerkiksi testauksen, toteutuksen ja loppukäyttäjien välillä. Periaatteella pyritään esimerkiksi erilaisten näkemysten vaihtoon ja keskusteluun.

Demonstrate Value Iteratively -periaate painottaa jatkuvaa integroimista eli ohjelmistoprojektin pitää tuottaa iteraatioissa syntyvänä lopputuotteena toimivaa ja suoritettavaa ohjelmistoa suhteessa edellisen iteraation lopputuotteeseen. Näiden kahden iteraation erotuksena syntynyttä tuotosta kutsutaan inkrementiksi. Jatkuvalla integroimisella tuotekokonaisuudeksi saavutetaan asiakkaan luottamus tuottamalla koko ajan loppukäyttäjälle esittelykelpoista ohjelmistoa tai sen osia. Tämä mahdollistaa myös nopean ja jatkuvan vasteen loppukäyttäjältä ja asiakkaalta heidän päästessä konkreettisesti käyttämään ohjelmiston kehitysversioita projektin edetessä.

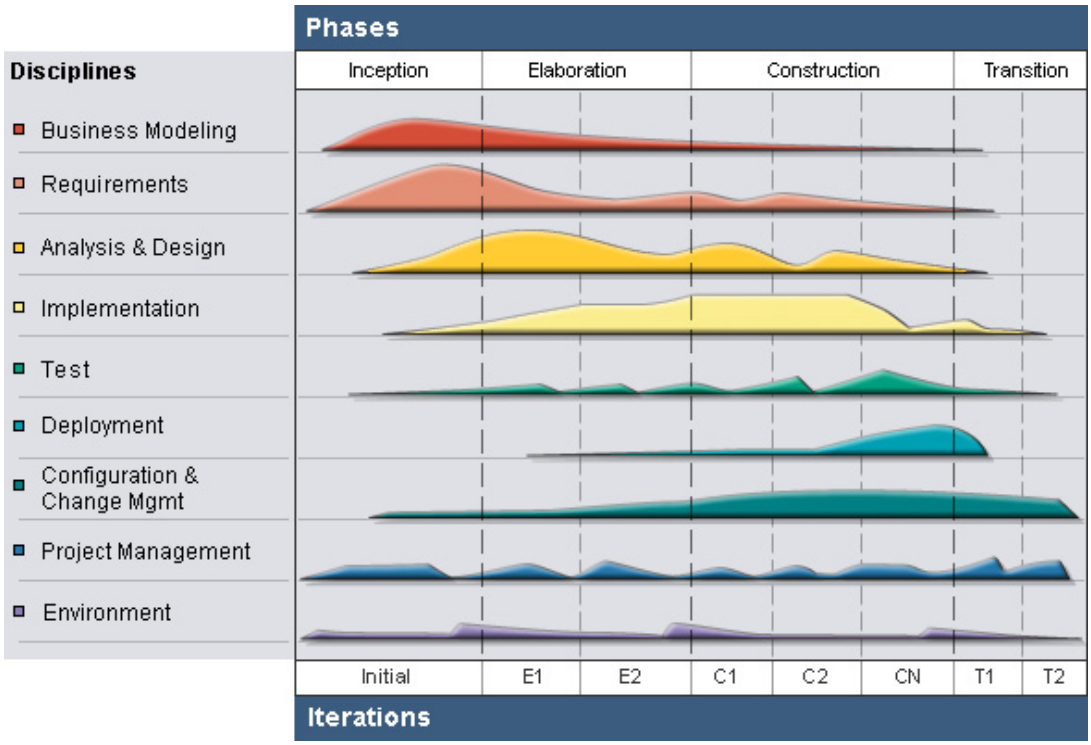
Elevate Level of Abstraction -periaate taas kehottaa nostamaan abstraktiotasoa. Ohjelmistoteollisuus toimii usein uusien teknologioiden ja vaatimusten parissa, jotka jo itsessään lisäävät toteutuksen monimutkaisuutta. Nostamalla perspektiiviä korkeammalle mahdollistetaan kokonaisvaltaisempi näkemys toteutukseen. Tällä on tavoitteena löytää toteutukseen jo olemassa olevia osia, kuten esimerkiksi uudelleenkäytettäviä komponentteja, avointa lähdekoodia tai muita aiemmin hyväksi ja toimiviksi havaittuja testattuja ratkaisuja.

Focus Continuously On Quality eli toteutetaan jatkuvasti laadukkaan ohjelmistotuotannon mukaisia periaatteita tavoitteena riskien pienentäminen. Laatu ei tarkoita ainoastaan vaatimusten täyttämistä tai että ohjelmisto läpäisee sille määritetyt testitapaukset, vaan laadulla viitataan tässä koko ohjelmiston elinkaareen.

3.3.2. RUP prosessina

Kuva 2 esittää RUP-menetelmän graafisessa muodossa. Yläpuolinen horisontaaliakseli kuvaa aika-akselia ja jakoa neljään eri peräkkäiseen vaiheeseen: aloitus (*inception*), yksityiskohtainen suunnittelu (*elaboration*), rakentamisvaihe (*construction*) sekä siirtovaihe. Vertikaaliakseli kuvaa sisältöä ja jakoa eri käytäntöihin (*disciplines*). Alempi horisontaaliakseli kuvaa jakoa yksittäisiin iteraa-

tiohin. Värillinen ala kuvaa vaiheen tai käytännön suhteellista osuutta iteraatiossa.



Kuva 2: RUP-menetelmä [IBM, 2005]

3.4. Scrum

Scrum nousi yleiseen tietoisuuteen 1990-luvun loppupuolella projekteissa, joihin liittyi suuri riski käytettävän teknologian tai projektiin kohdistuvien vaatimusten suhteen. Alun perin termi *scrum* viittaa rugbyn pelistrategiaan, jolla pyritään saamaan pallo takaisin peliin tiimityöskentelyn avulla. Termiä on alettu käyttää ohjelmistotuotannon yhteydessä 1986, kun Takeuchi ja Nonaka nimesivät artikkelissaan sopeutuvan, nopean ja itseohjautuvan kehitysprosessin termillä Scrum. [Abrahamsson et al., 2002]

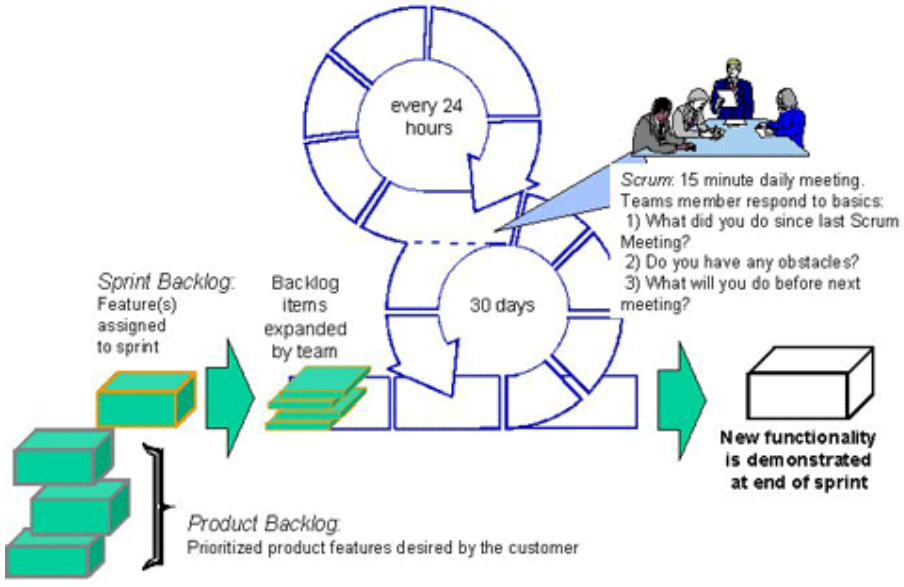
Scrum on ketterille menetelmille tyypillisesti iteratiivisesti inkrementaalinen menetelmä, ja se on evoluutio olemassa olevista ohjelmistotuotannon menetelmistä, käytännöistä ja parhaista toimintatavoista. Iteratiivisuus ja inkrementaalinen kehitys sopii etenkin ympäristöön missä useampi projektiin vaikuttava tekijä on potentiaalisen muutoksen alla. Näihin argumentteihin vedoten Scrum, kuten useampi muukin menetelmä, julistaa olevansa tapa maksimoida tuottavuus. [Scrum, 2006]

Scrum-menetelmän mukainen kehitysprosessi koostuu kolmesta vaiheesta. Ensimmäinen vaihe (*pre-game phase*) sisältää vaatimusmäärittelyä, suunnittelua, projektiryhmän määrittämisen ja arkkitehtuurin hahmottelua. Toisessa vaiheessa (*development/game phase*) tapahtuu varsinainen kehitystyö, joka toteutetaan *sprinteissä*. *Sprint* on Scrum-menetelmän iteraatio, joka tuottaa kehitettävään järjestelmään yhden toiminnallisen kokonaisuuden. Sprint kestää tavallisesti yhdestä viikosta kuukauteen. Viimeisessä vaiheessa (*post-game phase*) tapahtuu version julkaisu. Scrum-menetelmän vaiheet on esitetty Kuvassa 3. [Scrum, 2006]

Suunnitteluvaiheen alussa muodostetaan toteutettavien vaatimusten lista. Lista sisältää myös vaatimusten prioriteetit ja aikatauluarviot sekä muuta tietoa projektista kuten käytettävät työkalut, projektin henkilöt, riskit eli on siis eräänlainen projektisuunnitelma. Tätä kutsutaan Scrum-termistön mukaan nimellä *Product Backlog*. Lista tehdään yhteistyössä projektin sidosryhmien kanssa ja sitä päivitetään ja tarkennetaan tarvittaessa, kuitenkin niin, että lista on yhden henkilön vastuulla. [Scrum, 2006]

Kuvan 3 mukaisesti Scrum-menetelmässä jatkuva kommunikointi toteutuu päivittäisessä Scrum-tapaamisessa. Tämä formaalitapahtuma kestää viisitoista minuuttia, jonka aikana projektiryhmän suunnittelijoille esitetään kolme kysymystä:

1) Mitä teit sitten viime Scrum-tapaamisen?
2) Onko esteitä havaittavissa?
3) Mitä teet seuraavaan tapaamiseen mennessä?



Kuva 3: Scrum-menetelmä [Scrum, 2006]

4. Hallinta ja ketterät menetelmät

Riskien- ja resurssienhallinta ovat tärkeä osa ohjelmistoprojektia. Ketterissä menetelmissä nämä poikkeavat verrattuna esimerkiksi vesiputousmalliin. Tämä johtuu osittain siitä, että ketterien menetelmien tarve on ollut voimakas etenkin riskialtteissa ympäristöissä, joissa muutos on jatkuvaa ja vaikuttaa myös resurssienhallintaan. [Agile manifesto, 2001]

4.1. Riskienhallinta

Ohjelmistoprojektit ovat usein laajoja ja monimutkaisia. Projektit toteutetaan yleensä ympäristössä, joka muuttuu ja sisältää epävarmuustekijöitä. Usein vaatimukset projektin alussa eivät ole täysin selviä. Standish Groupin [2002] yli kymmenen vuotta jatkuneen tutkimuksen mukaan 28 prosenttia ohjelmistoprojekteista onnistuu. Kääntäen tämä tarkoittaa että yli 70 prosenttia ohjelmistoprojekteista epäonnistuu joko täysin tai osittain. Epävarmuustekijät usein konkretisoituvat riskeiksi. Riski on vahingonvaara tai –uhka, mutta myös menetyksen- tai tappionuhka. Riskeillä on todennäköisyys toteutumiseen sekä toteutuessaan laajuus tai vaikutus. Suomisen [2003] mukaan riski voidaan määritellä muotoon

Riski = todennäköisyys x riskin laajuus tai vakavuus.

Riskien laajuuden Suominen [2003] kategorisoi euromääräisesti ja todennäköisyyden arviona toteutumiskertojen mukaan suhteessa eri ajanjaksoihin. Riskien kategorisointi edellä kuvatulla tavalla antaa hyvän näkemyksen riskienhallintaan, vaikka Suomisen esittämä kategorisointiin pitääkin suhtautua suuntaa antavana.

Standish Groupin [2002] tutkimuksen mukaan ohjelmiston ominaisuuksista tai toiminnoista 45 prosenttia ei koskaan käytetä ja 25 prosenttia ominaisuuksista on tarpeellisia. Kaikkiaan 80 prosenttia ominaisuuksista käytetään joskus, harvoin tai ei koskaan. Tämä suuri osa sisältää tarpeettomia potentiaalisia riskejä. Riskit voivat liittyä esimerkiksi käytettyyn teknologiaan, arviointiriskeihin tai kehitystyökaluihin.

Tarve ketteriin menetelmiin sekä niiden toimintaympäristö on useissa menetelmissä riskilähtöinen. Verrattuna esimerkiksi vesiputousmalliin ketterissä menetelmissä riskienhallinta on usein hyvin monimuotoisesti sisäänrakennet-

tua ja tapahtuu menetelmän toimesta mekaanisesti. Riskienhallinta voidaan jakaa kahteen osaan, riskien kartoittamiseen ja niihin varautumiseen [Haikala ja Märijärvi, 2003]. Esimerkiksi Scrum-menetelmässä *Product Backlog* -listan luonnissa kartoitetaan riskit sekä suunnitellaan toimintatapa riskin toteutuessa. Riskien seuranta ja tarkennus Scrum-menetelmässä tapahtuu jokaisen iteraation eli sprintin alussa sekä päivittäisessä Scrum-tapaamisessa.

RUP-menetelmässä suositellaan käyttämään jo testattua ja toimivaksi havaittua olemassa olevaa toiminnallisuutta, kuten esimerkiksi avoimen lähdekoodin toiminnallisuutta. Avoin lähdekoodi kaupallisessa ympäristössä asettaa luonnollisesti omat tekijänoikeudelliset haasteensa, mutta muita vaihtoehtoja ovat esimerkiksi uudelleen käytettävyys ja niin sanonut perinnejärjestelmät (*legacy systems*). RUP-menetelmä itsessäänkin suosii jo valmiiksi todettuja hyviä ratkaisuja. RUP-menetelmässä mallit ja anti-mallit (*pattern, antipattern*) ovat merkittävässä osassa menetelmän prosessin eri kohdissa. Mallit ovat hyväksi havaittuja tapoja tai tapoja, joita on havaittu olevan suotavaa välttää.

RUP-menetelmän aloitusvaihe sisältää formaalin riskianalyysin. Menetelmä ohjaa kartoittamaan riskit sekä tekemään suunnitelman riskille varalle. RUP-menetelmässä riski voidaan pyrkiä välttämään, siirtämään tai se voidaan hyväksyä tapahtuvaksi. RUP ohjaa toteuttamaan suuririskisimmät osa-alueet ensimmäisissä iteraatioissa. Näin saadaan jo projektin alkuvaiheessa toimivaa ja testattavaa ohjelmakoodia. Inkrementissä tuotettu toimiva ohjelmiston osa on esitettävissä myös asiakkaalle ja loppukäyttäjälle. Tämä vähentää huomattavasti vaatimusmäärittelyvaiheessa mahdollisesti syntyviä sidosryhmien välisiä erilaisia näkemyksiä toiminnallisuudesta sekä auttaa havaitseman riskit ja virheet mahdollisimman ajoissa. Sama inkrementaalinen lähestymistapa on käytössä myös muissa ketterissä menetelmissä.

4.2. Resurssienhallinnasta

Scrum lupaa tarjota projektin johtajalle suoran näkyvyyden kehitykseen ja mahdollisuuden reaaliaikaiseen projektin johtamiseen. Se lupaa myös mahdollisuuden tehdä kompromisseja laadun, sisällön, kustannusten tai ajan suhteen, jotta paras mahdollinen asiakashyöty saavutetaan. Tämä tapahtuu esimerkiksi Sprint-tapaamisten kautta, *Backlog*-listojen (*Sprint-*, *Release-* ja *Product Backlog*) eli suunnitelmien läpikäymisen ja niitä vastaavan todellisuuden ja trendien kautta. Henkilö- ja projektijohtaminen tapahtuu huomattavalta osin päivittäisissä Scrum-tapaamisissa. Projektin henkilöt kertovat mahdollisista huolenaiheistaan projektin johdolle, joka ottaa asiat vastattavakseen. Tehtäviä ei siis Scrum-menetelmässä kirjata ylös vaan ne välitetään suullisesti. Tehtävät ja muu projektin tilanne ovat siis kaikkien osapuolten, myös asiakkaan, ulottuvilla. Jokaisen Sprintin jälkeen *Product Backlog* -lista tarkastetaan ja tarvittaessa tar-

kennetaan ennen seuraavan Sprintin alkua. Näin myös vaatimustenhallinta seuraa koko projektin ja sekä asiakas että muu projektiryhmä pääse osallistumaan tuotteenhallintaan. Scrum ei määrittele tarkasti miten esimerkiksi vaatimusten keruu tai suunnittelu pitäisi tehdä. Sen sijaan se ohjeistaa, miten projekti tulisi organisoida Scrum-menetelmän mukaisesti. Tässä suhteessa se eroaa huomattavasti esimerkiksi RUP-menetelmästä. Jokaisen Sprintin jälkeen myös ajankäyttö, aikataulu sekä näiden kautta myös kustannukset tarkentuvat. Tämä on argumentti, jolla etenkin tilaajaorganisaatio puolustaa vesiputousmallin käyttöä. Vesiputousmallissa edellä mainitut asiat on jo suunniteltu projektin alussa ja näin ollen myös projektin kokonaiskustannukset sekä aikataulu on tiedossa. Tämä, sekä mahdollisuus muutoksiin projektin aikana, on vesiputousmallin samalla sekä vahvuus että sen heikkous. [Scrum, 2006]

Scrum kuten RUP:kin ovat prosessikehyksiä. Tämä mahdollistaa niiden liittämisen hyvinkin erilaisiin ympäristöihin ja myös mahdollisuuden niiden räätälöintiin ympäristön vaatimusten mukaisesti. Tämä helpottaa siirtymistä yrityksen olemassa olevasta prosessista johonkin ketterään menetelmään. On mahdollista säilyttää osa yrityksen valmiista prosessista ja alkuvaiheessa korvata osia esimerkiksi RUP-menetelmän osilla. Hirsch [2002] kuvasi vastaavaa tilannetta. Tässä yritys etsi uutta ohjelmistoprosessia tarpeisiinsa. Tässä tapauksessa yrityksellä oli jo käytössään esimerkiksi UML, joten he pystyivät nopeasti muuttamaan olemassa olevaa prosessiaan ja muuttamaan sen kevyt-RUP:ksi, kuten Hirsch muodostamaansa räätälöityä menetelmää kuvaa.

XP@Scrum koostuu näiden kahden menetelmän yhdistelmästä. XP@Scrum sisältää Scrum-menetelmän hallinnan, kuten esimerkiksi backlog-listat ja sprint-ajattelun, ja XP-menetelmän ohjelmistotuotannon elementit, kuten esimerkiksi pariohjelmoinnin, ohjelmointistandardit ja uudelleenfaktoroinnin. Vastaavasti IBM:n Joe Krebs [2005] on kirjoittanut tutkielman RUP- ja Scrum-menetelmien yhdistämisestä. Useissa menetelmissä on paljon toimivia käytäntöjä ja tärkeätä on valita menetelmä, joka sopii juuri haluttuun tarkoitukseen. [Scrum, 2006]

Kaikki ketterät menetelmät jakavat päätösvaltaa organisaatiossa alaspäin. Tämä on ketterien menetelmien suurimpia eroja suhteessa perinteisiin menetelmiin. Tiimit ottavat vastuuta työstään ja seuraavat tehtävien etenemistä sekä tehtävän lopettamiseen vaadittavaa panosta jatkossa. Työn etenemisestä ja ongelmista raportoidaan useissa menetelmissä muulle projektiryhmälle ja projektin johdolle suullisesti suoraan. Tämä menettely on ketterien menetelmien idean mukaista ja koen pyrkimyksen tähän erittäin hyvänä tapana missä tahansa projektimuodossa teollisuuden alasta riippumatta. Tämä kuitenkin asettaa vaatimuksia myös projektipäällikölle ja muulle tapaamisiin osallistuville projekti-

johdon edustajille. Näissä tapaamisissa varsinaiset projektia johtavat henkilöt tulevat esiin riippumatta henkilön asemasta projektissa. Todennäköisesti näissä tapaamisissa projektipäällikön kyky henkilöjohtamisen alueella punnitaan, samoin myös hänen henkilökohtainen sosiaalinen älykkyytensä ja kommunikaatiotaidot.

Vaikka ketterissä menetelmissä korostetaankin tiimiä, yhdessä toimimista ja suullista kommunikaatiota, on taustalla kuitenkin voimakkaasti yksilöt. Näiden yksilöiden pitää olla kyvykkäitä toimimaan yksin, ottamaan valtaa ja vastuuta sekä olemaan myös luovia ja innovatiivisia, jotta kehitettävä tuote kehittyy. Tämän lisäksi heidän pitää olla kompetentteja suhteessa projektissa vaadittuun teknologiaan ja työkaluihin. Tämä asettaa suuret vaatimukset projektiin valittavilta henkilöiltä. Heidän pitää olla edellä mainittujen asioiden lisäksi sitoutuneita ja motivoituneita heille asetettuihin tavoitteisiin nähden ja tehdä sekä ylläpitoa että tutkimus- ja tuotekehitystä. Silti monet tutkimukset painottavat, että nämä menetelmät lähtevät suunnittelijoista itsestään, ja he haluavat toimia juuri näin. On mahdollista, että valitsemalla yritykseen suunnittelijoiden itsensä valitsema tapa toimia, saadaan esimerkiksi henkilöstön tyytyväisyyskyselyiden tilannetta parannettua. Asiakastyytyväisyyskyselyn tulokseen tällä uskoakseni on myös vaikutusta. Paine kohdistuu myös linjaorganisaation kykyyn rekrytoida, johtaa ja resursoida näitä henkilöitä parhaalla mahdollisella tavalla.

Useissa menetelmissä kollektiivinen vastuunotto on perusperiaatteena. Itse en näe kollektiivista vastuu pelkästään positiivisena asiana, vaan ennemminkin olen alalla vallinneen teesin ”jaettu vastuu on vastuuta ei-kenelläkään” kannalla. Uskon että, tiimi yhdessä saa aikaan enemmän kuin yksilö, mutta uskon ja kokemuksesta tiedän, että esimerkiksi asiakas, sisäinen asiakas tai vianhallinta menettely edellyttää, että on joku yksi kontaktipiste tai olemassa oleva henkilö kenelle voi lähettää sähköpostia tai tarvittaessa jopa soittaa. [Agile manifesto, 2001]

5. Yhteenveto ja jatkotutkimus

Näyttää siltä että, ohjelmistotuotannon prosessien kehittämisen saralla käynnissä oleva työ johtaa olemassa olevien menetelmävariaatioiden yhdistymiseen parhaimmaksi todetuksi toimintatavaksi. Vastaavia ilmiöitä on tietojenkäsittelytieteessä koettu aiemminkin esimerkiksi UML-standardin syntymänä. Tätä harmonisointia tukee myös alueella tehty tutkimustyö. Ketterät menetelmät ovat syntyneet tarpeesta ja palvelevat edelleen kattaen perinteisissä menetelmissä olevia puutteita. Olemassa olevat ketterät menetelmät kaipaavat hieman lisää konkretiaa ja tarkemmin määritettyä suhdetta ympäristöönsä, jotta sopivin menetelmä voidaan valita olemassa olevien vaatimusten toteuttamiseksi.

Ketterät menetelmät kaipaavat myös lisää tukea projektinhallinnalle [Abrahamsson et al., 2003]. Ilman tätä ketterät menetelmät eivät ole uskottavia eivätkä saavuta niiden, mielestäni ansaitsemaa, asemaa osana ohjelmistotuotannon prosesseja. Näiden lisäksi asiakas on saatava vakuutetuksi valitun menetelmän tuloksista lisäarvona. Myös asiakkaan kokemukset ketterien menetelmien käytöstä olisi mielenkiintoista jatkotutkimusta esimerkiksi CEM (*Customer Exprience Management*) lähtökohdista [Abrahamsson et al., 2003].

Viiteluettelo

[Abrahamsson et al., 2003]Pekka Abrahamsson, Juhani Warsta, Mikko T. Siponen and Jussi Ronkainen. New Directions on Agile Methods: A Comparative Analysis. VTT Electronics, Oulu, 2003.

[Abrahamsson et al., 2002]Pekka Abrahamsson, Outi Salo, Jussi Ronkainen and Juhani Warsta. Agile software development methods. Review and analysis. VTT Publications, Espoo, 2002.

[Agile manifesto, 2001] Manifesto for Agile Software Development 2001 Available as http://www.agilemanifesto.org. Checked 30.10.2006.

[Haikala ja Märijärvi, 2003] Ilkka Haikala ja Jukka Märijärvi. Ohjelmistotuotanto. Talentum, Helsinki, 2003.

[Hirsch, 2002] Michael Hirsh. Making RUP Agile. ACM Press, New York, 2002.

[Järvinen ja Järvinen, 2000] Pertti Järvinen ja Annikki Järvinen. Tutkimustyön metodeista. Opinpajan kirja, Tampere, 2000.

[IBM, 2005] IBM, RUP Online. Rational Unified Process 2005 Version 7.0.1. Available as: http://www-304.ibm.com/jct03002c/software/awtools/rup/idex.html>. Checked 22.11.2006.

[IEEE 610.12] ANSI/IEEE standard 610.12-1990. Glossary of Software Engineering terminology. Available as http://standards.ieee.org/catalog/olis/se.html.

[Krebs, 2005] Joe Krebs. RUP in the dialoque with Scrum. Available as: http://www.controlchaos.com/module/RationalEdge0205.pdf. Checked 28.12.2006.

[Lindberg, 2003]Harri Lindberg. Extreme Programming. Pro gradu-tutkielma, Tampereen yliopisto, Tietojenkäsittelytieteiden laitos, 2003.

[Raymond, 1997] Eric Raymond. The Cathedral and the Bazaar. Available as: http://www.catb.org/~esr/writings/cathedral-bazaar/. Checked 20.12.2006.

[Scrum, 2006] Scrum: It's about Common Sense. Available as: <URL: http://www.controlchaos.com/>. Checked 8.10.2006.

[Schmitt, 2003] Bernd H. Schmitt. Customer Experience Management: a Revolutionary Approach to Connecting with Your Customers. John Wiley & Sons, Hoboken, New Jersey, USA.

[SFS 1981] Suomen Standardisoimisliitto. Projektitoiminta, toimintaverkkosanasto ja piirrosmerkit. Suomen Standardisoimisliitto, 1981.

[Standish Group, 2002] Standish Group International. What are the requirements? In: CHAOS Report 2003. Available as: http://cclamino.dibe.unige.it/xp2002/talksinfo/johnson.pdf. Checked 26.12.2006.

[Suominen, 2003] Arto Suominen. Riskienhallinta. WSOY, Helsinki.

[UML, 2004] Object Management Group. UML-Unified Modeling Language Version 2.0. Available as: http://www.omg.org/. Checked 27.12.2006.

[Weinberg, 1982] Gerald Weinberg. Rethinking Systems Analysis & Design. Dorcet House Publishing, New York, 1982.

Anonymiteetti World Wide Webissä

Markus Kotilainen

Tiivistelmä

Tutkielmassa käsitellään erilaisia menetelmiä, joilla yksittäinen käyttäjä voidaan tunnistaa verkkosivulla ja kerätä hänestä tietoja, sekä menetelmiä, joilla tehdä käyttäjästä anonyymimpi ja pitää käytettyjen verkkosivujen sisältö salaisena. Tutkielmassa käsitellyt anonyymisointimenetelmät riittävät käyttäjän identiteetin ja luettujen verkkosivujen piilottamiseen hyvin pitkälle, sillä menetelmiä vastaan esitetyt hyökkäykset ovat luonteeltaan enemmän teoreettisia ja vaativat hyökkääjältä mittavia resursseja.

Avainsanat ja -sanonnat: WWW, Internet, anonymiteetti, tietosuoja, sensuuri, sananvapaus
CR-luokat: C.2.6, K.4.1, K.5.2

1. Johdanto

Internet ja erityisesti World Wide Web ovat nykyään olennainen osa arkipäivää ja varsinkin WWW:n käyttö on syrjäyttämässä monia perinteisiä kommunikaatiomuotoja. WWW tarjoaakin monia etuja verrattuna esimerkiksi kirjeeseen tai puhelinsoittoon; kirjeeseen verrattuna kommunikaatiossa ei ole viiveitä ja puhelinsoittoon verrattuna voidaan välittää tietoa muilla tavoilla kuin äänen avulla. WWW myös mahdollistaa kommunikaation useaan eri lähteeseen yhtä aikaa.

Ihmiset pitävät WWW:tä anonyyminä mediana, mikä tulee hyvin ilmi lukiessa mitä tahansa julkista keskustelupalstaa. Kuitenkin kirjeeseen tai puhelinsoittoon verrattuna WWW:n käyttö on huomattavasti julkisempaa. Tässä tutkielmassa käyn läpi yleisiä uhkia, joita liittyy WWW:n käyttöön, sekä erilaisia tapoja tehdä kommunikaatiosta yksityisempää ja anonyyminpää.

Anonymiteetti tietoverkoissa voidaan jakaa kolmeen luokkaan [13]:

- Vastaanottajan anonymiteetti
- Lähettäjän anonymiteetti
- Lähettäjän ja vastaanottajan yhdistämättömyys (yhteyden anonymiteetti).

Tutkielma käsittelee vastaanottajan ja yhteyden anonymiteettiä, mutta osaa käsiteltävistä tekniikoista voidaan käyttää myös lähettäjän anonyymisointiin.

Anonymiteetin tarve voi tuntua keinotekoiselta ja jopa vainoharhaiselta. Jotkin maat, kuten Kiina ja Iran, harjoittavat kuitenkin jo nyt valtiollista Internet-

sensuuria. Myös muualla maailmassa on peräänkuulutettu joidenkin sivustojen sensurointia joko valtion tai Internet-operaattoreiden toimesta. Vaikka usein halutaan sensuroida sivustoja joko lapsipornon tai terrorismin vuoksi, kumpikaan termi ei välttämättä ole selkeästi määriteltävissä ja sensuurin alle voi joutua sivustoja, joita ei sinne kuuluisi.

Vuonna 2003 tehdyssä tutkimuksessa [14], jossa haastateltiin yhdysvaltalaisten korkeakoulujen tietojärjestelmävastaavia, 13% vastaajista rajoitti pääsyä sivustoille kategorian, kuten pornografian, perusteella, ja 15% rajoitti pääsyä sivustoille tapauskohtaisesti. Vaikka luvut eivät ole kovin suuria, sensuuria harrastetaan siis nykyään jo länsimaissakin.

Myös erilaisten Internetissä toimivien tukiryhmien, kuten addiktio- ja hyväksikäyttötukiryhmien, käyttäjät haluavat ymmärrettävästi pysyä anonyymeinä, samoin kuin ihmiset, jotka raportoivat yrityksen tai valtiojohdon väärinkäytöksistä organisaation sisältä. Tässä tutkielmassa esiteltävät tekniikat sopivat sekä käyttäjän anonyymisointiin että sensuurin kiertämiseen. On todennäköistä, että WWW-käytön yhä yleistyessä tarve anonymiteetille ei tule ainakaan vähenemään.

2. World Wide Webin toiminnasta

2.1. TCP/IP-verkkojen perusteita

Internetin perustan muodostaa TCP/IP-protokollat [18]. Protokollat käyttävät hyväkseen pakettikytkentäisyyttä, jossa lähetettävä tieto jaetaan pienempiin osiin eli paketteihin, jotka sitten lähetetään erikseen kohteeseen. Tämän ansiosta paketit voidaan tarvittaessa lähettää eri reittejä, jos jossain kommunikaation vaiheessa jokin reitti lakkaa toimimasta. Toisaalta ei voida aina taata että kaikki paketit saapuvat oikeassa järjestyksessä tai lainkaan perille.

IP on verkkokerroksen [25] protokolla, joka vastaa pakettien toimittamisesta kohteeseensa. Jokainen IP-paketti pitää sisällään sekä lähde- että kohdeosoitteen. Nykyisin useimmiten käytetyissä IPv4-verkoissa se on 32-bittinen luku, joka esitetään usein neljänä kokonaislukuna väliltä 0-255 pisteillä erotettuna, kuten 123.10.113.254. Modeemi- ja laajakaistayhteyksissä IP-osoite on yleensä dynaaminen eli se voi vaihtua eri yhteyskertojen välillä, mutta käytännössä, etenkin laajakaistaa käytettäessä, osoite vaihtuu hyvin harvoin. Vaikka yhden julkisen IP-osoitteen alla voi olla useampi tietokone, yleensä osoite riittää melko pitkälti käyttäjän tunnistamiseksi ja osoitteen perusteella voidaan myös päätellä tietokoneen fyysinen sijainti maan ja jopa paikkakunnan osalta.

TCP on kuljetuskerroksen [25] protokolla, jonka päälle on rakennettu lähes kaikki Internetiä käyttävät protokollat kuten HTTP, SMTP, FTP ja SSH. TCP on yhteydellinen protokolla, jossa kahden koneen välille muodostetaan yhteys, jota pidetään auki kommunikaation ajan. Tämä mahdollistaa kaksisuuntaisen

kommunikaation. TCP numeroi paketit lähtemisjärjestyksessä, joten tieto saadaan palautettua alkuperäiseen muotoon, vaikka paketit eivät saapuisi samassa järjestyksessä. Vastaanottaja lähettää jokaisesta paketista kuittauksen, joten jos kuittausta ei saada määräajassa lähettäjä voi lähettää paketin uudelleen. TCP-paketteihin lisätään myös tarkistussummat, jotta siirrossa mahdollisesti syntyneet virheet voitaisiin huomata.

UDP on yhteydetön kuljetuskerroksen [25] protokolla, jolla voi siirtää tietoa vain yhteen suuntaan. TCP:n verrattuna sen käyttö on kuitenkin huomattavasti nopeampaa, koska ei tarvitse erikseen muodostaa yhteyttä tiedonsiirtoa varten ja vastaanottajan ei tarvitse kuitata saapuvia paketteja. Toisaalta ei ole mitään taetta siitä, että tieto siirtyy vastaanottajalle. Vaikka UDP:tä käytetään paljon vähemmän kuin TCP:tä, hyvin yleinen käyttö sille on DNS-pyynnöt. DNS [7] mahdollistaa sanallisten verkkotunnusten, kuten www.example.com, muuntamisen IP-osoitteiksi.

2.2. HTTP ja evästeet

HTTP [11] on sovelluskerroksen [25] protokolla, jota käytetään tiedonsiirtoon WWW-palvelimen ja selaimen välillä. HTTP käyttää suojaamatonta telnet-yhteyttä [19] porttiin 80, tosin yhteys voidaan myös tarvittaessa salata [21]. Tätä salausta pidetään tarpeeksi tehokkaana mm. verkkopankkikäyttöä varten. Tyypillisesti yhteys selaimen ja WWW-palvelimen välillä muodostuu seuraavasti:

- Käyttäjä kirjoittaa osoitteen osoiteriville tai seuraa hyperlinkkiä. Jos selaimen välimuistista ei löydy palvelimen verkkotunnusta vastaavaa IP-osoitetta, lähetetään DNS-pyyntö ja odotetaan vastausta.
- Muodostetaan telnet-yhteys WWW-palvelimen porttiin 80.
- Selain lähettää palvelimelle tiedon mm. siitä, minkä sivun käyttäjä haluaa ladata, mikä selain on kyseessä, mistä tälle sivulle päädyttiin ja minkälaista kommunikaatiota selain tukee. Tässä pyynnössä lähetään palvelimen mahdollisesti asettamien evästeiden sisältö.
- Palvelin lähettää ensin HTTP-vastauksen, josta tulee mm. ilmi, onko sisältö ladattavissa ja kuinka pitkään sisältö pysyy samana. Vastauksessa voidaan lähettää evästeitä.
- Vastauksen jälkeen lähetetään haettava sisältö, jos mahdollista.
- Suljetaan telnet-yhteys.

Evästeet [10] ovat tekstitiedostoja, joilla voidaan säilyttää tietoja usean eri sivun välillä. Ilman evästeitä tietojen säilyttäminen olisi vaikeaa, koska jokaisen sivun latauksen jälkeen yhteys suljetaan. Eväste sisältää vähintään jonkin avaimen ja sen sisältämän arvon. Tämän lisäksi se voi sisältää voimassaoloajan, verkkotunnuksen, polun, tiedon siitä, tuleeko eväste lähettää vain salattua yhteyttä käytettäessä ja versionumeron, jota käytetään hyvin harvoin. Jos evästeen verkkotunnus on .example.com ja polku /, lähetetään eväste palvelimelle ladat-

taessa mitä tahansa sivua jonkin example.com:in alitunnuksen alta, kuten foo.bar.example.com. Jos tunnus taas on foo.example.com ja polku /bar/, lähetetään eväste vain ladattaessa sivua, jonka osoitteen alkuosa on foo.example.com/bar/. Voimassaoloajan päätyttyä selaimen tulisi poistaa eväste. Jos muita arvoja kuin avain ja sen arvo ei ole määritelty, oletusarvoisesti verkkotunnus on nykyisen osoitteen verkkotunnus, polku on / ja evästeen voimassaoloaika päättyy, kun selain suljetaan. HTTP-vastauksen lisäksi evästeitä voidaan asettaa ja muokata Javascriptillä.

Evästeet voidaan jakaa kolmeen ryhmään:

- istuntoevästeet
- pysyvät evästeet
- kolmannen osapuolen evästeet.

Istuntoevästeet ovat lyhytaikaisia evästeitä, joita käytetään käyttäjän tunnistamiseksi istunnon ajan, kuten verkkokauppaa tai -pankkia käytettäessä. Niiden voimassaolo loppuu, kun selain suljetaan, eikä niiden käyttöön yleensä liity tietosuojariskejä. Pysyviä evästeitä käytetään usein asetusten ja käyttäjätunnuksien säilyttämiseen, ja niiden voimassaoloaika voi olla useita vuosia. Niiden avulla voidaan esimerkiksi poistaa kirjautuminen verkkosivuilta. Voimassaoloajan vuoksi niitä voidaan myös käyttää käyttäjän seurantaan.

Kolmannen osapuolen evästeet ovat pysyviä evästeitä, jotka asettaa jokin kolmas osapuoli, yleensä internetmainontaa harjoittava yritys. Vaikka monet haittaohjelmia poistavat ohjelmat varoittavat näistä evästeistä, ne eivät kuitenkaan ole varsinaisia haittaohjelmia vaan tekstitiedostoja. Koska sama yritys voi näyttää mainoksia usealla eri sivustolla, on mahdollista seurata käyttäjän selaustottumuksia näiden sivustojen välillä. Mainosbannerit tulevat usein mainostajan palvelimilta, joten ne voivat asettaa omia evästeitä jotka lähetetään aina mainosta ladattaessa. Joissain selaimissa on nykyään mahdollista kieltää kolmannen osapuolen evästeet ja estää joitain verkkotunnuksia asettamasta evästeitä.

3. Käyttäjän seuranta WWW-sivuilla

Nykyään on yhä yleisempää, että käyttäjän pitää rekisteröityä sivustolle päästäkseen lukemaan sisältöä. Vaikka rekisteröityminen onkin käyttäjälle usein nopeaa ja vaivatonta, rekisteröityessä saatetaan kysyä tietoja, jotka eivät millään tavalla liity sivustoon. Käyttäjän kirjauduttua palveluun hänestä voidaan alkaa kerätä myös muita tietoja mm. selaustottumusten perusteella, ilman käyttäjän varsinaista suostumusta tai tietämystä [24]. Pääosa seurantatavoista liittyy evästeiden käyttöön, mutta ilman evästeitäkin käyttäjä voi olla tunnistettavissa.

3.1. Seuranta evästeiden avulla

Evästeet ovat luettavissa olevaa tekstiä, mutta niiden sisältö on usein sellaisessa

muodossa, että se ei ole tulkittavissa tietämättä palvelinohjelmiston toimintaa. Tämän vuoksi ei voida aina sanoa, mihin tarkoitukseen evästettä käytetään, mutta pelkästään evästeiden määrä ja voimassaoloaika kertonee silti jotain.

Eniten käytettyjen sivustojen määrittäminen on hankalaa, joten Alexa.com:in [1] listatauksesta valittiin kymmenen käytetyintä verkkotunnusta maailmanlaajuisesti. Tulos ei välttämättä vastaa täysin käyttäjälukuja, mutta se lienee ainakin suuntaa antava. Jokaisesta verkkotunnuksesta avattiin etusivu Firefox-selaimella vanhojen evästeiden poiston jälkeen ja katsottiin lisättyjen evästeiden sisältö.

	istunto	pysyvä	muut	pisin aika
Yahoo	0	3	5	02.06.2037
MSN	2	19	5	04.10.2021
Google	0	2	0	17.01.2038
Baidu	0	1	0	08.12.2036
Myspace	1	5	0	18.01.2038
www.qq.com	1	4	3	01.01.2046
YouTube	2	2	1	05.12.2016
Windows Live	5	2	1	01.01.2021
Orkut.com	1	0	2	istunto
www.sina.com.cn	1	2	23	05.12.2016

Taulukko 1. Evästeet suosituimmilla sivustoilla maailmanlaajuisesti

Taulukossa 1 istunto-sarakkeessa on istuntoevästeiden määrä, pysyvä-sarakkeessa pysyvien evästeiden määrä, muut-sarakkeessa kolmannen osapuolen evästeiden määrä ja pisin aika-sarakkeessa pisin sivuston, ei kolmannen osapuolen, asettamien pysyvien evästeiden voimassaoloaika. Kuten taulukosta käy ilmi, pysyvien evästeiden voimassaoloaika on hyvin pitkä, eikä istuntoevästeitä käytetä yhtä paljon. Kolmannen osapuolen evästeitä on myös suurimmalla osalla sivuista, tosin osassa sivustoista kolmas osapuoli oli saman yrityksen omistama toinen verkkotunnus.

Myös suomalaisista verkkotunnuksista otettiin kymmenen suosituinta [1], jotka eivät sisälly maailmanlaajuisesti suosituimpiin verkkotunnuksiin ja suoritettiin edellisen kaltainen koe. Kokeen tulokset on kerätty Taulukkoon 2.

	istunto	pysyvä	muut	pisin aika
IRC-galleria	0	1	8	12.01.2007
Suomi24	0	0	5	-
Iltalehti	0	1	4	12.12.2011
Ilta-Sanomat	0	0	9	-
Wikipedia	0	0	0	-
Yle	0	0	2	-
Huuto.Net	2	0	3	istunto
Helsingin Sanomat	2	3	4	18.01.2038
MTV3 Internet	0	2	4	12.12.2011
Saunalahti	1	0	0	istunto

Taulukko 2. Evästeet suosituimmilla sivustoilla Suomessa

Verrattuna kansainvälisesti suosituimpiin sivustoihin suomalaisten käyttä-

mät sivustot asettavat paljon vähemmän sekä istunto- että pysyviä evästeitä. Lähes jokainen sivusto sisältää silti kolmannen osapuolen evästeitä, joista moni on saman yrityksen asettama. Huomattavaa kuitenkin on, että kaikki kolmansien osapuolien evästeitä sisältävät sivustot sisälsivät evästeen statistik-gallup.net:istä, joten käyttäjän selaustottumuksia voidaan seurata näiden sivustojen välillä.

3.2. Muita seurantamenetelmiä

Ulkopuolisen on mahdotonta tietää, mitä WWW-palvelin tekee sivua ladattaessa, joten ei voida sanoa, mitä kaikkia tietoja yksittäisestä kävijästä kerätään. Ilman evästeitäkin käyttäjästä on silti saatavilla melko paljon tietoa.

Jokaisen pyynnön mukana lähetetään käyttäjän IP-osoite, jota voidaan käyttää käyttäjän tunnistamiseksi, ja mm. Google käyttää tätä hyväkseen näyttäessään käyttäjälle lokaalin version sivuistaan maan perusteella. Pyynnöstä tulee myös ilmi käyttäjän käyttämä selain ja käyttöjärjestelmä, joskin nämä ovat käyttäjän muokattavissa, sekä tieto siitä, miltä sivulta tälle sivulle päästiin. Tämän ansiosta käyttäjän selaustottumuksia voidaan selvittää vähän myös ilman evästeitä.

Suurin osa selainliikenteestä on salaamatonta, joten käyttäjän lataamien sivujen sisältö on luettavissa verrattain helposti, jos voidaan tarkkailla liikennettä jossain reitityspisteessä selaimen ja WWW-palvelimen välillä. Vaikka selainliikenne salattaisiinkin, DNS-kutsut ovat salaamattomia, joten käytetylle DNS-palvelimelle menee tieto siitä, mitä verkkotunnuksia käytetään.

Erilaiset selainlaajennukset, kuten ActiveX, Java ja Flash, voivat myös kertoa käyttäjän järjestelmästä ja käyttäjästä tietoja, joita ei muuta kautta saataisi. Laajennuksissa olevat tietoturva-aukot voivat myös tehdä käyttäjästä haavoittuvan.

4. Välityspalvelimet

4.1. Välityspalvelimien käyttö

Välityspalvelimet mahdollistavat epäsuorat yhteydet palvelimen kautta. WWW:tä käytettäessä selain ottaa yhteyden välityspalvelimeen, joka sitten muodostaa uuden yhteyden WWW-palvelimeen. Tämä mahdollistaa sivujen sisällön tallentamisen välityspalvelimen muistiin lataamisen nopeuttamiseksi ja toisaalta sisältöä voidaan myös muokata tai sensuroida. Useat yritykset sallivat yhteydet Internetiin vain välityspalvelimen kautta.

Jos välityspalvelin ei kerro kohdepalvelimelle mitään tietoja sitä käyttävistä asiakkaista, kyseessä on ns. anonyymi välityspalvelin. Nämä voivat olla joko WWW-pohjaisia [16] kuten the-cloak.com tai tavallisia välityspalvelimia, joita käytetään muokkaamalla joko selaimen tai käyttöjärjestelmän verkkoasetuksia.

WWW-pohjaiset anonyymit välityspalvelimet muokkaavat sivulla esiintyviä hyperlinkkejä siten, että osoitteen alkuun tulee käytetyn välityspalvelimen osoite ja käytetyn CGI-ohjelman polku. Esimerkiksi the-cloak.com:ia käytettäessä *http://example.com* muuttuu muotoon *http://the-cloak.com/Cloaked/+cfg=32/http%3A//example.com/*. Käytetystä palvelusta riippuen evästeitä ei tallenneta tai ne tallennetaan sellaisessa muodossa, että sivusto toimii normaalisti, vaikka evästeet ovatkin peräisin välityspalvelimelta. WWW-pohjaiset anonyymit välityspalvelimet voivat myös poistaa mainoksia, muuttaa käyttäjästä ilmoitettavia selain- ja käyttöjärjestelmätietoja ja poistaa HTTP-REFERER-kentän HTTP-pyynnöistä, joka kertoo, miltä sivulta nykyiselle sivulle päästiin [16].

Perinteisemmät anonyymit välityspalvelimet eivät muokkaa linkkejä, vaan reitittävät WWW-liikenteen itsensä kautta ja väärentävät käyttäjän IP-osoitteen. Tällöin evästeet tallennetaan käyttäjän koneelle kuten tavallista yhteyttä käytettäessä.

Sekä WWW-pohjaisista että tavallisista välityspalvelimista on saatavilla lukuisia listauksia, mm. Open Directory Projectista [4,9].

4.2. Välityspalvelimien haitat

Vaikka välityspalvelimet tarjoavat helpon tavan tehdä kommunikaatiosta anonyyminpää, niiden käyttöön liittyy silti omat ongelmansa. Suurin osa WWW-pohjaisista välityspalvelimista ei tarjoa salattua yhteyttä ilmaiseksi, joten käyttäjän lataamien sivujen sisältö on luettavissa, jos voidaan tarkkailla verkkoliikennettä välityspalvelimen ja käyttäjän välillä [16]. Tämän lisäksi kohdepalvelimen osoite ja polku on osa ladattavan sivun osoitetta, joten analysoimatta sivun sisältöä voidaan selvittää, mikä sivu ladattiin [13].

Käyttösopimuksesta riippuen suurin osa WWW-pohjaisista välityspalvelimista säilyttää lokitietoja käyttäjästä jonkin aikaa [20]. Jos välityspalvelin takavarikoidaan tai joutuu tietomurron kohteeksi tänä aikana, käyttäjän selaushistoria on ulkopuolisen osapuolen käytettävissä [13].

Vaikka liikenne sekä käyttäjältä välityspalvelimelle että välityspalvelimelta WWW-palvelimelle olisi salattu, on mahdollista, että ulkopuolinen hyökkääjä voi tarkkailla molempia liikenteitä. Tällöin, etenkin jos WWW-palvelimelta ei ole paljoa muuta liikennettä, voidaan ilman salauksen purkua käyttää ajoitushyökkäystä analysoimaan, miltä palvelimilta käyttäjä hakee tietoa [16].

Tämän lisäksi jotkut sivustot estävät yhteydet tunnetuista WWW-pohjaisista välityspalvelimista. Tavallista välityspalvelinta käytettäessä DNS-pyynnöistä tulee ilmi, mitä osoitteita käytetään, vaikka WWW-palvelin ei voikaan yksilöidä käyttäjää.

Suurin ongelma välityspalvelimien käytössä on kuitenkin luottamus. Käyttäjän on mahdotonta tietää, mitä tietoja välityspalvelin tallentaa ja valvotaanko yhteyksiä. Internetistä löytyvään satunnaiseen IP-osoitteeseen voi olla vaikea

luottaa. Toisaalta suurin osa WWW-pohjaisista anonyymisointipalveluista kertoo käyttösopimuksissaan, että he luovuttavat välittömästi viranomaisen tai tuomioistuimen pyytämät lokitiedot [20]. Välityspalvelimien käyttöä voikin suositella vain, jos käyttäjän identiteetin tai ladattavien verkkosivujen paljastumisesta ei ole paljoa haittaa.

5. Tor

5.1. Sipulireitityksen toiminta

Välityspalvelimien puutteet anonyymisoinnissa ovat olleet jo kauan tiedossa, joten paremmalle järjestelmälle oli olemassa tarve. Ensimmäinen artikkeli, jossa kuvattiin sipulireititystä, ilmestyi vuonna 1996 [8].

Alkuperäisessä sipulireitityksessä kommunikaation alkuvaiheessa käyttäjä ottaa yhteyden reitityssolmuun, joka toimii myös välityspalvelimena. Tämä solmu määrittää käytetyn reitin kohdeosoitteeseen ja on täten myös haavoittuvin mahdollisille hyökkäyksille, joten anonymiteetistään huolehtivan käyttäjän tulee ajaa tätä solmua omalla tietokoneellaan. Reitin viimeisen solmun tulee myös toimia välityspalvelimena. Reittivalinnan jälkeen solmu muodostaa nk. sipulin, joka sisältää käytetyn reitin. Tämä sipuli salataan jokaisen solmun julkisella avaimella käänteisessä järjestyksessä viimeisestä (ulostulo-) solmusta alkaen ja lähetetään reittiä pitkin yhteyden muodostamiseksi [8].

Sipuli sisältää yksittäiselle solmulle seuraavat tiedot [8]:

- seuraavan solmun osoite ja sipulin voimassaoloaika
- kaksi symmetristä salausfunktiota ja -avainta
- komento, tunnus ja sipulin salattu loppusisältö.

Reitinmuodostuksen jälkeen reitti on käytössä sipulin voimassaoloajan verran. Salausfunktioita ja -avaimia käytetään reitin muodostuksen jälkeen tietoliikenteessä, toista funktio-avain -paria käytetään liikenteessä kohti ulostulosolmua ja toista kohti lähtösolmua. Komento voi olla *create*, *destroy* tai *data*. Sipulin mukana lähetettävä komento on *create*, jonka saadessaan solmu valitsee tunnuksen ja lähettää tämän tunnuksen sipulin mukana seuraavalle solmulle. Solmu tallentaa myös saamansa tunnuksen ja lähettämänsä tunnuksen parina. Yhteyden ollessa käytössä solmu lähettää toiselta tunnukselta saamansa tiedon automaattisesti toiselle tunnukselle. Koska sipulia ”kuorittaessa” sen sisältö lyhenee, saatuaan sipulista tarvittavat tiedot jokainen solmu lisää sen loppuun satunnaista tietoa kuoritun kerroksen verran, jotta seuraava solmu ei tiedä, kuinka monta solmua ketjussa on vielä jäljellä. Ainoastaan viimeinen solmu on tietoinen roolistaan, koska seuraavan solmun osoite on tyhjä [8].

Lähetettäessä tietoa kohti ulostulosolmua välityspalvelimena toimiva ensimmäinen solmu salaa viestin käänteisessä järjestyksessä ulostulosolmusta alkaen käyttäen lähetettyjen salausfunktioiden käänteisfunktioita. Tällöin jokai-

nen solmu kuorii salauksesta kerroksen pois, kunnes ulostulosolmussa viesti on salaamaton. Vastaavasti lähetettäessä tietoa kohti ensimmäistä solmua jokainen solmu salaa viestin toisella salausfunktiolla ja -avaimella. Ensimmäinen solmu käyttää sitten ko. funktioiden käänteisfunktioita ensimmäisestä solmusta alkaen saadakseen viestin siinä muodossa, kuin se oli ulostulosolmussa [8].

Sipulireititys ei tarjoa täydellistä vastaanottajan anonymiteettiä, koska voidaan todeta, että käyttäjä on lähettänyt tai vastaanottanut tietoa. Käyttäjän hakemien verkko-osoitteiden selvittäminen on kuitenkin paljon hankalampaa, koska liikenne reititetään usean eri solmun kautta, ja tiedon salauksen ansiosta kommunikaation sisältöä on vaikea tulkita.

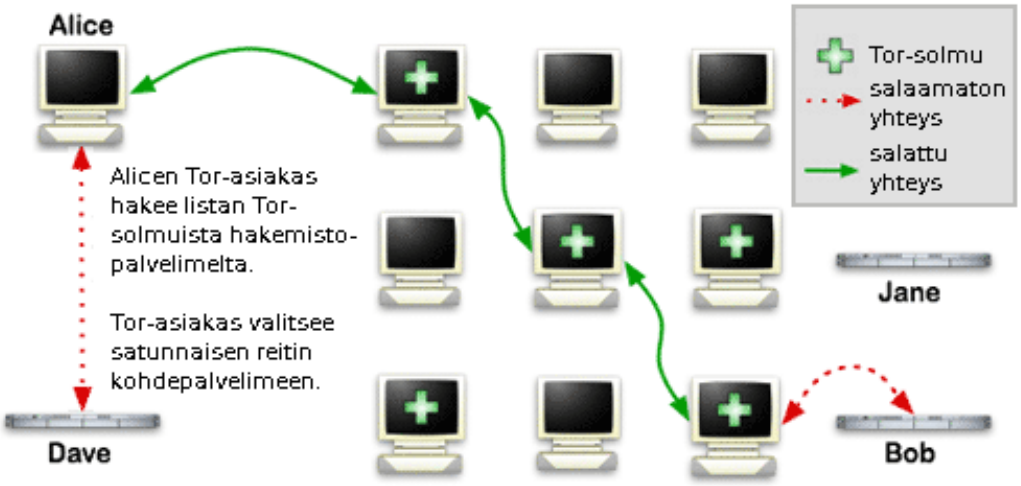
5.2. Tor ja Privoxy

Vaikka alkuperäinen sipulireititys olikin huomattava parannus tavallisiin välityspalvelimiin verrattuna, siinä oli lukuisia puutteita. Yksi näistä oli, että jokaista sovellusprotokollaa varten piti tehdä oma välityspalvelimensa, jonka vuoksi useat protokollat jäivät vaille tukea.

Tor [6] ei tue mitään yksittäistä sovellusprotokollaa, vaan toimii yleisenä SOCKS-välityspalvelimena [17], jonka kautta voidaan reitittää suurin osa TCP:tä tukevista protokollista. Alkuperäiseen sipulireititykseen verrattuna Tor tarjoaa myös mm. seuraavia etuja:

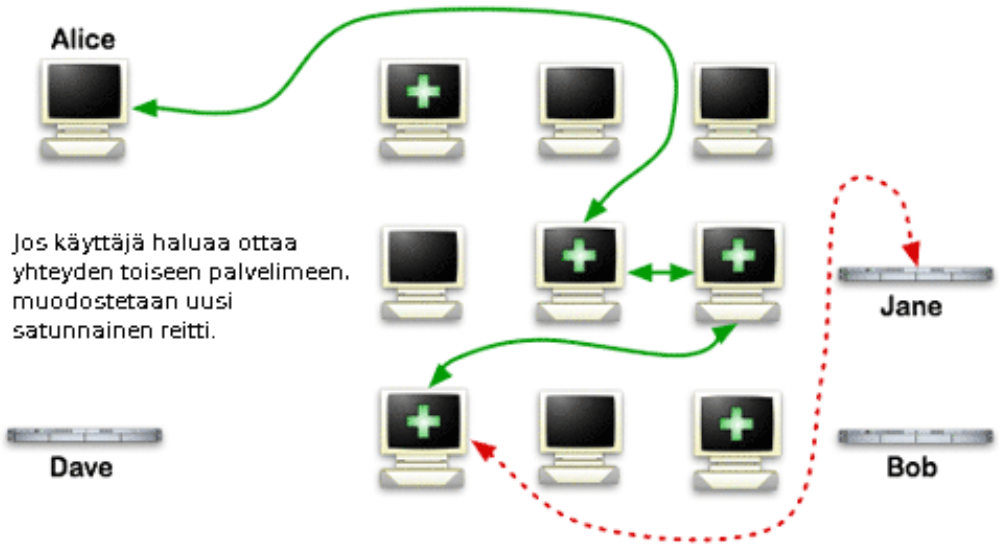
- Yhteyttä muodostettaessa jokaiselta solmulta pyydetään istuntoavain, jota käytetään julkisen avaimen sijaan. Tällöin yksittäinen solmu ei voi tallentaa liikennettä ja myöhemmin murtautumalla toisiin solmuihin saada selville käytettyjä salausfunktioita ja -avaimia ja niiden avulla purkaa salausta.
- Tor käyttää luotettuja solmuja hakemistopalvelimina, joilta käyttäjä saa tarvittaessa allekirjoitetun listan saatavilla olevista Tor-solmuista.
- Alkuperäinen sipulireititys ei sisältänyt minkäänlaista eheystarkistusta, joten yksittäinen solmu pystyi muokkaamaan siirrettyä sisältöä, esimerkiksi ensimmäinen solmu saattoi vaihtaa kommunikaation kohdeosoitetta. Tor varmistaa siirretyn tiedon eheyden.
- Tor-solmut ovat vapaaehtoisten ylläpitämiä, joten ei voida olettaa että jokainen solmu sallisi kaikenlaista liikennettä. Tämän vuoksi solmun ylläpitäjä voi päättää, mihin osoitteisiin ja portteihin verkkoyhteyksiä sallitaan [6].

Tor oli aluksi Yhdysvaltain laivaston tukemaa tutkimustyötä, joka siirtyi Electric Frontier Foundationin tukemaksi projektiksi vuonna 2004. Vaikka EFF ei enää anna projektille rahallista tukea, se tarjoaa projektille edelleen kotisivutilaa. Ohjelman jatkokehitys tapahtuu vapaaehtoisvoimin. Tyypillinen Tor-istunto tapahtuu Kuvassa 1 esitetyllä tavalla.



Kuva 1. Tor-solmujen haku ja reitinmuodostus

Kuvassa 1 esitetään kuinka Tor-asiakas hakee ensin listan voimassaolevista Tor-solmuista hakemistopalvelimelta käyttäen salaamatonta yhteyttä. Tämän jälkeen muodostetaan reitti kohdepalvelimeen, oletusarvoisesti kolmen Tor-solmun kautta. Lukuun ottamatta yhteyttä viimeisestä solmusta kohdepalvelimelle, yhteydet ovat salattuja. Kuten alkuperäisessä sipulireitityksessä, Tor-solmut ovat tietoisia vain edellisestä ja seuraavasta solmusta, eivät välttämättä kummastakaan loppupisteestä.



Kuva 2. Reitinmuodostus toiseen palvelimeen
(Kuvat osoitteesta http://tor.eff.org/overview.html.en, käytetty ja muokattu tekijän luvalla.)

Tor pitää yhteyden voimassa vain noin minuutin ajan. Tämän jälkeen muodostetaan uusi reitti kuten Kuvassa 2, jotta käyttäjän seuraaminen olisi hanka-

lampaa. Verrattuna alkuperäiseen sipulireititykseen Tor tarjoaa paljon pienemmät vasteajat, joten sitä voidaan käyttää paremmin esim. pikaviestimiin ja IRC-keskusteluun. Tor antaa käyttäjilleen myös mahdollisuuden pitää anonyymiä palvelinta käyttäen .onion-pseudoverkkotunnusta, johon voi muodostaa yhteyden vain Tor-verkon kautta.

Koska Tor reitittää vain TCP-liikennettä, DNS-kutsut kulkevat tavallista reittiä ja tämän vuoksi mahdollinen hyökkääjä voi saada selville mihin palvelimiin käyttäjä ottaa yhteyden. Tämän kiertämiseksi käyttäjä voi käyttää ohjelmia, jotka tukevat SOCKS5:tä, jolloin DNS-kutsut lähtevät ulostulosolmusta. WWW-selauksessa on käytetään kuitenkin useimmiten Privoxy-välityspalvelinta [15], joka voi reitittää DNS-kutsut Tor:in kautta. Privoxy osaa myös poistaa WWW-sivuilta mainoksia, evästeitä ja suojata paremmin käyttäjän yksityisyyttä. Privoxyn tai vastaavan HTTP-välityspalvelimen käyttö on muutenkin pakollista WWW-selauksessa, koska Tor on SOCKS-välityspalvelin.

Tor ja Privoxy ovat molemmat avoimen lähdekoodin ohjelmia, joten ainakin periaatteessa ne ovat luotettavampia kuin suljetut anonyymisointipalvelut, koska lähdekoodi on kenen tahansa nähtävissä.

5.3. Torin haitat

Koska Tor tarjoaa pienempiä vasteaikoja kuin alkuperäinen sipulireititys, se on, ironista kyllä, alttiimpi ajoitushyökkäyksille. Ajoitushyökkäys onnistuu todennäköisemmin, jos kohdepalvelimelta ei lähde paljoa muuta liikennettä. Tällöin käyttäjän yhdistäminen kohdepalvelimeen on mahdollista tilastollisella analyysillä [2]. Myös tarkkailemalla kohdepalvelimen ja käyttäjän liikennemääriä voi olla mahdollista havaita, että käyttäjä otti yhteyden palvelimeen [2].

Perinteisillä liikenneanalyysitekniikoilla mahdollisen hyökkääjän tulisi pystyä tarkkailemaan kaikkia mahdollisia reitityspisteitä. Koska Tor-solmut sijaitsevat ympäri maapalloa, tämä ei ole realistinen uhka. On kuitenkin osoitettu [12], että jos voidaan tarkkailla vain osaa verkosta ja kohdepalvelinta, pystytään määrittämään melko tarkkaan, minkä solmujen kautta käyttäjän liikenne kulkee. Vaikka tämä ei poista yhteyden salausta, se vähentää käyttäjän anonymiteettiä. Samoja tekniikoita käyttäen on myös mahdollista yhdistää useampi erillinen yhteys käyttäjään [12].

Koska Tor-solmujen ylläpitäjät ovat vapaaehtoisia, solmujen olemassaolo ei ole itsestäänselvyys. Joulukuun 2006 alussa solmuja oli kuitenkin olemassa jo yli tuhat, joskaan kaikki eivät olleet aktiivisia [22].

Toria kehitetään jatkuvasti. Vaikka ohjelma on nykyisellään toimiva ja vakaa, sitä ei suositella käytettäväksi, jos anonymiteetin tarve on ehdoton. Uusien ohjelmien asentaminen voi olla tavalliselle käyttäjälle hankalaa. Tämän vuoksi Tor on saatavilla Windows-käyttäjille valmiiksi paketoituna Privoxyn ja Firefox-selaimen kanssa sellaisessa muodossa, jota voidaan ajaa esim. USB-muistis-

ta [23].

6. Anonyymisoinnin haitat

Kuten mitä tahansa teknologiaa, myös anonyymisointitekniikoita voidaan väärinkäyttää, kuten vandalisoimalla julkisia keskustelupalstoja. Tätä tapahtuukin siinä määrin, että monet sivustot estävät käytön tunnettujen välityspalvelimien kautta. Esimerkiksi Wikipedia estää artikkeleiden muokkauksen tunnetuista Tor-solmuista. Anonyymisointitekniikoita käytetään myös mm. tietomurtoihin ja lapsipornon levitykseen.

Anonyymit yhteydet ovat hitaampia käyttää kuin tavalliset verkkoyhteydet, joten sivujen navigointi ei ole yhtä jouhevaa. Evästeiden estäminen saattaa myös estää joitakin sivustoja toimimasta normaalisti.

Davenport [5] esittää, että anonyymi kommunikaatio tulisi hylätä ja peräänkuuluttaa kommunikaation vastuullisuutta. Hänen mukaansa sananvapauden ja oikeusjärjestelmän kannalta on oleellista, että tarvittaessa kommunikaation lähde voidaan laittaa vastuuseen, vaikka kommunikaatiota ei tulisi valvoa ilman oikeuden päätöstä.

7. Yhteenveto

Täydellistä anonymiteettiä tietoverkoissa on vaikeaa, ellei mahdotonta, saavuttaa. Käyttäjä voi kuitenkin halutessaan tehdä itsestään tarpeeksi anonyymin, että käyttäjän tunnistaminen vaatii paljon resursseja ja aikaa. Onko tähän aihetta onkin eri kysymys.

Maissa, joissa on käytössä valtiollinen Internet-sensuuri, anonyymisointitekniikat ovat välttämättömiä vapaan tiedonkulun turvaamiseksi. Anonyymisointi voidaankin nähdä teknologisena keinona levittää sananvapautta ja demokratiaa silloin, kun poliittiset keinot eivät toimi. Länsimaisen käyttäjän näkökulmasta anonyymisointi voi tuntua tarpeettomalta, mutta sensuuria harrastetaan länsimaissakin, eikä sen määrä todennäköisesti tule vähentymään.

Tavalliselle käyttäjälle pakollisista rekisteröitymisistä ja kolmannen osapuolen evästeistä ei ole mitään hyötyä. Kukaan tuskin haluaa osallistua jatkuvaan markkinatutkimukseen ilman minkäänlaista korvausta. Rekisteröityminen voidaan kuitenkin helposti kiertää [3], eikä kolmannen osapuolen evästeiden estäminen yleensä vaikuta sivuston toimintaan.

Väärinkäyttöpotentiaalista huolimatta anonyymisointitekniikat ovat hyvin tärkeitä yksilön- ja sananvapauden kannalta. Niiden jatkokehitys on ehdottoman tärkeää, koska vanhoja tekniikoita vastaan löydetään koko ajan uusia hyökkäyksiä.

Viiteluettelo

[1] Alexa, *http://www.alexa.com*, [1.12.2006]

[2] Oliver Berthold, Hannes Federrath, Marit Köhntopp, Project "anonymity and unobservability in the internet". In: *Proceedings of the Tenth Conference on Computers, Freedom and Privacy* (Apr. 2000), 57-65.

[3] Bugmenot.com, *http://www.bugmenot.com* [1.12.2006]

[4] CGI proxy list, *http://dmoz.org/Computers/Internet/Proxying_and_Filtering/Hosted_Proxy_Services/Free/CGI_Proxy/* [1.12.2006]

[5] David Davenport, Anonymity on the Internet: why the price may be too high. *C. ACM* **45**, 4 (Apr. 2002), 33-35.

[6] Roger Dingledine, Nick Mathewson and Paul Syverson, Tor: the second-generation onion router. In: *Proceedings of the 13th USENIX Security Symposium* (Aug. 2004), 303-320.

[7] Domain name system, *http://en.wikipedia.org/wiki/DNS*, [1.12.2006]

[8] David M. Goldschlag, Michael G. Reed and Paul F. Syverson, Hiding routing information. In: *Proceedings of Workshop on Information Hiding*. Springer-Verlag (1996), 137-150.

[9] HTTP proxy list, *http://dmoz.org/Computers/Internet/Proxying_and_Filtering/Hosted_Proxy_Services/Free/Proxy_Lists/* [1.12.2006]

[10] HTTP State Management, *http://www.ietf.org/rfc/rfc2965.txt*, [1.12.2006]

[11] Hypertext Transfer Protocol – HTTP 1.1, *http://www.w3.org/Protocols/rfc2616/rfc2616.html*, [1.12.2006]

[12] Steven J. Murdoch and George Danezis, Low-cost traffic analysis of Tor. In: *Proceedings of the 2005 IEEE Symposium on Security and Privacy* (May 2005), 183-195.

[13] Rolf Oppliger, Privacy protection and anonymity services for the World Wide Web. *Future Generation Computer Systems* **16**, 4 (Feb. 2000), 379-391.

[14] Graham A. Peace, Balancing free speech and censorship. *C. ACM* **46** (Nov. 2003), 104-109.

[15] Privoxy, *http://www.privoxy.org* [1.12.2006]

[16] Anna M. Shubina and Sean W. Smith, Using caching for browsing anonymity. *ACM SIGE* **4**, 2 (Jun. 2003), 11-20.

[17] Socks Protocol version 5, *http://www.faqs.org/rfcs/rfc1928.html* [1.12.2006]

[18] William Stallings, *Data and Computer Communications*. Pearson Education, 2004.

[19] Telnet Protocol Specification, *http://www.ietf.org/rfc/rfc0854.txt*, [1.12.2006]

[20] the-cloak terms and conditions, *http://www.the-cloak.com/terms.html* [1.12. 2006]

[21] The TLS Protocol, *http://www.ietf.org/rfc/rfc2246.txt*, [1.12.2006]

[22] Tor network status, *https://tns.nighteffect.com/* [1.12.2006]

[23] Torpark, *http://torrify.com* [1.12.2006]

[24] Thomas Wright, Security, privacy, and anonymity. *ACM Crossroads* **11**, 2 (Winter 2004).

[25] Hubert Zimmermann, OSI reference model – the ISO model of architecture for open systems interconnection. *IEEE Transactions on Communications* **28**, 4 (Apr. 1980), 425-432.

Käyttäjäkeskeisyys tietojärjestelmäprojekteissa

Jouni H. Laitinen

Tiivistelmä.
Yhä useampi ihminen on tekemisissä tietojärjestelmien kanssa päivittäin sekä työssään että vapaa-ajalla. Ihminen tietojärjestelmän käyttäjänä on tietojärjestelmien olennaisimpia osia. Tietojärjestelmän käyttäjä on se, joka on tekemisissä järjestelmän kanssa jatkuvasti ja näin ollen järjestelmän on tällöin vastattava hänen tarpeitaan. Täten on loogista, että käyttäjät osallistuvat tietojärjestelmien kehittämiseen.

Tämä tutkielma sijoittuu tietojärjestelmien suunnittelun ja kehittämisen alueelle ja sen tarkoituksena on esitellä joitakin käyttäjäkeskeisen tietojärjestelmien suunnittelun ja kehityksen periaatteita sekä erilaisia tapoja toteuttaa käyttäjien osallistumista tietojärjestelmäprojekteihin.

Avainsanat ja -sanonnat: Käyttäjäkeskeisyys, tietojärjestelmien suunnittelu, suunnittelumenetelmät
CR-luokat: H.1.2, H.4.2

1. Johdanto

Tietojärjestelmä käsitteenä on kirjallisuudessa määritelty useilla tavoilla. Tässä yksi osuva määritelmä tietojärjestelmälle:

Tietojärjestelmä on

1. ihmisistä, tietojenkäsittelylaitteista, tiedonsiirtolaitteista ja ohjelmista koostuva järjestelmä, jonka tarkoitus on tietoja käsittelemällä tehostaa ja helpottaa jotakin toimintaa tai tehdä toiminta mahdolliseksi.
2. abstrakti systeemi, jonka muodostavat tiedot ja niiden käsittelysäännöt [Iivari et al., 2001].

Määritelmällisesti ihmiset ovat osa tietojärjestelmiä, näin ollen käyttäjä muodostaa olennaisen osan tietojärjestelmää. Useimmat tietojärjestelmät on rakennettu ihmisten käytettäväksi ja näin ollen on loogista, että järjestelmien tulisi ominaisuuksiltaan vastata käyttäjien tarpeita. Näitä järjestelmän käyttäjän tarpeita voidaan pyrkiä tyydyttämään käyttäjäkeskeisen tietojärjestelmäsuunnittelun avulla. Käyttäjäkeskeistä tietojärjestelmäsuunnittelua voidaan lähestyä useista eri näkökulmista ja suunnittelun toteuttamiseksi on olemassa useita erilaisia valmiita metodeja ja suunnittelumenetelmiä.

Tietojärjestelmien käyttäjäkeskeistä suunnittelua voidaan tarkastella erilaisten lähestymistapojen kautta. Nämä eri lähestymistavat katsovat tietojärjestelmäsuunnittelua kokonaisuutena ja tätä kautta myös siihen liittyvää käyttäjälähtöisyyttä eri näkökulmista.

2. Käyttäjäkeskeisen tietojärjestelmäsuunnittelun periaatteita

Käyttäjien osallistuminen on tärkeä tekijä tietojärjestelmäprojektien onnistumisessa. Käyttäjien osallistuminen tarkoittaa Ivesin ja Olsonin [1984] mukaan tietojärjestelmän tarkoitetun käyttäjäryhmän edustajien osallistumista järjestelmän kehittämisprosessiin. Debrabander ja Edstrom [1977] toteavat, että käyttäjien osallistuminen on muihin tekijöihin, kuten ylimmän johdon tukeen, tavoitteiden asetteluun tai tietojenkäsittelyhenkilökunnan pätevyyteen, verrattuna ainoa, joka on säännönmukaisesti ja merkittävästi vaikuttamassa projektien lopputuloksen laatuun. Johdon tietojärjestelmiä koskevassa kirjallisuudessa pidetään lähes aksioomana sitä, että käyttäjien osallistuminen on edellytys tietojärjestelmien menestyksekkäälle kehittämiselle [Ives and Olson, 1984].

Tietojärjestelmien käyttäjäkeskeisen suunnittelun ja kehittämisen toteuttaminen käytännössä vaatii tiettyjen periaatteiden noudattamista suunnittelu- ja kehittämistyössä. Jotkin periaatteet ovat erityisen tärkeitä ja niiden noudattaminen käytännön työssä ohjaa kehittämistyötä käyttäjäkeskeiseen suuntaan.

Gulliksen ja muut [2003] esittävät 12 käyttäjäkeskeisen suunnittelun avainperiaatetta, joiden noudattaminen tai noudattamatta jättäminen vaikuttaa tietojärjestelmäprojektin käyttäjäkeskeisyyden onnistumiseen. Nämä 12 periaatetta ovat [Gulliksen et al., 2003]:

1) Suunnittelun fokus on käyttäjissä

2) Käyttäjien aktiivinen osallistuminen

3) Evolutionäärinen järjestelmäkehitys

4) Yksinkertaiset suunnitelmaesitykset

5) Protoilu

6) Käytön arviointi kontekstissa

7) Eksplisiittiset ja tietoiset suunnittelutoimet

8) Ammattimainen asenne

9) Käytettävyysmestareiden käyttö

10) Holistinen suunnittelu

11) Prosessien kustomointi

12) Käyttäjäkeskeisen asenteen aikaansaaminen ja säilyttäminen suunnittelutyössä.

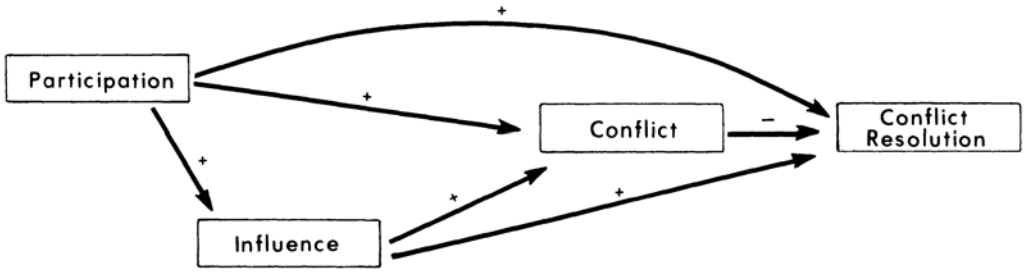
Näiden 12 avainperiaatteen perusteella tietojärjestelmän tavoitteiden, työympäristön ja käyttökontekstin tulisi ohjata kehitystyötä alusta alkaen. Käyttäjien tavoitteet, tehtävät ja tarpeet on otettava huomioon, minkä vuoksi edustavan otoksen järjestelmän käyttäjäjoukosta tulisi aktiivisesti osallistua kehitystyöhön jatkuvasti koko kehitysprosessin ajan ja koko järjestelmän elinkaaren läpi [Gulliksen et al., 2003].

Edellä esiteltyjen periaatteiden nojalla tietojärjestelmäsuunnittelun tulee olla holistista, jotta käyttäjäkeskeisyyttä voidaan pitää merkittävänä lähtökohtana kulloinkin kyseessä olevan järjestelmän suunnittelussa. Käyttäjäkeskeisyys lähtökohtana vaikuttaa tietojärjestelmän joka osa-alueeseen ja näin ollen sen on oltava mukana projektien alusta loppuun asti.

3. Prosessimallit ja käyttäjäkeskeisyys

Tietojärjestelmäprojektin elinkaarta voidaan kuvata erilaisten prosessimallien avulla. Vastaavia prosessimalleja voidaan käyttää myös esim. erilaisissa ohjelmistotuotannon projekteissa. Debrabanderin ja Edstromin [1977] mukaan tällaiset prosessia eteenpäin vievien toimintojen rakenteet ovat seurausta eri sidosryhmien välisestä kommunikaatiosta. He toteavat, että mitä tehokkaampaa tämä kommunikaatio on, sitä tarkoituksenmukaisempi on sen synnyttämä toimintorakenne ja tämän rakenteen toteuttaminen käytännössä.

Konstruktiiviseen konfliktiin perustuvan tietojärjestelmien kehitysmallin käsitteen esitti alun perin Deutsch [1969]. Mallilla voidaan hallita tilanteita, joissa tietojärjestelmien arviointiperusteita on useita ja tietojärjestelmän kanssa tekemisissä olevilla ihmisillä on toisistaan poikkeavia tavoitteita. Malli painottaa järjestelmän epäkohtien nostamista esiin konfliktien kautta, jotka näin kannustavat ongelmien korjaamiseen. Kuviossa 1 on kuvattu sitä, kuinka käyttäjien osallistuminen ja heidän vaikutuksensa järjestelmään saavat aikaan konflikteja, jotka sitten ratkaistaan. Kuvion nuolet kuvaavat kausaalisia suhteita ja plus- ja miinusmerkit oletettua vaikutusta. Kaikissa muissa suhteissa vaikutukset ovat positiivisia, mutta konfliktit vaikuttavat negatiivisesti niiden ratkaisuun [Robey and Farrow, 1982].



Kuvio 1. Konstruktiiviseen konfliktiin perustuva tietojärjestelmien kehitysmalli. [Robey and Farrow, 1982]

Gulliksen ja muut [2003] esittävät, että tietojärjestelmäprojekteissa tulisi käyttää evolutionääristä järjestelmän kehitysmallia. Evolutionäärinen malli on yhtä aikaa sekä iteratiivinen että inkrementaalinen. Malli mahdollistaa järjestelmän kehittämisen siten, että kehitystyö on jatkuvaa ja järjestelmä mukautuu käyttäjien kulloisiinkin tarpeisiin.

Käytännön näkökulmasta tietojärjestelmän tärkeimpien vaatimusten tulee täyttää kolme kriteeriä. Niiden on 1) tuotava parannuksia tarkoitettujen käyttäjien työhön, 2) oltava teknisesti toteutuskelpoisia ja 3) tehtävä se kustannustehokkaalla tavalla [Butler et al., 2000].

Nämä kolme kriteeriä tuntuvat loogisilta maalaisjärjelläkin ajateltuna, mutta itse asiassa ne sisältävät monimutkaisia suhteita, joita on vaikea nähdä ennalta. Näiden kriteerien yhteensovittaminen on monimutkaista. Butler ja muut [2000] tarjoavat ratkaisuksi mallia, jossa sovitetaan yhteen käyttäjäkeskeinen ja tekninen suunnittelunäkökulma sovittamalla yhteen näissä käytettyjä mallinnustekniikoita (prosessimallinnus ja tekninen mallinnus).

4. Tietojärjestelmien arviointi ja käyttäjäkeskeisyys

Tietojärjestelmiä arvioitaessa on käytettävissä oltava kriteerit, jotka määrittävät sen, kuinka onnistunut tietojärjestelmä on. Lisäksi arviointiin on oltava sopivat mittarit, jotta näiden kriteerien täyttymistä voidaan mitata.

Edellisessä luvussa esittelin kolme käytännön näkökulmasta tärkeää kriteeriä tietojärjestelmille asetettaville vaatimuksille. Nämä kriteerit voidaan siirtää suoraan koskemaan myös valmista tietojärjestelmää. Tietojärjestelmän on siis 1) tuotava parannuksia tarkoitettujen käyttäjien työhön, 2) oltava teknisesti toteutuskelpoisia ja 3) tehtävä se kustannustehokkaalla tavalla [Butler et al., 2000]. Näin tietojärjestelmiä pystytään arvioimaan esim. näiden kolmen kriteerin pohjalta. Käyttäjäkeskeisyys liittyy näistä kriteereistä ensimmäiseen selvästi kahta jälkimmäistä enemmän.

Melonen [1990] mukaan yleisimmät tietojärjestelmäarvioinnin mittarit ovat käyttäjätyytyväisyys ja järjestelmän käyttö, näistä käyttäjätyytyväisyys tärkeämpänä. Järjestelmän käytön mittaaminen on yleensä operaatioiden määrällistä mittaamista, esim. kuinka monta kertaa tiettyä toimintoa on käytetty tai tietty tiedosto avattu. Käyttäjätyytyväisyys kertoo mittarina enemmän, kun pyritään mittaamaan tietojärjestelmän organisaatiolle tuomia hyötyjä.

Tietojärjestelmän käyttö ja käyttäjätyytyväisyys vaikuttavat suoraan järjestelmästä saataviin hyötyihin organisaatiossa. Käytöllä ja käyttäjätyytyväisyydellä pystytään mittaamaan järjestelmän vaikutusta yksittäisten käyttäjien toimintaan. Yksittäisten käyttäjien toimilla taas on ilmiselvästi vaikutusta koko organisaatioon, joten käytön ja käyttäjätyytyväisyyden käyttäminen organisaation tietojärjestelmästä saaman hyödyn ja tietojärjestelmäprojektin onnistumisen mittarina on perusteltua [DeLone and McLean, 2003].

Bawden [1990] kirjoittaa käyttäjäkeskeisestä tietojärjestelmäarvioinnista. Käyttäjäkeskeisessä tietojärjestelmäarvioinnissa järjestelmää arvioidaan sen todellisessa käyttöympäristössä sen sijaan, että sitä arvioitaisiin erillisenä suljettuna systeeminä. Käyttäjillä tässä yhteydessä tarkoitetaan järjestelmän käyttäjiä siinä mielessä, että he ovat järjestelmän operaattoreita, toimittajia, ylläpitäjiä tms. kuitenkin niin, että käyttäjä on järjestelmää käyttävä informaatioammattilainen, ei siis esim. pankkiautomaattijärjestelmää käyttävä keskivertokansalainen.

Vain käyttäjäkeskeisellä arvioinnilla saadaan esiin järjestelmän tai palvelun todellinen ja käytännöllinen arvo sen oikeassa ympäristössä. Käyttäjäkeskeisessä järjestelmäarvioinnissa on kysymys ennen kaikkea holistisesta lähestymistavasta arviointiin. Arviointi toteutetaan käytössä olevassa järjestelmässä tai kehitteillä olevan järjestelmän prototyypissä. Käytännössä käyttäjäkeskeinen järjestelmän arviointi seuraa järjestelmän testausta laboratorio-olosuhteissa, ts. ”perinteistä” ohjelmistotestausta. Kun laboratorio-olosuhteissa testataan järjestelmän toiminnallisuutta, käyttäjäkeskeisessä arvioinnissa testataan järjestelmän toiminnallisuuden ja käyttäjätarpeiden vastaavuutta. Tämä saattaa johtaa muutoksiin ja parannuksiin järjestelmän toiminnallisuudessa, mikä taas johtaa uuteen laboratoriotestien kierrokseen jne. Arvioinnilla pyritään siihen, että aikaan saadaan konkreettisia parannuksia järjestelmään. [Bawden, 1990]

Bawdenin mallin mukainen käyttäjäkeskeinen arviointi keskittyy järjestelmän yksityiskohtiin ja sen toteuttaminen laajamittaisena vaatisi runsaasti taloudellisia ja henkilöresursseja. Yleensä arviointia toteutetaankin hyvin pienimuotoisesti. [Bawden, 1990]

Yksi näkökulma tietojärjestelmien arviointiin ovat järjestelmien implementaatio-ongelmat. Näiden ongelmien syitä ovat listanneet mm. Iivari ja Hirsch-

heim [1996]. He näkevät ensisijaisina syinä järjestelmien heikon teknisen laadun ja muutosvastarinnan organisaatioissa, teknisten ja sosiaalisten alisysteemien yhteensopimattomuuden sekä sosiaalisen inertian. Näistä syistä jotkut nousevat tärkeämmiksi sen mukaan, mistä näkökulmasta tietojärjestelmiä tarkastellaan. Esim. muutosvastarinta nousee erityisesti esiin, kun tietojärjestelmiä tarkastellaan puhtaan teknisestä näkökulmasta jättäen järjestelmien sosiaaliset ominaisuudet taka-alalle.

Jokisen [2005] mukaan muutosvastarintaa esiintyy organisaatioissa aina, kun organisaation ajetaan sisään jotakin muutosta. Muutosvastarintaa esiintyy aina, riippumatta siitä, onko muutoksesta organisaatiolle hyötyä vai ei. Jokisen mukaan muutosvastarinta ilmenee mm. erilaisten negatiivisten argumenttien etsimisenä, yhteisön ihmiset pyrkivät usein perustelemaan, miksi uusi innovaatio on tarpeeton ennemmin kuin miksi siitä voisi olla jotain hyötyä. Jokinen pitää koulutuksen järjestämistä erittäin tärkeänä muutosvastarinnan välttämisen kannalta [Jokinen, 2005].

5. Käyttäjäkeskeisen tietojärjestelmäsuunnittelun lähestymistavat

5.1. Lähestymistavan ja metodologian käsitteet

Tietojärjestelmäsuunnitteluun on tarjolla monia erilaisia metodeja ja työkaluja. Näistä metodeista ja työkaluista kokoamalla saadaan aikaan hieman laajempia kokonaisuuksia, joita kutsutaan tietojärjestelmien suunnittelumetodologioiksi. Metodologioita voidaan ryhmitellä edustamansa tietojärjestelmäsuunnittelun lähestymistavan mukaan.

Suunnittelun lähestymistavan ja metodologian käsitteellinen ero on siis hierarkkinen. Metodologia tulkitaan käsitteiden, metodien, uskomusten, arvojen ja normatiivisten periaatteiden organisoiduksi kokoelmaksi, jota aineelliset resurssit tukevat, kun taas lähestymistapa tulkitaan koostuvaksi useista metodologioista, jotka jakavat yhteisiä piirteitä [Iivari et al., 2001].

Boahenen [1999] mukaan tietojärjestelmien kehitystoiminta vaatii tiettyjä suuntaviivoja, joita seurata, jotta sen sisältämä työ saadaan ohjattua kohti ennalta asetettua tavoitetta. Jos tietojärjestelmät olisivat luonteeltaan ainoastaan fyysisiä systeemejä, niiden kehittämistyö olisi kohtuullisen yksinkertaista. Tietojärjestelmä on kuitenkin monimutkainen ympäristö kehitettäväksi, koska se sisältää fyysisiä, inhimillisiä ja abstrakteja elementtejä. Lähestymistavan Boahene määrittelee tavaksi, jolla eri elementit toimivat keskenään ”suoriutuakseen” kehitystyöstä.

5.2. Erilaisia lähestymistapoja

Iivari ja Hirschheim [1996] erottavat toisistaan kahdeksan erilaista tietojärjestelmäsuunnittelun lähestymistapaa:

1. Informaation mallinnus (Information Modelling)
   - korostaa organisaation ja informaatioteknologian välistä vuorovaikutusta.
2. Päätöksenteon tukijärjestelmät (Decision support systems)
   - johtajien kontrolloima tukityökalu
   - ensisijaisesti tekninen, mutta sisältää sosiaalisia aineksia.
3. Sosiotekninen (Socio-technical approach)
   - sisältää sekä teknisiä että sosiaalisia aineksia.
4. Infologinen lähestymistapa (Infological approach)
   - teknisen, sosioteknisen ja sosiaalisen näkökulman kompromissi.
5. Interaktionistinen lähestymistapa (Interactionist approach)
   - sosiaalinen näkökulma, jopa poliittinen
   - sosiaalisen inertian korostus.
6. Puhetoimipohjainen lähestymistapa (Speech Act –based approach)
   - tietojärjestelmät ovat teknisesti toteutettuja sosiaalisia systeemejä
   - tietojärjestelmät koostuvat puhetoimista, jotka luovat, asettavat, kontrolloivat ja ylläpitävät organisaation transaktiosopimuksia sekä raportoivat niiden statuksesta.
7. Pehmeä systeemimetodologia (Soft systems methodology)
   - datan manipulointi koneiden tehtävä
   - merkitysten antaminen ihmisen tehtävä.
8. Ammattiliitto-lähestymistapa (Scandinavian trade union approach)
   - tietojärjestelmä työkalu, joka on käyttäjien täydessä kontrollissa,
   - järjestelmän kehitys pitää nähdä osana laajempaa organisaation kehitysprosessia. [Iivari and Hirschheim, 1996]

Hellman [1989] määrittelee joitakin tietojärjestelmäsuunnittelun lähestymistapoja käyttäjäkeskeisestä näkökulmasta. Sosiotekninen lähestymistapa ottaa Hellmanin mukaan lähtökohdaksi työtehtävien suunnittelun. Sen modernit periaatteet sisältävät seuraavat työtehtävien ominaisuudet: vaihtelevuus ja haasteellisuus, jatkuva oppiminen, autonomisuus, tunnustus ja tuki, sosiaalinen osallistuminen sekä haluttavat tulevaisuudennäkymät. Nämä periaatteet johtavat työtehtävien suunnitteluun, joka johtaa haluttuihin työtehtävien ominaisuuksiin yksilö- ja ryhmätasolla. Yksilötasolla on määritelty seuraavia periaatteita: optimaalinen työtehtävien vaihtelu työn sisällä, työtehtävien sellainen rakenne, joka antaa joka työlle yksittäisen ja yleisen tehtävän luonteen, työnkierron optimaalinen pituus, sopiva tapa työn määrän ja laadun arvioimiseksi,

joka tarjoaa myös palautteen tuloksista, myös työtehtäviin liittyvien ja valmistavien tehtävien sisällyttäminen työhön, jonkinasteisen huolellisuuden, taidon, tiedon tai yrityksen sisällyttäminen, joka oikeuttaa kunnioitukseen yhteiskunnassa sekä oma panos tuotteen hyödyllisyyteen loppukäyttäjälle. Sosioteknisen lähestymistavan mukaan tietojärjestelmäympäristö tulisi nähdä työjärjestelmänä kokonaisuudessaan, eikä pelkkänä tietojärjestelmänä. Se korostaa siis sosiaalista järjestelmää sekä sen ja teknisen järjestelmän rajapintaa. [Hellman, 1989]

Hellmanin [1989] tietojärjestelmä työkaluna -näkökulma näkee tietokoneet työkaluina, joilla saadaan tietyt työtehtävät hoidettua. Hellmanin mukaan työkalu on objekti, joka ei pysty toimimaan itsenäisesti ja joka saa arvonsa vasta kun sille annetaan konteksti, jossa sitä käytetään. Tämä konteksti on käyttötilanne. Työkalua voidaan myös kontrolloida ja säätää. Lopputulos voidaan nähdä, ymmärtää ja arvioida, sekä prosessi että lopputulos on jotakin konkreettista ja kouriintuntuvaa. Hellmanin mukaan tietojärjestelmän keskivertokäyttäjä ei ymmärrä tietojärjestelmän rakennetta eikä toimintaa. Tietojärjestelmän käsittäminen työkalun kaltaisena on usein mahdotonta järjestelmien suuren koon ja abstraktisuuden vuoksi. Työkalunäkökulmaan sopivat järjestelmät ovat pienikokoisia. [Hellman, 1989]

Päätöksenteon tukijärjestelmät ovat Hellmanin [1989] mukaan sinällään yritys luoda käyttäjäkeskeisiä tietojärjestelmiä. Ne pyrkivät tuottamaan tietoa käyttäjien tekemien päätösten tueksi. Järjestelmän on tuettava rakenteisia ja rakentumattomia päätöksiä, päätöksentekoa kaikilla organisaatiotasoilla ja tasojen välistä integraatiota, päätöksentekijöiden välistä kommunikaatiota, kaikkia päätöksenteon vaiheita ja vaiheiden välistä vuorovaikutusta, erilaisia päätöksentekoprosesseja olematta riippuvainen mistään yksittäisestä prosessista sekä olla helppokäyttöinen ja helposti muunneltavissa tilanteissa, joissa järjestelmän käyttäjä, tehtävä tai ympäristö muuttuu. [Hellman, 1989]

Hellman [1989] näkee myös käyttäjätuen yhtenä lähestymistapana käyttäjäkeskeiseen tietojärjestelmäkehitykseen. Se edustaa näkemystä, että paras tapa ottaa käyttäjä huomioon on tarjoamalla konkreettista tukea järjestelmän käyttöön ja sen ymmärtämiseen.

Iivari ja Hirschheim [1996] tarkastelevat tietojärjestelmiä niiden organisatorisesta roolista sekä tietovaatimuksista käsin. Tietojärjestelmän organisatorista roolia tarkasteltaessa voidaan heidän mukaansa erottaa kolme eri näkökulmaa: tekninen, sosiotekninen ja sosiaalinen. Samoin tietojärjestelmän tietovaatimuksista puhuttaessa voidaan erottaa kolme eri näkökulmaa: objektiivinen, subjektiivinen ja intersubjektiivinen [Iivari and Hirschheim, 1996]. Nämä näkökulmat edustavat samalla myös näkökulmia käyttäjäkeskeisyyden tarkasteluun.

6. Käyttäjien osallistuminen tietojärjestelmäprojekteihin

Käyttäjien on mahdollista osallistua tietojärjestelmäprojekteihin sen kaikissa vaiheissa alusta loppuun saakka. Hartwick ja Barki [1994] jakavat käyttäjien osallistumisen tietojärjestelmäprojekteihin kahtia. He erottavat toisistaan käyttäjien osallistumisen projektiin fyysisellä ja psykologisella tasolla. Fyysisen tason osallistumisella he tarkoittavat 1) vastuullista osallistumista, kuten esim. projektin johtamista, 2) käyttäjän ja tietojärjestelmän väliseen suhteeseen ja vaikutukseen liittyvää kehitystyötä, kuten esim. järjestelmän kehitystyön suunnittelua ja tällaisten suunnitelmien arviointia ja 3) konkreettista kehitystyötä, esim. ohjelmointityötä. Fyysisen tason osallistumisesta he käyttävät termiä *user participation*. Psykologisen tason osallistumisella he tarkoittavat sitä, millainen on käyttäjien asenne tietojärjestelmää kohtaan. Asenne muodostuu kahdesta tekijästä. Nämä tekijät ovat tietojärjestelmän tärkeys (organisaatiolle) ja sen vaikutus käyttäjään henkilökohtaisesti. Psykologisen tason osallistumisesta he käyttävät termiä *user involvement* [Hartwick and Barki, 1994].

Ivesin ja Olsonin [1984] mukaan käyttäjien osallistumisella saadaan parannettua järjestelmän laatua ja käyttäjätyytyväisyyttä. He listaavat osallistumisesta odotettavissa olevia seurauksia järjestelmän laadun ja käyttäjätyytyväisyyden osalta seuraavasti:

Järjestelmän laadusta puhuttaessa käyttäjien osallistuminen

- tarjoaa tietojärjestelmäammattilaisille tarkempaa ja kattavampaa tietoa käyttäjien tietovaatimuksista,
- tarjoaa asiantuntemusta organisaatiosta, jota järjestelmän on tarkoitus tukea, asiantuntemus tästä puuttuu usein tietojärjestelmäryhmästä,
- auttaa välttämään tarpeettomien toimintojen tai toimintojen, joita ei haluta, kehittäminen ja
- parantaa käyttäjien järjestelmän ymmärrystä.

Käyttäjätyytyväisyyden ollessa kyseessä osallistuminen

- auttaa käyttäjiä kehittämään realistisen näkemyksen järjestelmän kyvyistä,
- tarjoaa foorumin suunnittelua koskevista asioista neuvotteluun sekä niitä koskevien erimielisyyksien ratkaisemiseen,
- johtaa siihen, että käyttäjät ottavat järjestelmän omakseen,
- vähentää käyttäjien muutosvastarintaa ja
- sitouttaa käyttäjät järjestelmään [Ives and Olson, 1984].

7. Yhteenveto

Käyttäjät muodostavat olennaisen osa tietojärjestelmää ja käyttäjien huomioonottaminen tietojärjestelmiä rakennettaessa nähdään tietojärjestelmäammattilaistenkin joukossa tärkeänä asiana.

Järjestelmän käyttäjä ei välttämättä omaa tietoa siitä, millä tavalla tietojärjestelmien suunnittelu ja kehittäminen teoreettisella ja käytännön tasolla toimii. Käyttäjällä ei läheskään aina ole käsitystä siitä, mikä tietojärjestelmän kehittämisessä on realistista käytettävissä olevien resurssien puitteissa. Lisäksi kysymys siitä, tarvitseeko järjestelmän käyttäjä todellisuudessa sitä, mitä haluaa, on olennainen. Järjestelmän käyttäjä ei välttämättä olekaan itse paras asiantuntija tunnistamaan omia tarpeitaan ainakaan ilman ulkopuolista apua. Tästä johtuen käyttäjäkeskeisyys vaatii jatkuvaa yhteistyötä tietojärjestelmäammattilaisten ja järjestelmän käyttäjien välillä.

Käyttäjäkeskeisyyden toteuttamiseksi tietojärjestelmäprojekteissa on noudatettava joitakin periaatteita, jotka tarjoavat pohjan käyttäjäkeskeiselle suunnittelulle ja kehittämiselle. Nämä periaatteet voivat olla esim. ne 12 avainperiaatetta, jotka Gulliksen ja muut [2003] esittävät.

Holistinen näkemys tietojärjestelmästä on tärkeä lähtökohta käyttäjäkeskeisessä tietojärjestelmäsuunnittelussa. Suunnittelun lähestymistapa, käytettävät metodologiat ja niiden kautta metodit ja työkalut on valittava siten, että ne mahdollistavat holistisen näkemyksen tietojärjestelmään koko projektin ajan.

Mahdollistamalla käyttäjien osallistuminen tietojärjestelmäprojekteihin saadaan monia etuja. Käyttäjälähtöisyyden näkökulmasta käyttäjien osallistuminen on erittäin tärkeää. Ilman käyttäjien osallistumista käyttäjäkeskeisyyden toteuttaminen perustuu tietojärjestelmäammattilaisten arvauksille. Ilman osallistumista ei käytännössä voida puhua todellisesta käyttäjäkeskeisyydestä.

Viiteluettelo

[Bawden, 1990] David Bawden, *User-oriented Evaluation of Information Systems and Services.* Gower, 1990.

[Boahene, 1999] Michael Boahene, Information systems development methodologies: Are you being served? In: *Proc. 10th Australasian Conference on Information Systems*, 88-99.

[Butler et al., 2000] Keith A. Butler, Ali Bahrami, Chris Esposito and Ron Hebron, Conceptual models for coordinating the design of user work with the design of information systems. *Data & Knowledge Engineering* **33** (2000), 191-198.

[Debrabander and Edstrom, 1977] Bert Debrabander and Anders Edstrom, Successful information system development projects. *Management Science* **24**, 2 (1977), 191-199.

[DeLone and McLean, 2003] DeLone and McLean, The DeLone and McLean model of information systems success: A ten-year update. *Journal of Management Information Systems* **19**, 4 (2003), 9-30.

[Deutsch, 1969] M. Deutsch, Productive and destructive conflict. *J. Social Issues* **25** (1969), 7-42.

[Gulliksen et al., 2003] Jan Gulliksen, Bengt Göransson, Inger Boivie, Stefan Blomkvist, Jenny Persson and Åsa Cajander, Key principles for user-centered systems design. *Behaviour & Information Technology* **22**, 6 (Nov.-Dec. 2003), 397-409.

[Hartwick and Barki, 1994] John Hartwick and Henri Barki, Explaining the role of user participation in information system use. *Management Science* **40**, 4 (1994), 440-465.

[Hellman, 1989] Riitta Hellman, Approaches to user-centered information systems. University of Turku, Dept. of Computer Science, Report **A 55**, December 1989.

[Iivari and Hirschheim, 1996] Juhani Iivari and Rudy Hirschheim, Analyzing information systems development: A comparison and analysis of eight IS development approaches. *Information Systems* **21**, 7 (1996), 551-575.

[Iivari et al., 2001] Juhani Iivari, Rudy Hirschheim and Heinz K. Klein, A dynamic framework for classifying information systems development methodologies. *Journal of Management Information Systems* **17**, 3 (2001), 179-218.

[Ives and Olson, 1984] Blake Ives and Margarethe H. Olson, User involvement and MIS success: A review of research. *Management Science* **30**, 5 (1984), 586-603.

[Robey and Farrow, 1982] Daniel Robey and Dana Farrow, User involvement in information system development: A conflict model and empirical test. *Management Science* **28**, 1 (1982), 73-85.

Yrityksen verkkosivut markkinointiviestinnän tukena

Eeva Luoto

Tiivistelmä

Tutkimus käsittelee yritysten verkkosivuja ja niiden käyttömahdollisuuksia. Siinä on keskitytty lähinnä verkkosivujen käyttömahdollisuuksiin yrityksen ulkoisessa viestinnässä ja markkinoinnissa. Yrityksillä on ollut omia verkkosivuja jo melko kauan, mutta niiden käyttö ja erilaiset mahdollisuudet markkinoinnin ja brändien rakentamisen osalta kehittyvät koko ajan. Lisäksi tutkimuksessa tarkastellaan erilaisia verkkosivujen tehokkuuden ja toimivuuden mittaustekniikoita.

Avainsanat ja -sanonnat: Internet, verkkosivut, yritykset, markkinointi.
CR-luokat: J.1, K.4

1. Johdanto

Internetillä on jo hyvin suuri levinneisyys etenkin Pohjois-Amerikassa, Euroopassa ja Oseaniassa, ja sen käytön kasvuvauhti on nopea myös muualla maailmassa. Nykypäivän ihminen käyttää päivittäin paljon sähköisiä yhteyksiä ja laitteita, ja tästä ajasta internetin osuus on merkittävä. Tästä kehityksestä johtuen on yritysten alettava yhä paremmin huomiomaan tämä kasvava media, ja parantamaan läsnäoloaan ja viestintäänsä siellä. Tällä hetkellä noin 80 prosentilla suomalaisista yli kymmenen henkeä työllistävistä yrityksistä on omat kotisivut, joten aihe koskettaa suurta osaa yrityksistä [Tilastokeskus, 2006]. Hiljattain tehdyssä vertailussa eurooppalaisten yritysten verkkosivuista paljastui että suomalaisyrityksillä on keskimäärin parhaat yritysverkkosivut Euroopassa, ja useita suuria suomalaisyrityksiä sijoittui vertailun kärkisijoille [Taloussanomat, 2006]. Jatkuvaa kehitystä kuitenkin tarvitaan, koska internet ja markkinointi muuttuvat koko ajan. Tässä tutkimuksessa selvitetään ensin internetin vaatimuksia ja mahdollisuuksia uutena liiketoimintaympäristönä. Sen jälkeen määritellään hyvien yritysverkkosivujen funktioita ja mahdollisuuksia sekä markkinointiviestintää verkossa. Tämän jälkeen käsitellään brändien toimintaa ja rakentamista verkossa, ja lopuksi määritellään hyvien verkkosivujen ominaisuuksia. Tutkimuksen tarkoituksena on selvittää verkkosivujen merkitystä ja tarvetta yrityksille sekä tutkia niiden erilaisia käyttötapoja etenkin markkinointiviestinnän näkökulmasta.

2. Internet: Uusi liiketoimintaympäristö

Internet on yhä vielä verrattain uusi toimintaympäristö perinteisille yrityksille. Tästä johtuen siellä vallitsevat toimintatavat ja sinne parhaiten sopivat toimintamuodot ovat yhä muotoutumassa. Taulukossa 1 esitellään uuden liiketoimintaorganisaation ominaisuuksia verrattuna vanhaan.

Organisaation elementit	Ennen 2000	Jälkeen 2000
Yrityksen visio	Lyhyen aikavälin, tuotekeskeinen	Pitkän aikavälin, asiakaskeskeinen
Organisaatio	Hierarkkinen rakenne	Uusia liiketoimintaprosesseja ja muuttuva rakenne, joiden avulla voidaan hyödyntää IT:n mahdollisuuksia
Myynti	Vähän asiakastietoa, myynti kasvotusten tapahtuvaa	Paljon asiakastietoa, massaräätälöinti, monikanavainen myynti, E-kauppaa
Myynnin jälkeiset toiminnot ja palvelu	Reagoivaa, Erilliset prosessit ja organisaatiot, Hidasta	Ennakoivaa, Integroidut prosessit ja organisaatio, Nopeat vastaukset

Taulukko 1 Uusi liiketoimintaorganisaatio [mukaillen Pétrissans, 1999]

Taulukosta 1 huomataan, että liiketoimintamallit ovat mielipiteestä riippuen joko informaatioteknologian yleistymisestä johtuen tai siitä huolimatta muuttuneet informaatioteknologian laaja-alaisesta käytöstä hyötyvään suuntaan. Asiakastietoa kerätään enemmän sen ollessa mahdollista uusien välineiden kautta, ja tätä tietoa hyväksikäyttäen pystytään erilaistamaan palvelua ja toimimaan asiakaslähtöisemmin. Lisäksi tietotekniikka on nopeuttanut prosesseja, lisännyt sekä asiakkaan että yrityksen tiedon saantia ja monipuolistanut liiketoiminnan kenttää. Tehokkaiden verkkosivujen avulla yrityksen on mahdollista parantaa asiakkaiden ja osakkaiden käsitystä itsestään, kasvattaa asiakasuskollisuutta ja näin ollen helpottaa asiakaslähtöisten e-kaupan strategioiden käyttöä [Heinze and Hu, 2006, p. 324].

Internetillä on monia erityispiirteitä verrattuna perinteisiin medioihin tai muihin toimintaympäristöihin. Näitä erityispiirteitä huomioitaessa liiketoimintaa sopeutettaessa uuteen ympäristöön, niistä merkittävimpiä ovat Brättön ja Gustavssonin [2001] mukaan informaation rikkaus, interaktiivisuus, anonyymius, globaalius, läpinäkyvyys, personointi ja käyttäjän valta. Koska internet on tekstipohjainen media, siellä on helppo esitellä laajemmin informaatiota kuin esimerkiksi televisiossa. Chaston [2001] puolestaan määrittelee internetin

erikoispiirteiksi informaation nopeat saanti- ja siirtomahdollisuudet, paikan ja ajan rajoitteiden puuttumisen, erilaisten asioiden, tapahtumien ja organisaatioiden vertailun helppouden, sekä interaktiivisuuden ja joustavuuden. Internet tarjoaa paljon mahdollisuuksia interaktiivisuuteen ja personointiin toisin kuin perinteiset mediat. Lisäksi se on hyvin läpinäkyvä ja tarjoaa tämän läpinäkyvyyden kautta enemmän valtaa kuluttajalle. Myös anonymiteetti lisää interaktiivisuuden helppoutta ja kasvattaa kuluttajan valtaa. Globaalius tuo asiat lähelle ja siten myös osaltaan edesauttaa läpinäkyvyyden toteutumista. Trendinä tällä hetkellä yritysten verkkosivujen osalta on että ne sisältävät enemmän informaatiota, palveluita sekä interaktiivisia ominaisuuksia kuin aiemmin [Heinze and Hu, 2006].

3. Verkkosivujen funktiot ja mahdollisuudet

Internet tarjoaa uusia mahdollisuuksia kilpailuedun saavuttamiseen sekä uusille että vanhoille yrityksille. Yritysten keskeisiä tavoitteita verkkosivujen käytölle ovat maailmanlaajuinen sähköinen läsnäolo, laajennettu markkina-alue, uudet liiketoimintamahdollisuudet, asiakaspalvelun parantaminen ja elektroninen kaupankäynti [Tao and Tan, 1998]. Nämä ovat kaikki strategisia tavoitteita, joilla on suora yhteys yrityksen menestykseen. Jo pelkkä läsnäolo verkossa lisää yrityksen ja sen tuotteiden tunnettuutta asiakkaiden keskuudessa ja harvalla yrityksellä on varaa jättää tätä jakelukanavaa käyttämättä elektronisen kaupankäynnin lisääntyessä jatkuvasti [Heinze and Hu, 2006]. Internetin yhtenä määräävänä piirteenä on vallan siirtyminen kuluttajalle, ja tästä johtuen kilpailuedun saavuttamiseksi verkossa on erittäin tärkeää, että asiakkaan ja yrityksen suhdetta rakennetaan huolella, jotta se olisi asiakastyytyväisyyden perusta [Gurâu, 2004].

Verkkosivujen funktiot kussakin yrityksessä ovat riippuvaisia monesta seikasta. Esimerkiksi Palmer ja Griffith [1996] ovat sitä mieltä, että verkkosivujen tarkoitus riippuu yrityksen tarjoamasta ja etenkin sen tietointensiivisyydestä. Heidän mielestään yritykset, joiden tuotteen ydinhyöty on informaatiokeskeistä (esimerkiksi vakuutusyhtiö), voivat hyvin ohittaa välikädet ja tuottaa palveluita asiakkailleen suoraan verkkosivujensa kautta, kun taas yritykset, joiden tuote on konkreettisempi (esimerkiksi bensa-asema), tarvitsevat yhä vielä väliporrasta tavoittaakseen asiakkaansa.

Pääasiallisesti verkkosivuja käytetään yrityksen markkinointiviestinnän tukena. Niiden avulla voidaan suorittaa monia markkinoinnin toimintoja kuten esimerkiksi suhdetoimintaa, mainontaa, markkinatutkimusta ja myyntiä. Markkinointiviestinnän työkaluna verkkosivut tarjoavat paljon uusia mahdollisuuksia verrattuna perinteisiin medioihin. Kanso ja Nelson [2004] ovat listanneet

seuraavat verkkosivujen tarjoamat hyödyt: 1. Yritysinformaation rikkaus, 2. Tuoteominaisuuksia koskevan tiedon rikkaus, 3. Päivittämisen helppous, 4. Tiedon keruun mahdollisuus, 5. Maailmanlaajuinen näkyvyys, 6. Räätälöinnin mahdollisuus, 7. Asiakkaan ja yrityksen välisen suhteen kehittäminen, 8. Roolipelien mahdollisuus ja 9. Asioinnin ja ostamisen helppous. Tutkimusten mukaan markkinoijat eivät kuitenkaan osaa hyödyntää kaikkia näitä verkkosivujen tarjoamia etuja, vaan lähinnä keskitytään yritysinformaation tarjoamiseen ja asioinnin ja ostamisen helpottamiseen. Erityisesti maailmanlaajuinen näkyvyys, päivityksen helppous, tiedonkeruun mahdollisuus ja roolipelien mahdollisuus ovat sellaisia ominaisuuksia, joita yritykset hyödyntävät vähän.[Kanso and Nelson, 2004]

Verkkosivuja voidaan myös käyttää ja tarkastella eräänlaisena elektronisena näyteikkunana yritykseen. Tutkimuksilla on osoitettu, että yrityksen verkkosivut vaikuttavat potentiaalisten asiakkaiden mielikuvaan ja käsitykseen yrityksestä, ja näin voivat luoda kilpailuetua [Winter *et al.*, 2003]. Harkitulla verkkosivujen rakentamisella yritys voi luoda itsestään haluamansa vaikutelman asiakkaille ja muille sidosryhmille hyvinkin pienillä kustannuksilla verrattuna esimerkiksi televisiomainontaan tai muunlaisin imagonluontikampanjoihin. Toisaalta pienillä virheillä tai epäjohdonmukaisuuksilla verkkosivujen luonnissa voidaan saada aikaan hyvin huono kuva yrityksestä katsojien mielessä. Tässä tulee useilla yrityksillä vastaan myös resurssien puute. Jos haluttaisiin hyödyntää edellisessä luvussa mainittua globaalia näkyvyyttä, niin myös verkkosivujen suunnittelussa pitäisi ottaa huomioon globaali yhteensopivuus sekä kielen että kulttuuritekijöiden suhteen, jotta ei aikaansaataisi huonoja mielikuvia vieraiden kulttuurien parissa.

4. Markkinointiviestintä ja verkkosivut

Nykypäivänä markkinoinnin vallitsevana suuntauksena on asiakassuhdemarkkinointi, jossa koko yrityksen ja etenkin markkinoinnin lähtökohtana on asiakas ja asiakkaan tarpeet. Tähän suuntaukseen informaatioteknologian hyödyntäminen sopii erittäin hyvin sen edesauttaessa vuorovaikutteisen suhteen rakentamista asiakkaaseen.

4.1. Perinteinen vs. internetin avulla tapahtuva markkinointiviestintä

Perinteiset markkinointikanavat ovat keskittyneet helpottamaan joko tuotteiden tai informaation virtausta, kun taas internetin myötä on syntymässä aivan uudenlainen vuorovaikutteinen markkinointikanava [Holland and Baker, 2001]. Taulukossa 2 on verrattu perinteistä markkinointiviestintää internetin avulla tapahtuvaan markkinointiviestintään.

Perinteiset mediat	Uusi media
Kommunikaatiomallina one-to-one	Kommunikaatiomallina one-to-one tai many-to-many
Massamarkkinointi	Yksilöllinen markkinointi tai massaräätälöinti
Monologi	Dialogi
Brändäys	Kommunikaatio
Tarjontakeskeinen ajattelu	Kysyntäkeskeinen ajattelu
Asiakas kohteena	Asiakas kumppanina
Segmentointi	Yhteisöt

Taulukko 2 Perinteisten ja uuden median erot. [Chaffey *et al.*,2000]

Taulukosta 2 käy ilmi, että vaikka internet onkin teoreettisesti ns. many-to-many -media, niin yrityksen viestinnässä asiakkaalle se parhaiten toimii one-to-one -periaatteella. Internet mahdollistaa personoinnin ja sisällön räätälöinnin yksilöllisesti, sekä lisää asiakkaan mukana oloa brändin luomisessa tehokkaan vuorovaikutteisen kommunikoinnin ja palautekanavien myötä. Myös asiakkaiden tarpeiden huomioon ottaminen helpottuu, ja asiakas toimii enemmänkin kumppanina kuin markkinointitoimenpiteiden kohteena. Internetissä muodostuvien yhteisöjen kautta saadaan helpommin kontaktia suureen joukkoon samanhenkisiä kuluttajia keinotekoisen ja tuotelähtöisen segmentoinnin sijaan. [Chaffey *et al.*, 2000]

4.2. Erilaiset kuluttajat ja viestinnän teoria

Holland ja Baker [2001] ovat luokitelleet kuluttajien orientaatiot netissä kahteen kategoriaan. Kuluttajat, jotka ovat verkossa etsimässä ajanvietettä ja ”surffailemassa”, muodostavat jatkuvien etsijöiden kategorian, johon kuuluville on luonteenomaista etsiä tietoa sen itsensä tai vaikka vain huvin vuoksi. Toiseen ryhmään kuuluvat tehtäväsuuntautuneet kuluttajat, jotka muistuttavat perinteistä oppikirjojen kuluttajaa, joka ratkaisee ongelmia rationaalisesti olemassa olevia resursseja hyödyntäen. Asiakaslähtöisesti toimivan yrityksen pitäisikin muuttaa toimintamallejaan palvelemaan etenkin verkkosivuillaan enemmän myös jatkuvien etsijöiden kohderyhmää, jota ei ole perinteisessä markkinoinnissa otettu huomioon. Singh ja Dalal [1999] esittelevät yksinkertaisen viestinnän mallin, jossa viestinnällä on kaksi eri tehtävää, tiedottaminen ja suostuttelu. Tätä mallia voidaan soveltaa myös verkkosivujen kautta tapahtuvaan viestintään. Sen mukaisesti verkkosivujen olisi tärkeää sisältää sopivaa tietosisältöä ja olevan ulkoisilta tai käyttöominaisuuksiltaan sellainen, että se suostuttelee käyttäjän viipymään sivuilla ja tulemaan sinne uudelleen. Jos näissä keskeisissä tehtävissä epäonnistutaan, kuluttajan on huomattavan helppoa siirtyä seuraa-

ville sivustoille. Yritettäessä taistella kuluttajan ajasta ja huomiosta on tärkeää panna merkille, että kuluttajan ensisilmäyksellä tekemät päätelmät ja niistä seurauksena oleva sivuille jääminen tai sieltä poistuminen ovat pääasiallisesti seurausta nopeasti muodostuvista mielikuvista. Näin ollen onkin erityisen tärkeää kiinnittää huomiota myös houkuttelevuuteen ja sivujen suostuttelutehtävän toteuttamiseen eikä täyttää yrityksen verkkosivuja pelkällä relevantilla tietosisällöllä [Singh and Dalal, 1999]. Kuten perinteiset mainokset, myös verkkosivut voidaan luokitella sen mukaan, kuinka rationaalisia tai emotionaalisia ne ovat. Se, kuinka hyvin rationaalinen tai emotionaalinen mainos tai verkkosivu suoriutuu suostuttelutehtävästään, riippuu taas siitä miten sitoutunut ja motivoitunut kuluttaja on kyseessä. Tässäkin kohtaa kuluttajat verkossa voidaan Singhin ja Dalalin [1999] mukaan jakaa kahteen eri ryhmään, joita ovat surffaajat ja etsijät. Hedonistiset surffaajat etsivät hauskanpitoa ja viihdykettä, ja heihin voidaan vedota emotionaalista lähestymistapaa painottavilla verkkosivuilla, kun taas vahvasti sitoutuneet ja päämäärätietoiset tiedon etsijät ovat huomattavasti paremmin suostuteltavissa rationaalisten ja paljon tietoa sisältävien verkkosivujen avulla. [Holland and Baker, 2001; Singh and Dalal, 1999]

4.3. Asiakaslähtöisen markkinoinnin malli

Kuten edellä on tullut esille, markkinointi ja sen toimintatavat ja pinnalla olevat suuntaukset ovat jatkuvassa muutoksessa, ja näitä muutoksia on pidettävä silmällä myös yrityksen verkkosivuja suunniteltaessa. Taulukossa 3 on esitelty eroavaisuuksia vanhan massamarkkinoinnin mallin sekä uuden asiakaslähtöisen räätälöinnin mallin välillä.

	Vanha massaräätälöinnin ja segmentoinnin malli	Uusi asiakaslähtöisen räätälöinnin malli
Suhde asiakkaisiin	asiakas on passiivinen osallistuja vaihtotapahtumassa	asiakas on aktiivinen ja tuottamassa tuotetta/palvelua yhteistyössä yrityksen kanssa
Asiakkaan tarpeet	selvästi ilmaistut	sekä ilmaistut että peitetyt
Segmentointi	massamarkkinointi ja kohdesegmentit	räätälöintiä ja yksilön segmentointia
Tuote- ja palvelutarjoama	linjalaajennukset ja muunnokset	räätälöidyt tuotteet, palvelut ja markkinointi
Uusien tuotteiden kehitys	markkinoinnin T&K johtaa uusien tuotteiden kehitystä	T&K keskittyy kehittämään alustoja joilla asiakkaat voivat räätälöidä tuotteitaan
Hinnoittelu	kiinteät hinnat ja alennukset	asiakkaiden määrittelemä hinnoittelu, arvoperustainen hinnoittelumalli
Viestintä	mainonta ja PR	integroitua, interaktiivista ja räätälöityä markkinointiviestintää
Jakelu	perinteinen vähittäiskauppa ja suoramarkkinointi	suora (online) jakelu ja kolmansien osapuolien logistiikkapalvelujen nousu
Brändäys	perinteistä brändäystä ja yhteisbrändäystä	asiakkaan nimi brändinä, minun brändini/brändi minulle
Kilpaluedun perusta	markkinavoima	markkinoinnin taidokkuus ja asiakkaan kiinnittäminen partneriksi samalla integroiden markkinointia, operaatioita, T&K:ta ja informaatiota

Taulukko 3 Markkinoinnin muuttuvat kasvot [Wind, 2001]

Taulukosta 3 käy ilmi, mitkä markkinointistrategian ja etenkin verkkosivujen suunnittelun kannalta tärkeitä tekijöitä. Verkkosivujen taitava hyväksikäyttö on yritykselle elintärkeää, jotta se voisi noudattaa näitä uuden markkinoinnin mallin ohjeita. Esimerkiksi asiakkaan tarpeita määriteltäessä olisi hyvä olla askeleen edellä ja tunnistaa asiakkaan vaiettujakin tarpeita ja niitä täyttämällä saavuttaa kilpailuetua. Näiden tarpeiden tunnistamiseksi esimerkiksi erilaisten

internetyhteisöiden ja yrityksen keskustelupalstojen luominen ja niiden seuraaminen on mahdollista. Antamalla asiakkaiden segmentoitua itse esimerkiksi yrityksen lanseeraamien ja sen verkkosivuilta löytyvien yhteisöiden kautta voidaan muodostaa huomattavasti todellisempi kuva asiakkaasta kuin perinteisin segmentointimenetelmin. Verkkosivujen interaktiivisuuden kautta asiakas voi itse osallistua tuotteen, brändin tai palvelun luontiin ja räätälöidä näitä makunsa mukaan. Myös mainonnan, eli tässä tapauksessa yrityksen verkkosivujen, räätälöinnin mahdollisuus auttaa luomaan vahvoja rakenteellisia siteitä, joilla lisätään asiakasuskollisuutta. Verkkosivujen kautta yrityksen on myös helppo tarjota huomattavasti laajempi valikoima palveluja, ja näin edistää asiakkaan oppimista tuotteen käyttöön. Esimerkiksi pyykinpesuaineen verkkosivuilla voi olla pyykinpesuohjeita erilaisiin tilanteisiin, ja näin jälleen saadaan luotua rakenteellisia siteitä asiakkaaseen, joka ongelman kohdatessa tukeutuu yrityksen tarjoamiin lisäpalveluihin.

4.4. Verkkosivujen ja perinteisen mainonnan integrointi

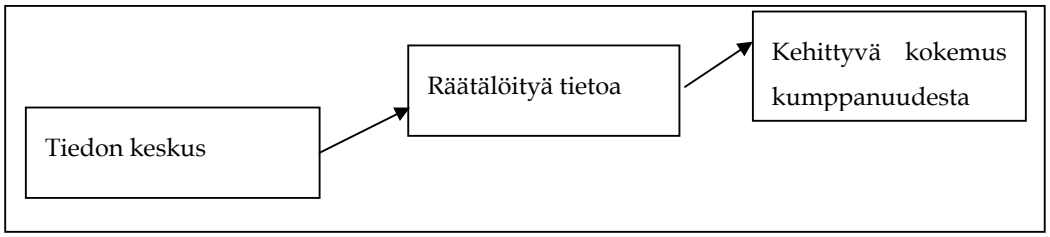
Käyttääkseen hyödyksi uutta teknologiaa ja ollakseen mukana kehityksessä on yritysten integroitava verkkosivunsa osaksi markkinointiviestintästrategiaansa. Verkkosivuista olisi saatava toimiva osa yrityksen markkinointimixiä, eli sen markkinoinnin kilpailukeinojen yhdistelmää. Markkinointimixin tavoitteena on tukea yrityksen liikeideaa, ja sen rakentamisen lähtökohtana ovat yrityksen tavoitteet ja kohdeasiakkaat. Erityisen tärkeää onnistunut integrointi perinteisen mainonnan kanssa on myös siksi, että verkkosivujen hyödyt jäävät vähäisiksi, jos kuluttajaa ei ensin saada löytämään tietään niille [Kanso and Nelson, 2004]. Yritysten verkkosivujen esille tuomisessa ja käyttäjien ohjaamisessa niille on perinteisellä mainonnalla tärkeä rooli. Internet eroaa perinteisestä mainonnasta olemalla ns. pull-media, kun taas perinteinen mainonta on yleensä ns. push-mediaa. Tämä tarkoittaa käytännössä sitä, että esimerkiksi yrityksen verkkosivut eivät työnny kuluttajan kasvoille kadunvarsimainonnassa, vaan hänen täytyy itse löytää tiensä sinne niiden vetovoiman ansiosta. On olemassa kahdenlaisia tapoja edistää ja tukea asiakkaan mahdollisuuksia löytää yrityksen verkkosivut. Voidaan käyttää joko erilaisia portaaleja, kuten hakukoneita ja hakemistoja, tai vaihtoehtoisesti mainostaa yrityksen verkkosivuja ja niiden osoitetta perinteisessä mainonnassa, muilla sivuilla olevien linkkien tai bannereiden kautta [Chaffey *et al.*, 2000].

5. Brändin rakentaminen verkossa

Internet tarjoaa runsaasti mahdollisuuksia brändeille ja niiden rakentamiselle. Lähtiessä viemään tuotteitaan tai palveluitaan internetiin yrityksillä on muu-

tamia erilaisia vaihtoehtoja. Voidaan joko viedä olemassa oleva perinteinen brändi verkkoon sinällään, tehdä perinteisestä brändistä internetiin soveltuva muunnos eli niin sanottu linjalaajennus, ottaa kumppaniksi joku jo olemassa oleva digitaalinen brändi tai luoda täysin uusi digitaalinen brändi [Chaffey *et al.*, 2000]. Toimintatavan valinta on hyvin tapauskohtaista, ja siihen vaikuttavat esimerkiksi brändin soveltuvuus verkkosivuille ja sen nykyaikaisuus sekä yrityksen resurssit ja osaaminen.

Jos ajatellaan yrityksen verkkosivuja brändinä, niin brändien kehitys internetissä on jatkuvaa ja paljon kehitystä on jo tapahtunut alkuvaiheen asetelmista. Kuviossa 1 esitetään brändien kehitystä internetissä.



Kuvio 1 Brändin kehitys internetissä [De Chernatony and McDonald, 2003]

Kuvio 1 havainnollistaa, kuinka alun perin internetsivut tarjosivat pelkkää yrityksen tärkeäksi määrittelemää tietoa, jota kuluttajan oli mahdollista selailla. Seuraavana vaiheena on räätälöity tieto ja sen hakumahdollisuudet, jolla tarkoitetaan esimerkiksi juna-aikataulujen hakumahdollisuutta omien matkustusaikataulujen mukaan. Kolmantena vaiheena on kehittyvä kokemus kumppanina, jolla tarkoitetaan asiakkaan mahdollisuutta olla kumppanina mukana luomassa yrityksen verkkosivujen sisältöä, esimerkiksi keskustelupalstojen tai yhteisöjen kautta, ja vaikuttamassa brändin kehitykseen mielipiteidensä kautta.

Menestyvän brändin määritelmään kuuluu myös vahva brändiuskollisuus. Asiakkaan uskollisuutta verkkosivuille voidaan parantaa monin tavoin ja näin saavuttaa yhä korkeampaa brändiuskollisuutta. Brändiuskollisuutta ja verkkosivujen vetovoimaa parantavat esimerkiksi aiemmin mainittu verkkosivujen personoinnin ja räätälöinnin mahdollisuus, verkossa toimivien yhteisöjen ja keskusteluryhmien ylläpito, palautteen vastaanottaminen ja hyödyntäminen, pelit sekä hypertekstilinkit muihin sivuston osiin [Holland and Baker, 2001].

Internet tuo myös haasteita brändeille ja niiden markkinoinnille. Internetin kehityksen ja sen tarjoaman vuorovaikutteisuuden ja uuden yhteisöllisyyden myötä valta on siirtymässä markkinoilla tuottajalta kuluttajalle. Internet tarjoaa kuluttajalle rajattomasti tietoa, ja siten myös valinnan mahdollisuudet lisääntyvät ja vertailu helpottuu. Etenkin brändien ja erilaisten vähittäiskaupan toimijoiden kohdalla hintavertailun helpottuminen voi aiheuttaa paineita. Hinnoittelu vaikeutuu myös muuten jakelukanavia ollessa huomattavasti enemmän kuin

aiemmin erilaisten huutokauppojen ja muiden kehittyvien kanavien myötä. Myös palvelun laadun tasolle ja sen ylläpidolle internetissä toimiminen aiheuttaa haasteita. Palvelun laatutason valvominen vaikeutuu, ja lisäksi sillä on vaikeampi erottua inhimillisen tekijän puuttuessa. [De Chernatony and McDonald, 2003]

6. Verkkosivujen ominaisuuksien tarkastelu ja tehokkuuden mittaus

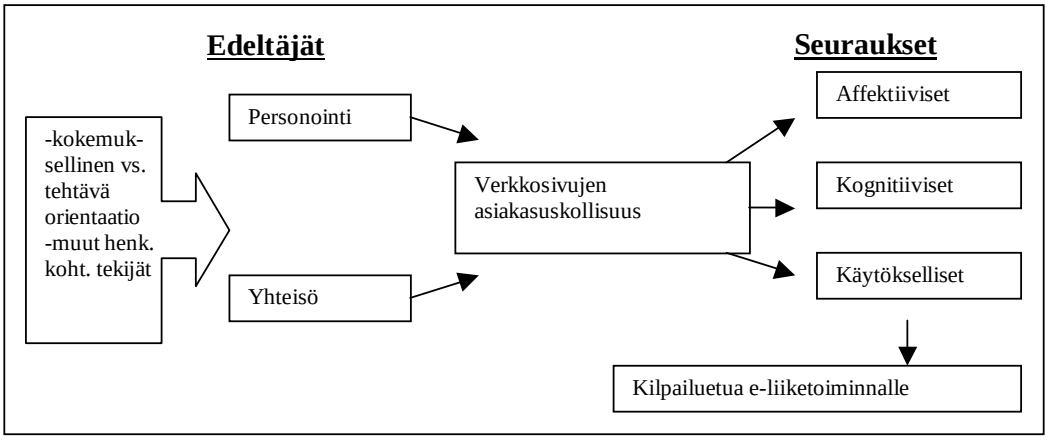
Verkkosivujen toimivuutta ja ominaisuuksia on tutkittu runsaasti, ja esimerkiksi Ivory ja muut [2001] sanovat että verkkosivujen tehokkuutta ja toimivuutta voidaan ennustaa tarkasti, ja lisäksi erityyppisten sivujen osalta kriteerit ovat erilaisia. Oikeiden mittareiden määrittäminen on kuitenkin vaikeaa, ja sekä verkkosivujen suunnittelun tueksi että niiden tehokkuuden mittaamiseksi on kehitetty lukuisia toisistaan poikkeavia apuvälineitä. Sen lisäksi että mitataan yrityksen verkkosivujen tehokkuutta ja toimivuutta niiden käyttöominaisuuksia ja ulkoasua arvioivien laadullisten mittareiden avulla, olisi tärkeää arvioida myös verkkosivujen tehokkuutta suhteessa muihin markkinointimixin osiin, verkkosivujen vaikutusta liiketoimintaan ja erityisesti myyntiin sekä niiden vaikutusta brändin kehitykseen. Tarkkaa tietoa verkkosivujen konkreettisesta käytöstä, kuten esimerkiksi tietoja vierailijoiden määrästä, vierailun keskimääräisestä kestosta ja vierailtujen sivujen määrästä on myös kannattavaa hyödyntää verkkosivujen kehityksessä.

6.1. Ulkoasua ja ominaisuuksia koskevat mittarit

Ivory ja muut [2001] ovat luokitelleet yritysten verkkosivut kolmeen kategoriaan sen mukaan mikä niiden sanamäärä on, ja määritelleet kriteerejä sille mikä kussakin luokassa on ominaista hyville sivuille. Pienimmän sanamäärän sisältäville sivuille tärkeitä ovat pienet sivukoot, suurempi määrä sisältöä, vähemmän grafiikkaa ja enemmän variaatioita kirjoitusasussa. Grafiikan ja sivukoon minimointi vähentävät sivujen latausaikaa ja näin parantavat käyttömukavuutta. Hyvillä sivuilla keskimmäisessä kategoriassa teksti on ryhmitelty esimerkiksi listoiksi tai taulukoiksi, ja tekstiosiosta on korostettu vain pieniä osia jotta ne todella tulevat esille. Myös värien käyttö otsikoiden erottuvuuden lisäämiseksi on hyvien sivujen ominaisuus keskimmäisessä kategoriassa. Hyvillä sivuilla korkean sanamäärän sivujen kategoriassa tekstin vartalo-osa on vähäisempi ja niillä käytetään paljon otsikoita ja tekstilinkkejä. Molemmat näistä ominaisuuksista helpottavat selailua ja tiedon etsintää paljon tekstiä sisältävillä sivuilla.[Ivory *et al.*, 2001]

6.2. Brändiuskollisuuden mittaus verkkosivuilla

Asiakasuskollisuutta verkkosivuille brändinä voidaan määritellä monin tavoin. Verkkosivujen brändäys ja uskollisuuden saavuttaminen ovat yritykselle hyvin tavoiteltavaa sillä onnistuneen brändin avulla voidaan saavuttaa kilpailuetua. Asiakasuskollisuutta ei kuitenkaan voida tarkastella samojen mittareiden avulla kuin esimerkiksi vähittäiskaupan tuotebrändien kohdalla, koska verkkosivuihin ei välttämättä liity esimerkiksi ollenkaan ostamisen mahdollisuutta, joten toistuvat ostot eivät ole toimiva mittari. Mittareina asiakasuskollisuudesta verkkosivuilla ovat Hollandin ja Bakerin [2001] mukaan sivuilla käytetty aika, vieraillut sivut ja toistuvat vierailut. Sivuilla käytetty aika kertoo siitä kuinka hyvin asiakas viihtyy sivuilla, mutta toisaalta erilaisten sivujen erilaisen luonteen vuoksi ei yksinään ole toimiva mittari. Asiakas voi esimerkiksi olla hyvinkin uskollinen verkossa käyttämilleen aikataulupalveluille, mutta niiden nopean toiminnan vuoksi käyttää niitä vain noin minuutin päivittäin. Vierailun syvyyttä kuvaa asiakkaan vierailemien sivujen määrä. Tämäkin mittari tarvitsee kuitenkin muita tuekseen, sillä se että asiakas on vieraillut useilla eri sivuilla saman asioinnin aikana saattaa jopa tarkoittaa että hän on pitänyt sivuja hyvin epäselvinä eikä löytänyt haluamaansa tietoa helposti. Toisaalta vierailun syvyys on positiivinen asia jos se kuvastaa sitä että asiakas on viihtynyt sivuilla ja löytänyt niiltä monia mielenkiintoisia asioita. Toistuvat vierailut kertovat asioinnin useudesta ja ovat sinänsä hyvä mittari asiakasuskollisuudesta, mutta tämänkin tekijän suhteen sivujen erilaisuus vaikuttaa siten, että tietyille sivuille asiakas ei koe tarvetta palata vaikka olisikin ollut niihin hyvin tyytyväinen.



Kuvio 2 Asiakkaan osallistuminen verkkosivujen asiakasuskollisuuden luomiseen. [Holland and Baker, 2001]

Kuviossa 2 esittää asiakasuskollisuuden muodostumista ja siihen vaikuttavia tekijöitä sekä näiden tekijöiden seurauksena muodostuvaa kilpailuetua. Personointi on kahdesta asiakasuskollisuutta edeltävästä tekijästä perinteisempi.

Yhteisöt taas ovat uudempi ja tällä hetkellä erittäin trendikäs tapa houkutella käyttäjiä. Esimerkiksi sellaisten maailmanmerkkien kuin Coca-Colan ja Niken verkkomarkkinoinnissa on aivan viime aikoina lähdetty lanseeraamaan yhteisöjä ja keskustelufoorumeita kuluttajille, yritysten aikaisempien verkkosivujen ollessa keskittyneitä enemmän konkreettiseen yritystietouteen ja sen esittelyyn [Brandweek, 2006; Holmes, 2006]. Kuviosta 2 käy myös ilmi kuinka verkkosivujen ominaisuuksiin suhtautumiseen vaikuttaa kuluttajan orientaatio ja muut henkilökohtaiset ominaisuudet kuten esimerkiksi kohdassa 4.2. mainitut tehtäväsuuntautuneisuus tai kuviossa 2 sille vaihtoehdoksi esitetty kokemuksellinen suuntautuneisuus jolla tarkoitetaan viihteen ja huvin vuoksi surffailua.

7. Johtopäätökset

Asiakaslähtöisen toimintatavan ollessa tämän hetken markkinatilanteessa useimmille yrityksille sopivin liiketoimintamalli, on internetin ja yrityksen verkkosivujen hyödyntäminen erittäin tärkeää, sillä niiden avulla voidaan saavuttaa huomattavaa kilpailuetua. Asiakaslähtöiselle toiminnalle on keskeistä toimia yhteistyössä asiakkaan kanssa, ja tähän verkkosivut ja muu informaatioteknologian kehitys tarjoavat paljon keinoja. Näiden keinojen käytössä täytyy kuitenkin huomioida myös erilaisten tuotteiden ja palveluiden sekä kuluttajan ja hänen orientaationsa vaatimukset, joten hyvien verkkosivujen ja markkinointiviestinnän kriteerit eivät ole yleispäteviä vaan hyvin tapauskohtaisia.

Brändeille verkkosivut tarjoavat ylivertaisia kehittymisen mahdollisuuksia perinteiseen mainontaan verrattuna. Läsnäolo internetissä mahdollistaa aitojen asiakassuhteiden rakentamisen keskusteluiden ja tiedon keruun ja hyödyntämisen helppouden myötä. Tämä on apuna sekä asiakaslähtöisyyden kehittämisessä että brändien rakentamisessa. Tämän lisäksi internet tarjoaa paremmat mahdollisuudet tarjonnan räätälöintiin koska siltä puuttuu perinteisten jakelukanavien jäykkyys. Verkkosivujen kautta yrityksellä on myös mahdollisuus aiempaa huomattavasti laajempaan interaktiivisuuteen [De Chernatony and McDonald, 2003].

Internetillä on myös suuri merkitys yritykselle globaalin näkyvyyden luojana ja mahdollisuutena saavuttaa ja luoda uusia markkina-alueita. Lisäksi pian ollaan siinä tilanteessa, että verkkosivujen puuttuminen tai yksinkertaisuus saavat yrityksen näyttämään vanhanaikaiselta. Nyky-yhteiskunnassa verkkosivut toimivat tietynlaisena näyteikkunana tai käyntikorttina yritykselle. Toisaalta tällä hetkellä ollaan myös mielenkiintoisessa murroskohdassa verkkosivujen kehityksen kannalta, sillä sivut alkavat sisältää pelkän yritysinformaation ja verkkokauppojen lisäksi yritysten lanseeraamia yhteisöjä ja muita kuluttajan vapaa-ajan viettoon tarkoitettuja toimintoja.

Verkkosivuja täytyy käsitellä kuten muitakin yrityksen strategisia työkaluja, eli ne pitää huolellisesti suunnitella ja rakentaa. Verkkosivujen sisällön tulisi olla laadukasta, ja palvelusta sekä sen laadusta on huolehdittava myös verkkoympäristössä. Koko yrityksen verkkoläsnäolon konsepti pitää rakentaa yhtenäiseksi muun yrityksen strategian ja brändin kanssa, jotta annetaan vakuuttava ja yhtenäinen kuva yrityksestä ja sen tuotteista sekä palveluista, eikä vahingoiteta yrityksen imagoa.

Viiteluettelo

[Brandweek, 2006] Coke's web formula is a work in progress. *Brandweek* **47**, 31 (2006), 9.

[Brattö and Gustavsson, 2001] Olof Brattö and Kennert Gustavsson, Branding online – leveraging industrial brands in the digital business environment. Department of Industrial Marketing, Chalmers University of Technology, 2001.

[Chaffey *et al.*, 2000] Dave Chaffey, Richard Mayer, Kevin Johnston, and Fiona Ellis-Chadwick, *Internet Marketing: Strategy, Implementation and Practice*. Pearson Education Limited, 2000.

[De Chernatony and McDonald, 2003] Leslie De Chernatony and Malcolm McDonald, *Creating Powerful Brands*. Butterworth Heinemann, 2003.

[Gurau, 2004] Calin Gurau, Customer-centric internet strategies: Achieving competitive advantage through CRM. In: Namchul Shin (ed.), *Strategies for Generating E-Business Returns on Investment.* Idea Group Publishing, 2004, 21-49.

[Heinze and Hu, 2006] Nathan Heinze and Qing Hu, The evolution of corporate web presence: a longitudinal study of large American companies. *International Journal of Information Management* **26**, 4 (Aug. 2006), 313-325.

[Holland and Baker, 2001] Jonna Holland and Stacey Menzel Baker, Customer participation in creating site brand loyalty. *Journal of Interactive Marketing* **15**, 4 (2001), 34-45.

[Holmes, 2006] Stanley Holmes, Nike it's not a shoe, it's a community. *Business Week 7/24/2006*, 3994 (2006), 50.

[Ivory *et al.*, 2001] Melody Y. Ivory, Rashmi R. Sinha, Marti A. Hearst, Empirically validated web page design metrics. In: *Proc. of ACM Conf. SIGCHI'01*, (2001).

[Kanso and Nelson, 2004] Ali M. Kanso and Richard Alan Nelson, Internet and magazine advertising: integrated partnerships or not? *Journal of Advertising Research* **44**, 4 (2004), 317-326.

[Palmer and Griffith, 1998] Jonathan W. Palmer and David A. Griffith, An emerging model of web site design for marketing. *Communications of the ACM* **41**, 3 (1998), 45-51.

[Singh and Dalal, 1999] Surendra N. Singh and Nikunj P. Dalal, Web home pages as advertisements. *Communications of the ACM* **42**, 8 (1999), 91-98.

[Taloussanomat, 2006] Ruotsalaistutkimuksen keskiarvo: Suomalaisyritysten verkkosivut Euroopan parhaat, Taloussanomat ITviikko. 23/11/2006 (2006), osoitteessa: http://www.itviikko.fi/page.php?page_id=15&news_id=200621470 (luettu 4.12.2006).

[Winter *et al.*, 2003] Susan J. Winter, Carol Saunders, and Paul Hart, Electronic window dressing: impression management with websites. *European Journal of Information Systems* **12** (2003), 309-322.

Digitaalisen musiikin verkkojakelu

Mikko Malinen

Tiivistelmä.
Tässä tutkielmassa käsitellään digitaalista musiikkia ja sen verkkojakelua. Työssä käydään läpi ilmiön historiaa, sen nykyistä tilaa sekä pohditaan lyhyesti, mihin suuntaan ollaan matkalla. Keskeinen osa tutkielmaa on jakelu- ja levityspalveluiden sekä -tapojen esittely. Myös näiden osaa ilmiön kehityksessä pyritään analysoimaan.

Avainsanat ja -sanonnat: Digitaalinen musiikki, MP3, vertaisverkot, MySpace.
CR-luokat: K.4

1. Johdanto

Viimeisen parinkymmenen vuoden aikana musiikin julkaisu, jakelu ja leviäminen on kokenut massiivisia mullistuksia. Koko musiikinkuuntelun ja -kulutuksen luonne on itse asiassa muuttumassa. Tässä tutkielmassa kerrataan digitaalisen musiikin suhteellisen lyhyttä historiaa ja pyritään katsomaan myös eteenpäin ja pohtimaan, miten ilmiö tulee kehittymään. Näiden ääripäiden väliin mahtuu luonnollisesti paljon asiaa. Nykytilannetta käydään läpi tilastollisesti ja pyritään löytämään syitä nykyiseen tilaan ja kehitykseen.

Näiden varsin yleisellä tasolla liikkuvien asioiden lisäksi syvennytään joihinkin tämän hetken merkittävimmistä digitaalisen musiikin jakeluun ja jakamiseen liittyvistä palveluista ja toimintatavoista. Esittelyssä ovat vertaisverkot, digitaalisen musiikin verkkokaupat sekä MySpace.com.

2. Digitaalisen musiikin historiaa

Noin sata vuotta vanhan ääniteteollisuuden digitaalinen aikakausi alkoi 1980-luvulla, kun äänitteitä alettiin vinyylilevyjen ja c-kasettien sijasta – tai aluksi rinnalla – myydä CD-levyillä. Compact Disc eli CD sisältää ääniraidat muunnettuina digitaaliseen muotoon, tarkoittaen käytännössä sitä, että kappaleet itse asiassa koostuvat vain ykkösistä ja nollista. Digitaalisen musiikin verkkojakelua kohti jo CD sinänsä oli askel siinä mielessä, että musiikin ollessa valmiina digitaalisessa muodossa sen siirto tietokoneelle helpottuu huomattavasti. (Alexander, 2002, pp. 2-3)

Digitaalisen musiikin jakelun todellisena alkusysäyksenä voidaan kuitenkin pitää MP3-formaatin syntyä. MPEG-1 layer 3 eli tuttavallisemmin MP3 on sak-

salaistutkimuskeskus Fraunhofer IIS:n kehittämä standardi. Formaatti sai nykyisen nimensä vuonna 1995. (Ulrich)

MP3 on audiopakkausmenetelmä, joka pyrkii pakkaamaan ääntä poistamalla siitä taajuuksia, joita ihmiskorva ei kykene erottamaan. Vaikka pakkaus heikentää – pakkaustehosta riippuen – jossain määrin äänenlaatua, sen etuna on mahdollisuus saada lopullisen äänitiedoston kokoa pienennettyä jopa alle kymmenykseen alkuperäisestä.

Jos siis kahden minuutin mittainen musiikkikappale CD-levyllä vie noin kaksikymmentä megatavua tilaa, vie se MP3 muodossa vain kaksi megatavua, jopa vähemmän. On selvää, että varsinkin hitailla verkkoyhteyksillä musiikkia siirrettäessä tämänkaltainen tiedostokokojen kehitys on lähes elintärkeää.

Verkkoyhteyksien ja sitä kautta Internetin käytön yleistymisen, sekä otollisen pakkausmenetelmän ilmestymisen myötä, digitaalisen musiikin verkkojakelu kasvoi räjähdysmäisesti vuosituhannen vaihteessa.

3. Vertaisverkot

Ensimmäinen musiikinlevitykseen yleisesti käytetty Peer-to-peer- eli vertaisverkkopalvelu avattiin vuonna 1999. Napster.com tarjosi internet-sivuillaan ladattavaksi ohjelmistoa, jonka asennettuaan käyttäjä avasi kovalevynsä mp3-arkistot kaikkien muiden käyttäjien nähtäväksi ja ladattavaksi. Jokaisella Napster-ohjelman käyttäjällä oli siis nähtävillä kaikkien muiden palvelun käyttäjien mp3-tiedostot. Tiedonsiirto vertaisverkossa tapahtuu suoraan käyttäjien välillä, ilman kiinteitä palvelimia. Kaikki verkon toimijat ovat siis keskenään tasavertaisia ja toimivat sekä palvelimina että asiakkaina.

Vaikkakin Napster jouduttiin oikeuden määräyksellä myöhemmin sulkemaan, oli kehitystä liian myöhäistä pysäyttää. Vertaisverkot ovat nykyään ehdottomasti suurin digitaalisen musiikin levityskanava.

Napsterin aloittama vertaisverkkojen valta onkin saanut valtavasti näkyvyyttä myös mediassa, ja useammillekin palveluille on sittemmin haettu loppua oikeusteitse – yleensä musiikkiteollisuuden toimesta. Viimeisimpänä kotimaisena esimerkkinä on Finreactor-niminen vertaisverkkopalvelu. Palvelun sulkemisen lisäksi sen ylläpitäjille määrättiin sakkoja ja tekijänoikeuskorvauksia yli puolen miljoonan euron edestä. Vertaisverkkopalvelujen suosiosta jonkinlaista kuvaa antaa se, että pelkästään Finreactorilla oli enimmillään noin 10 000 suomalaista käyttäjää ja kolmen ja puolen kuukauden ajanjakson aikana vuoden 2004 syksynä palvelusta ladattiin yli 15 teratavua musiikkia. Äänilevyiksi muutettuna kappalemäärä vastaa keskimäärin noin 270 000 albumia. (Tekijänoikeuden tiedotus- ja valvontakeskus, 2006)

Syytteet eivät enää koske ainoastaan palveluntarjoajia ja ylläpitäjiä, vaan myös tavallisia käyttäjiä. Vaikkakin joidenkin vertaisverkkopalvelujen käyttö jollain tapaa vaatii myös tiedostojen jakamista, on ainakin joistakin palveluista myös mahdollista ainoastaan ladata tiedostoja. Suomessa tekijänoikeuslaki muuttuikin vuoden 2006 alusta lähtien mm. siten, että laittomien tiedostojen jakamisen lisäksi myös lataaminen on nyt lainvastaista. (Kopiosto, 2005)

4. Musiikin lailliset verkkojakelukanavat

Napsteriin ja sitä seuranneisiin muihin vertaisverkkopalveluihin verrattuna, varsinainen musiikkiteollisuus on avannut omien ”uuden sukupolven jakelukanaviensa” portit pahasti jälkijunassa.

Laittomien kanavien tyrehdyttämistä on kokeiltu monen tahon toimesta oikeusteitse. Paniikinomainen reagointi ei ole kuitenkaan toiminut kuin viivytystekniikkana (Lam & Tam, 2001, p. 68) On selvää, että kenties tuhansien palvelujen ja niiden alla toimivien miljoonien ja taas miljoonien käyttäjien välillä kulkevia tiedostoja on mahdotonta pysäyttää rikkomalla yhtä verkon solmua sieltä ja toista täältä – puhumattakaan yksittäisten, satunnaisten käyttäjien tekemisiin keskittymisestä.

Sen sijaan on onneksi alettu rakentaa täysin laillisia palveluja, jotka kykenisivät tarjoamaan samoja etuja kuin tekijänoikeuksia kaukaa kiertäneet esisänsä: musiikin hankkimista vaivattomasti ja nopeasti.

Näiden ”elektronisten levykauppojen” uranuurtajana on toiminut – ja toimii edelleen – Apple-yhtiön verkkopalvelu iTunes. Vuonna 2003 perustettu palvelu on tähän päivään mennessä kasvanut megalomaanisiin mittoihin. Vuoden 2006 alkuun mennessä palvelu oli lokalisoitu – Suomi mukaan lukien – 21 eri maahan (International Federation of the Phonographic Industry [IFPI], 2006, p. 4) ja iTunesin päiväkohtainen latausmäärä oli kolmen miljoonan yksittäisen musiikkikappaleen luokkaa.

Vaikka voidaan ajatella, että valtavia määriä vannoutuneita käyttäjiä taakseen keränneitä vertaisverkkoja vastaan on mahdotonta taistella näiden omilla aseilla, voi asetelman myös helposti kääntää päälaelleen: potentiaalista asiakasainesta on valtavat määrät, kunhan vain onnistutaan tarjoamaan oikeat houkuttimet.

Verkkokauppojen tarjoamia etuja voisivat olla mm. niiden luotettavuus niin saavutettavuuden kuin laadunkin suhteen, laaja valikoima ja paikallisuus. Esimerkiksi suomalaista musiikkia on vertaisverkoissa saatavilla suhteellisen huonosti, joten tehokkaasti lokalisoituina Suomessa toimivien verkkokauppojen olisi mahdollista saada runsaasti käyttäjiä.

Suurin osa maksavista asiakkaista käyttää edelleen musiikin ostamiseen niin sanottuja tavallisia levykauppoja. Sen lisäksi, että levykaupat tarjoavat musiikkinsa säilöttynä tutuille ja turvallisille cd-levyille, tarjoavat ne parhaimmillaan paljon muutakin.

Verkkokauppojen ehkäpä suurin yksittäinen haaste lähestulkoon alaan tai myytävään tuotteeseen katsomatta onkin, miten verkkoon saadaan siirrettyä myyjien tarjoama palvelu ja ennen kaikkea asiantuntemus. Kuten verkkoon siirrettävät myymälät yleensäkin, myös levykauppa on erikoisliike, jossa myyjien automaattisesti oletetaan tietävän myytävistä tuotteista muutakin kuin vain hinnan.

Digitaalista musiikkia myyvät verkkokaupat voisivat yhtä lailla tarjota muutakin informaatiota kuin vain levyn tai yksittäisen kappaleen nimen ja julkaisuvuoden. Samaan palveluun voitaisiin kerätä erilaista tietoa yhtyeistä, ”henkilökunta suosittelee” tai ”viikon levy”-tyylisiä kampanjoita ja vaikkapa ilmaisia videohaastatteluja artisteista. Sen sijaan, että palvelun ja asiantuntemuksen siirtäminen verkkoon olisi mahdotonta, ovat mahdollisuudet itse asiassa rajattomat. Olisiko tällainen kokonaisvaltainen musiikkiportaalin ja verkkomusiikkikaupan risteytys hyvä valtti myös taistelussa laittomia vertaisverkkoja vastaan? Ainakin tietyt kohderyhmät ovat varmasti valmiita maksamaan palvelusta – olettaen, että hinta onnistutaan pitämään kohtuullisena.

Kun asiakas on vihdoin saatu houkuteltua digitaalista musiikkia tarjoavan palvelun sisään, täytyy pitää huolta siitä, että ostaminen on mahdollisimman vaivatonta. Tällä hetkellä ongelmia syntyy usein maksaessa, sillä verkko-ostosten tekeminen vaatii usein luottokorttia tai vastaavaa maksumenetelmää, mikä koituu usein ongelmaksi varsinkin nuorten käyttäjien kohdalla (Premkumar, 2003, p. 92). Täytyy ottaa huomioon, että nuoret ja nuoret aikuiset ovat erittäin tärkeää kohderyhmää tällaisille palveluille, ja jos ostaminen tuntuu vaikealta, ovat asiakkaat herkästi valmiita kääntymään korvaavien palvelujen puoleen. Siksi koko digitaalisen musiikin verkkomyynnin onnistumisen kannalta elintärkeä seikka on saada palvelut toimimaan aina vaivattomasti ja nopeasti.

5. Digitaalinen musiikin määrällinen tilanne tällä hetkellä

Digitaalisen musiikin myynti kasvaa tällä hetkellä kiihtyvällä vauhdilla. Vuoden 2005 aikana ladattiin yhteensä noin 420 miljoonaa yksittäistä musiikkikappaletta. Latausten määrä kaksikymmenkertaistui verrattuna vuoteen 2003. (IFPI, 2006, p.3) Kasvu on ollut siis hurjaa eikä vauhti näytä hiipuvan.

Vaikkakin vuonna 2005 digitaalisen musiikin myynnin osuus on vasta noin 6% koko maailmanlaajuisesti myytävästä musiikista, musiikki on selvästi siirtymässä perinteisestä levyformaatista kohti digitaalista muotoa.

Siinä missä CD-albumien myynti on moninkertaista CD-singlejen myyntiin verrattuna, on suhde digitaalisessa musiikissa täysin päinvastainen. Esimerkiksi Yhdysvalloissa noin 353 miljoonan yksittäisen digitaalisen musiikkiraidan rinnalla ladattiin vain noin 16 miljoonaa kokonaista albumia.

Selityksiä kehitykselle on varmasti monia. Yhtenä perustavaalaatua olevana tekijänä voidaan tietysti pitää laajakaistayhteyksien yleistymistä. Silti sekään ei yksin selitä ilmiötä, sillä siinä missä laajakaistaliittymien määrä esimerkiksi Euroopassa kasvoi vuosien 2004 ja 2005 välillä 46 %, yksittäisten musiikkiraitojen latausmäärä nousi huikeat 355% samaisen ajanjakson aikana. (IFPI, 2006, p.5)

Toisaalta tarjonnan sanotaan lisäävän kysyntää. Musiikkia laillisesti verkossa levittävien palvelujen määrä onkin kasvussa myös Euroopassa. Kuitenkin palvelujen lukumäärä kasvoi vuodesta 2004 vuoteen 2005 vain 33 %. (IFPI, 2006, p. 5)

Sen sijaan, että mikään yksittäinen tekninen tosiasia olisi syynä valtavalle kasvulle, voidaan sanoa koko musiikinkuuntelun – ja ennen kaikkea kulutuksen – luonteen selvästi ottavan uutta suuntaa.

Erilliset digitaalisen musiikin kuunteluun tarkoitetut kannettavat laitteet yleistyvät myös hyvää vauhtia. Niitä kutsutaan laajalti hieman harhaanjohtavasti MP3-soittimiksi, vaikka yleisesti ottaen kaikki soittimet kykenevät muidenkin tiedostomuotojen toistamiseen. Nimi on kuitenkin varsin vakiintunut, joten kutsuttakoon laitteita MP3-soittimiksi myös tässä tutkielmassa.

Mp3-soittimia myytiin vuoden 2005 aikana maailmanlaajuisesti noin 140 miljoonaa kappaletta. Edellisvuoteen verrattuna myynti nelinkertaistui. Kasvu on myös tällä saralla ollut räjähdysmäistä eikä sen odoteta pysähtyvän tulevaisuudessakaan. (In-stat, 2006)

Ylivoimaisesti suosituin yksittäinen soitinmalli on Applen valmistama iPod. Lähes joka toinen MP3-soitin maailmanlaajuisesti on iPod. (In-stat, 2006)

6. MySpace.com

MySpace.com on verkkosivusto, joka perimmäisenä tarkoituksena on toimia sosiaalisena verkkona, tai kuten palvelun motto kuuluu ”a place for friends” – ”paikkana kavereille”. Sen perustivat Chris DeWolfe ja Tom Anderson vuonna 2003. (Sellers, 2006) Nykyisin MySpace.com on käsittämättömän suosittu. Vain noin kolmessa vuodessa se on noussut maailman viidenneksi vierailluimmaksi internet-sivustoksi. Sen yläpuolelle listalla yltävät vain Yahoo!, MSN, Google sekä kiinalaishakukone Baidu (Alexa internet, 2006). Vuoden 2006 elokuussa

MySpace-palvelu ylitti 100 miljoonan rekisteröityneen käyttäjän rajapyykin. (Sellers, 2006)

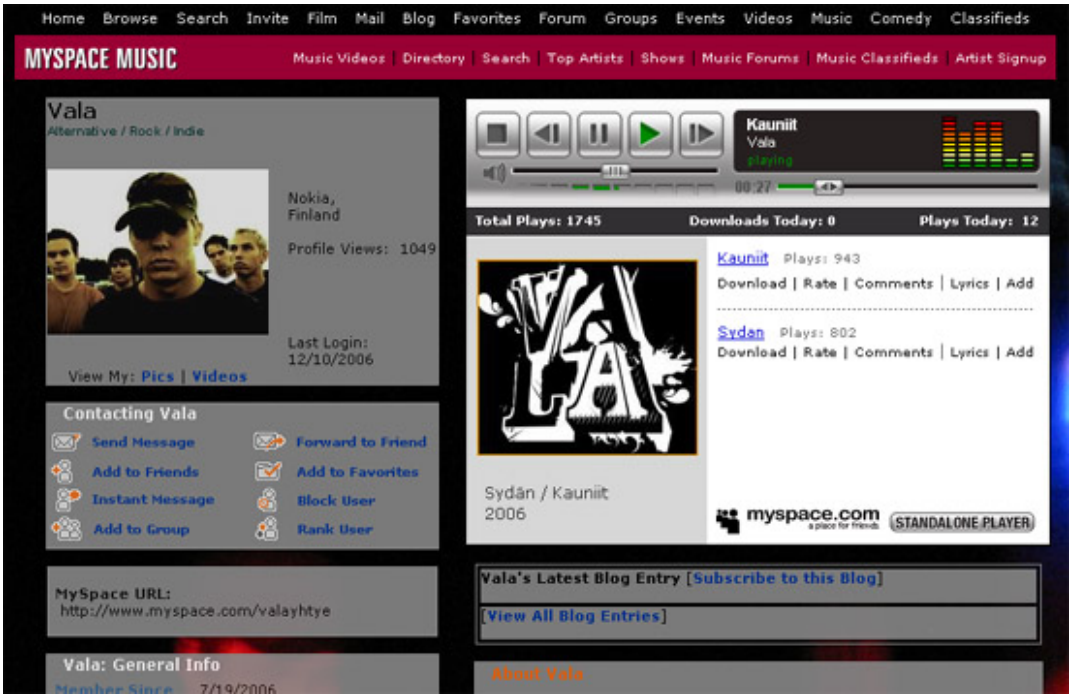
Heinäkuussa 2005 News corp-niminen yritys, johdossaan tunnettu australialainen ”mediamoguli” Rupert Murdoch, osti MySpace-palvelun omistaneen Intermix-yhtiön 580 miljoonan dollarin kauppahinnalla. (Sellers, 2006)

Käytännössä MySpace toimii siten, että kuka tahansa voi luoda palveluun oman sivunsa täysin ilmaiseksi. Sivulle voi lisätä kuva-, video- ja musiikkitiedostoja, joiden lisäksi käytössä on blog sekä koko palvelun perusidean mukaisesti kaverilista. Tälle listalle voi kutsua muita oman sivunsa luoneita käyttäjiä. Kutsuttu käyttäjä saa tiedon kutsustaan sähköpostiin, ja hän saa päättää hyväksyykö kutsujan kaverikseen. Hyväksynnän jälkeen käyttäjien nimimerkit näkyvät toistensa kaverilistoissa. Omille kavereilleen voi jättää kommentteja jokaisen profiilin sivulla näkyvään vieraskirjaan.

Syy MySpacen valtavaan menestykseen on monen mielestä sen kehittyminen musiikinlevityskanavaksi. Siihen luotu MySpace music-niminen ”alipalvelu” on tarjonnut yhtyeille ja artisteille mahdollisuuden levittää musiikkiaan tehokkaasti, ja toisaalta kuuntelijoille kappaleita kuunneltavaksi täysin ilmaiseksi. MySpacea musiikin levitykseen käyttävät musiikintekijät eivät suinkaan ole ainoastaan levy-yhtiöihin kuulumattomia pikkunimiä, vaan palvelusta löytyy omat sivut niin Madonnalta, Robbie Williamsilta kuin U2:ltakin. Luonnollisesti maailmankuulujen artistien lisäksi MySpacen voimaan luottavat myös lukemattomat tulevaisuuden nimet.

Käytännössä Music-osion sivut eivät eroa ns. tavallisista sivuista mitenkään muuten kuin että profiiliin on mahdollista lisätä omia musiikkikappaleitaan soittimeen (kuva1), joka mahdollistaa kuuntelun ns. suorasoittona, jolloin musiikkitiedostoa ei tallenneta käyttäjän koneelle. Profiilin luojan valittavissa on, onko kappaleita mahdollista tämän lisäksi ladata myös omalle koneelle. Kappaleista voi halutessaan tallentaa palveluun myös oheismateriaalia, kuten tiedon siitä, miltä julkaisulta kappale löytyy, kuvan levynkannesta sekä teoksen sanoitukset. Varsin yleistä on, että kappaleiden lisäksi omalle sivulle lisätään musiikkivideoita, kuvia sekä tiedot tulevista esiintymisistä.

Aivan kuten yksityisetkin, ns. tavallisen MySpacen käyttäjät, myös artistit voivat kutsua muita käyttäjiä kavereikseen.



Kuva 1: MySpace Music-palvelun artistisivu

Uusin suuri käänne MySpacen jo nyt värikkäässä historiassa tulee olemaan musiikkikappaleiden myymisen aloittaminen. Palvelun pitäjät ilmoittivat 2006 vuoden syyskuussa, että MySpace music tulee tulevaisuudessa tarjoamaan käyttäjille mahdollisuuden myydä omia kappaleitaan järjestelmän avulla. Perusideana on se, että jokainen käyttäjä voi itse määritellä hinnan myytäville kappaleille. Luonnollisesti MySpace ottaa jokaisesta myydystä tiedostosta tietyn summan, jonka suuruutta ei vielä ole paljastettu. (BBC, 2006)

7. Digitaalisen musiikin tulevaisuus

Digitaalinen musiikki on jo nyt käsitteenä varsin tuttu monille, mutta suurimmaksi osaksi yhteyksistä, joista artistit ja musiikkiteollisuus eivät ansaitse senttiäkään – ainakaan suoranaisesti. Musiikkiteollisuuden puolelta juuri kukaan ei ole ollut valmis myöntämään, että musiikkikappaleiden vapaasta jaosta olisi ollut sille mitään hyötyä. On hieman vaikea käsittää, minkä takia kappaleita nykyisin annetaan kuitenkin monessakin palvelussa, kuten MySpace.com, mp3.com, kuunnella ilmaiseksi, jos siitä ei ole pienintäkään hyötyä. Esimerkiksi laittoman musiikin levityksen vaikutusta esimerkiksi konserttilipunmyyntiin tai oheistuotteiden menekkiin on melko vaikea mitata. Yleisesti monissa yhteyksissä vedetään varsin suoria yhteyksiä laittoman verkkojakelun ja vähentyneen levymyynnin välille, vaikkakin vaikuttavia seikkoja voi tosiasiassa olla

useita. Esimerkiksi äänilevyjen hintojen nousu tuskin vaikuttaa levymyyntiin ainakaan positiivisesti.

Tietenkään ei voida kiistää sitä, ettei musiikkiteollisuus olisi kärsinyt ”musiikin varastamisesta”, mutta varmasti voidaan olla montaa mieltä siitä, onko se ollut välttämättä ainoastaan negatiivinen asia.

Toisaalta vertaisverkot ovat tutustuttaneet valtavan määrän käyttäjiä koko digitaalisen musiikin verkkojakelun ideaan. Niinpä pallon voidaan sanoa nyt olevan musiikkiteollisuudella: saadaanko näitä ihmisiä houkuteltua muuttumaan vain lataajista kuluttajiksi. Jos palveluihin onnistutaan tosissaan panostamaan niin, että hinnoittelu pysyy alhaisena, se on varmasti ainakin jonkinlaisessa mittakaavassa mahdollista.

Musiikkiteollisuudessa olisikin ehkä aika hieman tarkistaa asenteita sen suhteen, onko oikea tapa välttämättä suunnata valtavia resursseja kopiosuojausten kehittämiseen ja laittomien tiedostojen vainoamiseen, sen sijaan, että tarjottaisiin jotain oikeasti varteenotettavaa ja mielenkiintoista tilalle.

Tässä ”taistelussa” uudet innovaatiot ovat luonnollisesti tervetulleita. Etukäteen yksi mielenkiintoisimmista uusista tulokkaista on ehdottomasti Spiralfrog-niminen verkkopalvelu, jonka on määrä aloittaa toimintansa 2006 vuoden joulukuun aikana. Palvelun ideana on tarjota sivuillaan musiikkikappaleita ilmaiseksi ja siltikin täysin laillisesti. Kappaleet ”maksetaan” katselemalla sivuille asetettuja tarkasti kohderyhmälle suunniteltuja, aiheeseen liittyviä mainoksia. Palvelun kohderyhmäksi on ilmoitettu 13-34 -vuotiaat musiikista kiinnostuneet. Näiden ihmisten uskotaan olevan otollista kohderyhmää myös mainonnalle. (SpiralFrog, 2006)

Spiralfrog on jo onnistunut solmimaan sopimukset maailman suurimpiin levy-yhtiöihin kuuluvien Universal Music Groupin ja EMI Music Publishingin sekä hieman tuntemattomamman KOCH Records -yhtiön kanssa, joten sisältöä palvelussa tulee olemaan. (SpiralFrog, 2006)

Myös vertaisverkkoteknologian valjastamisesta lailliseen musiikin levittämiseen on puhuttu paljon, mutta ainakaan suuren mittakaavan konkreettisia esimerkkejä ei vielä ole nähtävissä.

Digitaalisen, verkkoteitse jaettavan musiikin kulutus tulee varmasti kasvamaan jatkossa entisestään. Täytyy kuitenkin muistaa, että koko ilmiö kerää edelleenkin kiinnostusta ainakin jossain määrin uutuudenviehätyksen avulla.

Onko konkreettiset levyt lopulta sittenkään täysin korvattavissa vain tiedostoilla? Vaikka palvelut toimisivat moitteettomasti, ei ruudulle ilmestyvät tiedostonimet koskaan voi saavuttaa sitä tunnetta, minkä juuri ostetun CD-levyn hypisteleminen saa aikaan levykaupasta pois kävellessä. Näyttävät kannet ja itse levy musiikin kanssa muodostavat taiteellisen kokonaisuuden, joka jossain

määrin rikkoutuu yksittäisiä tiedostoja ladattaessa. Eikä CD-levyn ja verkosta ladattujen tiedostojen paremmuutta ainakaan kaupallisena jakelumuotona edes voida pitää lähellekään niin itsestään selvänä kuin esimerkiksi CD-levyn ja jo lähes edesmenneen vinyylin välillä voidaan.

Siksi verkkoa tulisikin ajatella huomattavasti suurempana mahdollisuutena kuin vain pelkkänä kanavana, jonka avulla kappaleet saadaan kuljetettua julkaisijalta kuluttajalle ilman konkreettisia tallennusmedioita. Välttämättä verkkopalveluja ei edes tulisi kehittää siitä lähtökohdasta, että ne ovat suoranaisia rinnakkaisia vaihtoehtoja perinteisille levykaupoille.

Vaikka noin 6 % kaikesta musiikkimyynnistä on hyvä alku, ei se konkreettisena lukuna ole juuri mitään. Jotta lupaava kasvuvauhti saadaan pidettyä yllä, nyt jos koska on uusien, tuoreiden ideoiden aika.

8. Yhteenveto

Tässä tutkielmassa käytiin läpi musiikin digitalisoitumisen eri vaiheita sekä sen verkkojakelun ja -jakamisen välineitä ja tapoja. Vaikkakin musiikinlevitykseen on sen koko verkkoaikakauden ajan liittynyt valtavasti vastakkainasettelua, oli tämän tarkastelun puitteissa tarkoituksena katsoa asioita täysin sivustaseuraajan näkökulmasta, ottamatta kenenkään puolta. Digitaalinen musiikin verkkojakelu on vielä nuori ilmiö ja vain aika näyttää, mikä sen asema koko suuressa viihdeteollisuuden kentässä lopulta tulee olemaan.

Viiteluettelo

Alexa internet (2006). Top Sites. Retrieved 10.12.2006 from http://www.alexa.com/site/ds/top_sites?ts_mode=global&lang=none.

Alexander, P. J. (2002). Peer-to-peer file sharing: The case of the music recording industry. Review of Industrial Organization, 20(2), 151-161.

BBC. (5.9.2006). MySpace set to sell music online. Retrieved 10.12.2006 from http://news.bbc.co.uk/2/hi/entertainment/5316000.stm.

In-Stat. (2006). Portable Digital Audio Players: Market Growth Exceeds Expectation. Retrieved 10.12.2006 from http://www.instat.com/catalog/Ccatalogue.asp?id=27#IN0603155ID.

International Federation of the Phonographic Industry. (2006). Digital music report.

Kopiosto (2005). Mikä muuttuu 1.1.2006? Luettu 10.12.2006 osoitteesta http://www.kopiosto.fi/ index.php?cid=kopiosto&mid=32.

Lam, C. K. M., & Tam, B. C. Y. (2001) The internet is changing the music industry.. Communications of the ACM, 44(8), 62-68.

Premkumar, G. P. (2003). Alternate distribution strategies for digital music. Communications of the ACM, 46(9), 89-95.

Sellers, P. (29.8.2006). MySpace cowboys. Fortune 9.8.2006.

SpiralFrog. (20.9.2006). Press Room. Retrieved 10.12.2006 from http:// www.spiralfrog.com/press_release. aspx#sept19.

Tekijänoikeuden tiedotus- ja valvontakeskus. (26.10.2006). Finreactor-vertaisverkon ylläpitäjät tuomittiin tekijänoikeusrikkomuksista sakkoihin ja maksamaan korvauksia oikeudenhaltijoille yli puoli miljoonaa euroa. Luettu 10.12. 2006 osoitteesta http://www.antipiracy.fi/ajankohtaista/uutinen.html?id=8089.

Ulrich, R. MP3: MPEG Audio Layer 3. Retrieved 10.12.2006 from http:// www.iis.fraunhofer.de/amm/ techinf/layer3/.

Haptic Applications for Mobile Device

Xiaoqing Meng-Pitkänen

Abstract.
This study focuses on recent haptics applications and designs that have been developed for mobile devices. The main purpose is to find what the current development of haptics is in mobile context and what will be the future challenges. As I am involved in mobile industry and very closely deal with UI testing, I find intuitive and efficient interaction with mobile device very important. I'm constantly interested in new interaction techniques used in mobile devices to make the interaction more intuitive and more interesting. I selected three example applications and presented them in detail. The evaluation results show that the applications make interaction more dimensional but also more natural and enjoyable. As a consequence, they are accepted by the users. However, the future challenges are ahead with more complex haptic user interface and technology problems. People who are interested in current haptic technology development, particularly in mobile industry domain, would find this study in some ways useful.

Key words and phrases: haptics, haptics application, haptic interaction, mobile device, tactile, tacton, vibration.
CR-categories: H.5

1. Introduction

In world wide, mobile device has been becoming one of the most important accessories for individuals. Interaction with mobile device becomes part of people's daily life. Because mobile device has small screen, the mobile interaction is not so pleasant and efficient in many situations. The researchers are constantly developing new interaction techniques to improve efficiency of mobile interaction. Among all new interaction techniques, haptic interaction has received more and more attention from researchers and designers in HCI field.

Haptics is related to sense of touch. A haptic user interface is a user interface or device that allows users to interact via touching. There are two clear advantages of using haptics in mobile interaction. For people without ability of seeing and hearing, haptics is a good sensory channel which can help them use mobile devices. For example, vibration ringing tone helps audio-impaired users to recognize incoming call. The other advantage of using haptics is that haptic user interface can give users more real, more intuitive feeling than a non-haptic

user interface can. If we imagine playing game with haptic mobile device, it must be more fun when user can feel the hit or jump of the objects.

Particularly, for non-visual information, haptics can be additional or optional sensory modality to seeing and hearing. Therefore, not only existing functionalities of mobile applications can be enhanced by using haptics, but also totally new features and functionalities can be explored for mobile devices by using haptic technology. There are a lot of spaces for invention and development.

In recent years, some mobile devices have already implemented haptic features. For example, Nokia's phones feature vibration patterns which accompany MIDI ring tones, Immersion has developed VibeTonz technology for enhancing ring tones and games in mobile phones [Immersion VibeTonz System], and Motorola has its own Audio-Haptic approach enhancing ring tones with haptic effects using a multifunction transducer [Chang and O'Sullivan, 2005]. The haptics development is expanding in mobile industry and the potential for developing more haptic applications looks substantial.

In the beginning of this paper, definitions of haptic-related terms are introduced and current technology development is reviewed; after that, three haptic applications are presented with design background, design details and evaluations results. In the end, discussions and conclusions are made including discussions for future development.

2. Overview to Haptics

2.1. Definitions

Haptics means physical interaction via touch. Haptic user interface allows human to have haptic interaction with real or virtual environments. Under the scope of haptics, another important term is tactile. Haptics and tactile both relate to sense of touch. Tactile refers to the sense of contact with the objects, mediated by the response of low threshold mechanoreceptors innervating the skin, within and around the contact region [Salisbury, 1998]. Besides tactile feedback, there are other four haptic modalities: force, vibration, thermal and electrical. Tactile, force and vibration have been applied and studied more than thermal and electrical. In this study, the applications are based on tactile and vibration.

2.2. Usage

Compared to seeing and hearing, touching is much less used in human-computer interaction. However, touch is one of the most important sensory channels in the real world. Especially, for visual-impaired people, touching is very important and meaningful.

In recent years, various haptic applications have been developed in different fields, such as virtual gaming, automobile, and medical field. The benefits are obvious. Using haptic force feedback in medical operations is a very good example. In surgery training, if haptic force feedback is used, volunteers or dead bodies are not needed any more, doctors can get the same feeling from haptic force feedback as from real bodies.

In mobile device, haptics has been successfully used for enhancing audio effects and vibration alerts. Despite of these successes, haptics is still underused and the scope of haptic research is very limited. If we look at mobile phone vibration alert, which has been used in mobile phones for many years, but it still remains a simple feature. In the near future, at least haptics can be applied in the following three types of mobile usages: 1) designed specifically for visual-impaired people to be able to 'see' information. 2) presenting non-visual information to reduce amount of information or tasks that are delivered through small-screen mobile device. 3) virtual game or any other applications that require more interactions.

2.3. Haptic technology and current development

The most noticeable haptics technology pioneer is Immersion. It develops haptics technology in various fields, automotive, mobility, gaming, industrial controls, medical, digitizing and 3D interaction. In the field of mobility, Immersion's VibeTonz System is well-known [Immersion VibeTonz System], which advances vibration for mobile phone alerts that hasn't evolved much since the beginning. More specifically, VibeTonz System adds touch sensations to sight and sound cues allowing greater realism and intuition in the way people navigate and use their mobile phone [Immersion VibeTonz System]. Touch sensations can be used to prioritize, categorize, or highlight messages and content. For example, with touch sensation, user can create and respond to a friend with special handshake or secret language. Touch can add value to the mobile user interface and to applications including alerts, caller ID, messaging, ring tones, and gaming. [Immersion VibeTonz System]

Nokia is one of the other few companies that actively put efforts on exploring and developing haptics technology in mobile phones. As mentioned earlier, Nokia phones feature vibration patterns which accompany MIDI ring tones [Change and O'Sullivan, 2005]. Nokia also develops a digital pen [Nokia Digital Pen] which user vibrations for user interface features such as indicating battery status and confirming actions.

3. Haptic applications

3.1. Audio-haptic feedback in mobile phones

Chang and O'Sullivan [2005] stated that the problem with limited mobile device screen was becoming more obvious when more features and application have come to mobile phones and the small screen is not enough for displaying a large amount of contents. As a solution, sensorialism [Tollmar et al., 2001] in the device to offload information onto other physical spaces and modalities [Ishii and Ullmer, 2000; Les et al., 2001], e.g., by displaying information on the sidebars or through vibration [Brewster and Brown, 2004; Tan, 2000] as in Figure 1. They further explained that commercially tactile sensations are separate from user interface sounds and it would be a logical next step to study integration of vibration content into the user interface.

Figure 1: An example of moving information off the screen space. The e380 phone has tri-color sidebands and vibration to light up and shake depending on ring tone.

Chang and Sullivan analyzed three solutions that can present vibration in mobile devices and selected Multi-Function Transducer, (MFT), a speaker which can produce both audible and vibrotactile output from an audio signal. MFTs were favored for many reasons (size, power efficiency, and least lag time). Another important feature of MFT is the effect of the synchronicity of sound with vibration, audiohaptics. Chang and Sullivan explained that the significance of this effect is that, if designed correctly, special audio files can be played on both MFT-enabled and non-haptic phones [Change and O'Sullivan, 2005].

They studied two audio manipulation techniques specific to the MFT, which are 1) Haptic Inheritance 2) Synthesis and Matching. According to Chang and Sullivan, both methods require a preliminary analysis of the sound and its components to determine the appropriate path for generation of the haptic effect. The 'Haptic Inheritance' method is based on the authors' premise that the sound itself contains enough inherent haptic information that this method of processing can exploit to deliver a pleasing and appropriate tactile or haptic

icon. The 'synthesis and matching' method enables the designer to approximate a desired response by adding extra haptic frequencies, perhaps from sounds of a haptic library. [Chang and O'Sullivan, 2005]

Chang and Sullivan conducted an evaluation comparing audio-based haptic user interface feedback with audio-only feedback. The preliminary results showed that users were receptive to audio-haptic UI feedback and audio-haptics seems to enhance the perception of audio quality. Drawn from their studies, the two methods of haptic media generation allow simple creation of vibration content, and also allow for compatibility with non-haptic mobile devices. Moreover, the use of audio to drive vibration allows other applications to access the haptic channel by simply playing audio. This feature opens up the possibility of adding haptic feedback to mobile games without the need for extra software control.

3.2. Tactons for mobile phone alerts

Tactons are structured vibrotactile messages which can be used for non-visual information presentation when visual displays are limited, unavailable or inappropriate, such as in mobile phones and other mobile devices [Brown et al., 2006]. Basically, tacton tries to transform non-visual information into tactile messages; the advantage of using tacton in mobile phone is that users don't need to look at small phone screen to get the information. Instead, they can use tactile feeling to 'feel' the information.

Brown and Kaaresoja [2006] studied tactons for mobile phone alerts. They suggested two reasons for their study. The first reason is that vibration used for mobile phone alert is still very simple, not fully extend the potential of vibrotactile message functions. Secondly, the recent development in vibration alerts in mobile devices done by different IT companies has been mostly used to enhance audio effects. Therefore, they wanted to study the possibility of using multi-dimensional tactons for communicating complex information with current standard mobile phone vibration motors.

The study tests different levels of vibration roughness and intensity to present the priorities of incoming calls, which means, in practice, that users can feel who's calling by detecting from different vibrations. Brown and Kaaresoja used a mockup of a widely used mobile phone (Nokia 8210), containing a standard phone vibration motor. They created nine tactons to represent alerts which might occur when a message or a call arrives on a mobile phone. Two pieces of information were encoded in each tacton: the type of alert (voice call, text message, or multimedia message) and the priority (low, medium or high) of the alert. The types of alert were represented by three rhythms which had been shown to be distinguishable in a previous study [Brown et al., 2006].

Two different vibrotactile parameters (roughness and intensity of the vibration) were tested to encode the priority attribute of the tactons. It has been suggested that intensity is an unsuitable parameter for tactons as a vibration which is too strong that could be unpleasant, while a vibration which is too weak might make it difficult to detect other parameters [Brown and Kaaresoja, 2006]. However, their pilot testing indicated that the intensity levels chosen for the study were all perceived to be pleasant, and it was intuitive that the more important an alert was, the stronger it would feel.

Their experiments showed that higher recognition rates were achieved when tactons encoding two pieces of information (type and priority of alert) were created by encoding information in rhythm and intensity. These results are comparable to those achieved for two-dimensional tactons presented on a high-price vibrotactile transducer. Using tactons in mobile phone alerts would enable more information to be transmitted in these alerts, and offer personalization features, such as assigning personalized vibration "ring tones" to different callers.

3.3. Haptic gesture interaction for mobile devices

Linjama and Kaaresoja [2004] studied the opportunity of using haptics gesture for mobile interaction. Since the development for haptic applications started gradually shifting from desktop environment to mobile environment, more possibilities, opportunities and challenges have come along in the context of mobile haptics. Linjama and Kaaresoja wanted to explore possibilities for haptic interaction that are suitable for mobile devices, in particular, explore the limited gesture interaction with mobile devices.

The example application used in their study was a bouncing ball game. Like many other gesture recognition applications, controlling ball motion is done by tapping the device in horizontal or vertical directions. Unlike the other, the gesture input is minimal consisting of simple tactile feedback events. Linjama and Kaaresoja designed very simple haptic I/O events: "tap" input with fingers, and "kick" feedback with vibration, used in combination with graphical events. By tapping the device in X or Y direction the user can put the ball moving in corresponding direction on the screen, see Figure 2. The ball bounces from the walls, and these events are supported by short vibration bump feedback. Synchrony of graphical and tactile events (tap input and kick feedback) creates a kind of a kinesthetic illusion of a soft ball being tapped and bouncing inside the device.

Figure 2. Demonstration of device and haptic gesture interaction

According to the informal evaluations conducted by Linjama and Kaaresoja [2004], most users rated this illusion very natural, impressive, and enjoyable. This demo would help the CHI community to direct effort also towards the simplistic extreme of interfaces, and explore applications for them in the mobile context.

4. Discussions and summary

Haptics is a very important interaction channel in the real world. We explore the things via touching, feel via touching, communicate and interact via touching. While seeing and hearing are still primary interaction channels in HCI user interface, haptics is getting more and more attention as it can be used as additional or optional channel to seeing and hearing. With haptic user interface, interaction becomes more dimensional, more intuitive and more real. Without the doubt, haptics would bring the substantial benefits for visual-impaired or audio-impaired users.

Mobile device has small screen, but in the same time functionalities are becoming richer and richer, haptics seems a promising solution to solve this conflict. So far, researchers have developed haptic applications mostly for enhancing audio effects, vibration alerts and haptic feedback in mobile games. There are potential opportunities to be explored, but due to limitation of the haptic and mobile device technology as well as lacking of empirical study regarding use of haptics, the development for mobile haptic applications is still slow. Overall, this area is quite challenging and far behind the developed.

The positive side is that users have given good feedback in evaluations to the applications presented in this paper. Another good side we see from this study is that the researchers tend to create simpler interface and discover inexpensive technology for haptics.

5. Future Development

Thermal and electrical modalities are not presented in this paper, because there are little haptic applications for mobile device that use such modalities. It is more critical to apply thermal and electrical to haptics as they are very sensitive to users. When using thermal and electrical in haptic user interface, two things must be considered, first is how much heat or electrical power is used so that it doesn't cause unpleasantness to users and the second is that different uses have different response to temperature and electrical feedback. In the future, more empirical research needs to be done in this area.

In addition, technological problems in implementing force feedback need to be overcome, and tactile feedback depending on the actual situation such as driving, running and the contact the device has with the user need to be studied more in the future.

References

[Brewster and Brown, 2004] Brewster, S., and Brown, L. Demonstrations: Non-visual information display using tactons, In: *Ext. Abstracts CHI 2004*, ACM Press, 787-788.

[Brown and Kaaresoja, 2006] Brown, L.M., and Kaaresoja, T. Feel who's talking: using tactons for mobile phone alerts, In: *Extended Abstracts of CHI 2006* (Montreal, Canada), ACM Press, 604-609.

[Brown et al., 2006] Brown, L.M., Brewster, S.A., and Purchase, H.C., Multidimensional tactons for non-visual information display in mobile devices, In: *Proc. Mobile HCI 2006*, ACM Press (2006), 231-238.

[Chang and O'Sullivan, 2005] Chang, A., and O'Sullivan, C., Audio-haptic feedback in mobile phones, In: *Proc. CHI 2005*, ACM Press (2005), 1264 - 1267.

[Immersion VibeTonz System] Immersion web pages
http://www.immersion.com/mobility/.
http://www.immersion.com/mobility/technology/#haptics
http://www.immersion.com/mobility/in_the_news.php.

[Ishii and Ullmer, 2000] Ishii, H., and Ullmer, B., Emerging frameworks for tangible user interfaces, *IBM Systems Journal* **39**, (3&4) (2000), 915-931.

[Les et al., 2001] Nelson, L, Bly, S., and Sokoler, T., Quiet calls: talking silently on mobile phones, In *Proc. CHI 2001*, ACM Press, 174-181.

[Linjama and Kaaresoja, 2004] Linjama, J., and Kaaresoja, T., Novel, minimalist haptic gesture interaction for mobile devices. In: *Proc. CHI 2004*, ACM Press (2004), 457 - 458.

[Nokia Digital Pen] Nokia UK, Nokia Digital Pen SU-1B support page, http://www.nokia.co.uk/A4222118. Checked 20.12.2006.

[Salisbury, 1998] Ken Salisbury. *SIGGRAPH 98 Tutorial on Physical Interaction.* The Artificial Intelligence Lab in Massachusetts Institute of Technology.

[Tan, 2000] Tan, H. Z., Perceptual user interfaces: haptic interfaces, *Communications of the ACM* **43**, 3 (Mar. 2000), 40–41.

[Tollmar et al., 2001] Tollmar, K., Junestrand, S. and Torgny, O., Virtually living together, In: *Proc. DIS 2000*, ACM Press (2001), 83-91.

Lääkitystiedon hallinta – kansallinen ja kansainvälinen tilanne

Joonas Mäkinen

Tiivistelmä.
Lääkitystieto ja sen saatavuus ovat keskeisessä asemassa terveydenhuollon arjessa. Organisaatiorajat ylittävät hoito- ja palveluketjut edellyttävät kehittynyttä tietojärjestelmien yhteistoiminnallisuutta ja tiedonvälitystä. Tutkimus kartoittaa kansallisen ja kansainvälisen lääkitystiedon hallinnan ja lääkitystiedon integraation tilannetta sekä tapahtuvia muutoksia. Lopuksi tarkastellaan lääkitystiedon käsittelyä Turun terveystoimen Pegasos-potilastietojärjestelmässä ja raportoidaan käyttäjien kokemista hyödyistä ja kehitystarpeista. Pegasoksen lääkitystietoja arvioidaan myös suhteessa tuleviin kansallisiin uudistuksiin, ennen kaikkea ydintietomäärittelyjen käyttöönottoon.

Avainsanat ja -sanonnat: Sähköinen resepti, sähköiset potilaskertomusjärjestelmät, terveydenhuollon tietojärjestelmät, kokonaislääkitys.
CR-luokka: J.3

1. Johdanto

Suomessa käynnistettiin vuonna 2001 Kansallinen TerveysHanke terveydenhuollon tulevaisuuden turvaamiseksi. Hankkeen taustalla on väestön ikärakenteen muutos ja tarve saada kuriin jatkuvasti nousevat terveydenhuollon kustannukset. Yhtenä keskeisenä keinona on nähty sosiaali- ja terveydenhuollon palveluiden tehostaminen muun muassa organisaatiorajat ylittäviä saumattomia hoito- ja palveluketjuja kehittämällä. Edellytyksenä saumattomalle yhteistoiminnallisuudelle yli organisaatiorajojen on kuitenkin saumaton tiedonvälitys, jonka haasteisiin on vastattu käynnistämällä erilaisia alueellisia ja kansallisia kehityshankkeita sekä muun muassa sopimalla käytetyistä standardeista ja uudistamalla lainsäädäntöä. Kansallisella tasolla on pilotoitu esimerkiksi sähköistä reseptiä sekä suunniteltu kansallista arkkitehtuuria. Alueellisesti on viime vuosina käyttöönotettu terveydenhuollon alueellisia tietojärjestelmiä ja aluetietojärjestelmiä. Tulevaisuudessa keskeiset potilastiedot tullaan arkistoimaan keskitettyyn kansalliseen sähköiseen arkistoon, josta tiedot ovat saatavissa valtakunnallisesti. Sähköinen resepti pyritään ottamaan käyttöön lähivuosina.

Tiedonvälitys yli organisaatiorajojen järjestelmien välillä edellyttää laajaa yhteistoiminnallisuutta, standardeja ja semanttista yhteensopivuutta. Tätä varten on tehty standardointityötä ja ydintietomäärittelyt potilastietojärjestelmille.

Terveydenhuollossa lääkitystieto on usein hyvin keskeisessä asemassa päivittäisessä työssä. Potilaan kokonaislääkitys olisi voitava saada selville uutta lääkettä määrätessä tai kirjattaessa potilasta osastolle. Saumattomien hoito- ja palveluketjujen myötä potilas on tekemisissä yhä useamman eri terveydenhuollon yksikön kanssa, minkä seurauksena lääkitystieto voi hajaantua ja sen välittyminen mutkistua. Siksi on entistäkin tärkeämpää ratkaista, miten ajantasaiset lääkitystiedot voidaan tuoda kaikkien intressiryhmien – lääkärin, kotisairaanhoitajan, potilaan tai omaisen – saataville helposti. Tällä hetkellä lääkitystiedon käsittelyn kysymykset ovat monilta osin ratkaisematta. Ongelmia on sekä tiedon sisällössä että tiedon välittämisessä organisaatioiden välillä ja jopa toimittaessa organisaation sisällä.

Tämä tutkimus liittyy Tekes-rahoitteiseen *Lääkitystiedon hallinta* -hankkeeseen. Hanke on jaettu viiteen osahankkeeseen, joista Tampereen yliopistolle on delegoitu *Lääkitystiedon integrointi potilastietojärjestelmissä*. Hankkeen tutkimustehtävänä on tukea lääkitystiedon hallinnan muita kansallisia hankkeita kansallisiin infrastruktuuriratkaisuihin pohjautuvin sovellus- ja prosessitason ratkaisuin. Tutkimukseni on osa nykytilanteen kartoitusta ja pohjustaa siten osahankkeen jatkoa.

Tutkimusta on tehty Turun terveystoimessa, jossa on käytössä Pegasos-potilastietojärjestelmä. Lääkitystiedon käsittelyn nykytilaa Turun terveystoimessa on kartoitettu haastatteluin sekä aiempiin tutkimuksiin, Pegasos-ohjelmiston käyttöohjeeseen ja demo-ohjelmaan tutustuen.

2. Lääkitystiedon integrointi potilastietojärjestelmissä – tilannekartoitus

2.1. Kansallinen tilanne

Lähes jokaisessa perusterveydenhuollon ja erikoissairaanhoidon yksikössä on käytössä jokin potilastietojärjestelmä [Winblad et al., 2006]. Järjestelmään kirjataan potilaan perustiedot, allergiat, määrätyt lääkkeet, hoitotapahtumat, diagnoosit ja muuta hoitotyöhön liittyvää. Potilastiedot ovatkin helposti saatavilla ilman perinteisen potilasarkiston penkomista ja paperien siirtelyä. Järjestelmä toimii yleensä hyvin niin kauan kuin kaikki potilaan tiedot ovat samassa järjestelmässä. Ongelmia voi tulla, mikäli potilas on liikkunut esimerkiksi erikoissairaanhoidon ja perusterveydenhuollon tai avo- ja osastohoidon välillä. Tällöin hoidon kannalta olennaista tietoa voi olla useassa eri järjestelmässä ja käytännössä tietoja saatetaan joutua kyselemään puhelimitse tai muilla keinoin. Tieto ei kulje järjestelmä- ja organisaatiorajojen yli. Potilastietojärjestelmäratkaisuja on käytössä useita ja jokainen asennus on erillinen, vaikka järjestelmäntoimitta-

ja olisikin sama. Pelkästään yhden kunnan alueelle mahtuu jo useita järjestelmiä, Tampereella on käytössä ainakin WM-datan Pegasos ja Medici Datan Miranda sekä epäilemättä useita muita järjestelmiä yksityisellä puolella.

Viime vuosina Suomessa on rakennettu erilaisia alueellisia tietojärjestelmäratkaisuja, kuten viitetietokantaan perustuvia aluetietojärjestelmiä. Tällainen on esimerkiksi Satakunnassa sekä Pirkanmaalla käyttöönotettu Fiale-informaatiopalvelu. Aluetietojärjestelmään voi olla kytkeytyneenä muun muassa erikoissairaanhoidon ja perusterveydenhuollon yksiköitä. Järjestelmän kautta saadaan nähtäville järjestelmään kytkeytyneiden potilastietojärjestelmien sisältämät tiedot potilaan hoidosta. Potilastiedot säilytetään paikallisesti, eikä niihin ole muutosoikeutta ulkopuolisesta järjestelmästä käsin. Tietojen katselu vaatii myös potilaan suostumuksen. Aluetietojärjestelmiä on otettu tuotantokäyttöön muutamassa sairaanhoitopiirissä, mutta noin puolessa sellaista ei ole lainkaan. Alueellisesti on kuitenkin käytössä tätä runsaammin esimerkiksi lähete–palautejärjestelmiä, joiden avulla lähetteet ja palautteet kulkevat sähköisesti terveyskeskuksen ja erikoissairaanhoidon välillä. Myös muun muassa röntgen- ja magneettikuvien digitaaliseen tallentamiseen tarkoitetun PACS-kuvantamisjärjestelmän etäkäytön liittymiä on käytössä yhä enemmän [Winblad et al., 2006].

Sosiaali- ja terveysministeriön koordinoiman kansallisen terveyshankkeen sähköisten potilasasiakirjojen määrittelytyössä on haluttu luoda perusta valtakunnalliselle potilastietojärjestelmien yhteistoiminnallisuudelle. Hankkeelle strategian vuonna 2003 laatinut työryhmä on esittänyt tavoitteeksi, että vuoden 2007 loppuun mennessä potilastietojärjestelmien tulee käyttää yhdenmukaisia rakenteisia ydintietoja [STM, 2003]. Rakenteinen ja yhteisesti sovittuja standardeja hyödyntävä malli mahdollistaa saumattoman tiedonsiirron alueellisella ja kansallisella tasolla järjestelmien välillä sekä keskitetyn kansallisen sähköisen arkiston käyttöönoton. Määrittelytyössä on sovittu yksityiskohtaisesti käytetyistä luokituksista ja semantiikasta sekä käytetyistä otsikkotiedoista ja näyttömuodoista.

Tietojärjestelmäarkkitehtuuri ja kansallistenpalveluiden määritystyö on vielä osittain kesken ja sen loppuun viejäksi on kilpailutuksen perusteella valittu WM-data. Työssä on mukana laaja arviointiryhmä. Työn tavoitteena on tarkentaa ja yhdenmukaistaa dokumentaatiota, tuottaa kuva kansallisesta kokonaisarkkitehtuurista ja määrittelyt Kelan ylläpitämistä valtakunnallisista palveluista. Määrityksien on määrä olla valmiina 28.2.2007 mennessä [STM, 2006a]. Kela tulee ottamaan vastuulleen reseptikeskuksen ja kansallisen keskitetyn arkiston ja myös muita palveluita. Arkistoon tullaan tallentamaan keskeiset potilaskertomustiedot tulevien määritysten mukaisesti.

Rakenteisilla ydintiedoilla tarkoitetaan ”yhteisesti sovittua tietosisältöä keskeisimmille potilaskertomuksen tiedoille” [STM, 2006b]. Rakenteisten ydintietojen määrittely kattaa kaikki potilaskertomuksen keskeiset osat sisältäen siten myös potilaan lääkehoidon. Ydintietomäärittelyissä opastetaan, miten lääkitystietoa kirjataan ja millainen on sen tietosisältö.

Kuntaliiton koordinoimissa klusterihankkeissa valmistellaan valtakunnallisten vaatimusten käyttöönottoa eri toimittajien potilastietojärjestelmissä. Klusterihankkeet jakautuvat käytetyn potilastietojärjestelmän mukaisesti: Proxit (Effica), Pegasos (WM-data), Miranda (Medici Data) ja Kaisa (Esko). Hankkeiden välinen yhteistyöelin on POKANEN-työryhmä. Järjestelmien täytyy tulevaisuudessa hyödyntää rakenteisia ydintietoja, valittuja koodistoja, kuten OID, ja HL7 CDA R2 -rajapintaa, joka on yksi edellytys tulevan potilaskertomusten keskitetyn arkiston käytölle. Lisäksi on toteutettava lain ja määritysten edellyttämällä tavalla kansallinen varmennepalvelu, sähköinen allekirjoitus ja suostumuksenhallinta. [STM, 2006a; STM, 2006d]

Tiedon rakenteisuudesta seuraa järjestelmien yhteistoiminnallisuuden lisäksi monia muitakin hyötyjä. Muun muassa päällekkäinen kirjaaminen vähenee, kun hoitotilanteessa voidaan kirjata vain kyseisen hoitotapahtuman kannalta olennainen tieto. Rakenteisuus tuo myös hyötyä uusien tiedon jäsentämismahdollisuuksien myötä, kun samasta tiedosta voidaan luoda erilaisia näkymiä käyttötarpeen mukaisesti [STM, 2006b, 6-7]. Tiedonhaku tehostuu yhä suuremman osan tiedosta ollessa rakenteisessa muodossa. Lääkitystiedon kannalta ydintietojen käyttöönotto tarkoittaisi sitä, että lääkityksen ollessa kaikissa järjestelmissä tai tulevassa kansallisessa keskitetyssä arkistossa samalla tavalla rakenteisesti ilmaistuna, voidaan tiedoista helpommin muodostaa yksi yhdenmukainen kokonaisuus, esimerkiksi lääkityslista potilaalle annettavaksi. Tällä hetkellä Turun terveystoimen Pegasos-järjestelmästä voidaan lääkityslistaan tulostaa vain ne lääkkeet, jotka on kirjattu kyseiseen järjestelmään. Tietoja naapurikunnassa määrätyistä lääkkeistä ei voida poimia esimerkiksi aluetietojärjestelmästä. Tilanne vaikuttaisi olevan pitkälti sama myös muualla.

2.1.1. Sähköinen resepti

Suomessa otetaan lähivuosina käyttöön *sähköinen resepti*, joka tunnetaan myös nimillä eResepti ja e-resepti. Muutosta ohjaa valtioneuvoston 2.11.2006 hyväksymä esitys sosiaali- ja terveydenhuollon asiakastietojen sähköistä käsittelyä koskevaksi lainsäädännöksi [HE, 2006a] sekä erityisesti esitys laiksi sähköisestä lääkemääräyksestä ja lääkelain muuttamisesta [HE, 2006b]. Lait astuvat voimaan vuoden 2007 alusta. Lääkemääräykset tallennetaan valtakunnalliseen Kelan ylläpitämään reseptikeskukseen, jota neljän vuoden siirtymäajan jälkeen tulevat käyttämään kaikki julkisen terveydenhuollon yksiköt ja apteekit. Myös

kansalaisen mahdollisuutta katsoa reseptejään omatoimisesti esimerkiksi kotoa internetin yli on ehdotettu toteutettavaksi. Perinteinen paperiresepti säilyy vaihtoehtoisena lääkemääräyksen tekotapana, jota voidaan käyttää esimerkiksi silloin, jos potilas ei salli sähköisen reseptin käyttöä. Sähköisen reseptin ja yhteisen reseptitietokannan odotetaan tuottavan runsaasti hyötyjä muun muassa käsialaongelmien poistuessa, päällekkäisen tai tarpeettoman lääkityksen ja interaktioiden eli haitallisten yhteisvaikutusten riskin vähentyessä sekä yleisesti koko toimitusketjun parantumisen myötä. Reseptit eivät myöskään pääse enää hukkumaan, sillä potilas saa jatkossa mukaansa vain reseptiksi kelpaamattoman potilasohjeen varsinaisen lääkemääräyksen säilyessä tietokannassa.

Sähköinen resepti parantaa merkittävästi terveydenhoidon osapuolten mahdollisuuksia selvittää potilaan *kokonaislääkitys*. Kokonaislääkityksen käsitteelle ei liene vakiintunut vielä määrittelyä. Yleensä sillä tarkoitetaan sitä kokonaisuutta, joka muodostuu yksilön resepti- ja käsikauppalääkkeistä sekä lääkkeisiin verrattavista luontaistuotteista. Määrittelyyn liittäisin ehkä vielä myös lääkityshistorian, sillä se on usein keskeinen määrättäessä uutta lääkettä tai uusittaessa vanhaa reseptiä. Sähköisen lääkemääräyksen lakiesityksestä ei mielestäni tule tarpeeksi korostuneesti ilmi, että reseptikeskuksen tarjoama tieto ei ole aivan sama asia kuin kokonaislääkitys. Yksikön käyttämän potilastietojärjestelmän tai kansallisen sähköisen arkiston lääkitystiedoilla on edelleen keskeinen rooli samoin kuin potilaan haastattelulla. Vaikka lääkityksen riittävään kartoittamiseen saattaa riittää pelkkien vanhojen tai voimassaolevien reseptien huomioiminen, ainoastaan näin saatuun tietoon ei voida aina tukeutua. Potilas on voinut saada tavallisesti syömiensä reseptilääkkeiden lisäksi muuta lääkitystä osastohoidossa, ehkä lääkemääräyksen paperireseptinä tai hän saattaa käyttää käsikauppalääkkeitä ja luontaislääkityksiä, joilla voi olla suurikin merkitys uutta lääkemääräystä tehtäessä. Tällaisista muista lääkityksistä voi olla merkintä potilastietojärjestelmässä, muttei aina reseptiä. Reseptitietokanta on kuitenkin hyödyllinen esimerkiksi, jos potilas on asioinut eri terveydenhuollon yksiköissä, mahdollisesti toisessa kunnassa, tai selvitettäessä, mitkä lääkkeet potilas on todellisuudessa noutanut apteekista. On myös hyvin käytännöllistä, mikäli tulevaisuudessa voi apteekista tai lääkäriltä saada mukaansa lääkelistan käyttöohjeineen. Se myös parantaa mahdollisuuksia havaita mahdolliset interaktiot viimeistään apteekissa.

Suomessa on pilotoitu sähköisen reseptin käyttöä vuosina 2002–2006. Pilotoinnin yhteydessä on tullut esiin erilaisia ongelmia nykyisessä lääkitystiedon käsittelyssä. Pilotoinnin raporteista [Hyppönen, 2005; Hyppönen et al., 2006] ilmenee, miten eri potilastietojärjestelmissä lääkitystieto on toteutettu vaihtelevin tavoin ja että käytössä on erilaisia luokituksia ja tietokantoja. Tällaiset on-

gelmat vaikeuttavat merkittävästi valtakunnallisen eReseptin käyttöönottoa. Epäyhtenäiset ja lukuisat erilaiset tietojärjestelmätoteutukset jarruttavat kaiken kaikkiaan kansallisen tietoarkkitehtuurin toteutumista ja keskitetyn sähköisen arkiston käyttöönottoa.

2.2. Kansainvälinen tilanne

Eri maiden terveydenhuollon tietojärjestelmähankkeita vertaillessa voidaan havaita arkkitehtuurien olevan lähtökohdiltaan selvästi erilaisia. Sosiaali- ja terveysministeriön selvityksessä [STM, 2006c] vertaillaan kansainvälisesti arkkitehtuuriratkaisuja ja todetaan tietoarkkitehtuurin näkökulmasta löytyvän ainakin kolme ratkaisumallia. *Keskitetyssä mallissa* kansalaisen terveystiedot tallennetaan keskitettyyn valtakunnalliseen tietokantaan tai arkistoidaan keskitetysti. *Hajautetussa mallissa* tiedot sijaitsevat erillään perusjärjestelmien tietokannoissa ja niihin päästään käsiksi valtakunnallisen tason linkityksen kautta. *Terveyskorttimallissa* potilas-, lääkitys- ja viitetiedot kulkevat kortilla potilaan mukana. Tällöin osa tiedoista on kortilla ja osa mahdollisesti kortin viitetietojen avulla saatavilla muista järjestelmistä. Ratkaisumallin valintaan vaikuttaa luonnollisesti se, miten kyseisen maan terveydenhuolto on järjestetty. Vakuutuspohjainen terveydenhuollon malli on käytössä esimerkiksi Saksassa, mikä selvityksen mukaan lienee vaikuttanut siihen, että Saksassa on valittu ratkaisuksi jokaiselle kansalaiselle annettava sähköinen terveyskortti. Toisaalta Hollannissa, jossa on myös vakuutuspohjainen järjestelmä, potilastiedot säilytetään paikallisissa järjestelmissä. Suomi on valinnut valtakunnallisen keskitetyn potilastietojen arkistointiratkaisun, jolloin keskeiset osat potilastiedoista tallennetaan saataville koko maassa. Myös monissa muissa maissa on suunnitteilla siirtyminen keskitetympään malliin, mutta missään maassa ei vielä ole sellaista täysimittaisessa tuotantokäytössä.

Sähköisen reseptin pilotoinnin ensimmäisessä vaiheessa kartoitettiin Ruotsin, Tanskan ja Saksan sähköisen reseptin silloinen tilanne. Tarkoituksena oli vertailla Suomessa valitun mallin perusteltavuutta muihin nähden. Tuloksena oli, että Suomen valitsema sähköisen reseptin ratkaisu on perusteltu ja sitä kannattaa viedä eteenpäin [Hyppönen, 2005]. Vertailuista ilmenee sähköisen reseptin toteutusratkaisun lisäksi paljon muutakin siitä, miten lääkitystiedon on tarkoitus kulkea toimijoiden välillä, kenellä on siihen pääsy ja mihin tietoa tallennetaan.

2.2.1. Ruotsi

Suomen sähköisen reseptin pilotoinnin yhteydessä on tutkittu Ruotsin ratkaisua erityisesti pilotin toisen vaiheen dokumentaatiossa, jossa on arvioitu Suomen ja Ruotsin sähköisen reseptin ratkaisuja sekä niiden toimivuutta [Hyppö-

nen et al., 2006]. Sähköisen reseptin käyttö on Ruotsissa kasvanut nopeasti vuosina 2000–2006, ja yli puolet lääkemääräyksistä toimitetaan nykyisin sähköisesti, vaikkakin maakuntien välillä on käyttöasteessa suuria eroja [HE, 2006b].

Lääkemääräys kirjoitetaan potilastietojärjestelmässä tai erillisellä ohjelmalla, mistä se kulkee maakuntaverkon reseptipalvelimen kautta valtakunnalliseen reseptipostilaatikkoon apteekin noudettavaksi. Resepti voidaan lähettää myös ”point-to-point” suoraan tiettyyn apteekkiin. Eri maakuntien ratkaisut eroavat toisistaan, esimerkiksi joissain maakunnissa resepti lähetetään suoraan valtakunnalliseen palveluun käyttämättä maakunnallista reseptipalvelinta. Suunnitteilla on kansallinen reseptitietokanta, kuten myös Suomessa.

Sähköisen reseptin toisen vaiheen raportissa [Hyppönen et al., 2006] keskeiseksi eroksi Suomen ja Ruotsin järjestelmien suunnittelun välillä katsotaan se, että Suomen malli on suunniteltu kattamaan kerralla kaikki toiminnot kun taas Ruotsin ratkaisua on rakennettu vaiheittain. Tulevissa Ruotsin sähköisen reseptin kehityssuunnitelmissa on joitakin merkittäviä eroja Suomen ratkaisuihin nähden. Suomessa lääkärillä tulee olemaan oikeus tarkastella potilaan koko reseptihistoriaa, Ruotsissa tämän saavat tehdä vain farmaseutit. Ruotsissa ei myöskään ole suunnitteilla korvauskäsittelijöille pääsyä reseptitietoihin, Suomessa aiotaan toteuttaa Kelalle sähköinen korvauskäsittely. Huomionarvoista on myös, että Ruotsissa ei ole suunniteltu potilaalle mahdollisuutta tarkastella kaikkia reseptipostilaatikon tietoja, vaan potilaalla on pääsy ainoastaan toimitettujen lääkkeiden rekisteriin, josta näkee toimitetut tai osatoimitetut lääkkeet. Suomessa on tarkoitus rakentaa kansalaiselle pääsy kaikkiin reseptikeskuksen tietoihin [Hyppönen et al., 2006].

Potilaan lääkitystiedot säilytetään Ruotsissa läänitasolla. Mikäli potilaalla on lääkeyliherkkyys, tartuntatauti tai hän on raskaana, järjestelmä varoittaa näistä tarpeen tullen. Kaikissa potilastietojärjestelmissä on käytössä kontraindikaatiotietokanta sekä tuotelista, jossa on kaikki ne lääkkeet, joita voi määräyksellä saada apteekista. Tuotelistoista tullaan muodostamaan läänikohtaisia lääkitystietokantoja, joiden päivityksestä vastaa Apoteket AB. [HE, 2006b]

Sähköisen reseptin käyttöönottoa on Ruotsissa helpottanut monikin seikka. Yksi tekijä lienee vaiheittainen ”bottom-up”-lähestymistapa, kun Suomessa on lähdetty suunnittelemaan valtakunnallista kokonaisuutta ”top-down”. Ruotsissa on käytössä valtakunnallinen terveydenhuollon erillisverkko Sjunet, johon ovat nykyisin liittyneet kaikki Ruotsin maakunnat. Sähköisen reseptin tietoliikenne voidaan hoitaa sen kautta jo sovituilla standardeilla, joita sähköisen reseptin osalta ovat EDI ja XML. Myös se, ettei Ruotsissa ole yksityisiä apteekkeja ja vain yksi ohjelmisto, on yksinkertaistanut käyttöönottoa.

2.2.2. Tanska

Tanska on ollut edelläkävijä sähköisen reseptin käyttöönotossa. Sähköisen reseptin lanseeraaminen aloitettiin osana kansallista MedCom-projektia 1995–1996. Resepti lähetetään lääkärin ja potilaan ennalta valitsemaan apteekkiin postilaatikon kautta, josta apteekki noutaa sen. Apteekista lähtee lääkkeen noutamisesta kuittaus reseptin kirjoittaneelle lääkärille. Resepteistä 81 % kulkee sähköisenä. Raportin [Hyppönen, 2005] mukaan myös Tanskassa on suunnitteilla keskitetty reseptipalvelin. Vanha järjestelmä toimii, mutta siinä on myös puutteita. Ongelmiksi koetaan muun muassa, ettei asiakas voi vapaasti valita apteekkia ja että reseptien toimitustietoja ei ole saatavilla. Myös uusimiskäytännössä on puutteita. Vaikka järjestelmä on toimiva, tekniikka alkaa olla vanhentunutta ja Tanskassa on tehty päätös uuteen järjestelmään siirtymisestä. Health Data Network tulee yhdistämään eri terveydenhuollon toimijat.

Tanskassa on avattu portaali http://www.sundhed.dk, jonka englanninkielisen esittelymateriaalin [Sundhed.dk, 2006] perusteella palvelun avulla kansalaisella näyttäisi olevan lähitulevaisuudessa pääsy ainakin joihinkin potilaskertomustietoihin, esimerkiksi hoitohistoriaan ja lääkitysprofiiliin. Ammattilaiset puolestaan voivat tulevaisuudessa potilaan luvalla katsoa potilaan tietoja, kuten lääkitystietoja tai laboratoriotuloksia. Portaalissa on myös yhteystietoja, tietoa laeista, hoitoonpääsyajoista ja yleensä terveydestä.

2.2.3. Slovenia

Slovenian lääkitystiedon hallintaa ja sähköisen reseptin hanketta on esitelty World of Health IT 2006 -konferenssissa [Sušelj, 2006]. Sloveniassa on käytössä HIC (Health Insurance Card), joka on pilotin jälkeen saatu nopeasti valtakunnalliseen käyttöön vuoden 2006 keväällä. Kortti jaetaan kaikille siihen oikeutetuille asukkaille, kortteja tarvitaan kaikkiaan noin kaksi miljoonaa. Kortille tallennetaan tällä hetkellä tieto potilaan saamasta hoidosta, tieto elinluovutuksen sallimisesta ja tieto määrätyistä lääkkeistä. Työn alla on lisätä kortille mahdolliset allergiat ja lääkeyliherkkyydet. Käyttöön otetaan pian myös sähköinen resepti. Valtaosalla potilaista on jo lääkityshistoria tallennettuna kortille. Vuoden 2006 touko- ja lokakuun välillä on korteille tallennettu 18 miljoonaa lääkemääräystä.

Tekninen muutos vuosina 2006–2008 on PKI:n käyttöönotto, jolloin HIC:sta tulee avainkortti, joka mahdollistaa tunnistautumisen. Lisäksi aiotaan mahdollistaa online-pääsy vakuutus- ja lääkitystietoihin.

Lääkärillä on korttiin lukuoikeus, farmaseutilla myös kirjoitusoikeus. Noudettaessa lääkettä apteekista kirjautuu sekä kortille että kansalliseen keskitettyyn tietokantaan tieto määrätystä lääkkeestä. Käytössä on myös itsepalvelu-

päätteitä. Päätteen avulla kortille voi päivittää keskitetyn tietokannan lääkitystiedot.

Kortille tallentuvat määrätyt lääkkeet HIV-lääkitystä lukuun ottamatta, ja merkintöjä voi olla enintään 46 kappaletta. Lääkemääräykseen merkitään lääkkeen tunniste (drug code), määrä, määräyspäivä, lääkkeen määrääjän tunniste ja lääkkeen luovuttavan apteekin tunniste.

Pääosin kokemukset ovat olleet hyviä niin potilaiden, lääkärien kuin farmaseuttienkin näkökulmasta. Potilaiden positiivista vastaanottoa uudelle järjestelmälle kuvastaa hyvin se, että vain kourallinen potilaita on kieltänyt farmaseuttia tai lääkäriä katsomasta korttinsa tietoja. Lääkkeiden määräämisen osalta ongelmia on epäyhtenäisissä työtavoissa ja myös puutteellinen tekninen varustus: lääkäreiltä puuttuu koneita, joilla työskennellä.

Farmaseutit näkivät tarpeellisena merkitä myös hankitut käsikauppalääkkeet kortille, mitä ei vielä ole toteutettu. Tietoa käsikauppalääkkeistä ei liene suunniteltu kirjattavaksi Suomen järjestelmässä.

2.2.4. Saksa

Saksan malli perustuu sähköiseen terveyskorttiin (eGesundheitskarte, eGK), joka korvaa vaiheittain vanhemman sairausvakuutuskortin. Myös vanhempi kortti oli sirullinen, mutta teknisesti vaatimattomampi. Vuoden 2005 lopussa aloitettiin uuden kortin laboratoriotestaus, jonka jälkeen seuraa vaiheittain laajeneva testaus valituilla alueilla. Valtakunnallinen käyttöönotto organisoidaan asteittain testialueista lähtien [Federal Ministry of Health, 2006]. Kortti on tarkoitus jakaa kaikille noin 80 miljoonalle asukkaalle. Terveysalan ammattilaisille jaetaan noin 300 000 korttia, joilla voi tehdä sähköisen allekirjoituksen [Hyppönen, 2005]. Toistaiseksi näyttäisi olevan epäselvää, mitä kaikkia tietoja tullaan tallentamaan kortille ja vastaavasti miltä osin tallennetaan vain viite. Sähköinen resepti tallennetaan joko suoraan kortille tai vaihtoehtoisesti tallentuu viite tai koodi, jolla reseptin saa auki apteekissa [HE, 2006b].

Suostumuskäytäntö on potilaan näkökulmasta yksinkertainen: potilaan tietoihin päästäkseen lääkäri tarvitsee myös potilaan kortin, joka on lisäksi suojattu PIN-koodilla. Akuutissa hoidossa on kuitenkin mahdollista päästä hoidon kannalta tärkeiksi määriteltyihin tietoihin myös ilman PIN-koodia [Federal Ministry of Health, 2006]. Yksinkertaisen suostumuskäytännön lisäksi kortin hyväksi puoleksi voi laskea sen konkreettisuuden potilaan näkökulmasta. Riippumatta siitä teknisestä ratkaisusta, tallennetaanko kortille varsinainen sähköinen lääkemääräys vai viite, sähköisen reseptin käyttö on potilaalle helppo ymmärtää.

Ensimmäisiä uuden kortin myötä käyttöönotettavia ominaisuuksia on sähköinen resepti, joka tallennetaan kortille. Jatkossa kortille tullaan tallentamaan

myös lääkitys reseptien perusteella luovutettujen lääkkeiden osalta joko lääkärin tai farmaseutin luovuttaessa lääkkeen. Apteekista ostetut käsikauppalääkkeet voidaan merkitä kortille. Kortille voidaan tallentaa myös tietoa akuuttihoitoa varten muun muassa lääkeyliherkkyyksistä, allergioista ja kroonisista vaivoista. Viimeisenä kortin ominaisuuksiin aiotaan lisätä ”sähköinen potilasasiakirja” (Elektronische Patientenakte, EPA), jossa voi olla referenssejä yksittäisiin hoitotapahtumiin, laboratoriotuloksia, röntgenkuvia ja tietoa muista tutkimuksista. Vain yksilön henkilö- ja vakuutustiedot, sähköinen resepti sekä eurooppalainen sairausvakuutuskortti ovat pakollisia ominaisuuksia, ja kansalainen voi kieltäytyä muiden ominaisuuksien käyttöönotosta. [Hyppönen, 2005; Federal Ministry of Health, 2006]

3. Lääkitystiedon käsittely Pegasos-potilastietojärjestelmässä

Lääkitystiedon käsittelyyn on perehdytty Turun terveystoimessa, jossa on käytössä Pegasos-potilastietojärjestelmä sekä terveyskeskuksissa että vuodeosastoilla. Tässä luvussa selvitetään, mitä lääkitystietoa järjestelmässä on, mistä se löytyy ja miten sitä käsitellään avohoidossa ja pitkäaikaissairaanhoidon osastolla. Lisäksi on vertailtu Pegasoksen lääkitystietoja sähköisen potilaskertomuksen ydintietomäärittelyn suosituksiin. Selvityksen tukena on käytetty raporttien ja muun dokumentaation lisäksi haastatteluja avoterveydenhuollossa ja pitkäaikaissairaanhoidon osastolla sekä Pegasos Tutor 7.3 -ohjelmaa [WM-data, 2006b]. Kuvat ovat peräisin tutor-ohjelmiston osiosta *Onlineohje*.

Aineiston hankkimiseksi on haastateltu lääkäreitä ja hoitajia Turun terveystoimessa. Haastateltujen tehtävä ja haastattelun päivämäärä ovat listattuina taulukkoon 1. Osa haastatteluista tehtiin Turun yliopiston osahankkeen toimesta, joista on ollut käytettävissä translitteraatiot. Lukuun ottamatta lokakuista osastolääkärin haastattelua, samoja haastatteluita on käytetty myös Järven ja Tuomisen [2006] raportissa. Haastattelujen pituus vaihteli noin puolesta tunnista tuntiin.

Ammattinimike/Rooli	**Päivämäärä**
Terveyskeskuslääkäri	19.4.2006
Osastolääkäri	9.5.2006* ja 16.10.2006
Apulaisosastohoitaja	10.5.2006* ja 1.6.2006*
	**translitteraatio*

Taulukko 1: Haastateltujen ammattinimikkeet.

3.1. Pegasos-potilastietojärjestelmä

Pegasos on WM-data Oy:n potilastietojärjestelmä perusterveydenhuollon potilastietojen hallintaan esimerkiksi terveyskeskuksissa, erikoissairaanhoidossa ja sosiaalitoimessa. Pegasoksella on merkittävä markkina-asema Suomen perusterveydenhuollossa. Selvityksen mukaan Pegasosta käyttää tällä hetkellä lähes puolet niistä terveyskeskuksista, joissa on jokin potilaskertomusjärjestelmä tuotantokäytössä [Winblad et al., 2006].

Pegasoksen keskeiset osat ovat *avohoito*, *osastohoito* ja *kotihoito*. Ohjelma koostuu useista komponenteista, joita järjestelmän tilaaja voi ottaa käyttöön tarpeidensa mukaan [WM-data, 2006a]. Tämän tutkimuksen kannalta kiintoisia ovat avohoidon käyttämä *Vastaanoton työasema* ja osastohoidon *Osastohoidon työasema*. Samoihin ohjelmistoihin viitataan myös silloin, jos työaseman sijaan puhutaan esimerkiksi *osastohoito-ohjelmistosta*.

Pegasos-potilastietojärjestelmässä lääkitystietoa käytetään erityisesti kahdessa eri kontekstissa: avohoidon lääkitys ja osastohoidon lääkitys. Yksinkertaistaen voidaan sanoa, että avohoidossa keskeistä on uusien reseptien kirjoittaminen ja vanhojen reseptien uusiminen potilaalle, osastolla lääkkeiden jakelu ja lääkityksen helppo selvittäminen päivittäisen hoitotyön yhteydessä. Avo- ja osastohoitoa on toki monenlaista, potilaiden hoitojaksot ovat eripituisia ja tarpeet vaihtelevat muutenkin.

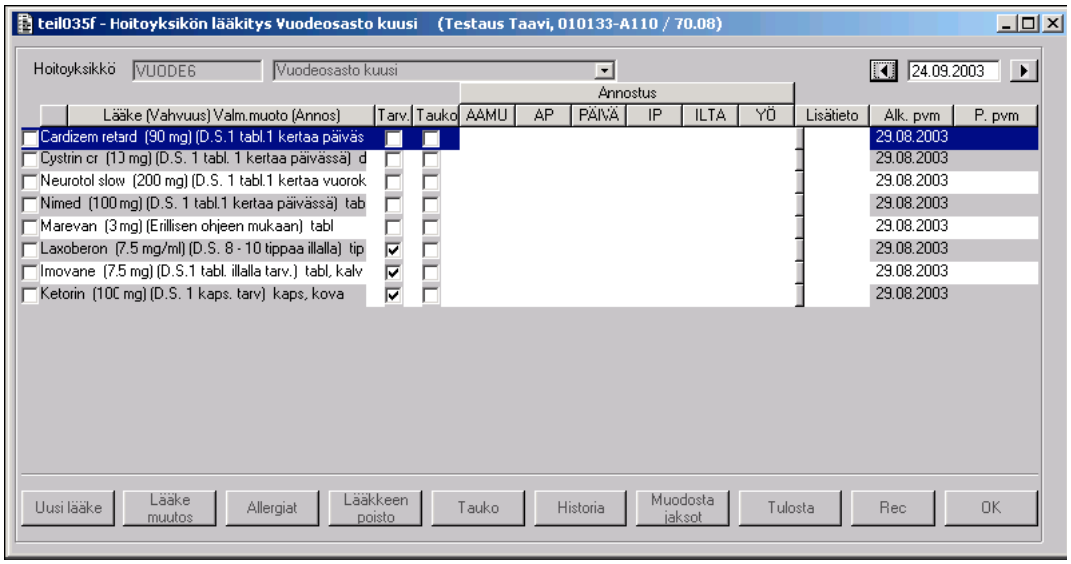
Avohoidon vastaanotolla tai reseptiä uusittaessa tarkistetaan potilaan voimassaoleva lääkitys ja mahdollisesti selataan muuta lääkityshistoriaa. Tarpeen tullen voidaan tarkastella myös potilaan osastohoidon lääkitystietoja. Lääkitystietoa voi selvitellä lisäksi selaamalla potilaskertomusta tai tekemällä siihen hakuja.

Osastohoidon malli perustuu potilaan hoitojaksoihin, ja lääkitys määritellään hoitojaksokohtaisesti. Osastohoidossa potilaan lääkitys selvitetään potilaan saapuessa osastolle joko lääkekortista, avohoidon määrätyistä lääkkeistä tai muilla keinoin, esimerkiksi haastattelemalla potilasta. Saadun tiedon pohjalta potilaalle tehdään uusi *lääkekortti*, johon hoidon osapuolet tukeutuvat myöhemmin hoitojakson jatkuessa.

Kahteen käyttötilanteeseen perustuva ohjelmien jaottelu näkyy järjestelmän käyttäjälle erityisesti siinä, että lääkitystietoa on kahdessa eri paikassa – lääkekorttien hoitojaksokohtainen lääkitys antaa vain ne lääkkeet, joita käytetään osastolla, eivätkä tiedot välttämättä riitä uutta lääkemääräystä tehdessä. Vastaavasti avohoidon määrätyissä lääkkeissä ei välttämättä näy kaikkia niitä lääkkeitä, joita potilaalla on ollut hoitojaksolla, ja silloin on tutkittava lääkekortteja. On perusteltua esittää potilaan lääkitys sellaisessa muodossa, joka parhaiten

tyydyttää käyttäjän tarpeet. Kokonaislääkityksen selvittäminen ei saisi tästä kuitenkaan vaikeutua.

Osastohoidolle ja vastaanotolle on Pegasoksessa omat osionsa, joissa lääkitystiedon hallinta on toteutettu käyttötarkoituksen mukaisesti eri tavalla. Avohoitoa varten on *Vastaanoton työasema*, josta saa näkyviin näytön *Määrätyt lääkkeet*. *Osastohoidon työasemasta* (osastohoito-ohjelmisto) löytyy *osastonlääkitys* (Kuva 1), jossa lääkitystä voi katsella osasto- tai potilaskohtaisesti [WM-data, 2006b]. Ohjelmistojen välillä on yhteistoiminnallisuutta, esimerkiksi osastohoidon työasema toimii saumattomasti osana avohoidon järjestelmää [WM-data, 2006a]. Osastohoidon ohjelmiston kautta on mahdollista nähdä avohoidossa määrätyt lääkkeet ja vastaavasti vastaanoton työasemalta pääsee käsiksi osastohoidon lääkitykseen. Tämä ei kuitenkaan ole aivan ongelmatonta: osastolääkäri kertoi avaavansa usein kaksi Pegasosta, jolloin potilaan tietojen käsittely helpottuu pitämällä näitä kahta ohjelmistoa auki yhtä aikaa erillisinä kokonaisuuksina. Ensimmäisessä ohjelmassa voi olla auki esimerkiksi osastohoidon lääkekortti ja toisessa vastaanoton työaseman lääkeyhteenveto.



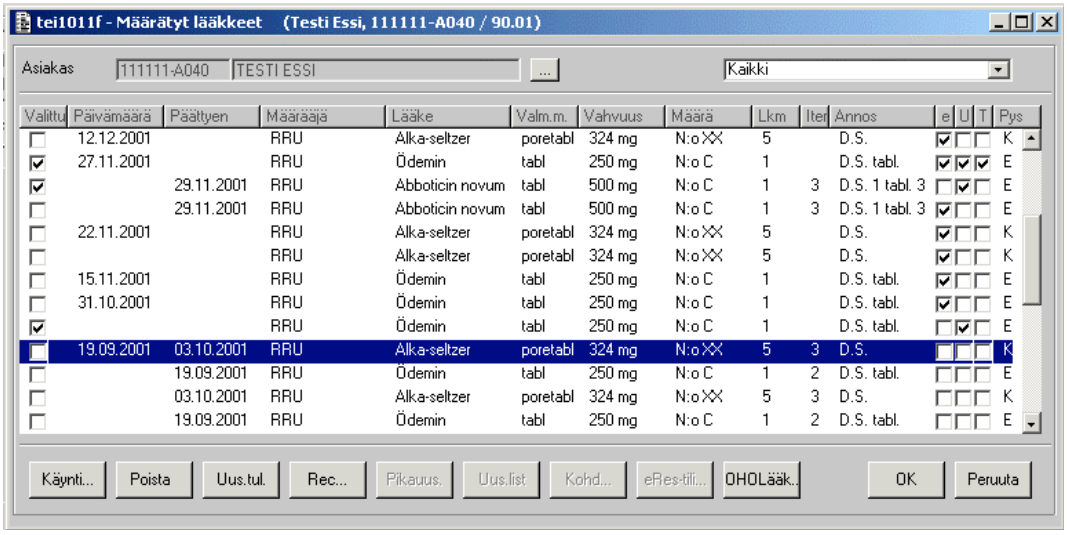
Kuva 1: Hoitoyksikön lääkitys

3.2. Avohoidon lääkitys

Avohoidon lääkitys sisältää terveyskeskuksessa tai muualla Turun terveystoimessa kirjoitettuja reseptejä (lääkemääräyksiä) sekä uusittuja reseptejä. Vastaanoton ohjelmistossa on näyttö ”Määrätyt lääkkeet” (Kuva 2), jossa on lääkeyhteenveto. Näytön lääkitystiedot perustuvat järjestelmässä kirjoitettuihin resepteihin ja mahdollisiin muihin lääkärin tai hoitajan toimesta merkittyihin lääkityksiin. Näitä voivat olla potilaan kertomat muualta saadut voimassaolevat lääkitykset. Tehdessään lääkemääräyksen lääkäri poimii lääkkeen lääkere-

kisteristä. Kun resepti on valmis, järjestelmä lisää sen lääkeyhteenvetoon ja potilaskertomukseen lääkehoito-otsikon alle.

Lääkeyhteenvetoon voidaan lisätä myös potilaan käsikauppalääkkeet tai luontaislääkitykset. Nämä, kuten myös erikoislääkitykset voidaan lisätä listaan erityisellä #-merkinnällä. Merkintää käytetään silloin kun lääkettä ei ole poimittavissa lääkerekisteristä. [Järvi ja Tuominen, 2006]



Kuva 2: Määrätyt lääkkeet

3.3. Osastohoidon lääkitys

Osastohoidossa on käytössä lääkekortti tai vanhempi lääkelista, joka on poistumassa käytöstä. Haastatellulla osastolääkärillä oli jo hoitoyksikössään käytössä lääkekortti, joten keskitymme tässä siihen.

Kun potilas tulee osastolle, hoitaja luo hänelle lääkekortin. Lääkitystä jatketaan entisellään, kunnes lääkäri tarkistaa kortin ja tekee siihen mahdolliset muutokset, mikä tapahtuu yleensä seuraavana arkipäivänä. Usein hoitaja kirjaa lääkärin tekemät muutokset. Lääkkeet voidaan lisätä kortille joko poimimalla ne yksitellen viimeisimmästä lääkekortista, määrätyistä lääkkeistä tai myös poimimalla uusi lääke lääkerekisteristä samaan tapaan kuin avohoidossa. Osastohoidon lääkitys kirjataan käsin potilaskertomukseen.

Jos potilaalla ei ole vielä lääkekorttia aiemmista hoitojaksoista, tietoja voi selvitellä avohoidon lääkeyhteenvedosta (Kuva 2). Selvittäminen on kuitenkin ongelmallista, sillä listasta ei aina selviä, mitkä lääkkeet ovat oikeasti käytössä. Tämän taustalla on ainakin se, että monet lääkärit eivät kirjaa lääkitykselle päättymispäivää tai eivät merkitse järjestelmään lääkitystä lopetetuksi. Usein suoraan avohoidosta tulevilla potilailla on mukanaan reseptejä ja esimerkiksi lääkepurkkeja tai dosetti, jonka pohjasta löytyy lääkkeiden tiedot.

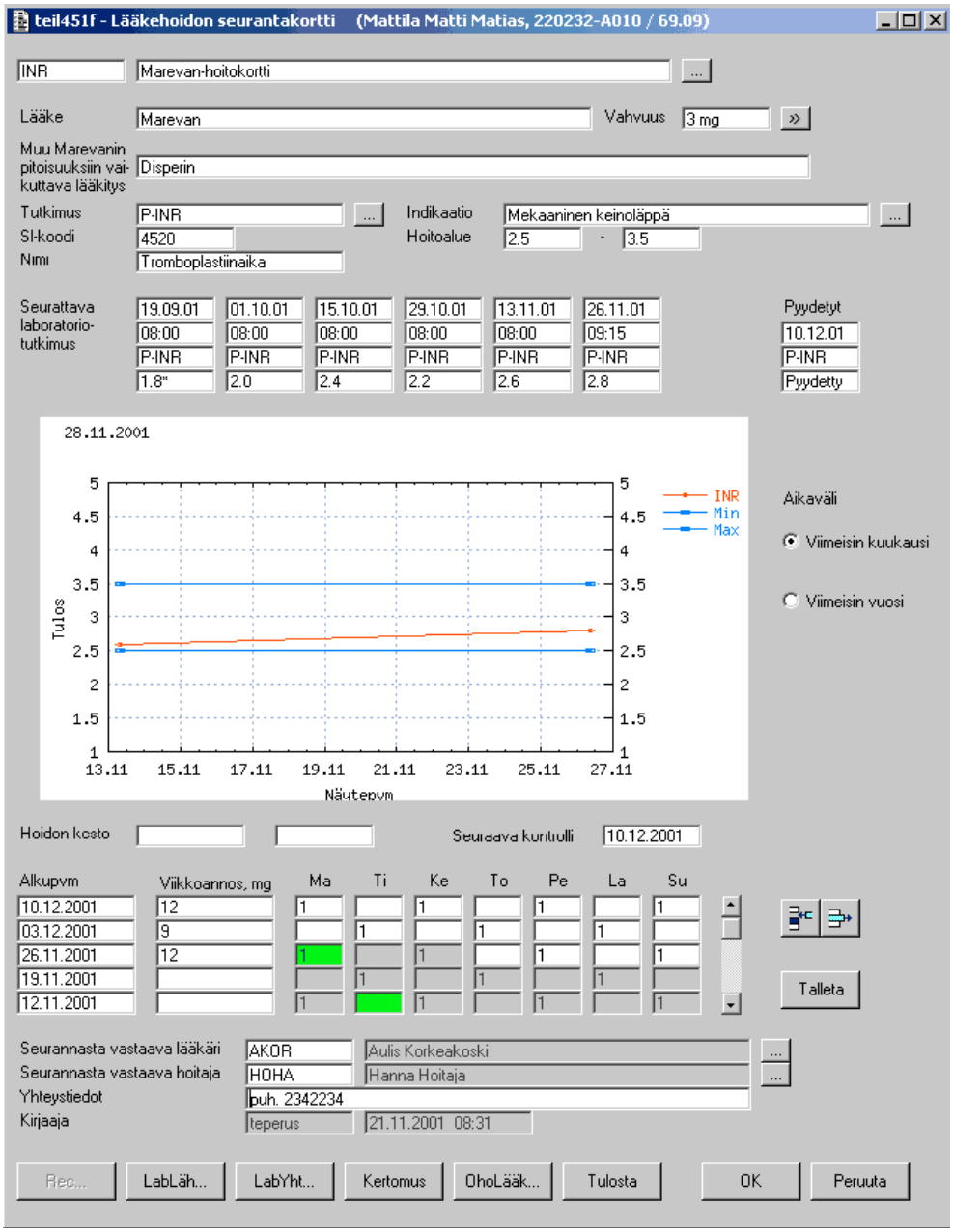
Lääkekortti on hoitojaksokohtainen, eikä sitä voi suoraan ottaa käyttöön uudelle hoitojaksolle. Lääkekortin saa kuitenkin nähtäville ja lääkkeet voi poimia uudelle kortille. Usein potilas siirtyy pitkäaikaissairaanhoidon vuodeosastolle kotihoidosta ja siellä käytetyn lääkekortin tiedot eivät välttämättä muutu merkittävästi, jolloin uuden lääkekortin luominen alusta asti teettää tarpeetonta työtä. Jos viimeisimmän lääkekortin voisi ”kuitata” käyttöön uudelle hoitojaksolle ja merkinnät siirtyisivät potilaskertomukseen suoraan, työvaiheet vähenisivät muun muassa siksi, että lääkäri motivoituisi tekemään koko työn kerralla ilman, että hoitaja kirjaa tiedot lääkekortille ja potilaskertomukseen. Nykyisin kirjaamistyö on hoitajien vastuulla lähinnä sen vuoksi, että se vie niin paljon aikaa. Kaikkiaan lääkityksen tai lääkekortin löytyminen tietojärjestelmästä koetaan hyödylliseksi ja paperille kirjaamiseen nähden nykyinen työtapa on puutteistaan huolimatta parempi.

Potilaan lähtiessä osastolta lääkekortin tiedot siirretään käsin takaisin resepteiksi. Turhaa työtä aiheutuu uloskirjaamisen yhteydessä siitä, että hoitajat poistavat työnsä kannalta tarpeettomia tietoja siirtäessään tietoja lääkekortille. Tämä tehdään siksi, että se on helpoin tapa tehdä lääkekortista helppolukuinen, mikä puolestaan tukee virheetöntä annosjakelua. Potilaan lähtiessä osastolta lääkäri joutuu palauttamaan poistetut tiedot siirtäessään kortin lääkkeet takaisin resepteiksi.

3.4. Muut lääkitystiedot

Sekä määrätyt lääkkeet (reseptit) että osastohoidon lääkitykset kirjataan osaksi potilaskertomusta. Haastatteluista ilmenee, että aina tiedot näiden välillä eivät täsmää. Syynä tällaiseen tilanteeseen voi olla se, että työ tehdään osaksi käsin automaation puuttuessa. Lääkekorttiin laitetut tiedot tai annostukseen tehdyt muutokset merkitään potilaskertomukseen käsin, ja sen tekeminen on lopulta hoitajan tai lääkärin muistamisen varassa.

Pegasoksen osastohoito-ohjelmassa on erityisiä *lääkehoidon seurantakortteja*, joilla voidaan seurata tietyn vaivan lääkitystä ja vaikutusta esimerkiksi veriarvoihin. Korttiin merkitään seurantaa koskeva lääkitys ja annostus. Lääkkeitä ei poimita suoraan määrätyistä lääkkeistä, vaan ne lisätään käsin. Turun terveystoimessa on otettu päivittäiskäyttöön veren hyytymistä ehkäisevän Marevan-lääkkeen seurantakortti (Kuva 3). Myös diabeteksen seurantakorttia on kokeiltu, muttei vielä otettu käyttöön.



Kuva 3: Lääkehoidon seurantakortti (Marevan-hoitokortti)

3.5. Lääkitystiedon sisältö ja rajoitukset

Potilastietojärjestelmän kautta on saatavilla kaikki ne lääkitystiedot, jotka on sinne järjestelmää käyttävissä Turun terveystoimen yksiköissä kirjattu. Järjestelmästä ei ole liitäntää esimerkiksi Tyksin (Turun yliopistollisen sairaalan) tai kolmannen sektorin järjestelmien lääkitystietoihin. Keskeisin rajoite lääkitystiedon käsittelyn kannalta onkin se, ettei potilastietojärjestelmään tukeutuen voi selvittää potilaalle muualla määrättyä lääkitystä. Turun terveystoimessa ei ole vielä käytössä aluetietojärjestelmää. Sellaiset tilanteet, joissa lääkitystietoja ei verrattain helposti saataisi selville, eivät olleet haastatellun osastohoidon henkilöstön mukaan kovin yleisiä, sillä potilaat ovat yleensä olleet terveystoimen asi-

akkaana jo jonkin aikaa ja tulevat joko kotihoidon piiristä tai Tyksistä. Potilaan tullessa osastolle suoraan Tyksistä tiedot tulevat mukana paperilla. Usein myös suoraan kotoa tulevan potilaan tiedot ovat saatavilla tai hän saattaa tuoda mukanaan esimerkiksi dosetin, jonka pohjassa lääkitys on merkittynä.

Lääkitystiedon jakautuneisuus järjestelmän sisällä tuottaa ylimääräistä vaivaa. Lääkitystä on tarpeen tullen selvitettävä paitsi määrätyistä lääkkeistä, myös osastohoidon lääkityksestä ja potilaskertomuksesta. Järven ja Tuomisen [2006] mukaan Pegasokseen ollaan tämän vuoden lopulla ottamassa käyttöön uusi lääkitysnäyttö, johon on keskitetty osastohoidon ja avohoidon lääkitys. Uudelta lääkitysnäytöltä voi myös kirjoittaa reseptejä. Uuden näytön tuomat mahdolliset parannukset tai muutokset nykyiseen eivät ehtineet mukaan vielä tähän tutkimukseen.

Lääkkeen tiedot poimitaan lääkerekisteristä, josta reseptille siirtyy lääke, valmistemuoto, vahvuus, määrä (pakkauskoko) ja valittu vakioindikaatio. Lääkkeiden poiminnan yhteyteen on integroitu interaktiotietokanta, joka varoittaa, mikäli lisättävällä lääkkeellä on tunnettuja haitallisia yhteisvaikutuksia potilaan voimassaolevien lääkkeiden kanssa.

3.6. Sähköisen potilaskertomuksen ydintiedot ja Pegasos

Tässä kohdassa selvitetään, missä määrin Pegasoksen lääkitystiedot nykyisin vastaavat kansallista suositusta potilaskertomuksen rakenteisista ydintiedoista. Vertailuun on käytetty ydintietojen, otsikoiden ja näkymien toteuttamisoppaan liitteen 4 kohtaa ”Lääkehoito” [STM, 2006b]. Pegasoksen lääkitystietojen selvittäminen perustuu Pegasos Tutor -ohjelman sisältämiin esittelyvideoihin, demoohjelmaan ja manuaaliin [WM-data, 2006b]. Vertailukohteeksi on Pegasoksesta otettu avohoidon määrätyt lääkkeet -näyttö sekä reseptinkirjoitus. Näiden pohjalta on päätelty ne tiedot, joita lääkkeestä määrätessä, uusiessa tai lopetettaessa tallentuu Pegasoksen tietokantaan. Lääkekorttia ei vertailla ydintietoihin, sillä kortti sisältää samat tiedot kuin määrätyt lääkkeet sillä erotuksella, että tietoja on karsittu ja lääkkeen annostus ilmoitetaan kuudella kellonajalla aamukahdeksasta iltakymmeneen, joista valitaan halutut ajankohdat ja annostus kullekin kierrolle – esimerkiksi illalla lääkettä voidaan näin merkitä annettavaksi enemmän kuin iltapäivällä.

Rakenteisten lääkehoidon ydintietojen vertailu suoraan Pegasoksen lääkitystietoihin osoittautui osittain riittämättömäksi tavaksi vertailuun. Vertailun selkeyttämiseksi ja täydentämiseksi on taulukkoon 2 tiivistetty vasempaan riviin lääkehoidon rakenteisten ydintietojen oppaan *kuvaukset* lääkehoidon osion ydintiedoista [STM, 2006b, 27–28]. Oikealle puolelle on taulukkoon laitettu kuvausta vastaava tieto Pegasoksessa. Taulukossa 3 on verrattu saman oppaan liitteessä olevia varsinaisia ydintietoja suoraan Pegasoksen tietoihin. Mikäli

vastaavaa tietoa ei ole tarjolla toisessa tietojoukossa, on solu jätetty tyhjäksi. Kahdesta vertailutavasta huolimatta vertailu on joiltain osin epätäydellinen, sillä käytettävissä olevan tutkimusmateriaalin pohjalta on hankala selvittää esimerkiksi, mitä standardeja on Pegasoksen toimintojen taustalla käytössä. Samanlainen tilanne on myös ydintietojen moninaisten päivämäärien kanssa – ei ole täysin selvää mihin Pegasoksen päivämäärään niitä tulisi tai voi verrata.

Lääkehoidon aloituksen ja lopetuksen syyn merkintään ydintieto-oppaassa suositellun MedDRA-luokituksen (Medical Dictionary for Regulatory Activities) käytöstä ei löytynyt tietoa. Pegasoksessa lääkettä poimittaessa lääkkeen aloitukselle voi valita joko vakioindikaation tai kirjoittaa muun indikaation. Se, ovatko vakioindikaatiot MedDRA-luokituksen mukaisia, jäi epäselväksi. Osastohoidon osiossa voidaan kirjata lopetuksen syy lääkettä poistettaessa, mutta vastaavaa paikkaa ei näytä löytyvän avohoidon puolella.

Lääkehoidon ydintietojen kuvaus [STM, 2006b, 27–28]	**Pegasos: Määrätyt lääkkeet**
Lääkevalmisteen kauppanimi	Lääke
Lääkemuoto	Valmistemuoto
Lääkkeen rooli: jatkuva, pysyvä, tarvittaessa	Pysyväislääkitys: kyllä/ei
Lääkkeen annostus	
Lääkkeen annostus (tekstimuoto)	Annos
Aloituspäivä	Päivämäärä
Vaikuttavan aineen ATC-koodi(t)	*Käytössä*
Lääkehoidon aloituksen syy (MedDRA)	*Ei tietoa*
Lääkehoidon lopetuksen syy (MedDRA)	Lopetuksen syy (osastohoito, vapaamuotoinen)
	Päättyen
	Lääkkeen määrääjä
	Lääkkeen vahvuus
	Määrä (pakkauskoko)
	Lukumäärä (pakkausten määrä)
	Iteraatio (uusintakerrat)

Taulukko 2: Lääkehoidon rakenteisten ydintietojen kuvauksen vastaavuus Pegasoksen lääkitystietoihin

Potilaskertomuksen ydintiedot: Lääkehoito (Lääkityslistan CDA R2-rakenne, v1.07)	**Pegasos: Määrätyt lääkkeet**
Lääkehoito **Lääkkeen nimi ja vahvuus**	
Lääkekoodi (ATC-luokitus)	ATC-luokitus käytössä
Lääkkeen rooli	Pysyväislääkitys: kyllä/ei
Lääkeannostus	Annos
Päiväys, lääkityksen aloittaminen tai muuttaminen	Päivämäärä
Vaikuttavan aineen ATC-koodi(t)	*Käytössä*
Lääkehoidon aloituksen tai muutoksen syy	Vakioindikaatio tai muu indikaatio
Lääkeindikaation nimi	*Ei tietoa*
Lääkeindikaation koodi (IDC-10 / ICPC / MedDRA)	*Ei tietoa*
Päiväys, lääkityksen lopetus	*Ei tietoa*
Lääkkeen vaihdettavuus	*Ei tietoa*
Lääkitysohjeistuksen antaminen	
Terveydenhuollon ammattihenkilö ja organisaatio	Lääkkeen määrääjä
Päiväys	*Ei tietoa*

Taulukko 3: Lääkehoidon ydintietojen vastaavuus Pegasoksen lääkitystietoihin

Ydintietoja ja Pegasosta verrattaessa huomataan joitakin eroja kumpaankin suuntaan. Pegasos ei vaikuttaisi noudattavan kaikkia ydintietoja ja toisaalta Pegasoksessa on sellaisia tietoja, joita ei ydintiedoissa ole. On tosin huomattava, että esimerkiksi pakkauksen tiedot saadaan poimittua ATC-koodin perusteella, vaikkei niitä ole ydintiedoissa erikseen mainittu. Ydintiedoista kuitenkin puuttuu esimerkiksi iteraatio (uusintakerrat), joka saattaa olla hyödyllinen tieto selvitettäessä sitä, kauanko samaa reseptiä on jo uusittu. Mikäli uusitulla reseptillä on selkeä suhde vanhaan reseptiin, tämän tiedon selvittäminen ei tarpeen tullen liene vaikeata. Huomionarvoinen on myös se seikka, ettei Pegasoksessa voi tällä hetkellä määritellä lääkkeen rooliksi ”tarvittaessa”. Haastatteluissa tätä ominaisuutta kaivattiin Pegasokseen.

Lääkehoidon ydintiedoista voidaan koota lääkityslista, jossa on tieto senhetkisestä lääkityksestä sekä lääkehistoria niiltä osin kuin se on hoitopäätösten kannalta olennaista. Lisäksi lääkityslistaan merkitään allergiat, haittavaikutuksia aiheuttaneet lääkkeet ja immunisaatiot. [STM, 2006b, 28]

4. Yhteenveto

Lääkitystiedon hallinnan kehitystyö on ajankohtainen alue paitsi Suomessa myös maailmalla. Ratkaisumallit ovat erilaisia, mutta lähtökohdat ja tavoitteet ovat perimmiltään samat: tiedonsaantia ja -kulkua parantamalla voidaan vä-

hentää tarpeetonta lääkitystä ja lääkitysvirheitä vaikuttaen lopulta myös lääkityksen kokonaiskustannuksiin yhteiskunnalle.

Vuoden 2007 alusta voimaan astuvat lait sähköisestä arkistoinnista ja sähköisestä reseptistä tulevat vauhdittamaan muutosta Suomessa. Lakiin kirjatut linjaukset toimintatavoista ja vastuunjaosta mahdollistavat valmiin kokonaisuuden luomisen. Tällä hetkellä suunnitelmat alkavat olla sähköisen reseptin ja arkiston osalta kypsässä vaiheessa ja eri klusterihankkeissa valmistellaan jo muun muassa ydintietojen käyttöönottoa potilastietojärjestelmiin.

Tarkastelluissa maissa on joiltain osin ehditty Suomea pidemmälle esimerkiksi sähköisen reseptin tuotantokäyttöön ottamisessa. Toisaalta missään ei ole kansallisen tason palveluita käytössä siinä laajuudessa kuin Suomessa on tällä hetkellä tavoitteena. Suomen lähestymistavassa on panostettu laaja-alaiseen ja perusteelliseen kokonaisuuden suunnitteluun, ja se vie aikaa.

Muiden maiden terveydenhuollon kansallisten arkkitehtuurien ja tietojärjestelmien kehitystyössä ratkotaan pitkälti samanlaisia ongelmia kuin Suomessa. Potilastietojärjestelmien ja standardien laaja kirjo on yksi kansallisen infrastruktuurin kompastuskivistä. Jo pelkästään lääkkeiden luokituksia voi olla useita, puhumattakaan eriävistä kirjaustavoista, rajapinnoista ja kommunikointiratkaisuista. Alan tutkimus on kansainvälistä samoin kuin standardit. Suomen käyttämä HL7-standardi on valittu käyttöön myös monessa muussa maassa.

Luultavasti jo lähivuosina tulee tapahtumaan merkittäviä muutoksia terveydenhuollon tietojen käsittelyssä. Kansalaiselle tämä voi parhaimmillaan näkyä oman lääkityksen selkiytymisenä tai vähentymisenä ja vähintäänkin siten, että resepteistä ei tarvitse enää kantaa huolta apteekkiin tai lääkäriin mennessä. Terveydenhuollon ammattilaisille lääkitystiedon parempi välittyminen tuo helpotusta arkirutiineihin ja varmuutta potilaan lääkityksen hallintaan.

Viiteluettelo

[Federal Ministry of Health, 2006] Federal Ministry of Health, *The Electronic Health Card*. Available as http://www.die-gesundheitskartte.de/download/dokumente/broschuere_elektronische_gesundheitskarte_engl.pdf. Checked 8.11.2006.

[HE, 2006a] Hallituksen esitys n:o 253 Eduskunnalle sosiaali- ja terveydenhuollon asiakastietojen sähköistä käsittelyä koskevaksi lainsäädännöksi. HE n:o **253/2006** vp.

[HE, 2006b] Hallituksen esitys n:o 250 Eduskunnalle laiksi sähköisestä lääkemääräyksestä sekä laiksi lääkelain 57 ja 57 a §:n muuttamisesta. HE n:o **250/2006** vp.

[Hyppönen, 2005] Hannele Hyppönen (toim.), *Sähköisen reseptin pilotoinnin arviointi vaihe I – Loppuraportti*, Osaavien keskusten verkoston julkaisuja 1/2005, Helsinki.

[Hyppönen et al., 2006] Hannele Hyppönen (toim.), Kirsi Hännikäinen, Marja Pajukoski, Pekka Ruotsalainen, Lauri Salmivalli, Emmi Tenhunen, *Sähköisen reseptin pilotin arviointi II (2005–2006).* Stakes Raportteja 11/2006, Helsinki.

[Järvi ja Tuominen, 2006] Jussi Järvi, Martti Tuominen, *Turun terveystoimi: Reseptin elinkaari.* Turun yliopisto, Informaatioteknologian laitos Laboris, 2006, Turku.

[STM, 2003] Sosiaali- ja terveysministeriö, *Sähköisten potilasasiakirjajärjestelmien valtakunnallinen määrittely ja toimeenpano.* Sosiaali- ja terveysministeriö työryhmämuistioita 2003:38, Helsinki.

[STM, 2006a] Sosiaali- ja terveysministeriö, Kansallinen terveyshanke – sähköiset potilasasiakirjat vuoteen 2007. PowerPoint-esitys. Saatavilla http://www.terveyshanke.fi. Haettu 9.11.2006.

[STM, 2006b] Sosiaali- ja terveysministeriö, *Opas – Ydintietojen, otsikoiden ja näkymien toteuttaminen sähköisessä potilaskertomuksessa.* Versio 1.1. Saatavilla http://www.terveyshanke.fi. Haettu 30.10.2006.

[STM, 2006c] Sosiaali- ja terveysministeriö, *Alueellisista ratkaisuista kansalliseen kokonaisuuteen.* Sosiaali- ja terveysministeriön selvityksiä 2006:8, Helsinki.

[STM, 2006d] Sosiaali- ja terveysministeriö, Klusterihankkeet. Saatavilla http://www.stm.fi/Resource.phx/vastt/tietoh/klusterit.htx. Haettu 9.11. 2006.

[Sundhed.dk, 2006] *The Danish eHealth portal*. Available as http://www.sundhed.dk/wps/alias?alias=in_english. Checked 9.11.2006.

[Sušelj, 2006] Marjan Sušelj, Recording of Medication History – Field experiences from pilot and national implementation in Slovenia. PowerPoint presentation. In: *The World of Health IT 2006 Conference & Exhibition*.

[Winblad et al., 2006] Ilkka Winblad, Jarmo Reponen, Päivi Hämäläinen, Maarit Kangas, *Informaatio- ja kommunikaatioteknologian käyttö Suomen terveydenhuollossa 2006 – Tilanne ja kehityksen suunta.* Stakes Raportteja 7/2006, Helsinki.

[WM-data, 2006a] WM-data Oy, Potilastietojärjestelmä Pegasos. Saatavilla http://www.wmdata.fi/wmwebb/Services/files/Pegasos_yleisesite2005.pdf. Haettu 30.10.2006.

[WM-data, 2006b] WM-data Oy, *Pegasos Tutor 7.3.* CD-ROM. WM-data, 2005.

Lähtökohtia käytettävyysnäkökulmaiseen lasten tietokonepelien äänimaailmojen suunnitteluun ja arviointiin

Pauliina Paarlahti

Tiivistelmä.
Tietokoneiden käyttö ja tietokonepelien pelaaminen on nykyisin tuttua yhä useammalle lapselle. Peliteollisuus tuottaa vuosittain lukuisan määrän tälle käyttäjäryhmälle suunnattuja tuotteita. Samalla tuotteet kehittyvät tarjoten entistä mukaansatempaavampia pelikokemuksia alati kehittyvine audiovisuaalisine maailmoineen.

Tietokoneen tarjoamissa pelikokemuksissa näkö- ja kuuloaisti ovat edelleen keskeisimmässä roolissa. Silti valtaosa annetusta suunnittelu- ja arviointiohjeista koskee vain visuaalisuutta. Äänellä on kuitenkin erityinen merkitys peleissä – myös lasten erityispiirteiden osalta. Äänet ovat paitsi osa pelituotteiden viihteellisyyttä, mutta myös osa niiden käytettävyyttä. Erilaiset äänet, kuten puhuttu teksti ja ikoneihin liitetyt äänet, ohjaavat ja auttavat lasta käyttämään tuotetta esimerkiksi silloin, kun häneltä vielä puuttuu taito lukea kirjoitettua tekstiä tai tulkita esitettyjä symboleja.

Tässä tutkielmassa tarkastellaan tietokonepelejä koskevia peliheuristiikkoja erityisesti äänen osalta ja esitetään ohjeisto, joka tarjoaa lähtökohtia lasten tietokonepelien äänimaailmojen suunnitteluun ja arviointiin. Tarkastelu tehdään tietokonepelien käytettävyyden näkökulmasta.

Avainsanat ja -sanonnat: lasten tietokonepelit, käytettävyys, peliheuristiikat, äänimaailma
CR-luokat: H.1.2, H.5.2

1. Johdanto

Käytettävyystutkimuksen alalla kiinnostus lasten vuorovaikutteisten teknologiatuotteiden suunnittelua ja arviointia kohtaan on voimistunut merkittävästi 1990-luvun jälkipuoliskolta asti (Höysniemi, 2005). Nykyisin lasten tietoteknologiatuotteita ja lapsia käyttäjäryhmänä koskevaa tutkimustietoa onkin saatavilla jo melko paljon (Chiasson & Gutwin, 2005; Höysniemi, 2005). Lisääntynyt tutkimustieto ja kiinnostus tuotteiden kehittämistä kohtaan ovat johtaneet aivan uudenlaisiin laatuvaatimuksiin lapsille suunnattujen tietoteknologiatuotteiden osalta. Laatuvaatimukset koskevat niin tuotteiden sisältöä, teknistä toteutusta kuin käytettävyyttäkin.

Tietokonepelit ovat yksi merkittävimmistä lasten käyttämistä tuoteryhmistä. Pelien suuresta suosiosta huolimatta akateeminen pelitutkimus laajeni lasten

teknologiatuotteita koskevan ja lapset käyttäjäryhmänä huomioivan käytettävyystutkimuksen tavoin vasta 1990-luvun lopulla ja erityisesti 2000-luvun alussa (Eskelinen, 2005). Nykyisin pelejä koskevaa tutkimusta tehdään jo melko laaja-alaisesti.

Pelien kehittämistä ja käytettävyystutkimusta yhdistää niiden tavoite tarjota käyttäjälle sitä, mitä nämä haluavat. Tästä huolimatta peli- ja käytettävyysalojen välinen vuorovaikutus on ollut viime vuosiin asti vähäistä. Vasta viime aikoina kiinnostus alojen väliseen yhteistyöhön on kasvanut. (Jørgensen, 2004).

Yksi varsin vähän tutkimusta osakseen saaneista pelien – ja ylipäätään käyttöliittymien – osa-alueista käytettävyyden näkökulmasta on pelien äänimaailma. Vaikka monet sovellukset luottavat edelleen pääosin visuaalisuuteen informaation välittämisessä, on äänillä peleissä yhä keskeisempi osa. Ääniä hyödynnetään peleissä eri rooleissa: toiminnasta kertovina efekteinä, puhuttuina dialogeina, musiikkina, taustahälynä, palautteena jne. Äänillä on keskeinen merkitys pelien tunnelman ja peleille ominaisen viihteellisyyden luonnissa. Kehittyvä tekniikka, kuten 3D-äänen mukaantulo, ovat tuoneet äänen pelien suunnittelijoiden ja kehittäjien kasvavan kiinnostuksen kohteeksi (Röber & Masuch, 2004). Äänen merkityksen tähänastista tutkimuksellista toisarvoisuutta käytettävyys- ja pelitutkimuksen alalla kuvaa esimerkiksi äänen huomioinnin satunnaisuus ja huomiointiin liittyvä täsmentymättömyys niin pelejä kuin myös osin käyttöliittymiä koskevissa heuristiikoissa ja suunnitteluohjeissa. Tutkimuksen painopiste on selvästi ollut visuaalisuudessa.

Tässä tutkielmassa keskitytään tietokonepelien äänenkäytön osa-alueeseen käytettävyyden näkökulmasta. Tarkoituksena on selvittää lähinnä kirjallisuuskartoituksen perusteella sitä, miten äänet on huomioitu pelisuunnittelua ja -arviointia koskevissa heuristiikoissa ja ohjeissa. Erityishuomion kohteena ovat pienille, alle kouluikäisille lapsille suunnatut tietokonepelit. Tutkielman lopussa esitetään kartoituksen perusteella muodostettu ohjeisto, joka tarjoaa lähtökohdan lasten tietokonepelien äänimaailmojen suunnitteluun ja arviointiin.

Tutkielman alussa tehdään lyhyt katsaus lasten tietokoneiden käytön tilanteeseen ja lapsiin tietokoneiden ja tietokonepelien käyttäjinä. Tämän jälkeen tarkastellaan tietokonepelien äänimaisemia ennen siirtymistä peliheuristiikkojen kartoittamiseen ja ääninäkökulmasta tehtyyn tarkasteluun sekä kartoituksen perusteella laaditun ohjeiston esittelyyn.

2. Lapset tietokoneiden ja tietokonepelien käyttäjäryhmänä

2.1. Lasten tietokoneiden ja tietokonepelien käytöstä

Tietokoneiden ja tietokonepelien käyttö on nykyisin monille lapsille tuttua hyvin varhaisesta iästä asti. Yleinen tietoteknologiatuotteiden määrän kasvu ympäröivässä yhteiskunnassa on tuonut nämä tuotteet kiihtyvällä vauhdilla myös lasten toimintaympäristöihin. Tilannetta kuvaa esimerkiksi se, että nykyisin suurimmalla osalla suomalaislapsista on kotona käytössään tietokone tai pelikonsoli (Latva, 2004). Huomionarvoinen asia on myös se, että tietokonepelejä suunnitellaan eri-ikäisille lapsikäyttäjäryhmille aina puolitoista vuotiaista lapsista alkaen. Tietokoneet erilaisine ohjelmineen ja oheislaitteineen ovat myös osa melkein jokaisen päiväkodin ja koulun käytössä olevaa välinevalikoimaa. Tietoteknologian käytön voidaankin sanoa olevan osa lasten päivittäistä arkea – leikkimistä, oppimista ja vapaa-ajanviettoa.

Valtaosa lasten tietokoneen käytöstä on erilaisten tietokonepelien pelaamista. Merkittäväksi taloudelliseksi toimialaksi muodostunut peliteollisuus myykin pelimarkkinoilla erilaisia pelituotteita miljardeilla euroilla vuosittain (Eskelinen, 2005). Tarjolla on laaja kirjo erilaisia pelityyppejä seikkailu- ja taistelupeleistä opetuspeleihin, joiden parissa lapset viihtyvät jopa useita tunteja päivittäin.

Tietokoneiden ja tietokonepelien käytön käyttäjäkunnan kasvu laajan lapsikäyttäjäryhmän kattavaksi on edellyttänyt tekniikan kehittymistä. Erityisesti graafisten käyttöliittymien ja äänikorttien mukaantulo ja kehitys ovat olleet merkittävässä osassa. Graafiset käyttöliittymät vapauttivat käyttäjät komentokielistä ja mahdollistivat käytön myös luku- ja kirjoitustaidottomille (Räty, 1999). Äänikortit puolestaan mahdollistivat esimerkiksi palautteen ja ohjeiden antamisen äänen avulla aiemman visuaalisuuden sijaan. Tekniikan kehityksen rinnalla erityisesti lapsille suunnattujen pelituotteiden markkinoille tulo 1980-luvun alussa (Räty, 1999) oli luonnollisesti voimakas sysäys lasten pelaamisen yleistymiselle. 1990-luvulla CD-ROM ja sen myötä pelien edullisempi levittäminen ja PC-yhteensopivien tietokoneiden suorituskyvyn nopea kehitys tukivat trendiä (Kasvi, 2001). Lasten mahdollisuudet toimia tietokoneiden kanssa lisääntyivät merkittävällä tavalla 1990-luvulla mikrotietokoneiden yleistyessä kotitalouksissa. Esimerkiksi Suomessa mikrotietokoneen omaavien kotitalouksien määrä kasvoi vuoden 1990 kahdeksasta prosentista vuoteen 1998 mennessä 30 %:in (Tilastokeskus, 2006). 2000-luvulla luku on noussut entisestään.

2.2. Lapset tietoteknologiatuotteiden käyttäjäryhmänä

Käyttäjinä lapset muodostavat hyvin heterogeenisen käyttäjäryhmän – vaihteleehan lasten osaamis- ja kehitystaso paljon niin yksilöiden kuin ikäryhmien

välillä. Lasten kognitiiviset, fyysiset, emotionaaliset, sosiaaliset ja kielelliset ominaisuudet ja taidot kehittyvät ja muuttuvat koko ajan (Höysniemi, 2006). Vuoden ikäero merkitsee lasten kehityksessä ja taidoissa huomattavaa eroa. Yhtälailla yksilöiden väliset kehityserot voivat olla suuria tullen esiin esimerkiksi eroissa pituuskasvussa, luku- ja kirjoitustaidon oppimisessa ja kielellisissä valmiuksissa (esim. sanavaraston laajuus).

Lasten teknologiatuotteiden suunnittelussa ja kehityksessä lasten erityispiirteet jätettiin pitkään huomioimatta ja suunnittelussa hyödynnettiin samoja menetelmiä ja sääntöjä kuin aikuisten tuotteissa. Kuten Druinkin (2002) huomauttaa, lapset eivät kuitenkaan ole pieniä aikuisia. Esimerkiksi jo tietokoneen keskeisimpien oheislaitteiden kuten näppäimistön- ja hiirenkäyttö eroavat aikuisten vastaavasta mm. lasten pienempien raajojen ja kehittymättömämmän motoriikan asettamien rajoitteiden vuoksi (Hietala & Ovaska, 2002). Samoin lasten mielenkiinnon kohteet ja motivaation lähteet ovat erilaisia kuin aikuisilla (Chiasson & Gutwin, 2005). Äänet, kuvat, liikkeet, monipuolinen multimedian käyttö, hauskuus ja värikkyys ovat esimerkkejä lasten arvostamista tekijöistä tuotteissa (Hanna et al., 1999; Hietala & Ovaska, 2002; Nielsen, 2002).

Nykyisin lasten erityispiirteet pyritään huomioimaan käyttöliittymissä yhä paremmin. Tuttua on esimerkiksi tekstipainikkeiden korvaaminen toimintoa kuvaavin kuvin varustetuin ikonein, pyrkimys lasten tuntemien metaforien käyttöön aikuistenmaailmassa käytettyjen metaforien sijaan sekä animaatioiden käyttö havainnollistajana toiminnan opastamisessa. Lukutaidon puutteen korvaavat puhutut ohjeet ovat myös melko yleisiä lapsille tarkoitetuissa tietokonesovelluksissa, erityisesti peleissä. Erittäin tärkeää on lapsille soveltuvien tuotteiden testaaminen oikealla käyttäjäryhmällä: aikuisten on välillä vaikeaa hahmottaa sitä maailmaa, jossa lapset tietojensa ja taitojensa kanssa elävät. Puuttuvat aiemmat kokemukset, vaihtuvat kulttuuriset metaforat, teknistyvät elinympäristöt ja lasten muuttuva suhde tietotekniikkaan ovat kaikki esimerkkejä tekijöistä, jotka tulee huomioida suunnittelu-, toteutus- ja arviointityössä.

Tietokonepelien osalta pyrkimys kunkin ikäryhmän yleisten tietojen ja taitojen huomioonottamiseen voidaan katsoa ilmeiseksi, ovathan peliteollisuuden tuottamien pelien kohderyhmät paikoin hyvinkin tarkkaan harkittuja. Tämä käy ilmi esimerkiksi useiden opetuspelien koteloista, joiden tekstit ilmaisevat selkeästi ikäryhmän, jolle peli on tarkoitettu. Kohderyhmä vaikuttaa mm. pelin grafiikkaan, tehtävätyyppeihin sekä pelin vaikeustasoon mukaan lukien niin tekninen vaikeus (esimerkiksi hiirenkäytön tai näppäimistökomentojen antamisen vaatimustaso) kuin tietotasoinen (esimerkiksi kohteiden tunnistaminen, metaforien hallinta, juonen ymmärtäminen) vaikeus.

3. Tietokonepelien äänimaailmoja

Ääni on saanut osakseen vain vähän huomiota ihmisen ja koneen välisenä vuorovaikutuselementtinä (Lee et al., 2000), vaikka äänen rooli ihmisten välisessä vuorovaikutuksessa on huomattava. Huomio on hämmentävä, sillä kuten esimerkiksi Nielsen (1993) on todennut, ihmisen ja koneen välisessä vuorovaikutuksessa tulisi pyrkiä mahdollisimman luonnolliseen dialogiin. Ihmisten välisessä vuorovaikutuksessa ääni on mukana varhaisesta lapsuudesta asti: ääni on mukana välittämässä informaatiota ihmisten tarpeista, tunteista ja mielialasta niin sanattoman kuin sanallisenkin viestinnän yhteydessä. Huomionarvoista vuorovaikutuksessa on myös esimerkiksi se, että äänen avulla myös sanallinen viestintä on mahdollista ilman luku- ja kirjoitustaitoa.

Ihmiset saavat tietoa ympäristöstään aistiensa kautta. Aistit – näkö-, kuulo-, tunto-, haju-, maku-, liike- ja tuntoaisti – välittävät omanlaisia signaalejaan aivojen tulkittavaksi. Kokonaiskuva ympäristöstä muodostuu aivoissa näistä viesteistä muodostettujen havaintokokemusten yhteisvaikutuksena. (Sinkkonen ym., 2002) Tietokonepelejä pelattaessa näkö ja kuulo ovat useimmiten aktiivisimmat aistit. Perinteisesti näistä näköaisti on saanut dominoivan osan visuaalisuuden painotuksen myötä äänten jäätyä marginaaliseen rooliin (Ekman et al., 2005). Kuulo- ja näköaisti tuottavat toisistaan erillistä informaatiota, joka kuitenkin on usein valjastettu palvelemaan saman kohteen havainnointia. Tämä tarkoittaa yleensä sitä, että kuuloaistilla pyritään ohjaamaan tarkkaavaisuutta. Nykypeleissä pääpaino onkin edelleen visuaalisuudessa, ja tyypillinen tilanne on se, että äänten lähteet valitaan vuorovaikutuksessa hiiren kanssa ja parametrit visuaalisen käyttöliittymän kautta. (Röber & Masush, 2004) Kuten Ekman ja muut (2005) esittävät, on tilanne usein se, että monet pelit ovat täysin pelattavissa ilman ääntä.

Nykyaikaisissa tietokonepeleissä käytetään 3D-audioteknologiaa ja pelit ovat vähitellen lähestymässä tyyliltään ja laadultaan elokuvien audiovisuaalisia ominaisuuksia. Aiemmin pelien äänet rajoittuivat usein erilaisiin piippauksiin, mutta tekniikan kehityksen myötä monimutkaisempi ja edistyneempi äänentoisto ovat tulleet mahdollisiksi. Nykyisin kolmiulotteinen äänimaisema on keskeinen tekijä monissa peleissä ja samalla jotakin, jota pelaajatkin pitävät standardinomaisena asiana. (Menshikov, 2003)

Tietokonepeleissä äänet esiintyvät eri rooleissa ja muodoissa. Jensen (2001 Manninen, 2004 mukaan) luokittelee äänen neljään tyyppiin: puheeseen, musiikkiin, ääniefekteihin ja hiljaisuuteen. Äänet voivat olla diegeettisiä eli pelimaailman sisäisiä (tarinaan liittyviä, visuaalisesti nähtäviin tapahtumiin liittyviä) tai ei-diegeettisiä eli pelimaailman ulkopuolisia (esim. taustamusiikki, musiikkiääniraita) (Jensen, 2001 Manninen, 2004 mukaan; Järvinen, 2002). Pelien

äänet kertovat pelien objekteista, toiminnasta, tilanteissa tapahtuvista muutoksista, hahmojen välisistä dialogeista, tunnelmasta jne. (Röber & Masuch, 2004; Järvinen, 2002) Ne välittävät pelaajalle tietoa pelin spatiaalisista ja ajallisista ulottuvuuksista (Jensen, 2001 Manninen, 2004 mukaan): Mistä lähteestä äänet tulevat? Mihin samanaikaisiin visuaalisiin tapahtumiin äänet liittyvät? Järvinen (2002) lisää äänen tehtäviin roolin pelin teemaan sopivan pelitilan tunnun luomisessa. Äänillä voidaan vahvistaa haluttua tilavaikutelmaa esimerkiksi tuomalla pelin äänimaailmaan pelin simuloimalle ympäristölle ominaisia ääniä (esim. kaupunki). Myös palaute käyttäjän käskyistä ja valinnoista annetaan usein äänen muodossa. Käytettävät äänet voivat olla luonnollisia, reaalimaailmaa ja sen kohteita representoivia tai ne voivat olla fiktiivisiä.

Äänten roolit ovat suurelta osin samoja niin aikuisille kuin lapsille suunnatuissa tietokonepeleissä. Tietenkään pelien äänimaailmat eivät ole tyyleiltään samanlaisia ja myös äänen funktioiden painotuksessa voidaan nähdä eroja. Pienille lapsille suunnatuissa peleissä äänen voidaan olettaa olevan merkitsevämmässä roolissa pelin konkreettisessa oppimisessa ja pelin toiminnan ymmärtämisessä, aikuisille tarkoitetuissa peleissä enemmän pelikokemuksen viihteellisyydessä ja nautinnollisuudessa. Yhtäkaikki, oikeanlaisen äänimaailman luominen (so. esim. tunnistettavat, ymmärrettävät, mielenkiintoiset ja odotuksia vastaavat äänet) on hyvin tärkeää onnistuneen pelikokemuksen luomisessa.

4. Pelien käytettävyydestä ja peliheuristiikoista

4.1. Käytettävyys tietokonepelien kontekstissa

Käytettävyydellä käsitteenä on useita määritelmiä, joista tunnetuimpia ovat Nielsenin (1993) ja standardi ISO 9241-11 (ISO, 1998) määritelmät. Nielsenin määritelmässä käytettävyys määrittyy viiden osatekijän, opittavuuden, tehokkuuden, muistettavuuden, virheettömyyden ja miellyttävyyden kautta. ISO 9241-11 (ISO, 1998) standardi puolestaan määrittelee käytettävyyden tuloksellisuuden, tehokkuuden ja käyttäjän tyytyväisyyden kautta: kuinka hyvin tietty käyttäjä voi saavuttaa tavoitteensa tuotteen avulla tietyssä kontekstissa em. attribuuttien osalta.

Pelit eroavat hyötysovelluksista, joiden käytettävyyteen edellä esitetyt käytettävyyden määritelmät ensisijaisesti sopivat, joiltakin keskeisiltä ominaisuuksiltaan. Perinteisten arviointikriteerien ja käytettävyysheuristiikkojen soveltaminen suoraan tietokonepeleihin ei näin ollen ole mielekästä. Tietokonepelien pelaamisen tavoitteena on yleensä hauskanpito ja nautittava sekä sopivan haasteellinen pelikokemus (Lazzaro & Keeker, 2004; Korhonen & Koivisto, 2006). Myös pelin oppiminen, ongelmien ratkaiseminen, uusien asioiden löytäminen

ovat osa tätä kokemusta (Korhonen & Koivisto, 2006). Pagulayan ja muut (2003) lisäävät hyötysovellusten ja pelien keskeisiin eroihin vielä erot tavoitteiden asettamisen lähtökohdassa, käyttöliittymälle ja sen toiminnoille asetetuissa yhdenmukaisuus- ja toimintavaatimuksissa, haasteellisuudessa, äänen ja grafiikan käytössä sekä innovaatiomyönteisyydessä.

Toki peleillä ja hyötysovelluksilla on myös yhteisiä tekijöitä käytettävyyden kannalta. Jørgensen (2004) esittää tällaisiksi mm. oppimisen, motivaation, mentaaliset mallit, kontrollin, vuorovaikutuksen, palautteet, spatiaalisen navigoinnin sekä kielelliset ja visuaaliset ilmaukset.

Pelien arvioinnissa on alettu puhua arvioinnin yhteydessä käytettävyyden ohella tai rinnalla pelien pelattavuudesta. Pelattavuus on toistaiseksi määritelmältään melko jäsentymätön käsite, mutta joitakin akateemisia määritelmiä on tehty. Esimerkiksi Järvinen ja muut (2002) ovat määritelleet pelattavuuden laadulliseksi suunnittelun ja arvioinnin käyttötarkoituksiin soveltuvaksi käsitteeksi. Heidän mukaansa pelattavuudella viitataan yhtäältä välttämättömien elementtien toteutukseen tietynlaisen pelinomaisuuden aikaansaamisessa, toisaalta taas käytettävyyden kaltaiseen arviointityökaluun, jolla voidaan arvioida tuotteen pelinomaisuutta ja vuorovaikutusta. Hanski ja Kankainen (2004) ovat hyödyntäneet Idean Researchin määritelmää, jossa pelattavuus määrittyy viiden osatekijän, käytettävyyden, vuorovaikutuksen, teknologian, tarinan ja kontekstin summana.

Kuten edellisille määritelmille, myös useimmille muille pelattavuuden määritelmille, on yhteistä se, että käytettävyys huomioidaan yhtenä pelattavuuteen vaikuttavana tekijänä muiden pelinomaisuuden tuottavien tekijöiden joukossa. Hyvä käytettävyys ei yksin riitä tekemään pelistä hyvää. Sama pätee toki myös toisinpäin: käytettävyydeltään huono peli on tuskin hyvä pelin haasteellisuuden noustessa käyttöliittymästä pelitilanteiden sijaan.

4.2. Yleisiä peliheuristiikkoja ja arviointiohjeita

Pelien arvioinnin eroavaisuudet hyötysovellusten arvioinnin vastaaviin käyvät ehkä parhaiten ilmi olemassa olevia peliheuristiikkoja tarkastelemalla. Heuristiikat huomioivat peleistä eri osa-alueita ja korostavat siten pelien arvioinnin erityisyyttä. Samoin kuin kaikki perinteiset käytettävyysheuristiikat ovat osin päällekkäisiä eivätkä suoraan kaikkiin hyötysovelluksiin sovellettaessa, myös peliheuristiikat eivät ole kaikkiin peleihin sovellettavissa. Onkin mahdollista, että jokaiselle pelityypille saatettaisiin tarvita omat heuristiikkansa, jotka voisivat toimia yleisten, usein melko pinnallisiksi ja yleistasoisiksi jäävien peliheuristiikkojen tukena.

Peliheuristiikkoja ovat tähän mennessä laatineet peliteollisuuden kanssa tekemisissä olevat yksittäiset henkilöt, ryhmät ja ammattilaiset sekä HCI-alan

tutkijat (Desurvire et al., 2004). Erilaisia heuristiikkoja on löydettävissä alan kirjallisuudesta useita. Esimerkkejä peliheuristiikoista ovat Malonen (1982) varhaisimpiin peliheuristiikkoihin lukeutuvat opetuspelien arviointiin tarkoitetut heuristiikat, Federoffin (2002) kirjallisuuden ja tapaustutkimuksen pohjalta kokoamat heuristiikat, Desurviren ja muiden (2004) suunnitteluvaiheen HEP-peliheuristiikat (*Heuristic Evaluation for Playability*), Järvisen ja muiden (2002) pelattavuusheuristiikat sekä Sweetserin ja Wyethin (2005) flow-tilaan perustuva pelin nautinnollisuuden arviointiin tarkoitettu kriteeristö. Eräs maininnan arvoisista peliheuristiikkojen kokoamisyrityksistä on myös Falsteinin ja Barwoodin (2006) pyrkimys koota 400 peliheuristiikka. Tällä mennessä he ovat listanneet noin 110 heuristiikka.

4.3. Lasten tietokonepelejä koskevia peliheuristiikkoja ja arviointiohjeita

Yleisten peliheuristiikkojen lisäksi tarve lasten tietokonepelien suunnittelua ja arviointia koskeviin heuristiikkoihin on tunnustettu. Esimerkiksi Barendregt ja Bekker (2004) peräänkuuluttavat yksityiskohtaisia suunnitteluohjeita pienten lasten tietokonepeleille, jotka mahdollistavat suunnittelijoiden luoda sekä helppoja että hauskoja tuotteita. He ovatkin esittäneet joitakin pelien osa-alueita (esim. pelin aloittaminen ja lopettaminen, kursorin muoto ja toiminnalliset kohteet, oikean modaliteetin käyttöön opastaminen sekä oikean perspektiivin hahmottaminen), joiden kohdalla erityisistä kohderyhmän erityispiirteet huomioivista heuristiikoista olisi suunnittelijoille huomattavaa apua pelien suunnittelussa. (Barendregt & Bekker 2004) He eivät kuitenkaan itse esitä varsinaisia heuristiikkoja, vaan antavat esimerkkejä lasten kohtaamista ongelmista pelien käyttöliittymissä.

Baauw ja muut (2005) esittävät, että lasten tietokonepelejä koskevat arviointiheuristiikat puuttuvat vieläkin. He arvioivat olemassa olevien heuristiikkojen koskevan ennemmin tuotteiden suunnittelua kuin arviointia. Baauw ja muut laativat varsinaisten heuristiikkojen sijaan Structured Expert Evaluation Method –nimisen (SEEM) metodin, jonka avulla asiantuntijat voivat arvioida lasten tietokonepelejä. Metodissa hyödynnetyt kysymykset on esitetty taulukossa 1.

Jokaisen näytön arviointia koskevat kysymykset:
Ymmärtävätkö lapset pelin tavoitteen?
Tietävätkö lapset, mitä tavoite voidaan saavuttaa?
Suoriutuvatko lapset helposti pelin vaatimista fyysisistä toiminnoista?
Saavatko lapset palautetta? Huomioidaanko sekä oikeat että väärät vastaukset ja voivatko lapset keskeyttää palautteen?
Ymmärtävätkö lapset saamansa palautteen? Koskeeko tämä sekä visuaalisesti että äänen avulla annettua palautetta ja niin oikeista kuin vääristä toimista annettua palautetta?
Jatkavatko lapset toimintaa tavoitteen saavuttamiseen asti? Pitävätkö lapset alipeleistä ja onko vaikeustaso pienille lapsille sopiva?
Ovatko navigointimahdollisuudet ja poistumistiet pelistä ja alipeleistä selkeät?
Onko pelin käyttöliittymässä muita ongelmia aiheuttavia kohteita?
Globaalit kysymykset:
Onko pelin haaste kohderyhmälleen sopiva? Herätetäänkö lasten mielenkiinto? Noudattavatko juoni ja käyttöliittymä lasten mielikuvitusmaailmaa/fantasioita?
Käykö alipelin vapaaehtoisuus/pakollisuus selvästi ilmi? Vastaako pelin 'flow' eli optimaalinen pelikokemus odotuksia? Onko pelin juoni looginen? Ilmaistaanko lapsen aktiivisen ja passiivisen roolin vaihtelu pelissä selvästi?

Taulukko 1. Lasten tietokonepelien arviointiin liittyvät kysymykset SEEM-metodin mukaan (Baauw et al., 2005).

Myös Barendregt ja muut (2006) ovat tutkineet lasten tietokonepelien käytettävyyttä. He tutkivat lasten tietokonepelien käytettävyyttä käytettävyystestien kautta ja käyttivät ongelmien jaotteluperusteena lasten tietokonepeleissä kohtaamien käytettävyysongelmien luokittelua. Luokittelun ryhmät ovat: tietämykseen liittyvät ongelmat, ajatteluun liittyvät ongelmat, muistiin liittyvät ongelmat, päätöksentekoon liittyvät, tapoihin liittyvät ongelmat, tekemiseen tai tekemättä jättämiseen liittyvät ongelmat, tunnistamiseen liittyvät ongelmat ja sensomotoriset ongelmat. Hekään eivät esittäneet varsinaisia käytettävyysheuristiikkoja pelien arvioimiseen.

Lasten tietokonepelejä koskevien heuristiikkojen täsmentymättömyydestä johtuen tässä tutkimuksessa hyödynnetään ns. yleisiä peliheuristiikkoja, joista keskeisimmiksi nostetaan Federoffin (2002) ja Desurviren ja muiden (2004) heuristiikkalistat, jotka kokoavat lukuisan määrän eri lähteissä esitettyjä heuristiikkoja. Näiden ohella otetaan huomioon mahdollisuuksien mukaan sellaisia lasten peleihin sovellettavissa olevia heuristiikkoja, joita em. heuristiikkalistat eivät huomioi.

5. Tietokonepelien käytettävyys ja peliheuristiikat äänenkäytön näkökulmasta

5.1. Äänestä käytettävyyden elementtinä

Pelien äänien useista rooleista huolimatta äänien keskeisimpiä tehtäväksi nähdään yleensä pelikokemukseen liittyvän hauskuuden ja mielihyvän lisääminen (Gal et al., 2002). Pelien musiikki, äänitehosteet ja hahmojen puheäänet ovatkin usein peleistä mieleen jääviä asioita. On äänillä peleissä toki muitakin funktioita, joista käytettävyyden tukeminen ei ole vähäisimmässä roolissa. Sinänsä ei ole yllättävää, että vaikka äänen merkitys pelikokemuksen luomisessa on tunnustettu, ei äänen käytettävyysomaisuuksia ole silti juurikaan tutkittu. Sinkkosen ja muiden (2002) mukaan tämä pätee laajemminkin äänen käytettävyysominaisuuksien tutkimiseen ja tutkimustietojen tuntemiseen suunnittelijoiden keskuudessa.

Miten äänellä siten voidaan vaikuttaa käytettävyyteen? Monissa yhteyksissä on totta, että äänestä ei ole – ainakaan helposti – syrjäyttämään tekstiä tai kuvaa, vaan ääni ja kuva toimivat toisiaan täydentävästi. Ääni on usein parhaimmillaan signaaleina, muistuttamassa tai hälyttämässä asioista erimerkiksi silloin, kun käyttäjän tarkkaavaisuus on muualla (Sinkkonen ym., 2002). Ääni on myös hyvä keino välittää palautetta tehdyistä valinnoista ja toimintojen suorittamisesta – kertoa vuorovaikutuksesta pelin kanssa. Ääni voi olla mukana myös mm. ohjeiden ja käytönaikaisten neuvojen ja vihjeiden antamisessa sekä pelin tilasta ja pelaajan sijainnista pelissä kertomisessa.

Pelit ovat osoittaneet, että käyttäjän tarkkaavaisuutta koskevien vaateiden vähentäminen tukee primaarisen tehtävän suorittamista. Ääni onkin tehokas tapa välittää informaatiota käyttäjälle esimerkiksi äänitiedotteiden ja erityisten ympäristöllisten äänten kautta lisäämättä visuaalista taakkaa tai rikkomatta peleissä tavoiteltavaa flow-tilaa (Dyck et al., 2003). Flow-tilalla viitataan alun perin Csikszentmihalyin määrittämään optimaaliseen pelikokemuksen käsitteeseen (ks. esim. Csikszentmihalyi, 1991). Myöhempiä pelin flow-tilaan liittyviä tulkintoja ovat esittäneet esimerkiksi Sweetser ja Wyeth (2005).

5.2. Yleisohjeita äänenkäyttöön

Sinkkonen ja muut (2002) ovat koonneet joitakin sovelluksissa tapahtuvaa äänen hyödyntämistä koskevia käytettävyysohjeita. Vaikka nämä ohjeet eivät ole kohdennettuja nimenomaan tietokonepeleihin ja niistä osan alkuperäinen kohdealue ovat äänivalikot, voidaan ohjeita soveltaa luonnollisesti peleihin sekä laajentaa ohjeistoa koskemaan laajemminkin sovelluksissa tapahtuvaa äänenkäyttöä. Ohjeet on esitetty taulukossa 2.

Esitä aina ensin ehto, sitten vasta sen edellyttämä toiminta.
Kerro aina valinnan jälkeen käyttäjän sijainti ennen uuden vaihtoehtosarjan antamista.
Anna yhdessä kehotteessa enintään neljä vaihtoehtoa.
Huolehdi käyttäjän kuuleman tekstin selkeydestä.
Anna vaihtoehtosarjojen lopuksi mahdollisuus alkuun palaamiseen.
Huolehdi tekstin eloisuudesta ja ymmärrettävyydestä myös puhuttuna esitettynä.
Älä käytä pitkiä lauseita.
Käytä sekä ääni- että visuaalisia vihjeitä, jos on odotettavissa, että käyttäjän mielenkiinto on täysin muualla.
Huolehdi siitä, että käyttäjän ei tarvitse koskaan samanaikaisesti lukea yhtä ja kuunnella toista viestiä.

Taulukko 2. Äänenkäyttöön liittyviä käytettävyysohjeita (Sinkkonen ym., 2002).

5.3. Äänen huomiointi peliheuristiikoissa

Peliheuristiikoissa äänen huomioiminen on toistaiseksi ollut melko vähäistä. Joidenkin heuristiikkojen yhteyteen on lisätty maininta heuristiikassa esitetyn asian toteuttamisesta ääntä hyödyntämällä, mutta yleisesti ottaen äänen suora huomiointi on vähäistä. Taulukkoon 3 on koottu peliheuristiikoista esimerkkejä, joissa äänen osallisuus mainitaan suoraan tai sen mukanaolo on suoraan tulkittavissa. Kuten huomataan, pääpaino on äänen avulla tapahtuvassa palautteen annossa ja pelitunnelman- ja innostuksenluonnissa. Myös heuristiikkojen yleisluonteisuus käy ilmi.

Hyödynnä ääntä merkityksellisen palautteen antamisessa.	Federoff, 2002
Taiteen tulee yhdistyä tehtäväänsä ja olla tunnistettavaa.	Federoff, 2002; Desurvire et al., 2004
Herätä pelaajan kiinnostus kuva- ja ääniefekteillä.	Federoff, 2002
Pelin pitää reagoida pelaajan toimintoihin mielenkiintoisella, mutta aina samalla tyylillä, esim. sopiva musiikki liittyen aina tiettyyn tilanteeseen.	Desurvire et al., 2004
Kytke pelin äänet johonkin toimintoon tai pelaajan tunteeseen.	Desurvire et al., 2004

Taulukko 3. Esimerkkejä äänen huomioivista heuristiikoista.

Todellisuudessa ääni on mukana useamman heuristiikan kohdealueella. Kuten aiemmin todettiin, on ääni usein muita kanavia tukevassa roolissa. Tietokonepelien kohdalla tämä tarkoittaa yleensä yhteistoimintaa visuaalisen kanavan kanssa. Näin tulkittuna ääni tulee huomioiduksi huomattavasti useamman heuristiikan kohdalla, mikä kuvastaa paremmin äänen roolia ja mahdollisuuksia

tietokonepelien käytettävyyden tukemisessa. Taulukkoon 4 on koottu esimerkkejä peliheuristiikoista, joiden huomioinnissa suunnittelussa ja arvioinnissa äänen roolia ei nykyaikaisissa peleissä voida useinkaan ohittaa. Heuristiikkojen yhteyteen on lisätty esimerkkisovellutamistapojen kautta lyhyet perustelut sille, miksi ko. heuristiikan yhteydessä myös äänellä on oma merkityksensä käytettävyyden näkökulmasta.

Minimoi käyttöliittymän valikkotasojen määrä ja järjestä valikot hyvin.	Federoff, 2002; Desurvire et al., 2004
Perustelu: Valikkotasojen ja niissä esitettyjen vaihtoehtojen määrä sekä esitysjärjestys ovat ääninäkökulmasta keskeisiä erityisesti puheeseen tai merkkiääniin pohjautuvissa valikoissa. Ihmisen työmuistin rajallisuus vaikuttaa muistettavien vaihtoehtojen määrään merkittävästi.	
Tarjoa keinoja virheiden välttämiseen ja niistä toipumiseen varoitusviestien avulla.	Federoff, 2002
Perustelu: Varoitukset voidaan esimerkiksi puhuttuina viesteinä, varoittavina äänin tai ääniin yhdistettyinä teksti- tai kuvamuotoisina ilmoituksina. Ääni kiinnittää pelaajan huomion, vaikka hänen tarkkaavaisuutensa olisi kohdistettuna muualle.	
Pelin pitää antaa vihjeitä, muttei liikaa.	Federoff, 2002; Barendregt & Bekker, 2004
Perustelu: Osa pelin antamista vihjeistä voidaan toteuttaa äänen avulla. Vihjeet voivat olla esimerkiksi sijaintiin tai suuntaan liittyviä tai suorempia puhuttuja vihjeitä. Vihjeiden esittämisherkkyydellä ja selvyydellä voidaan vaikuttaa esimerkiksi pelin haasteellisuuteen.	
Anna pelaajan tekemille toimille välitön palaute.	Desurvire et al., 2002; Barendregt & Bekker 2004
Perustelu: Ääni on hyvä nopean ja jatkotyöskentelyä häiritsemättömän palautteen välittämiskanava.	
Tarjoa mielenkiintoinen ja mukaansatempaava opaste.	Federoff, 2002; Desurvire et al., 2004; Pagulayan et al., 2003
Perustelu: Äänen merkitys opasteessa voi olla esimerkiksi puhutussa tekstissä, toiminnan kuvaamisessa ja toimintojen esiintuomisessa.	
Peliä on miellyttävä pelata uudelleen.	Desurvire et al., 2004
Perustelu: Pelin äänimaailma tulee rakentaa sellaiseksi, että se soveltuu niin ensikertalaisen kuin peliä useamman kerran pelanneen pelattavaksi. Tämä tulee huomioida esimerkiksi ääneen annettujen ohjeiden mahdollisena valinnaisuutena, vaihtuvuutena jne. Käytetyt äänet eivät saa olla käyttäjää kyllästyttäviä tai ärsyttäviä.	
Tarjoa pelaajalle tapauskohtaista apua niin, ettei hän jää jumiin tai tarvitse turvautua ohjekirjaan	Desurvire et al., 2004
Perustelu: Esimerkiksi käytönaikaiset ohjeet voidaan toteuttaa ääneen puhuttuina tai pelaajaa voidaan tukea äänivihjein.	

Tauluko 4. Esimerkkejä heuristiikoissa, joiden kohdealan arviointiin ja toteutukseen äänellä on usein selvät kytkökset.

6. Äänimaailmojen suunnittelun ja arvioinnin lähtökohtia lasten tietokonepelien kontekstissa

6.1. Onnistuneen äänimaailman merkityksestä

Äänellä voi olla lasten tietokonepeleissä erityinen merkitys pelikokemuksen muodostumisessa. Ääni voi olla mukana paitsi viihteellisyyttä tuottavassa roolissa musiikkina ja ääniefekteinä, myös lasta kokonaisvaltaisesti pelissä opastavana ja ohjaavana. Parhaassa tapauksessa ääni voi olla myös avain lapsen itsenäiseen pelikokemukseen, jossa lukutaitoinen, pelin tarkoitukseen, toimintojen sijaintiin, tarkkaavaisuuden ohjaamiseen yms. perinteisesti vaadittu avustaja voidaan jättää taka-alalle ja vahvistaa näin lapsen oman tekemisen ja oman osaamisen tunnetta.

Onnistuneen äänimaailman luominen – ja arviointi – vaatii huomionkiinnittämistä niin yleisiin äänenkäyttöä koskeviin ohjeisiin, peliominaisuudet huomioiviin suunnitteluohjeisiin sekä kohderyhmän erityispiirteisiin. Pelien äänimaailman kohdalla kohderyhmän tuntemus on, kuten sovellusten kohdalla ylipäätään, erittäin tärkeää.

Seuraavassa esitetään äänen huomioimista vaativista peliheuristiikoista, ääntä koskevista yleisistä käytettävyysohjeista sekä lastenpelien arviointia koskevista ohjeista koottu ja johdettu ohjeisto äänimaailmojen arvioinnin lähtökohdaksi. Jokaisen ohjeen yhteyteen on liitetty huomioita siitä, miksi ohjeen huomioiminen on tärkeää nimenomaan lapsikohderyhmän osalta.

6.2. Ohjeisto lasten tietokonepelien äänimaailmojen suunnitteluun ja arviointiin

Huolehdi kuultavan tekstin selkeydestä (Sinkkonen ym, 2002). Tekstin selkeys on sen sisällön ymmärtämisen perustekijä. Epäselvä teksti lisää pelaajalle aiheutuvaa kognitiivista kuormaa ja häiritsee keskittymistä tekstin kokonaissisältöön. Tämä on merkittävä asia erityisesti lasten kohdalla, joiden kyky esim. yhdistää osittain kuulemaansa tekstiä kokonaisuudeksi on aikuisia heikompi.

Selkeyteen tuleekin kiinnittää erityistä huomiota esimerkiksi pelihahmojen äänten valinnassa. Lasten peleissä hahmojen äänillä tavoitellaan usein lapsenomaisuutta ja äänet ovat usein fiktiivisiä (vs. ihmisääni). Pienten lasten kohdalla liika erilaisuus normaalista puheesta tekee siitä kuitenkin vaikeammin ymmärrettävää.

Käytä kohderyhmän ymmärtämää kieltä (Sinkkonen ym., 2002; Baauw et al., 2005). Lasten omaama sanavarasto on aikuisten sanavarastoa suppeampi. Esimerkiksi rinnakkaistermien, vertauskuvallisen ilmaisun sekä lyhenteiden ymmärtäminen on lapsille huomattavasti hankalampaa. Myös aikuisten normaaliin kielenkäyttöön liittyvät ilmaisut kuten ”vasen”, ”oikea”, ”pohjoinen”, ”ete-

lä" jne. ovat pienille lapsille tuntemattomia. Kielenkäytön kohdalla tavoitellulla kohderyhmällä suoritettu testaus onkin tärkeää. Myös lauserakenteet ja lauseiden pituus tulee sovittaa kohderyhmälle sopivaksi.

Älä käytä pitkiä lauseita (Sinkkonen ym., 2002). Kuullun tekstin osalta pitkät lauseet ovat erityisesti pienille lapsille hankalia. Lauseiden ongelmat saattavat liittyä paitsi ymmärtämiseen liittyviin ongelmiin, myös siihen, etteivät lapset malta kuunnella pitkiä lauseita loppuun asti.

Pyri tekemään äänistä tunnistettavia (Federoff, 2002; Desurvire et al., 2004). Äänten tunnistettavuuteen voidaan nähdä lasten tietokonepeleissä useita ulottuvuuksia. Käytettävillä äänillä voidaan jäljitellä reaalimaailman kohteita ja tarjota siten vinkkejä esimerkiksi painikkeisiin liittyvästä toiminnallisuudesta. Äänillä voidaan tavoitella myös tilanteen tunnistamista, esimerkiksi tarvetta johonkin toimintaan, tunnelman tai paikan tunnistamiseen. Tunnistettavuus voi liittyä myös pelin sisäiseen tunnistettavuuteen, kuten kytkeytyminen tietynlaisiin toiminnallisuuksiin, tiettyyn hahmoon, tapahtumasarjaan jne. Yksi tunnistettavuuden ulottuvuuksista on äänen tunnistaminen merkitykselliseksi ja mahdollisesti toimintaa edellyttäväksi.

Lapset pitävät käyttöliittymissä tutuista asioista (Hanna et al., 1999). Tämä tulee huomioida myös äänisuunnittelussa.

Anna yhdessä kehotteessa enintään neljä vaihtoehtoa (Sinkkonen ym., 2002). Ihmisen työmuisti on hyvin rajallinen. Lapsille suunnatuissa kehotteissa ja ohjeissa pätee sama sääntö kuin aikuistenkin vastaavien osalta: pyri minimoimaan vaihtoehtojen määrä ja huolehdi vaihtoehtojen poissulkevuudesta.

Esitä aina ensin ehto, sitten vasta sen edellyttämä toiminta (Sinkkonen ym., 2002). Niin lasten kuin aikuisten kohdalla on tärkeää, että toiminnanopastus tapahtuu oikeassa järjestyksessä. Jos ehto esitetään vaadittavan toimenpiteen jälkeen, on toimenpide usein jo unohtunut ehdon kuulemisen jälkeen (Sinkkonen ym., 2002).

Herätä pelaajan kiinnostus ääniefekteillä (Federoff, 2002). Ääniefektit ovat erinomainen keino tarkkaavaisuuden kiinnittämiseen. Lapset ovat kiinnostuneita erilaisista äänistä ja nauttivat usein toiminnoista, joiden seurauksena on ääniefektejä. Pelien käytettävyyden kannalta tarkasteltuna ääniefektien käytössä tulisi kuitenkin pysyä kohtuudessa. Liian useat efektit laskevat lopulta niiden merkitysarvoa ja voivat vaikeuttaa pelaajan tarkkaavaisuuden saantia tilanteissa, joissa tarkkaavaisuuden ohjaaminen olisi tarpeellista. Hanna ja muut (1999) ovat esittäneet esimerkiksi 0.1-0.5 sekunnin viivettä äänen kuulumiseen tilanteissa, joissa kursorin olo toiminnon päällä aikaansaa äänipalautteen. Tällä pyritään vähentämään käytön kannalta merkityksettömiä ääniä, jotka aiheutuisivat lapsen liikuttaessa hiirtä ympäri näyttöä.

Kytke pelin äänet johonkin toimintoon tai pelaajan tunteeseen. Huolehdi tyylin kontekstisesta yhdenmukaisuudesta. (Desurvire et al., 2004) Tietokonepeleissä tulee pyrkiä äänien tarkoituksenmukaisuuteen ja välttää ns. hälyääniä, jotka laskevat merkityksellisten äänten huomioarvoa. Pelin äänet taustamusiikkia myöten olisi hyvä valita pelitilanteen mukaan. Musiikilla voidaan esimerkiksi viestiä erilaisista tunnetiloista tai ilmaista siirtyminen tilasta toiseen. Ääniefektit taas ovat tehokkaimmillaan tiettyihin toimintoihin yhdistettynä, jolloin pelaaja saa samalla palautetta tekemisistään. Tämä pätee niin lapsille kuin aikuisille suunnattuihin peleihin.

Äänten kohdalla tulisi huomioida äänten tyylillinen yhdenmukaisuus pelitilojen, -tilanteiden ja -tapahtumien mukaan. Esimerkiksi lapset voivat käyttää tiettyyn pelitilaan liittyviä ääniä muistivihjeenä siitä, millaisia toiminnallisuuksia ko. tilaan liittyy ja mikä heidän sijaintinsa pelissä ylipäätään on. Äänten yhdenmukaisuus esimerkiksi samojen toimintojen kohdalla luo pelissä kaivattavaa johdonmukaisuutta, mikä auttaa pelin oppimista.

Tarjoa mielenkiintoinen ja mukaansatempaava alkuohjeistus/tutoriaali, jossa pelin päätavoite ja perustoiminnot esitetään selkeästi (Federoff, 2002; Desurvire et al., 2004; Pagulayan et al., 2003, Baauw et al., 2005). Tietokonepelin pelaamisen perusedellytyksiä ovat pelin pelaamistavan oppiminen sekä pelin tavoitteen ymmärtäminen. Helppo opittavuus ja ohjeiden yksinkertaisuus ovatkin Druinin ja muiden (1999) mukaan lapsiin vetoavia tekijöitä. Lapset ovat usein toiminnassaan kärsimättömiä, mikä edelleen korostaa nopean ja oikeasuuntaisen liikkeelle pääsyn merkitystä.

Ohjeiden antaminen puhuttuina on pienten lasten peleissä erityisen suositeltavaa. Suositeltavaa yhdistää ohjeiden esittämisessä kuva ja ääni, jolloin pelaajan huomio saadaan paremmin herätettyä. Esimerkiksi Hanna ja muut (1999) ovat havainneet lasten kiinnittävän enemmän huomiota hahmon lukemiin ohjeisiin kuin pelkkään ääneen.

Anna toiminnasta välitön, ymmärrettävä palaute (Barendregt & Bekker, 2004; Desurvire et al., 2004; Baauw et al., 2005). Erityisesti lapset kaipaavat toiminnastaan palautetta. Havaittavan palautteen puuttuminen saa lapsen usein olettamaan, ettei vuorovaikutus pelin kanssa esimerkiksi painikkeen painamisen osalta ole onnistunut. Lapset saattavat toistaa saman toiminnon useita kertoja odottaen palautteen saamista. Ääni on monipuolinen kanava palautteen antamiseen. Yksi sen hyvistä ominaisuuksista on riippumattomuus käyttäjän senhetkisestä tarkkaavaisuuden kohteesta. Ääni ei myöskään sopivassa muodossa annettuna häiritse käyttäjän muuta työskentelyä.

Huomioi palautteenanto sekä oikeasta että väärästä toiminnasta (Baauw et al., 2005). Lapset tarvitsevat usein vahvistusta tiedolle siitä, onko heidän valitse-

mansa toiminto tai vaihtoehto oikea. Tämän johdosta peleissä tulisikin huomioida erilaisen palautteen antaminen oikean ja väärän toiminnan ja valinnan kohdalla. Tämä tulee huomioida myös ääneen annetun palautteen yhteydessä. Äänen ominaisuudet mahdollistavat palautteen laadun ilmaisemisen tunnistettavalla tavalla usealla eri tavalla.

Anna pelaajalle pelin haasteellisuustason huomioivia vihjeitä (Federoff, 2002; Barendregt & Bekker, 2004). Ääneen annetut vihjeet voivat olla hyvin erilaisia. Vihjeet voivat olla lasten tietokonepeleissä esimerkiksi toimintojen sijaintipaikoista kertovia äänimerkkejä kursorin ollessa painikkeen päällä, pelissä etenemisen kannalta merkityksellisiä suuntaa ilmaisevia ääniä tai puhuttuna tekstinä ilmaistuja konkreettisempia vihjeitä. Vihjeet tulee sovittaa pelin vaikeustasoon siten, etteivät vihjeet ole liian paljastavia tai minimalistisia. Lapsille on hyvä antaa vihjeitä pelissä jumiutumisen estämiseksi, joka johtaa helposti lapsen turhautumiseen ja pelaamisen keskeyttämiseen.

Tarjoa pelaajalle käytönaikaisia ohjeita (Federoff, 2002; Desurvire et al., 2004). Ääneen annetut käytönaikaiset ohjeet tarjoavat mahdollisuuden opastaa pelaajaa keskeyttämättä kokonaan hänen pelisuoritustaan. Lukutaidottomien lasten kohdalla ohjeiden esittäminen puhuttuna on muutoinkin hyvin merkittävä tekijä pelin käytön itsenäisyydessä. Ohjeet voivat olla esimerkiksi johonkin pelitilanteeseen liittyviä aina kuultavia ohjeita tai ne voivat aktivoitua pelaajan jouduttua pelissä ongelmiin. Ohjeiden saaminen on hyvä mahdollistaa myös pelaajan toimesta esimerkiksi ohjepainiketta painamalla. Hanna ja muut (1999) ovat lisäksi esittäneet, että puhuttujen ohjeiden yhteydessä pitäisi mahdollistaa niiden toistaminen sekä ohjeiden pysäyttäminen.

Auta pelaajaa välttymään ja toipumaan pelitapahtumiin liittymättömistä virheistä (Federoff, 2002). Äänen avulla voidaan antaa erilaisia varoituksia ja virheilmoituksia. Toivottavaa toki on, ettei virhetilanteita pääse syntymään. Lasten kohdalla virheitä aiheutuu usein esimerkiksi epäselvästä navigointirakenteesta ja toisistaan erottumattomista painikkeista. Lapsi saattaa esimerkiksi poistua pelistä tulkitessaan navigointipolut eri tavalla kuin niiden toteuttaja (Barendregt et al., 2006). Äänen yhdistäminen painikkeisiin on yksi mahdollisuus tällaisten virheiden estämiseen. Painikkeen merkityksen kuuleminen vähentää tällaisissa tilanteissa virhevalintoja. Sama voidaan toteuttaa puheen ohella muunlaisin äänimerkein. Myös virheilmoitusten antaminen puhuttuna auttaa lasta toipumaan virhetilanteesta, jotka usein ovat lapsille ylitsepääsemättömiä.

Huomioi pelin miellyttävyys myös uudelleenpelaamisen yhteydessä (Desurvire et al., 2004). Lapset pelaavat usein samoja tietokonepelejä yhä uudelleen. Pelikokemuksen miellyttävyyden ja sujuvuuden kannalta peleissä tulisikin kiinnittää huomiota tähän liittyen myös äänen osa-alueeseen. Gal ja muut (2002) varoitta-

vat esimerkiksi musiikin osalta lyhyiden luuppien pitkästyttävästä vaikutuksesta – saati sitten pelattaessa peliä yhä uudelleen ja uudelleen. Pelin äänet tulisi ylipäätään suunnitella sellaisiksi, että ne herättävät mielenkiinnon yhä uudelleen. Yhtenä keinona tähän voisi olla esimerkiksi osin pelikerroittain vaihtuvat äänet ja hahmojen kommentit. Uudelleenpelaaminen on haasteellista myös pelin sisältämien ohjeiden kannalta: pelissä aina kuuluvat ohjeet menettävät toistuvasti kuultuina merkityksensä, eikä lapsi välttämättä kuuntele ohjeita sellaisissakaan tilanteissa, joissa hän ei ole aikaisemmin pelissä ollut. Eräänlainen turtumus voi johtaa siirtovaikutukseen myös muiden pelien kohdalla.

7. Yhteenveto

Ääni voi olla merkittävä osa tietokonepelin käytettävyyttä. Tähän mennessä pelien suunnittelu- ja arviointiheuristiikoissa ja ohjeissa on kuitenkin painotettu enemmän pelien visuaalista puolta. Tämän tutkielman yhtenä tarkoituksena on ollut osoittaa, että äänelläkin on väliä – myös käytettävyyden näkökulmasta.

Lapset ovat heterogeeninen käyttäjäryhmä, jonka erityispiirteet on huomioitava sille suunnattujen tuotteiden suunnittelussa ja arvioinnissa. Ääni tarjoaa tähän useita mahdollisuuksia. Samalla se voi lisätä lapsille mahdollisuuksia pelituotteiden itsenäiseen käyttöön, on tämä sitten kasvatuksellisista lähtökohdista toivottavaa tai ei. Joka tapauksessa lapsille voidaan äänen avulla mahdollistaa paremmin käyttäjäryhmälle sopiva ohjeiden, opastuksen ja palautteen saaminen entistä hauskemmasta, mukaansatempaavasta ja nautittavasta pelikokemuksesta puhumattakaan.

Äänen huomioivat peliheuristiikat ovat toistaiseksi melko vähäisiä, joskin osa nykyisistä peliheuristiikoista on laajennettavissa koskemaan myös pelien ääniä. Tietokonepelien äänimaailma on kuitenkin osa-alue, jonka ominaisuuksien tutkiminen ja sitä koskevien peliheuristiikkojen laatiminen on jatkossa tarpeellista. Äänen hyödyntäminen entistä tehokkaammin ei alenna sen viihteellisyyttä, mutta voi vaikuttaa merkittävällä tavalla yleiseen pelikokemuksen laatuun.

Tässä tutkielmassa on esitetty ohjeisto, jota on mahdollista hyödyntää lasten tietokonepelien äänimaailmojen suunnittelussa ja arvioinnissa. Esitys on luonteeltaan lähtökohtia tarjoava. Ohjeiston käyttökelpoisuutta tulisi jatkossa testata esimerkiksi vertaamalla asiantuntija-arvioinnin ja käytettävyystestausten antamia tuloksia lasten tietokonepelien äänimaailmoista. Testauksen myötä ohjeistoa olisi mahdollista tarkentaa ja laajentaa. Samalla voitaisiin selvittää esimerkiksi sitä, millainen äänentoistolaitteisto lapsilla on pelatessa käytettävissään.

Tässä tutkielmassa laadittu ohjeisto antaa kuvan siitä, millaisia mahdollisuuksia – ja toisaalta sudenkuoppia – äänenkäytöllä tietokonepelien kontekstis-

sa on. Näiden seikkojen välittäminen laajempaan tietoisuuteen olisi varmasti hyödyllinen lisä tietokonepelien suunnitteluun ja arviointiin.

8. Lähteet

Baauw, E., Bekker, M. M., & Barendregt, W. (2005). A structured expert evaluation method for the evaluation of children's computer games. *Human-Computer Interaction - INTERACT 2005.* Lecture Notes in Computer Science 3585, 457-469.

Barendregt, W., & Bekker, M. M. (2004). Towards a framework for design guidelines for young children's computer games. In *Entertainment Computing - ICEC 2004.* Lecture Notes in Computer Science 3166, 365-376.

Barendregt, W., Bekker, M. M., Bouwhuis, D. G., & Baauw, E. (2006). Identifying usability and fun problems in a computer game during first use and after some practice. *International Journal of Human-Computer Studies,* 64(9), 830-846.

Chiasson, S., & Gutwin, C. (2005). *Design principles for children's software*. Technical Report HCI-TR-05-02, Computer Science Department, University of Saskatchewan.

Csikszentmihalyi, M. (1991). *Flow: psychology of optimal experience*. New York: HarperPerennial.

Desurvire, H., Caplan, M., & Toth, J. A. (2004). Using heuristics to evaluate the playability of games. *CHI '04 Extended Abstracts on Human Factors in Computing Systems,* Vienna. 1509-1512.

Druin, A. (2002). The role of children in the design of new technology. *Behaviour & Information Technology, 21*(1), 1-25.

Druin, A., Bederson, B., Boltman, A., Miura, A., Knotts-Callahan, D. & Platt, M. (1999). Children as our technology design partners. In: Druin, A. (Eds.) *The design of children's interactive technologies* (pp. 51-72). San Francisco: Morgan Kaufmann.

Dyck, J., Pinelle, D., Brown, B., & Gutwin, G. (2003). Learning from games: HCI design innovations in entertainment software. Proceedings of Graphics Interface. Retrieved December 15, 2006, from http://www.graphicsinterface.org/cgi-bin/DownloadPaper?name=2003/159/ paper159.pdf.

Ekman, I., Ermi, L., Lahti, J., Nummela, J., Lankoski, P. & Mäyrä, F. (2005). *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, Valencia, Spain. 110-116.

Eskelinen, M. (2005). *Pelit ja pelitutkimus luovassa taloudessa*. Helsinki: Sitra.

Falstein, N. & Barwood, H. (2006). *The 400 Project*. Retrieved December 15, 2006 from http://theinspirancy.com/400_project.htm.

Federoff, M. (2002). Heuristics and usability guidelines for the creation and evaluation of fun in videogames. Master's thesis, Indiana University.

Gal, V., Le Prado, C., Merland, J. B., Natkin, S. & Vega, L. (2002). Processes and tools for sound design in computer games. *International Computer Music Conference*, Gothenburg. Retrieved December 15, 2006, from http://cedric. cnam.fr/ PUBLIS/RC326.pdf.

Hanna, L., Risden, K., & Alexander, K. J. (1999). The role of usability research in designing children's computer products. In: Druin, A. (Ed.), *The design of children's technologies* (pp. 3-26). San Francisco: Morgan Kaufman.

Hanski, M.-P. & Kankainen, A. (2004). Pelien laadun kehittäminen käyttäjien näkökulmasta. Teoksessa: Kankaanranta, M., Neittaanmäki, P. & Häkkinen, P. (toim.), *Digitaalisten pelien maailmoja*, (ss. 67-76). Jyväskylä: Jyväskylän yliopistopaino.

Hietala, P. & Ovaska, S. (2002). Lapset ja käyttöliittymät – johdatus aihepiiriin. Teoksessa: Hietala, P. & Ovaska, S. (toim.), *Lasten käyttöliittymät*. (ss. 1-19). Tampereen yliopisto: Tietojenkäsittelytieteiden laitos. Saatavissa 15.12. 2006 myös: http://www.cs.uta.fi/reports/bsarja/B-2002-2.pdf.

Höysniemi, J. (2005). Käytettävyystestaus lasten kanssa. Teoksessa: Ovaska, S., Aula, A. & Majaranta, P. (toim.), *Käytettävyystutkimuksen menetelmät* (ss. 259-282). Tampereen yliopisto: Tietojenkäsittelytieteiden laitos.

ISO (1998). International Organisation for Standardazion, *ISO 9241-11: Ergonomic requirements for office work with visual display terminals (VDTs). Part 11: Guidance on Usability.*

Jørgensen, A. H. (2004). Marrying HCI/Usability and computer games: A preliminary look. *Proceedings of the Third Nordic Conference on Human-Computer Interaction*, Tampere, Finland. 393-396.

Järvinen, A. (2002). Kolmiulotteisuuden aika. Audiovisuaalinen kulttuurimuoto vuosina 1992-2002. Teoksessa: Huhtamo, E. & Kangas, S. (toim.), *Mariosofia: elektronisten pelien kulttuuri*. (ss. 70-92). Helsinki: Gaudeamus.

Järvinen, A., Heliö, S. & Mäyrä, F. (2002). *Communication and community in digital entertainment services*. Prestudy research report. University of Tampere: Hypermedia Laboratory. Retrieved December 15, 2006 from http://tampub. uta.fi/ tup/951-44-5432-4.pdf.

Kasvi, J. (2001). Lasten tietokonepelit. Teoksessa: Kangassalo, M. & Suoranta, J. (toim.), *Lasten tietoyhteiskunta*, (ss. 106-123). Tampere: Tampereen yliopistopaino.

Korhonen, H., & Koivisto, E. M. I. (2006). Playability heuristics for mobile games. *Proceedings of the 8th Conference on Human-Computer Interaction with Mobile Devices and Services*, Helsinki, Finland. 9-16.

Latva, S. (2004). Pelisuunnittelun tematiikka - lapsille tarkoitettujen digitaalisten pelien suunnittelun lähtökohtia. Teoksessa: Kankaanranta, M., Neittaanmäki, P. & Häkkinen, P. (toim.), *Digitaalisten pelien maailmoja* (ss. 33-50). Jyväskylä, Koulutuksen tutkimuslaitos: Jyväskylän yliopisto.

Lazzaro, N., & Keeker, K. (2004). What's my method?: A game show on games. *CHI '04 Extended Abstracts on Human Factors in Computing Systems,* Vienna, Austria. 1093-1094.

Lee, C., Kim, S., Chae, C. & Chung, K. (2000). Sound: an emotional element of interactions a case study of a microwave oven. *Proceedings of the conference on Designing interactive systems: processes, practices, methods, and techniques*, New York City, New York. 174-182.

Malone, T. W. (1982). Heuristics for designing enjoyable user interfaces: Lessons from computer games. *Proceedings of the 1982 Conference on Human Factors in Computing Systems,* Gaithersburg, Maryland. 63-68.

Manninen, T. (2004). *Rich interaction model for game and virtual environment design*. Tietojenkäsittelytieteiden väitöskirja. Oulu: Oulun yliopisto.

Menshikov, A. (2003). *Modern audio technologies in games*. Retrieved December 15, 2006, from http://www.digit-life.com/articles2/sound-technology/index. html.

Nielsen, J. (1993). *Usability engineering*. Academic Press, Inc.

Nielsen, J. (2002). *Kid's corner: Website usability for children.* Retrieved October 7, 2006, from http://www.useit.com/alertbox/20020414.html.

Pagulayan, R., Keeker, K., Wixon, D., Romero, R. L. & Fuller, T. (2003). User centered design in games. In: Jacko, J. & Sears, A. (Eds.), *Handbook for Human Computer Interaction.* Mahwah: Lawrence Erlbaum & Associates.

Räty, V. (1999). *Pelien leikki lasten tietokonepelien suunnittelusta sekä käytöstä erityisesti vammaisten lasten kuntoutuksessa*. Helsinki: Taideteollinen korkeakoulu.

Röber, N. & Masuch, M. (2004). Auditory Game Authoring: From virtual Worlds to auditory Environments. *Computer Games: Artificial Intelligence, Design and Education 2004*. 114–121.

Sinkkonen, I., Kuoppala, H., Parkkinen, Jarmo & Vastamäki, Raimo (2002). *Käytettävyyden psykologia*. Helsinki: Edita, IT Press.

Sweetser, P. & Wyeth, P. (2005). GameFlow: a model for evaluating player enjoyment in games. *Computers in Entertainment*, 3(3).

Tilastokeskus (2006). *Suomen tilastollinen vuosikirja 2006*. Hämeenlinna: Karisto Oy.

Katsaus tunkeutumisen havaitsemiseen

Jukka Pitkänen

Tiivistelmä.
Tunkeutumisen havaitsemisesta on tullut tärkeä tietoturvan osa-alue, jonka avulla voidaan havaita tunnettuja ja tuntemattomia tunkeutumisia järjestelmiin. Tunkeutumisen havaitsemismenetelmät tarjoavat perustan, jonka avulla voidaan erottaa järjestelmän väärinkäyttö ja poikkeava käyttäytyminen normaalista käytöstä. Tunkeutumisen havaitseminen perustuu lähtökohdiltaan tietämys- ja käyttäytymispohjaisiin menetelmiin. Tietämyspohjaiset menetelmät tarjoavat mahdollisuuden havaita tunkeutuminen ennalta olevan tietämyksen perusteella tunkeutumistavoista ja järjestelmän haavoittuvuuksista. Käyttäytymiseen perustuvissa menetelmissä puolestaan verrataan järjestelmän käyttöä sen aiempaan käyttöön. Tunkeutumisen havaitsemisjärjestelmät toimivat näiden lähtökohtien pohjalta isäntäkonekohtaisina, verkkokohtaisina tai hybrideinä. Arkkitehtuurillisesti tunkeutumisen havaitsemisjärjestelmät voidaan jakaa monoliittisiin, hierarkkisiin ja yhteistoiminnallisiin. Tässä tutkielmassa tarkastellaan yleisimpien tunkeutumisen havaitsemismenetelmien ja -järjestelmätyyppien toimintaa.

Avainsanat ja -sanonnat: tunkeutumisen havaitseminen, tunkeutumisen havaitsemismenetelmät, tunkeutumisen havaitsemisjärjestelmät, tietoturva.
CR-luokat: K.2, K.6.5, D.4.6

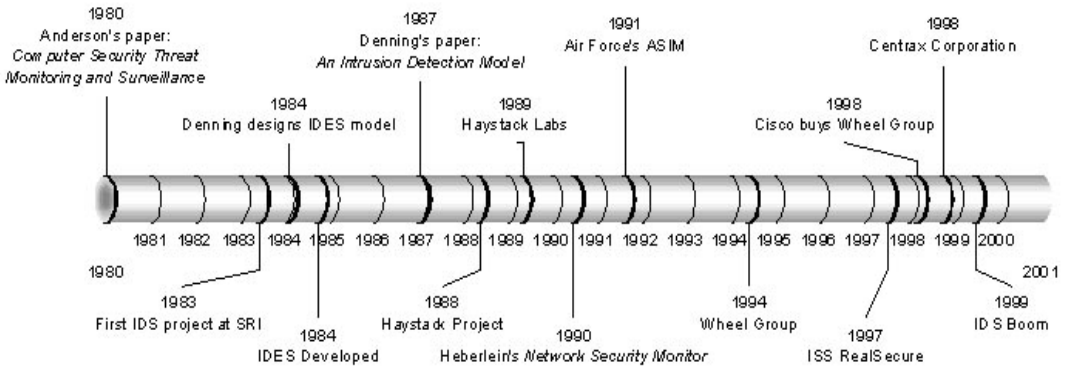
1. Johdanto

Ulkopuolisten uhkien torjuminen ja havaitseminen on historian aikana ollut tärkeää kulttuurien parissa. Tämä pätee myös nykyään, sillä modernien tietojärjestelmien ja -verkkojen suojaaminen on tärkeää ulkoisia ja sisäisiä uhkia vastaan. Tietojärjestelmien ja -verkkojen suojaaminen ei rajoitu pelkästään uhkien estämiseen, vaan myös tietojärjestelmiin ja -verkkoihin kohdistuneiden tunkeutumisten havaitsemiseen. Yleisimpiä ja tunnetuimpia keinoja suojautua uhkia vastaan ovat esimerkiksi palomuurit, tietoliikenteen suodatus, toimintojen estäminen ja virustorjunta. Näiden lisäksi on myös syntynyt tarve havaita tunkeutuminen, jos tietojärjestelmän etulinja on pettänyt ja sinne on päästy tunkeutumaan. Tunkeutuminen järjestelmään luo epämääräistä ja normaalista poikkeavaa liikennettä ja tapahtumia.

Tunkeutumisen havaitsemisen on oltava tehokasta ja tunkeutuminen on kyettävä havaitsemaan mahdollisimman nopeasti, jotta tunkeutuja, oli se sitten

henkilö tai haittaohjelma, ei pääse tekemään tuhojaan tai kuten monessa tapauksessa keräämään hakemaansa tietoa, esimerkiksi luottokorttien tietoja tai muuttamaan tietoja, jolloin tietojen eheys menetetään. Tätä tarkoitusta varten on kehitetty tunkeutumisen havaitsemisjärjestelmiä, jotka toimivat eri toimintaperiaatteiden pohjalta, mutta kuitenkin niiden avulla pyritään samaan lopputulokseen, havaitsemaan tunkeutuminen ja ottamaan selville mitä tunkeutuja on tehnyt ja minimoimaan vahingot.

Tunkeutumisen havaitsemisen tutkiminen alkoi 1980-luvulla ja ensimmäiset tutkimukset aiheesta syntyivät 80-luvun alkupuolella. Tunkeutumisen havaitsemisen tutkimisen aloitti James P. Anderson [1980]. Hän painotti sitä, että kirjausketjujen pohjalta pystyttiin havaitsemaan luvattomat järjestelmän resurssien ja tiedostojen käytöt, eli aiheet eivät ole tästä muuttuneet ajan kuluessa. Vuonna 1987 Dorothy Denning [1986] esitti viitekehyksen tunkeutumisen havaitsemisjärjestelmille. Tuolloin luotu viitekehys pätee yhä taustana tunkeutumisen havaitsemisen tutkimiselle ja havaitsemisjärjestelmille. Ensimmäistä kertaa oli olemassa viitekehys, jonka pohjalta pystyttiin luomaan ohjelmallisesti toimivia reaaliaikaisia järjestelmiä tunkeutumisen havaitsemiseen. Alkuun 1980 -luvulla havaitseminen perustui paperilla olleiden käytöstä syntyneiden lokitiedostojen seuraamiseen, joita järjestelmäasiantuntija tarkastelivat. Nykyään tunkeutumisen havaitsemisjärjestelmät toimivat verkkojen eri segmenteissä tarkkaillen verkkoliikennettä sekä palvelimissa ja yksittäisissä työasemissa tarkkaillen väärinkäyttöä ja tapahtumien poikkeavuuksia normaalista. Kuvasta 1 nähdään. miten tunkeutumisen havaitseminen on kehittynyt 1980-luvun alusta ensimmäisten kirjallisten tuotosten kautta ensimmäiseen ohjelmalliseen järjestelmään, ja milloin tunkeutumisen havaitsemisjärjestelmät ovat lyöneet itsensä läpi kaupallisilla markkinoilla.



Kuva 1. Tunkeutumisen havaitsemisen kehitys [Innella, 2001].

Vaikka tunkeutumisen havaitsemista suoritetaan eri ympäristöissä tietojärjestelmässä, tarkkailemalla tietoverkkoa ja sen liikennöintiä sekä isäntäkoneen tapahtumia ja kirjausketjuja, niin tunkeutumisen havaitsemisjärjestelmät pohjautuvat kuitenkin johonkin menetelmään, eikä pelkästään siihen missä ne toimivat. Tunkeutumisen havaitseminen voidaan jakaa menetelmien perusteella kahteen kategoriaan, tietämys- ja käyttäytymispohjaisiin menetelmiin. Tietämyspohjainen tunkeutumisen havaitsemisen lähtökohta perustuu nimensä mukaisesti ennalta tiedettyyn tietoon, jonka perusteella tunkeutuminen voidaan havaita, kun puolestaan käyttäytymispohjaiset menetelmät perustuvat poikkeavan käyttäytymisen havaitsemiseen. Tässä tutkielmassa tarkastellaan yleisellä tasolla, millaisia ovat tunkeutumisen havaitsemismenetelmät toimintaperiaatteiltaan sekä miten havaitsemisjärjestelmät toimivat verkko- ja isäntäkonepohjaisissa tunkeutumisen havaitsemisjärjestelmissä.

2. Tunkeutumisen havaitsemismenetelmät

Tunkeutumisen havaitsemismenetelmiä on monenlaisia ja niitä voidaan luokitella niiden käyttämien menetelmien perusteella. Tunkeutumisen havaitsemismenetelmät voidaan yleisesti jakaa kahteen pääluokkaan, tietämyspohjaisiin ja käyttäytymispohjaisiin havaitsemismenetelmiin. Tietämyspohjaisille menetelmille on tyypillistä, että niiden avulla voidaan havaita tunkeutuminen olemassa olevan tietämyksen pohjalta. Käyttäytymispohjaisille menetelmille puolestaan on tyypillistä, että niiden avulla voidaan havaita tuntemattomia uhkia. Kumpaa tahansa menetelmää tunkeutumisen havaitsemisjärjestelmä käyttää, saa se tietoja, jonka perusteella havaitsemisjärjestelmä havaitsee mahdollisen tunkeutujan, jonka toiminta poikkeaa normaaleista järjestelmän tapahtumista [Kemmerer & Vigna, 2002].

2.1. Tietämyspohjaiset menetelmät

Tietämyspohjaiset tunkeutumisen havaitsemismenetelmät perustuvat ennalta tiedettyyn tietoon hyökkäystyypeistä ja järjestelmän haavoittuvuuksista [Kemmerer & Vigna, 2002]. Näiden tietojen pohjalta luodaan havaitsemisjärjestelmälle tietämystietokanta, jonka avulla tunkeutumisen aiheuttama järjestelmän väärinkäyttö pyritään havaitsemaan. Tietämyspohjaisia menetelmiä kutsutaan yleisesti myös väärinkäytön havaitseviksi menetelmiksi. Tietämyspohjaisia tunkeutumisen havaitsemismenetelmiä ovat asiantuntijajärjestelmiin (experts systems), malliin (model based), kaaviovertailuun (pattern matching) ja tiedon tilan analysointiin (state transition analysis) perustuvat menetelmät [Bai & Kobayashi, 2003].

Asiantuntijajärjestelmiin perustuvat tunkeutumisen havaitsemismenetelmät käyttävät hyödykseen aiempaa järjestelmän käyttöä ja siihen kohdistuneita tunkeutumisen yrityksiä sekä onnistuneita tunkeutumisia. Asiantuntijajärjestelmät periaatteessa toimivat sekä tietämyspohjaisina että käyttäytymispohjaisina, koska aiemmasta käytöstä kerätään profiilitietoa, jotta voidaan luoda tunkeutumisen havaitsemiselle säännöt [Lunt, 1993]. Käyttäytymispohjainen osa asiantuntijajärjestelmässä havaitsee tunkeutumisen silloin, kun järjestelmän odotettu käyttö poikkeaa oletetusta eli jotain resurssia yritetään käyttää luvatta. Tällöin käytetään tilastollista lähestymistapaa. Tietämyspohjainen osa asiantuntijajärjestelmästä puolestaan luo tunkeutumisskenaarioista ja hyökkäysmenetelmistä sääntötietokannan [Lunt, 1993]. Olettaen, että asiantuntijajärjestelmiin pohjautuvat tunkeutumisen havaitsemismenetelmät olisivat tehokkaita, niiden tulee toimia reaaliaikaisesti, päivittäen tietämyskantaa ja käyttäjien toiminnoista profiilijoukkoa, joilla väärinkäytökset voidaan havaita.

Malliin pohjautuva tunkeutumisen havaitseminen perustuu määritelmiin käyttäytymisestä tiettyjen tapahtumien seurauksena. Näin voidaan päättelemällä havaita tunkeutuminen, joka toteuttaa malliin määritellyn skenaarion tunkeutumisesta. Malliin perustuva tunkeutumisen havaitseminen koostuu kolmesta osasta [Lunt & Garvey, 1991]. Ensimmäinen osa hoitaa käyttö- ja skenaariomalleja, joihin on määritelty tietoja tunkeutumisskenaarioista, ja näiden avulla yritetään ennustaa oletettava seuraava tapahtumaketju. Toinen osa ottaa vastaan oletuksen seuraavasta tapahtumaketjusta ja muuntaa sen käyttäytymiseksi, joka voisi esiintyä tapahtumaketjussa. Näin tämä osa voi käyttää ennustettua tietoa seuraavan tapahtuman etsimiseen. Menetelmän kolmas osa suorittaa varsinaisen haun ja etsii vastaavaa dataa tapahtumaketjusta. Näin tietoa keräämällä voidaan havaita järjestelmään tunkeutuminen päättelemällä malliin määritellyillä skenaarioilla ja tapahtumilla onko kyseessä tunkeutuminen [Lunt & Garvey, 1991].

Kaaviovertailuun pohjautuvassa tunkeutumisen havaitsemisessa tunnetuista tunkeutumisjäljistä koostetaan kaavioita ja säännöllisiä lauseita. Tunkeutumisen havaitsemiseksi näitä kaavioita ja lauseita verrataan käytöstä kertyneeseen tietoon [Kumar, 1995]. Menetelmää voidaan käyttää esimerkiksi määrättyjen merkkijonojen etsimiseen kirjausketjusta. Kun jäljet on hyvin määritetty järjestelmälle, niin menetelmä on tehokas tunnettujen tunkeutumismenetelmien havaitsemiseen. Menetelmän heikkous piilee siinä, että sen avulla ei voida tehokkaasti havaita muita kuin tunnettuja tunkeutumismenetelmiä. Myös huonosti määritellyt kaaviot tunkeutumisista voivat heikentää menetelmän tehokkuutta.

Tilan muuttumisen analysointiin pohjautuva menetelmä perustuu siihen, että tarkkailtava järjestelmä on mallinnettu tilan muutosdiagrammina. Kun järjestelmästä saatu tieto on analysoitu, suoritetaan tiedon tilan muunnos tilasta toiseen [Bai & Kobayashi, 2003]. Menetelmä perustuu siis siihen, että tilan muuttuminen turvallisesta tilasta vaaralliseen tilaan voidaan havaita annettujen tunnettujen hyökkäysmenetelmien pohjalta. Tilan muuttumisen analysointiin pohjautuvat menetelmät tarkkailevat järjestelmän tapahtumaketjujen ja tiedostojen muutoksia ennalta annettujen määriteltyjen mallien pohjalta, joilla tunnistetaan, onko muutos turvallinen vai ei.

Tietämyspohjaisten menetelmien eduksi voidaan laskea se, että ne kykenevät havaitsemaan ne tunkeutumiset, jotka ovat tunnettuja. Täten ne suojaavat järjestelmän haavoittuvuuksia. Tässä piileekin suurin heikkous tietämyspohjaisessa tunkeutumisen havaitsemisessa, koska tuntemattomat tunkeutumistyypit jäävät havaitsematta. Käyttäytymiseen pohjautuviin menetelmiin verrattuna tietämyspohjaisten menetelmien etuna on myös se, että ne antavat vähemmän vääriä hälytyksiä tunkeutumisista.

2.2. Käyttäytymiseen pohjautuvat menetelmät

Käyttäytymisen pohjalta tunkeutumisen havaitsevat menetelmät perustuvat lähtökohtaan, jonka perusteella pyritään havaitsemaan normaaleista tapahtumista poikkeavat tapahtumat [Debar]. Tämän perusteella oletetaan, että kaikki normaalista poikkeavat tapahtumat ovat välittömästi tunkeutumisesta johtuvia. Käyttäytymiseen pohjautuvien havaitsemismenetelmien perustaksi luodaan tietokanta järjestelmän aiemmista tapahtumista ja käyttäjäprofiileista. Käyttäytymiseen pohjautuvat järjestelmät oppivat aikaa myöten, mitkä tapahtumat tai millainen verkkoliikenne on normaalia, kun ne toistuvat tarpeeksi usein. Tämän perusteella voidaan käyttäytymiseen pohjautuvia havaitsemismenetelmiä nimittää myös poikkeavuuden havaitsemiseksi. Normaalista käyttäytymisestä poikkeavien tapahtumien havaitsemiseen voidaan käyttää tilastollisen päättelyyn (statistical approach), ennustavaan malliin (predictive pattern generation) ja neuroverkkoihin (neural networks) perustuvia menetelmiä [Bai & Kobayashi, 2003].

Tilastollisen päättelyn menetelmät perustuvat käyttäytymisprofiileihin, jotka luodaan olemassa olevista käyttäjäprofiileista. Menetelmän avulla luodaan varianssi käytössä olevasta profiilista ja verrataan sitä alkuperäiseen profiiliin [Bai & Kobayashi, 2003]. Taustalla olevaan käyttäjäprofiilin voidaan sisällyttää useita eri mittayksiköitä järjestelmän käytöstä kuten esimerkiksi muistinkulutus, prosessorin kuormitus, verkkoyhteyden aika, käytössä olevien prosessien määrä ja verkkoprotokollien käyttö. Näiden perusteella luodaan tilastollinen

malli tunkeutumisen havaitsemisjärjestelmälle, jonka kautta tunkeutumien voidaan havaita. Tärkeintä ja myös vaikeinta tilastollisen päättelyn menetelmässä on miten valita oikeat mittaluvut, joita tarkkaillaan. Suurimpana vahvuutena ja heikkoutena menetelmässä on se, että se oppii käyttäjän käyttäytymisestä oikeanlaisen käyttäytymismallin. Oikeiden käyttäytymismallien tunnistaminen on toisaalta vahvuus, mutta jos tunkeutuja on kyennyt tunkeutumaan järjestelmään, se voi opettaa havaitsemisjärjestelmälle oman käyttäytymisensä normaaliksi käyttäytymiseksi. Menetelmän kautta voidaan saada suuri määrä vääriä hälytyksiä, jolloin hälytykset vastaanottavan osapuolen tulee olla riittävän tehokas suodattaakseen menetelmän pohjalta saadut oikeat hälytykset.

Ennustaviin kaavioihin perustuvat menetelmät pyrkivät ennustamaan tulevia tapahtumia perustuen tapahtumiin, joita on aiemmin tapahtunut. Tunkeutumisen havaitseminen tämän menetelmän mukaan tapahtuu siten, että tapahtumien jälkeen arvioidaan tulevien tapahtumien todennäköisyydet, joiden avulla voidaan päätellä, onko käyttäytyminen poikkeavaa ja täten tunkeutuminen järjestelmään [Bai & Kobayashi, 2003]. Jos menetelmän avulla ei pystytä ennustamaan käyttäytymistä oikein ja se poikkeaa normaalista käyttäytymisestä sekä havaitsemissäännöistä, menetelmän pohjalta suoritetaan hälytys tunkeutumisesta. Koska menetelmä perustuu aiempiin käyttäytymismalleihin, tulee menetelmää käyttävälle järjestelmälle opettaa normaali käyttäytyminen sekä normaalista poikkeava käyttäytyminen, jotta todennäköisyyksien avulla ennustaminen olisi tehokasta. Suurimpana etuina menetelmässä on se, että sääntöpohjaiset määritykset kykenevät havaitsemaan poikkeavia tapahtumia, joita ei muuten kyettäisi havaitsemaan, sekä järjestelmälle luotujen sääntöjen sopeutuvuus muutoksille [Bai & Kobayashi, 2003].

Neuroverkkoihin perustuvissa menetelmissä pyritään kouluttamaan neuroverkkoa ennustamaan käyttäjän seuraavaksi aikomia komentoja ja tapahtumia [Bai & Kobayashi, 2003]. Neuroverkon kouluttaminen tapahtuu siten, että sille annetaan käyttäjän komentoja kuvaava joukko. Tunkeutumisen havaitseminen menetelmän avulla suoritetaan siten, että neuroverkko peilaa käyttäjän antamia komentoja neuroverkkoon annettua profiilia vastaan ja siten jokainen väärin ennustettu tapahtuma itseasiassa mittaa poikkeamaa käyttäjän komennoista verrattuna profiilin skenaarioihin. Menetelmän etuna on, että se havaitsee paremmin tunkeutumisen kuin tilastolliseen päättelyyn perustuva menetelmä ja sillä on parempi kyky käsitellä hajanaista tietoa. Heikkona puolena voidaan pitää sitä, että tässäkin menetelmässä tunkeutuja voi kouluttaa järjestelmää ja tällöin järjestelmä ei kykene havaitsemaan poikkeamia käyttäjän käyttäytymisestä. Jos on annettu liiallisen pieni määrä käyttäjän komentoja profiiliin, niin

siitä aiheutuu turhia vääriä hälytyksiä, koska poikkeamat vertailtavaan profiiliin voivat pitää sisällään myös laillisia toimintoja. Liiallisen suuri määrä profiilille annettuja komentoja aiheuttaa sen, että joukossa on myös turhaa dataa ja näin ollen tunkeutujan aiheuttamia poikkeamia ei voida ennustaa oikein ja hälytykset tunkeutumisesta voivat jäädä vähäisiksi.

Muitakin menetelmiä käyttäytymisen pohjalta suoritettavaan tunkeutumisen havaitsemiseen on olemassa. Esimerkkeinä voidaan mainita immuuniin järjestelmään pohjautuva menetelmä (immune systems approach) sekä protokollan varmistus -menetelmä (protocol verification). Immuuniin järjestelmään pohjautuvassa menetelmässä järjestelmä on mallinnettu järjestelmäkutsujen sekvensseistä eri ehtoja varten, kuten normaali käyttäytyminen, virhetilanteet ja tunkeutumisen yritykset. Protokollan varmistus -menetelmässä tarkastellaan protokollien käyttöä annettuja odotuksia vastaan. Odotuksista poikkeavaa protokollien käyttöä kohdellaan epäilyttävänä.

Käyttäytymiseen perustuvan tunkeutumisen havaitsemisen ehdottoman hyvänä puolena voidaan pitää sitä, että se kykenee havaitsemaan poikkeamia normaalista käytöstä ja täten havaitsemaan ennalta tuntemattomia tunkeutumisia järjestelmään. Käyttäytymisestä päätellyn tunkeutumisen heikkoutena voidaan pitää vääriä havaintoja, koska kaiken normaalista poikkeavan liikenteen ja toimintojen oletetaan olevan tunkeutumista järjestelmään. Jos otetaan käyttöön uusi ohjelma, tai jos ärjestelmää tulee käyttämään uusi käyttäjä, voivat ne aiheuttaa turhia hälytyksiä, koska niille ei ole aiempaa käyttöä vastaavaa normaalia käyttöä.

3. Tunkeutumisen havaitsemisjärjestelmät

Tunkeutumisen havaitsemisjärjestelmät toimivat havaitsemismenetelmien pohjalta, mutta ne voidaan jakaa kolmeen eri kategoriaan niiden toimintaympäristön ja -tarkoituksen perusteella. Yksittäisessä järjestelmässä, kuten palvelimissa ja työasemissa, toimivista tunkeutumisen havaitsemisjärjestelmistä käytetään nimitystä Host-based IDS (Intrusion detection system) eli isäntäkonekohtainen tunkeutumisen havaitsemisjärjestelmä, joita aivan ensimmäiset tunkeutumisen havaitsemisjärjestelmät olivat. Tietoverkoissa ja niiden eri segmenteissä toimivia havaitsemisjärjestelmiä kutsutaan termillä Network IDS eli verkkokohtainen tunkeutumisen havaitsemisjärjestelmä. Usein käytetään myös näistä kahdesta muodostettua hybridijärjestelmää, joka tarkkailee tunkeutumisten varalta verkon tasolla järjestelmää sekä yksittäisen järjestelmän tasolla yleisimmin yksittäistä työasemaa.

3.1. Toimintaperiaate

Tunkeutumisen havaitsemisjärjestelmien toiminta voidaan jaotella kolmeen eri osaan, sensoriin, analysoijaan ja manageriin [Endorf, 2004]. Sensoriosa hankkii tietoa lokitiedostoista, verkkoliikenteestä ja järjestelmän käytöstä, ja lähettää niistä eteenpäin tiedot analysoitavaksi. Analysointiosa vastaanottaa sensorin lähettämät tiedot, jotka testataan tunkeutumisen havaitsemismenetelmän mukaan. Näistä tiedoista tunnistetaan onko tunkeutuminen järjestelmään tapahtunut, tai onko se parhaillaan tapahtumassa. Analysointiosa lähettää käsittelyn jälkeen epäilyttävistä tapahtumista raportit manageriosalle, joka päättelee raporttien tietojen pohjalta sopivat toiminnot, tehdäänkö hälytys ja minne kirjataan tiedot tunkeutumisesta. Manageriosa myös hallinnoi muiden osien toiminta virhetilanteiden tapahtuessa.

3.2. Arkkitehtuurit

Tunkeutumisen havaitsemisjärjestelmät toimivat jonkin arkkitehtuurin perusteella, tällaisia ovat monoliittiset, hierarkkiset ja yhteistoiminnalliset arkkitehtuurit [Endorf, 2004]. Monoliittisissa järjestelmissä tunkeutumisen havaitsemisjärjestelmä on yksi ohjelma, joka muodostuu sensorista, analysoijasta ja managerista. Hierarkkisissa järjestelmissä manageriosa, joka tekee tunkeutumisesta päätöksen, sijaitsee hierarkian juuressa tai huipulla. Tapahtumista analysoinnin tekevät osat sijaitsevat välittömästi managerin ala- tai yläpuolella rinnakkain ja jokaisen analysointiosan ala- tai yläpuolella puolestaan ovat tietolähteistä datan hankkivat sensorit. Yhteistoiminnalliseen arkkitehtuuriin perustuvassa mallissa tunkeutumisen havaitsemisjärjestelmät toimivat keskenään laajassa yhteistyössä vaihtaen tietoa keskenään. Muitakin arkkitehtuureja tunkeutumisen havaitsemisjärjestelmissä voidaan käyttää kuten esimerkiksi hajautettuihin ratkaisuihin perustuvia.

3.3. Isäntäkonekohtaiset järjestelmät

Työasemissa ja palvelimissa toimivat tunkeutumisen havaitsemisjärjestelmät ovat pieni osa järjestelmää ja yleensä määriteltävissä palomuureiksi ja agenttipohjaisiksi työkaluiksi. Isäntäkoneissa toimivat järjestelmät tutkivat järjestelmän tapahtumaketjuja ja lokitiedostoja sekä muita käytöstä jääviä jälkiä [Zirkle]. Isäntäkonekohtaiset järjestelmät tarkkailevat työasemasta sisään ja ulospäin kulkevaa liikennettä, järjestelmätiedostojen eheyttä, epäilyttäviä prosesseja sekä käyttäjäoikeuksien muutoksia.

Isäntäkoneen lokitietojen ja tapahtumaketjujen perusteella saadaan kerättyä pohjatieto järjestelmän normaalista käyttäytymisestä. Lokitietoihin vertaamalla voidaan havaita järjestelmään tunkeutuminen sekä selvittää, mistä tunkeutumi-

nen on johtunut. Käyttäjän profiilin mukaiset toiminnot oletetaan normaaleiksi ja kaikki epänormaali käyttäytyminen ja väärinkäyttö, kuten yritys saada laajempi oikeus käyttää järjestelmää, oletetaan tunkeutumiseksi. Verkkoliikenteen epäilyttävä liikenne ulos tai sisäänpäin voidaan olettaa myös tunkeutumiseksi.

Isäntäkonekohtaisten tunkeutumisen havaitsemisjärjestelmien vahvuutena voidaan pitää sen kykyä havaita sisältäpäin tulevat tunkeutumiset. Heikkoina puolina voidaan pitää sitä, että se on asennettava jokaiselle yksittäiselle koneelle, jotta kaikki laitteet olisivat varustettuna tunkeutumisen havaitsemiseen. Myös informaatio, jota lokitietojen perusteella saadaan, ei välttämättä tarjoa kovinkaan tärkeää tietoa mahdollisista tunkeutumisista.

3.4. Verkkokohtaiset järjestelmät

Verkkokohtaiset tunkeutumisen havaitsemisjärjestelmät tarkkailevat joko koko verkon tai sen segmentin verkkoliikenteen paketteja joiden perusteella se pyrkii havaitsemaan esimerkiksi porttiskannereita, palvelunestohyökkäyksiä, murrettuja järjestelmiä ja matoja [Northcutt]. Verkkokohtainen tunkeutumisen havaitsemisjärjestelmä on siis osa verkkoa joko verkkolaitteena, isäntäkoneena tai jonain muuna verkkoresurssina keräten umpimähkäisesti tietoa verkkoliikenteestä. Täten tämä osa toimii tietolähteenä havaitsemisjärjestelmälle. Tietolähteen lisäksi tarvitaan sensori, joka havaitsee poikkeavan ja luvattoman verkkoliikenteen tutkimalla pakettien allekirjoituksia ja vertaamalla niitä tunnettuihin tunkeutumismenetelmiin. Nämä epäilyttävien pakettien tiedot välitetään havaitsemisjärjestelmän ylemmille osille, jotka ratkaisevat, onko kyseessä tunkeutumisen.

Verkkoliikenteestä tunkeutumisen havaitseva järjestelmä on siinä suhteessa vaivalloisempi kuin isäntäkonekohtaiset järjestelmät, että sen asentaminen ei ole yhtä helppoa. Verkkokohtainen tunkeutumisen havaitsemisjärjestelmä voidaan asentaa verkkosegmentin reunoille molemmin puolin palomuuria tai palvelimen tai verkkolaitteen eteen ja taakse.

Verkkokohtaisten tunkeutumisen havaitsemisjärjestelmien etuina ovat niiden pienemmät kustannukset verrattuna isäntäkonekohtaisiin järjestelmiin, koska jokaiselle palvelimelle ja työasemalle ei tarvitse asentaa omaa tunkeutumisen havaitsemisjärjestelmää. Verkkokohtaisten tunkeutumisen havaitsemisjärjestelmien etuna on yleensä myös käyttöjärjestelmäriippumattomuus.

3.5. Samantyyppiset järjestelmät

Tunkeutumisen havaitsemista lähellä olevia tietoturvamenetelmiä ovat hunajapurkit (honeypots) ja tunkeutumisen estämiseen pyrkivät järjestelmät (intrusion prevention).

Hunajapurkkimenetelmässä tunkeutuja pyritään houkuttelemaan hunajapurkin kimppuun. Kun tunkeutuja yrittää tunkeutua järjestelmään, pyritään se ohjaamaan hunajapurkin kimppuun, jotta organisaation toimintajärjestelmät pysyisivät koskemattomina. Tunkeutujan ollessa hunajapurkissa pitäisi se eristää muusta järjestelmästä, että tunkeutuja ei pääse sieltä pois muihin järjestelmiin. Hunajapurkkiin tunkeutumisen perusteella voidaan tutkia, mitä menetelmiä tunkeutuja on käyttänyt. Hunajapurkit voivat toimia monella tavalla, ohjelmana isäntäkoneessa, olla koko isäntäkone tai kokonainen verkko, jossa on useita hunajapurkkeja. Pääasia menetelmässä on kuitenkin, että hunajapurkki vaikuttaa tunkeutujalle mielenkiintoiselta ja tunkeutumisen arvoiselta, ja näin muut tietojärjestelmän osat pysyvät suojassa.

Tunkeutumisen estämiseen pyrkivät menetelmät suojaavat järjestelmiä tunkeutumisia vastaan. Tunkeutumisen estäminen rajoittaa pääsyä järjestelmään kuten palomuuritkin. Nykyisissä palomuureissa on osittain otettu käyttöön tunkeutumisen estämiseen perustuvia menetelmiä. Tunkeutumisen estäminen eroaa siinä palomuureista, että siinä ohjelman sisällön mukaan kontrolloidaan pääsyä järjestelmään, eikä IP-osoitteiden ja porttien mukaan. Samoin kuin tunkeutumisen havaitsemisessa myös tunkeutumisen estämisessä järjestelmät voivat toimia isäntäkone- tai verkkokohtaisina.

4. Yhteenveto

Koska nykyiset tietojärjestelmät ja -verkot eivät ole täysin tietoturvallisia, tarjoavat tunkeutumisen havaitsemisjärjestelmät lisää tietoturvaa, vaikka niiden käyttämissä havaitsemismenetelmissä onkin omat heikkoutensa. Näitä heikkouksia ovat tietämyspohjaisten menetelmien kyky havaita vain tunnettuja tunkeutumismenetelmiä ja käyttäytymiseen pohjautuvien menetelmien antamat väärät hälytykset. Tunkeutumisen havaitsemisjärjestelmät eivät tarjoa suojaa tunkeutumisia vastaan, mutta ne kykenevät tarjoamaan keinon, jolla tunkeutuminen voidaan havaita ja ryhtyä tarvittaviin toimenpiteisiin.

Tunkeutumisen havaitseminen ei yksistään riitä, kun tunkeutuminen on tapahtunut, vaan on oltava lisäksi sovittuna menettelytavat mitä tehdään tunkeutumisen havaitsemisen jälkeen. Tällöin täytyy tietää tunkeutumisen laajuus ja mitä vahinkoja on tapahtunut, onko tietojen eheys menetetty, onko tietoa kadonnut ja miten saadaan tunkeutuja poistettua tietojärjestelmästä ja -verkos-

ta. Näihin kohtiin vastausten saaminen on edellytys tunkeutumisen havaitsemisen onnistumiseen ja jälkitoimenpiteiden hoitoon.

Viiteluettelo

[Anderson, 1980], James P. Anderson, Computer security threat monitoring and surveillance, Technical report, James P. Anderson Company, Fort Washington, (1980).

[Bai & Kobayashi, 2003] Yuebin Bai, Hidetsune Kobayashi, Intrusion detection systems: Technology and development, In: *IEEE Advanced Information Networking and Applications, 2003. AINA 2003. 17th International Conference* (March, 2003), 710-715.

[Debar] Herve Debar, Intrusion detection faq: What is knowledge-based intrusion detection? *Sans IDS faq,* http://www.sans.org/resources/idfaq/knowledge_based.php checked 03.11.2006.

[Denning, 1986] Dorothy E. Denning, An intrusion detection model. In: *IEEE Symposium on Security and Privacy* (1986), 118-131.

[Endorf et al., 2004] Carl Endorf, Jim Mellander, Eugene Schultz, *Intrusion Detection and Prevention*, McGraw-Hill, 2004.

[Innella, 2001] Paul Innella, The evolution of intrusion detection systems, Securityfocus, (Nov, 2001), http://www.securityfocus.com/infocus/1514 checked 26.3.2006.

[Kemmerer & Vigna, 2002] Richard A. Kemmerer and Giovanni Vigna, Intrusion detection: a brief history and overview, *Computer*, **35**, 4, supplement (Apr, 2002), 27-30.

[Kumar, 1995] Sandeep Kumar, Classification and detection of computer intrusions, Ph.D dissertation, Purdue University (Aug, 1995).

[Lunt, 1993] Teresa F. Lunt, Detecting intruders in computer systems, In: *Conference on Auditing and Computer Technology*, (1993).

[Lunt & Garvey, 1991] Teresa F. Lunt, TD Garvey, Model based intrusion detection, *In Proceeding of the 14th National Computer Security Conference* (1991), 372-385.

[Northcutt] Stephen Northcutt, Intrusion Detection FAQ: What is network based intrusion detection? *Sans IDS faq,* http://www.sans.org/resources/idfaq/network_based.php checked 03.11.2006.

[Zirkle] Laurie Zirkle, Intrusion detection faq: What is host-based intrusion detection? *Sans IDS faq,* http://www.sans.org/resources/idfaq/host_based.php. Checked 03.11.2006.

Kielioppipohjainen mallintaminen tietokonegrafiikassa

Samu Ristkari

Tampereen yliopisto

Tiivistelmä Tässä tutkielmassa tarkastellaan polygoniverkkojen tuottamista mallinnuskohteiden käsitteistöä varten sovitetuilla kieliopeilla.

Avainsanat- ja sanonnat: tietokonegrafiikka, L-järjestelmä, proseduraalinen mallintaminen, kielioppi
CR-luokat: F.4.2, I.3.5, I.3.7

1 Johdanto

Tietokonegrafiikassa kolmiulotteiset mallit kuvataan polygoniverkkoina, jotka piirrettäessä kuvioidaan (teksturoidaan) ja varjostetaan. Useimmiten mallit muodostetaan neliö- tai kolmioverkoista. Malleja rakennetaan mallinnusohjelmistoilla, kuten *Blender*illä [1], joissa on monipuolisia työkaluja polygoniverkkojen tuottamiseen ja muokkaamiseen. Blenderissä mallintaja voi rakentaa polygoniverkkoja esimerkiksi

- muokkaamalla ja yhdistämällä geometrisia primitiivejä, kuten kuutioita ja sylintereitä,
- määrittämällä splinikäyrällä polygonin kaksiulotteiselle tasolle, josta polygoni laajennetaan kolmiulotteiseksi,
- laajentamalla kolmiulotteisen käyrän ympärille geometrisen primitiivin, esim. sylinterin.

Mallinnusohjelman työkaluilla mallinnettaessa esimerkiksi puun polygoniverkon mallintaminen voidaan aloittaa sylinteristä. Mallinnusohjelmassa sylinteriä muokataan käytössä olevien työkalujen avulla, kunnes se muistuttaa puun runkoa. Haarautuvat oksat voidaan mallintaa lisäsylintereillä, jotka liitetään edeltävään haaraan kärkipiste- ja polygonitasolla. Työ on suhteellisen suoraviivaista, mutta aikaa kuluu toistuviin työvaiheisiin.

Ihmisvoimin tehtävään mallinnukseen kuluu aikaa, kun tavoitellaan realismia hyvin laajassa ympäristössä. *Superman Returns* -elokuvan kaupunkimaisemien mallintamiseen kului yhteensä 15 miestyövuotta, vaikka niiden tarkkuus ei yllä vastaavien proseduraalisesti tuotettujen maisemien tasolle [10].

Proseduraalisesti mallinnettaessa puu luodaan parametrisoidun mallin pohjalta. Mallintajan tehtäväksi jää sopivien parametrien valinta. Parametreja voivat olla esimerkiksi haarautumistiheys, oksien määrä haarautumista kohti ja oksien kapeutumisnopeus. Mallin suoritus annetuilla parametreilla antaa tuloksena polygoniverkon, jota voidaan tarvittaessa muokata.

Kasvillisuuden ja arkkitehtuurin kuvaamiseen on luotu kielioppipohjaisia ratkaisuja, joita voidaan käyttää proseduraalisina malleina.

2 Muodolliset kielet

2.1 Chomskyn kieliopit

Kieliopin G määrittelemä eli tuottama kieli L koostuu niistä sanoista, jotka voidaan johtaa lähtemällä alkumerkistä S ja soveltamalla sääntöjoukon R sääntöjä, kunnes saadaan jono, jossa on pelkkiä päätemerkkejä. Kieliopin G päätemerkit vastaavat kielen L aakkosia [9].

Määritellään kieliopin G sääntöjoukko R seuraavasti:

$$A \rightarrow Ba$$
$$B \rightarrow b.$$

Symbolit A ja B ovat välikemerkkejä ja b on päätemerkki. Kieliopilla G voidaan tuottaa esimerkiksi seuraavanlaiset merkkijonot, kun aloitetaan aksioomamerkkijonosta $ABBA$:

$$ABBA$$
$$BaBBA$$
$$baBBA$$
$$babBA$$
$$\ldots$$
$$babbba.$$

Välikemerkki voidaan poistaa säännöllä, jossa välikemerkistä seuraa tyhjä merkki ϵ.

2.2 L-järjestelmä

L-järjestelmä on tuottavien kielioppien muunnos, joka perustuu merkkijonojen rinnakkaiseen uudelleenkirjoitukseen. L-järjestelmän sääntöjen soveltaminen aloitetaan merkkijonosta, jota kutsutaan aksioomaksi. Merkkijono uudelleenkirjoitetaan korvaamalla sääntöjen edeltäjät niiden seuraajilla. Olkoon L-järjestelmässä G merkit a ja b sekä uudelleenkirjoitussäännöt

$$a \rightarrow ab$$
$$b \rightarrow ba.$$

Alla on lueteltu G:n neljä ensimmäistä uudelleenkirjoituskierrosta alkaen aksioomasta a:

$$a$$
$$ab$$
$$abba$$
$$abbabaab$$
$$abbabaabbaababba.$$

Jokaisella kierroksella edeltäjien a ja b kaikki esiintymät on korvattu niiden sääntöjen mukaisilla seuraajillaan. Aristid Lindenmayer kehitti L-järjestelmän biologisen kasvun mallintamiseen [12], joten merkkijonon pituus kasvaa nopeasti, vaikka järjestelmässä on vain kaksi sääntöä. Kasvien tai fraktaalien mallintamisessa tämä on tarkoituksenmukaista, mutta ei välttämättä muissa L-järjestelmien sovelluksissa, kuten rakennusten mallintamisessa [10].

3 Kasvillisuuden mallintaminen

L-järjestelmää voidaan käyttää niin puiden kuin muunkin kasvillisuuden mallintamiseen. Prusinkiewicz ja Lindenmayer kuitenkin suosittelevat L-järjestelmän käyttämistä lähinnä ruohomaisten kasvien mallintamiseen, joissa L-järjestelmän käyttämisestä on merkittäviä etuja [12]:

- Erilaisten kehitysvaiheiden toistuminen kasvin eri osissa on mallinnettavissa.
- Samoilla säännöillä voidaan mallintaa kasvia sen eri kasvuvaiheissa.

Pienillä L-järjestelmän sääntöjoukoilla voidaan tuottaa monimutkaisia rakenteita. Sääntöjen määrittely ei kuitenkaan ole triviaali tehtävä [12].

Kielioppien tuottamia merkkijonoja voidaan tulkita käskyinä tasolla kulkevalle kilpikonnalle. Kilpikonna piirtää viivaa ja tekee käännöksiä, kun merkeille valitaan järjestelmän ulkopuolella sopivat tulkinnat. Pinon avulla voidaan mallintaa esim. haarautumista kasvien varsissa. Haarojen pituuden ja paksuuden muutosta voidaan mallintaa laajentamalla L-järjestelmän kielioppia.

Parametrisoitujen sääntöjen seuraajissa voidaan tehdä artitmeettisiä laskutoimituksia. Kontekstisessa L-järjestelmässä sääntö valitaan ainoastaan silloin, kun sääntömerkin vasemmalla ja oikealla puolella on vaaditut merkit tai merkkijonot. Stokastisessa L-järjestelmässä säännön seuraaja valitaan satunnaisesti seuraajavaihtoehtojen joukosta. Satunnaisuuden ongelmana on kuitenkin vähäinen kontrolli lopputulokseen, joten säännöissä on mahdollista käyttää konditionaalisia lausekkeita.

Kasvun interaktiota ympäristön kanssa voidaan mallintaa kyselysäännöillä. Esimerkiksi kaksiulotteisella tasolla liikkuvalle kilpikonnalle tehty kysely $?P(x, y)$ korvaa parametrit x ja y kilpikonnan sijaintikoordinaateilla. Kyselyillä voidaan selvittää esimerkiksi kasvin jossakin pisteessä saaman valon määrän, jolloin kasvin muiden osien ja muun kasvillisuuden aiheuttama varjostus voidaan ottaa huomioon [4].

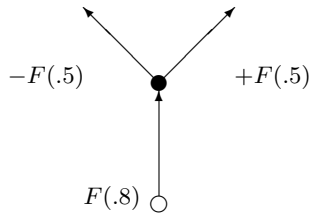
3.1 Kaksiulotteinen graafinen tulkinta

Otetaan esimerkiksi L-järjestelmä G, jolla on seuraava parametrisoitu sääntö:

$$F(x) \rightarrow F(.8x)[-F(.5x)][+F(.5x)].$$

Olkoon järjestelmän G aksiooma $F(1.0)$. Ensimmäinen uudelleenkirjoitus tuottaa seuraavan merkkijonon:

Kuva 1. L-järjestelmän tuottaman merkkijonon graafinen tulkinta.

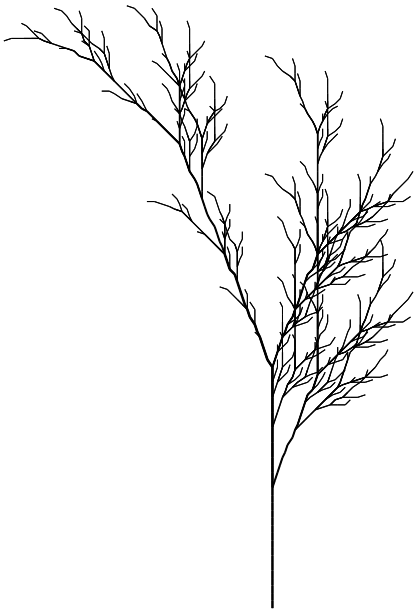


$$F(0.8)[-F(0.5)][+F(.5)].$$

Järjestelmän G tuottamalle merkkijonolle on järjestelmän ulkopuolella graafinen tulkinta, joka on esitetty Kuvassa 1. Merkkijonoa luetaan vasemmalta oikealle. Kilpikonna aloittaa matkansa kuvan alaosasta, joka on merkitty avoimella ympyrällä. Kilpikonnan etenemissuunnaksi on alustettu $\boldsymbol{S} = (0, 1)$.

$F(x)$ merkitsee x:n mittaisen viivan piirtämistä kilpikonnan etenemissuuntaan $\boldsymbol{S}$. Merkki $-$ tulkitaan kilpikonnan 45° käännöksenä vasempaan ja $+$ vastaavasti oikeaan. Vektorilla $\boldsymbol{P}$ merkitään kilpikonnan sijaintia. Suunta ja sijainti muodostavat yhdessä kilpikonnan tilan, joka tallennetaan pinoon merkillä [ja palautetaan merkillä].

Kuva 2. L-järjestelmän tuottaman merkkijonon graafinen tulkinta, joka muistuttaa kasvin vartta.



Otetaan laajemmaksi esimerkiksi merkkijonon tulkinnasta esimerkkisääntöjoukko [12], joka on muokattu parametrisoiduksi. L-järjestelmällä G' on seuraavat kaksi sääntöä:

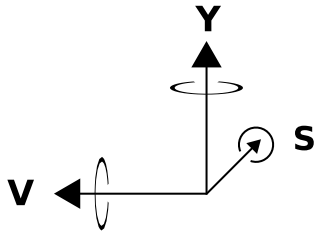
$$\begin{aligned} S(x) &\rightarrow F(x) - [[S(x * 0.8)] + S(x * 0.8)] + F(x) \ldots \\ &\quad [+F(x)S(x * 0.8)] - S(x * 0.8) \\ F(x) &\rightarrow F(x)F(x). \end{aligned}$$

G':n tuottaman merkkijonon graafinen tulkinta on Kuvassa 2. Tulkittu merkkijono on saatu viidellä uudelleenkirjoituskierroksen jälkeen, kun aksioomana käytettiin merkkijonoa $S(1.0)$. Merkin $F(x)$ parametri x:n arvo tulkittiin piirtoviivan paksuutena. Merkki $-$ tulkittiin kilpikonnan 22.5° käännöksenä vasempaan ja $+$ vastaavasti oikeaan.

3.2 Kolmiulotteinen graafinen tulkinta

Prusinkiewic ja Lindenmayer mallintavat kilpikonnan liikkumista kolmiulotteisessa avaruudessa Kuvan 3 tapaan kolmella suuntavektorilla [12]. $\boldsymbol{S}$ merkitsee kilpikonnan etenemissuuntaa, $\boldsymbol{V}$ osoittaa katseesta vasemmalle ja $\boldsymbol{Y}$ ylöspäin siten, että $\boldsymbol{S} \times \boldsymbol{V} = \boldsymbol{Y}$. Kilpikonnan suunta voidaan siten merkitä suuntamatriisilla $[\boldsymbol{S}\ \boldsymbol{Y}\ \boldsymbol{V}]$.

Kuva 3. Kilpikonnan määrittely 3D-avaruudessa. S on kilpikonnan katseen suunta, Y osoittaa ylöspäin ja V vasemmalle.

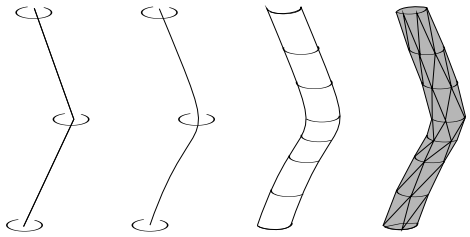


Prusinkiewiczin ja Lindenmayerin L-järjestelmäsovelluksien säännöissä on monipuolinen valikoima merkkejä, jotka järjestelmän ulkopuolella tulkitaan piirtokomentoina. Merkeillä voidaan määritellä konvekseja polygoneja ja splinikäyrien määräämiä lappuja (patches). Piirtokomentojen sijasta voidaan käyttää esim. mallinnusohjelmilla luotuja polygoniverkkoja [12].

Bloomenthal tuottaa vaahteran kolmiulotteisen geometrian haarautumiskohtia lukuunottamatta yleistetyillä sylintereillä, jotka ovat splinikäyrän ympärille laajennettuja sylintereitä. Bloomenthalin tekniikassa puun runko on määritelty C2-jatkuvana splinikäyränä. Käyrä on C2-jatkuva, kun se on kahdesti derivoituva. Tällöin käyrän ympärille voidaan muodostaa Frenet-kehys kaikissa käyrän kulkemissa pisteissä [2].

Kuvan 4 esimerkissä on seuraavat vaiheet:

Kuva 4. Esimerkki puun rungon polygonisoinnista. Vasemmalta oikealle: reittipisteet, käyrä, segmentointi ja kolmioverkko.



1. Määritetään reittipisteet, joiden kautta rungon muodostava käyrä kulkee. Ympyrät havainnollistavat puun rungon paksuutta reittipisteiden kohdalla.
2. Funktio $r(t)$ määrittää käyrän, joka kulkee reittipisteiden kautta.
3. Käyrä $r(t)$ jaetaan segmentteihin, joiden välille kolmiot muodostetaan. Jakopisteissä lasketaan Frenet-kehys, jossa $\boldsymbol{S}$ on käyrän tangentti ja $\boldsymbol{V}$, $\boldsymbol{Y}$ ovat sen normaalit (Kuva 3). Kolmioiden kärkipisteet jaetaan vektoreiden $\boldsymbol{V}$, $\boldsymbol{Y}$ muodostamalle tasolle määritellyn ympyrän kehältä.
4. Kärkipisteet on yhdistetty kolmioverkoksi.

3.3 Reaaliaikaiset sovellukset

Kasvien ja puiden mallintaminen ja piirtäminen vaatii suurten polygonimäärien tuottamista ja prosessointia. Geometriaa asteittaisesti yksinkertaistamalla voidaan vähentää piirrettävien polygonien määrää. Puusta voidaan piirtää katseluetäisyydestä riippuen vähiten polygoneja sisältävä versio. Geometrian yksinkertaistamiseen pohjautuvat menetelmät eivät kuitenkaan säilytä puun rakennetta [8].

Lluchin ja muiden menetelmässä puuta mallintavan L-järjestelmän tuottaman merkkijonon haarautumiskohdille lasketaan pituusmitat, jotka tallennetaan haarautumiskohtien mukaiseen puutietorakenteeseen [8]. Eri resoluutioiden versioiden laskminen aloitetaan käymällä puutietorakenetta läpi juurisolmusta alkaen aina siihen lehtisolmuun asti, johon johtavat pisimmät haaraumat. Reitti tallennetaan resoluutioversioksi ja matkalla olevat solmut merkitään käydyiksi. Myöhempien reittien juurisolmuina käytetään juurta lähimpänä olevaa solmua, jolla on pisin käymätön haarauma.

Piirrettäessä haluttu tarkkuus saadaan piirtämällä riittävä määrä resoluutioversioita alkaen ensimmäisestä tallennetusta reitistä.

4 Virtuaalikaupungit

4.1 CityEngine

Parish ja Müller ovat soveltaneet L-järjestelmää virtuaalikaupungin mallintamiseen [11]. *CityEngine* rakentaa kokonaisen virtuaalikaupungin 2D-bittikarttasyötteen pohjalta. Virtuaalikaupungin maantieteellisiä ominaisuuksia voidaan

kontrolloida annetuilla korkeus-, vesi-, maa- ja kasvillisuuskarttasyötteillä. Yhteiskunnallisia piirteitä voidaan kontrolloida ja rajoittaa kartoilla väestöntiheydestä, asuin- ja liikealueista, tietyypeistä ja rakennuskorkeudesta.

Virtuaaliaupungin geometrian rakentaminen aloitetaan tuottamalla tieverkkograafi, joka polygonisoidaan kortteleiksi. Stokastisesti tuotetut rakennukset sijoitetaan kortteleihin jaetuille tonteille.

Tieverkkograafi luodaan L-järjestelmällä, jonka yhdeksän sääntöä toimivat tuottavina lähtökohtina tieverkon rakentamiselle. Tienpätkien rakentamisen suuntaa ohjaavat tietyyppi, ympäristön paikalliset ja globaalit rajoitteet ja haarautumisen hidastuslaskuri.

Tieverkkoa tuottava algoritmilla on seuraavat säännöt välikemerkkejä R ja B varten:

- Tiemerkin $R(laskuri, ohjeet, ominaisuudet)$ kohdalla kysytään merkin oikealla puolella olevalta ympäristökyselymerkiltä $?I(ominaisuudet, tila)$, voidaanko tietä jatkaa.
  1. Kyselyn onnistuessa R korvataan kuudella merkillä, joiden parametrien arvot laskevat järjestelmän ulkopuolella olevat rajoitusfunktiot.
     - Suunnanmuutos $+(kulma)$
     - Askel eteenpäin $F(pituus)$
     - Kaksi uutta tienhaaraa $B(laskuri, ohjeet, ominaisuudet)$
     - Päähaaraa jatkavat R ja I.
  2. R ja I poistetaan, jos ympäristökysely epäonnistuu tai haarautumislaskurin arvo on < 0.
- Haaramerkin B kohdalla tarkistetaan hidastuslaskuriparametrin arvo.
  1. > 0, vähennä laskurin arvoa yhdellä.
  2. $= 0$, tallenna tila pinoon ja aloita uusi tie korvauksella $[RI]$.
  3. < 0, poista haara.

Moottoriteiden rakentamisen suuntaa ohjaa väestöntiheyskartta, joka on globaali rajoitin. Tienpätkän alkupisteestä otetaan väestöntiheysotoksia tienpätkän suunnan ja pituuden perusteella. Alkupiste, suunnan ympäristö ja pituus muodostavat kaksiulotteisia säteiden joukon. Säteiden hyvyys ratkaistaan laskemalla jokaisen säteen kulkemalta matkalta väestöntiheyden summa, jota tasapainotetaan suhteuttamalla otokset etäisyyteen alkupisteestä.

Tavalliset kadut noudattavat ennalta määrättyjä asemakaavaohjeita. Perusohjeen mukaan rakennettavat tiet noudattavat moottoriteiden tapaan väestöntiheyttä, jonka seurauksena katuverkot muistuttavat vanhoja kaupunkeja, joissa ei ole ollut asemakaavaa. Ruutukaavaa noudattava katuverkko muistuttaa New Yorkin asemakaavaa. Sädekaavaa noudattava katuverkko on matkittu Pariisin asemakaavasta. San Francisco -mallissa katu rakennetaan siihen suuntaan, jossa on vaihtoehdoista pienin korkeusvaihtelu.

Paikallisilla rajoittimilla tarkennetaan tienpätkän rakentamisen alkuperäistä suuntaa, jonka globaalit rajoittimet ovat antaneet. Teiden on kuljettava paikallisessa ympäristössään laillisiksi luokitelluilla alueilla. Tienpätkien on laillista kulkea maan päällä ja ne saavat poikkeustapauksissa ylittää vesialueita. Tieverkossa jo olevien teiden ylittäminen ei ole laillista.

Teillä on seuraavat rajoitussäännöt:

1. Jos tienpätkän päätepiste ei ole lailliseksi luokitellua alueella, niin tienpätkä on sopeutettava tilanteeseen.
   (a) Tienpätkän pituutta vähennetään ennalta määrättyyn pituuteen asti.
   (b) Tienpätkän suuntaa muutetaan ennalta määrätyn säteen sisällä.
2. Moottoritieksi luokitellut tienpätkät saavat ylittää laittomiksi luokiteltuja alueita, kuten merenlahtia ja jokia, ennalta määriteltyyn pituteen asti.
3. Tienpätkän leikatessa toisen tienpätkän ylitse niiden leikkauspisteeseen muodostetaan risteys.
4. Tienpätkän päätepisteen ja jonkin toisen tienpätkän etäisyyden alittaessa ennalta määritellyn kynnysarvon on luotava niitä yhdistävä tienpätkä.

Tieverkkograafi polygonisoidaan ja samalla risteyksien teräviä kulmia pehmennetään. Teiden rajaamat polygonit muodostavat kortteleita, jotka jaetaan rakennusten tonttipolygoneiksi. Kortteliin jää sisäpiha, jos korttelin keskelle on jaettu tonttipolygoni, joka ei jaa yhtään sivua tien kanssa.

4.2 Reaaliaikainen sovellus

Greuter ja muut ovat ottaneet virtuaalikaupungin mallintamisen tavoitteeksi reaaliaikaisen piirtonopeuden ja välivaihetiedon tallentamisen minimoinnin [5]. Virtuaalikaupungin geometriasta luodaan kerrallaan vain katselupisteestä näkyvissä oleva osa.

Pseudoloputon kaupunki matkii Manhattanin ruutuasemakaava. Tonttiruuduille lasketaan tiivistysfunktiolla siemenluku, jota käytetään tontin sisältämän rakennuksen ominaisuuksien määrittelyyn. Siemenluvun deterministisyyden ansiosta rakennusten geometriaa ei tarvitse tallentaa, vaan se voidaan aina tarvittaessa laskea uudelleen. Käytännön toteutuksessa rakennusten geometrian tallentamiseen on kuitenkin käytetty välimuistitietorakennetta.

5 Arkkitehtuurin mallintaminen

5.1 Konstruktiivinen kielioppi

Muinainen kiinalainen arkkitehtuuri noudattaa orjallisesti kaanonia, jossa rakennuksen pohjapiirroksen määrää shakkilautakuvioitu suorakulmio. Perinteistä rakennusoppia on siten helppo matkia konstruktiivisella kieliopilla [7], joka tuottaa kaanonia noudattavia rakennuksia.

Rakennus voidaan tuottaa valitsemalla ainoastaan suorakulmion vaaka- ja pystyruutujen (*kaijian*) lukumäärät. Rakennuksen muut ominaisuudet määräytyvät stokastisilla säännöillä. Rakentaminen alkaa rakennuksen korkean tason ominaisuuksien määrittelystä ja vähitellen laskeutuu yksittäisten elementtien muokkaukseen.

Pohjapiirrosruudukolle valitaan yksi neljästä pylväskuviosta, jotka ovat rakennuksen kantavat rakenteet. Seinäruudut valitaan pylväskuvion perusteella.

Kattovaihtoehtoja on kuusi erilaista. Rakennuksen perustuksille ja pilareille on johdettavissa useita erilaisia ulkoasuja sääntöjen soveltamisesta riippuen.

Esimerkiksi pilareiden rakentaminen aloitetaan lattiatasolta rajauslaatikoita pinoamalla. Pilarin jalustan välikemerkki varaa rajauslaatikolla tarvitsemansa tilan. Stokastisten sääntöjen soveltamisesta riippuen jalustan päälle asetetaan tyyny- ja pilarivälikemerkit, jotka varaavat omat rajauslaatikkonsa. Lopulta välikemerkit korvataan päätemerkeillä, jotka tulkitaan rajauslaatikkonsa täyttävinä primitiiveinä tai kolmioverkkoina.

5.2 Jakavat kieliopit

CityEnginellä rakennetun virtuaalikaupungin rakennuksten geometria tulkitaan stokastisen ja parametrisoidun L-järjestelmän tuottamasta merkkijonosta. Globaalit rajoittimet asettavat rakennuksen korkeuden ja käyttötarkoituksen määrittävät parametrit. Tontista riippuva järjestelmän aksiooma on rakennuksen pohjapiirroksesta laajennettu rajauslaatikko [11].

Järjestelmän sääntöinä ja päätemerkkeinä on tarjolla monenlaisia laajennuksen muokkausoperaatioita. Muokattavan laatikon piirteet tarkentuvat jokaisella uudelleenkirjoituksella lähemmäs rakennuksen julkisivua. Uudelleenkirjoituksen välivaiheita on mahdollista hyödyntää piirtovaiheessa. Rakennusta esittämään voidaan valita katseluetäisyyden mukaan vähiten polygoneja sisältävä polygoniverkko.

Wonka et al. ovat laajentaneet rajauslaatikkotekniikan ideaa. Jakavassa kieliopissa (split grammar) aksioomaprimitiivi johdetaan loogisia rakennuselementtejä kuvaavilla säännöillä satunnaiseen lopputulokseen. Aksioomaprimitiiveinä voidaan käyttää esim. sylintereitä ja laatikoita. Primitiivien kolmiulotteisella jakamisella voidaan saavuttaa erikoisia rakenteita, kuten esimerkiksi ovenkarmeja muistuttavia kerrostaloja.

Järjestelmän satunnaisuutta kontrolloidaan kontrollikieliopilla, joka huolehtii elementtien ominaisuuksien yhtenäisyydestä. Ominaisuudet vaikuttavat järjestelmän tuottaman merkkijonon lopulliseen geometriseen tulkintaan.

Esimerkiksi kerrostalorakennus voidaan aloittaa rajauslaatikosta, jonka julkisivut jaetaan vertikaalisesti kerrosten ja saumakohtien joukoksi. Kerrosjoukon suorakulmiot jaetaan edelleen horisontaalisesti ikkuna- ja ovipaloihin. Kontrollikielioppilla voidaan määrätä rakennuksen ikkunapuitteiden ominaisuudeksi puumateriaali.

5.3 CGA shape

CityEngineen integroitu *CGA shape* on jatkoa Wonkan ja muiden kehittämälle jakavalle kieliopille [13], jota Müller ja muut nimittävät joukkokieliopiksi (set grammar) [10]. Kuvauskielessä on piirteitä L-järjestelmistä ja Chomskyn kieliopeista. Sääntöjen merkintätapa ja laajennukset lainaavat L-järjestelmistä. Säännöt ovat muotoa [10]

$$id : predecessor : cond \rightarrow successor : prob.$$

Säännöillä on yksilöllinen nimi *id*, edeltäjä *predecessor*, soveltamisehto *cond*, seuraaja *successor* ja soveltamistodennäköisyys *prob*. Sääntöjen soveltaminen aloitetaan aksioomajoukosta A, josta valitaan välikemerkki $B \in A$. Järjestelmän sääntöjoukosta etsitään seuraaja B', jolla on edeltäjä B. Nyt B voidaan korvata aksioomajoukossa joukolla B'. Menettelyä jatketaan niin kauan kuin aksioomajoukossa on pelkkiä päätemerkkejä.

Mallintaminen tapahtuu kolmi- ja kaksiulotteisesti geometristen muotojen pohjalta. Muotoja voivat esittää niin välikemerkit kuin päätemerkit. Muotojen rajoitelaatikon määräävien ominaisuuksien joukossa ovat vähintään sijainti P, koordinaattijärjestelmän ja orientaation ilmaisevat vektorit X, Y ja Z ja koon ilmaiseva vektori S.

Sääntöjen seuraajissa voidaan käyttää L-järjestelmille tyypillisiä haarautumisrakenteita. Parametreina voidaan käyttää edeltäjän ominaisuuksia, kuten sijaintia ja kokoa. Jakavien kielioppien tapaan säännön edeltäjän kolmiulotteinen muoto voidaan jakaa osiin akselin suhteen funktioilla *Subdiv* ja *Repeat*. Funktiolla *Comp* voidaan jakaa säännön edeltäjä osiin esimerkiksi särmien ja tahokkaiden perusteella.

Esimerkiksi kolmiulotteinen sylinteri, jonka pitkittäissivut koostuvat kahdeksasta tahokkaasta, voidaan palauttaa kaksiulotteisella tasolla muokkausta varten funktion $Comp(menettely)\{merkit\}$ avulla. Esimerkissä sääntö on muotoa

$$s \rightarrow Comp(''sidefaces'')\{julkisivu\},$$

jolloin kahdeksan sivutahkoa jaetaan julkisivumerkeiksi. Julkisivua voidaan muokata eteenpäin jakavien kielioppien tapaan. Kaksiulotteisesta tasosta voidaan palata takaisin kolmiulotteiseen mallintamiseen esim. laajentamisen avulla.

Sovellettaessa on mahdollista tehdä ympäristölle näkyvyys- ja kohdistyskyselyjä. Näkyvyyskyselyillä voidaan varmistaa, ettei esimerkiksi näkymättömiin jääneelle seinälle aseteta ikkunoita ja ovea. Kohdistyskyselyllä (*Snapping*) voidaan määrätä taso tai "kohdistuslanka", joihin lähelle osuvien rajoitelaatikoiden ja muun geometrian on loksahdettava kiinni. Kohdistamista käyttämällä monimutkaisissakin rakennuksissa esim. kerrokset pysyvät samalla korkeudella.

6 Mallintaminen käytännössä

Müllerilla ja muilla on jokseenkin positiivisia kokemuksia arkkitehtuurikielen käytöstä [10]. Ammatikseen mallintava testihenkilö oli muutamassa päivässä onnistunut luomaan pienen kaupungin annettujen esimerkkisääntöjen pohjalta. Arkkitehtuurikieltä voisi siis teoriassa käyttää mallinnusohjelmistojen skriptikielten sijaan, mutta käyttäjän olisi kuitenkin hyvä tuntea tietojenkäsittelytieteen perusteita.

Prusinkiewicz ja Lindenmayer [12] ja Prusinkiewicz ja muut [4] määrittelevät kasvien mallit biologisesti eksaktisti, mutta jättävät samalla biologiaa tuntemattomat kylmiksi. L-järjestelmät määrittelevät kasvit paikallisilla kasvusäännöillä,

joka saattaa olla biologeille intuitiivista, mutta mallintajien näkökulma on kiinnostuneempi kasvin ominaisuuksista kokonaisuutena [6]. Deussen ja Lintermann ehdottavat vaihtoehtoiseksi mallinnustekniikaksi kasvien kuvausta kieliopin ja graafin yhdistelmänä. Tällöin kasveja muokataan graafisella käyttöliittymällä, jossa mallintaja saa välitöntä palautetta parametrien muutoksista [6].

Boudon ja muut ovat myös törmänneet kasvien mallinnusongelmaan [3] ja jatkaneet samoilla linjoilla kuin Deussen ja Lintermann. Menetelmässä L-järjestelmä säilyy tuottajan roolissa. Järjestelmän tuottama kasvi pilkotaan haarautumiskohdistaan puutietorakenteeseen, jossa mallintaja voi muokata haaran sisältämien merkkien parametreja. Kun järjestelmä tuotetaan uudelleen, muokatut parametrit noudetaan puutietorakenteesta. Haittapuolena ongelmia ilmenee, jos tuottavan L-järjestelmän parametreja muokataan siten, että kasvin topologia muuttuu. Tällöin muokatut parametrit saattavat osua väärille haaroille.

Viitteet

1. Blender Foundation, 3D Software Package, 2006. URL: http://blender3d.org/.
2. Jules Bloomenthal, Modeling the mighty maple. *SIGGRAPH Comput. Graph.* **19**, 3, (1985), 305–311.
3. Frederic Boudon, Przemyslaw Prusinkiewicz, Pavol Federl, Christophe Godin and Radoslaw Karwowski, Interactive design of bonsai tree models. *Computer Graphics Forum* **22**, 2003, 591–599.
4. Pavol Federl, Radoslaw Karwowski, Radomir Mech and Przemyslaw Prusinkiewicz, L-systems and beyond. *SIGGRAPH 2003 Course Notes*, 2003, Przemyslaw Prusinkiewicz, Ed.
5. Stefan Greuter, Jeremy Parker, Nigel Stewart and Geoff Leach, Real-time procedural generation of 'pseudo infinite' cities. In: *GRAPHITE '03: Proceedings of the 1st International Conference on Computer graphics and Interactive Techniques in Australasia and South East Asia*, ACM Press, 2003, 87–ff.
6. Bernd Lintermann and Oliver Deussen, Interactive modeling of plants. *IEEE Comput. Graph. Appl.* **19**, 1, (1999), 56–65.
7. Hua Liu, Qing Wang, Wei Hua, Dong Zhou and Hujun Bao1, Building Chinese Ancient Architectures in Seconds. *Lecture Notes in Computer Science* **3515**, Springer, 2005, 248–255.
8. Javier Lluch, Emilio Camahort and Roberto Vivo, Procedural multiresolution for plant and tree rendering. In *AFRIGRAPH '03: Proceedings of the 2nd International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, ACM Press, 2003, 31–38.
9. Jorma Merikoski, Ari Virtanen ja Pertti Koivisto, *Johdatus diskreettiin matematiikkaan.* WSOY, 2004.
10. Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer and Luc Van Gool, Procedural modeling of buildings. *ACM Trans. Graph.* **25**, 3, (2006), 614–623.
11. Yoav I. H. Parish and Pascal Müller, Procedural modeling of cities. In *SIGGRAPH '01: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press, 2001, 301–308.
12. Przemyslaw Prusinkiewicz and Aristid Lindenmayer, *The algorithmic beauty of plants.* Springer, 1996.
13. Peter Wonka, Michael Wimmer, Francois Sillion and William Ribarsky, Instant architecture. *ACM Trans. Graph.* **22**, 3, (2003), 669–677.

On Face Recognition Algorithms

Outi Räihä

Abstract

This study introduces the concept of face recognition, and implores face recognition algorithms in four main categories: PCA, LDA, Kernel implemented with PCA, LDA and ICA techniques, and 3D face recognition. Some other techniques are also briefly introduced. The study is based on a literary survey. Conclusions on different algorithms are drawn.

Key words and terms: Face recognition, principal component analysis, eigenface, linear discriminant analysis, Kernel, independent component analysis, 3D face recognition.
CR classes: I.5.4, I.4.7, F.2

1. Introduction

Face recognition has become a more and more popular research area over the past few years. Recognition algorithms have developed from basic 2D recognition based on still images to recognizing faces from 3D images and moving video, and the technology is developing concurrently as new methods are published almost daily [Ding and Fang, 2004].

To humans face recognition is relatively easy; observations of a face are compared to those found in each individual's memory, and the conclusions are made. People are able to make these comparisons regardless of the viewpoint, pose, illumination, expression and the time passed since the last observation [Ding and Fang, 2004], which are exactly the variables that are causing the most trouble for computers.

Computers need to be taught what people are able to analyze intuitively. A face recognition algorithm can only be made to recognize as small particles of the face as possible, to find some pinpoints in every possible lighting, to find the common features of the same face in different expressions, and to recognise the same face again after a significant period of time in between.

The recognition of a face begins with extracting the discriminative features. The fundamental idea in recognition is to make a person's face features as close to each other as possible, but as far away as possible to those of a different person [Ding and Fang, 2004]. To try to eliminate the effects of lighting and other

changing factors on the face, the image goes through a set of normalizing operations.

Face recognition algorithms are largely based on vector representations of the image. The very fundamental algorithms in face recognition are based on principal component analysis (PCA), a statistical method, and linear discriminant analysis (LDA). A third member in the same family is independent component analysis (ICA). These methods can be made more efficient by applying some other technique in addition to them, and using a hybrid algorithm to get the most accurate and efficient result. A wide range of completely new algorithms has also emerged from the vast research done during the past years.

This paper introduces the most common methods based on PCA and LDA, and how these can be used with applications of the kernel method. These are compared to ICA using the kernel method. Since 3D recognition is the new trend in face recognition, 3D recognition with PCA will also be introduced, as well as two other techniques applied with 3D face recognition. A set of alternative methods, such as the hidden Markov model, elastic bunch graph matching and the Bayesian model with Gabor features are also briefly described. Finally, conclusions on the different techniques are drawn.

2. Principal Component Analysis

2.1. Principal component analysis in general

Principal component analysis, or PCA, is considered to be the very foundation of face recognition. PCA itself is a statistical method for reducing the dimensionality of a data set while retaining the majority of the variation present in the data set [Moon and Phillips, 2000]; thus, PCA is only a building block to be used with the actual face recognition algorithm.

Moon and Phillips [2000] introduce a generic modular PCA-based face recognition system which consists of three modules: normalization, PCA projection and recognition. With this system it is intended to avoid the problem of insufficiently explained design decisions regarding the algorithm which is common with PCA-based methods.

PCA-based face recognition algorithms are first given a training set, $\boldsymbol{t}_1,\ldots,\boldsymbol{t}_N$ of N facial images such that the ensemble mean of the training set is zero ($\sum_i t_i = 0$) [Moon and Phillips, 2000]. Each image of the size of n by m pixels is then considered to be a point in $\mathbf{R}^{nxm}$ to compute the PCA representation. PCA then finds the optimal linear least-square representation in (N-1)-dimensional space while the representation preserves variance. The PCA representation consists of a set of N-1 eigenvectors ($\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{N-1}$) and eigenvalues ($\lambda_1, \ldots, \lambda_{N-1}$)

[Moon and Phillips, 2000]. In Moon and Phillips' presentation the eigenvectors are normalized so that they are orthonormal, and ordered in a way that $\lambda_i > \lambda_{i+1}$. Thus the λ_i's are equal to the variance of the projection of the training set onto the ith eigenvector. Thus, the smaller the index of the eigenvector, the larger the variation that it encodes in the training set.

Faces are represented by their projection onto a subset of $M \leq N$-1 eigenvectors, which is called a *face space*. K facial images are represented as K points $\{\boldsymbol{g}_1, ..., \boldsymbol{g}_K\}$ in face space [Moon and Phillips, 2000]. An image can then be recognized by first projecting it into face space and then comparing the projection to those that are already known. The recognition is decided on a similarity measure, two possible similarity measures being the Euclidean and the L_1 distances between the test image and the known one. The known image which gives the minimum of all similarity measures is decided to be the one that is desired [Moon and Phillips, 2000].

The generic system of Moon and Phillips [2000] consisted of three modules as stated before. The first module is for normalizing the input image. Here the facial image is transformed into a standard format so that all the necessary information can be gathered for the recognition. The second module, PCA projection, gathers the eigenvectors that are needed, and the third module compares the eigenvectors in face space.

This generic system well describes the steps that are needed in order to recognise a face with the use of PCA. It takes full advantage of the principles of PCA techniques while still leaving room for different algorithmic designs for normalizing and recognizing the face.

2.2. Eigenfaces

In face recognition the eigenvectors described in Section 2.1 are often referred to as *eigenfaces*, and the foundation of face recognition algorithms is considered to be the eigenface technique. As Turk and Pentland [1991a] state: "The eigenvectors can be thought of as a set of features which together characterize the variation between faces." Every point of the image contributes to the eigenvector in some way, so the eigenvector can be displayed as a kind of ghostly face, called the eigenface.

Where Moon and Phillips described a generalized view of using PCA as a base for face recognition, Turk and Pentland give a specific algorithm designed to use PCA that describes the basic ideas of the eigenface technique.

Turk and Pentland [1991a, 1991b] divide the procedure of recognising a face with eigenfaces into four steps. The first step is the initialisation, which is the step using PCA. Here the eigenvectors are calculated from the training set and a face space is formed just as in the second module of the model by Moon

and Phillips [2000]. The second step is to calculate a set of weights based on the input image and the M eigenfaces by projection. The third step is determining whether the image is a face at all, and the fourth step is to classify the weight pattern as a known or unknown person if the image was decided to be a face.

In Turk and Pentland's eigenface technique [1991a], the initialisation phase includes calculating and using the average face from the training set as well as its corresponding vector to achieve the desired eigenfaces for the actual recognition. When the training set is $\mathbf{t}_1, ..., \mathbf{t}_M$, the average face of the set is defined by $\mathbf{\Psi} = \frac{1}{M}\sum_{n=1}^{M} \mathbf{t}_n$. Each face then differs from the average face by the vector $\mathbf{\Phi}_i = \mathbf{t}_i - \mathbf{\Psi}$ [Turk and Pentland, 1991a]. This set of vectors is then submitted to PCA, the result being a set of M orthonormal vectors $\mathbf{u}_n$, which give the best description of the distribution of the data.

The vectors $\mathbf{u}_k$ and their respective scalars λ_k achieved from PCA are the eigenvectors and eigenvalues of the covariance matrix

$$C = \frac{1}{M}\left(\sum_{n=1}^{M} \mathbf{\Phi}_n \mathbf{\Phi}_n^T\right) = AA^T \qquad (1)$$

where A is the matrix $[\mathbf{\Phi}_1\mathbf{\Phi}_2...\mathbf{\Phi}_M]$. However, the matrix C is N^2 by N^2, and to determine N^2 eigenvectors isn't computationally very feasible for typical images [Turk and Pentland, 1991b]. Fortunately, the calculations can be simplified by solving an M x M matrix first; if the number of data points in the image space is significantly less than the dimension of the space (and it usually is), there will only be M - 1 meaningful eigenvectors, since the remaining eigenvectors will have associated eigenvalues of zero [Turk and Pentland, 1991b]. After solving the M x M matrix, the eigenvectors for matrix C can be formed from the linear combinations of the face image $\mathbf{\Phi}_i$.

Turk and Pentland [1991b] end their initialisation by describing how the M x M matrix can be constructed as $L = A^TA$, where $L_{mn} = \mathbf{\Phi}_m^T \mathbf{\Phi}_n$ and the M eigenvectors $\mathbf{v}_i$ can be found for the matrix L. These vectors determine then the eigenfaces $\mathbf{u}_l$:

$$\mathbf{u}_l = \sum_{n=1}^{M} \mathbf{v}_{lk}\mathbf{\Phi}_k \qquad l=1, ..., M . \qquad (2)$$

This analysis greatly reduces the calculations that are needed for the recognition process.

After the eigenfaces are calculated, the second step is the actual recognition. The eigenfaces now span an M' dimensional subspace of the original N^2 image space. The M' significant eigenvectors of the L matrix are chosen to be the ones

with the largest associated eigenvalues [Turk and Pentland, 1991a]. A new image **t** is transformed into its eigenface components and projected into the face space by the operation

$$\omega_k = \mathbf{u}_l^T (\mathbf{t}-\Psi) \qquad k = 1, ..., M' \tag{3}$$

[Turk and Pentland, 1991a], which demonstrates how the average eigenface is used in the recognition process. These weights form the vector $\mathbf{\Omega}^T = [\omega_1 \omega_2 \ldots \omega_{M'}]$ that describes the contribution of each eigenface in representing the input face image. This vector is further used to determine whether the image belongs to a face class, and if it does, to which one [Turk and Pentland, 1991a].

The simplest way to find out the corresponding face class is to calculate the Euclidian distances $\varepsilon_k = \| \mathbf{\Omega} - \mathbf{\Omega}_k \|$ between the vectors and find the minimum of these values; just as Moon and Phillips suggested in their generic model. A face is classified as belonging to class k when ε_k is below some chosen threshold. If such a value is not found, the face is classified as unknown. Figure 1 illustrates the face space and projection of images. In Figure 1, $\mathbf{u}_1$ and $\mathbf{u}_2$ represent eigenfaces, and the vectors $\mathbf{\Omega}_1$, $\mathbf{\Omega}_2$ and $\mathbf{\Omega}_3$ represent known individuals to which the new images are compared.

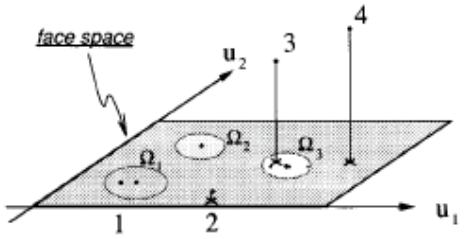


Figure 1: Projecting images to face space [Turk and Pentland, 1991a]

The points 1, 2, 3 and 4 in Figure 1 represent the cases that can happen when comparing a new image to the known face space. In case 1 the pattern vector of the image is near a face class and the face space, and is recognized as the individual represented by $\mathbf{\Omega}_1$. In case 2, the pattern vector of the image is close to the face space but far from all known face classes. This labels the image as an unknown face. In cases 3 and 4 the image is far from the face space, and in this case, even though the image might be close to a face class as in case 3, the image is classified as something else than a face. This procedure counts for the steps three and four in Turk and Pentland's study [1991a].

As a summary, principal component analysis can be used as a basis for various face recognition techniques and algorithms, the general idea of which was presented by Moon and Phillips [2000]. The purest and most straightforward technique using PCA is the eigenface technique as illustrated by Turk and

Pentland [1991a, 1991b]. Other PCA techniques include techniques such as evolution pursuit, feature lines and probabilistic eigenfaces [Zhao et al., 2003]. The common feature on all PCA techniques is the usage of eigenvectors and their corresponding eigenvalues as a label for the image to be studied.

3. Linear Discriminant Analysis

Linear discriminant analysis (LDA) is a close relative to PCA, in fact Zhao et al. [2003] even classify LDA under principal component analysis. LDA is often referred to as the *Fisherface technique* due to the fact that the very founding applications of LDA use Fisher's linear discriminant. Over time many other variations of LDA-based face recognition algorithms have emerged, and Fisherfaces are called Fisher linear discriminant analysis (FLDA or FLD) [Yin et al., 2006; Zhou et al., 2006] in recent references. In this paper however, LDA is approached from the classical point of view that was first introduced by Belhumeur et al. [1996], and the terms LDA and Fisherfaces are used as equivalents in face recognition.

LDA itself is a classical statistical approach for feature extraction and dimension reduction [Ye, 2005], just like PCA. LDA computes the optimal transformation, which minimizes the within-class distance of the data set and maximizes the between-class distance, thus achieving maximum discrimination [Ye, 2005].

Belhumeur et al. [1996] define the scatter matrices to be used for Fisherface analysis as follows: the between-class scatter matrix is defined as

$$S_B = \sum_{i=1}^{c} \left| \chi_i \right| (\mu_i - \mu)(\mu_i - \mu)^T \tag{4}$$

and the within-class scatter matrix is defined as

$$S_w = \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu)(x_k - \mu_i)^T , \tag{5}$$

where μ_i is the mean image of class χ_i and $\left| \chi_i \right|$ is the number of training images in class χ_i.

As a comparison, more recent studies take into account the amount of images in the whole training set, and Zhou et al. [2006] and Yin et al. [2006] define scatter matrices as $G_B = \frac{1}{n} S_B$ and $G_W = \frac{1}{n} S_W$, where n is the total number of images in the training set.

The scatter matrices are used to calculate the *Fisher criterion function* $J(W)$ defined as

$$J(W) = \left| \frac{W^T S_B W}{W^T S_W W} \right| . \tag{6}$$

The criterion function $J(W)$ is maximized when W consists of the eigenvectors of the matrix $S_W^{-1} S_B$ [Zhou et al., 2006], $W = [w_1, w_2..., w_n]$. In this way, W gets selected such that the ratio of the between-class scatter and the within-class scatter is maximized, which, in turn, brings reliability into classification [Belhumeur et al., 1996]. After obtaining the optimal discriminating vectors that form W, a feature vector can be extracted for any input pattern, and classification can be made based on that feature vector [Yin et al., 2006]. The comparison can be done by a similar nearest neighbour technique using the Euclidean distances as discussed with eigenfaces.

All in all, LDA is quite similar to PCA. Just like PCA, it requires a mean feature vector to be calculated from the training set and a feature vector to be calculated for each training image. The actual calculation of feature vectors can be done using, e.g., the discrete cosine transformation technique [Zhou et al., 2006], where the image is divided into increasingly small sub images, and finally the location information and knowledge of face structure are used in order to create feature vectors which mainly concentrate on the eyes and the nose. LDA then provides a way to reduce dimensionality, and provides a projection matrix with which the comparison between different images is made.

4. Kernel Techniques

Kernel techniques are a combination of the *kernel trick* and a statistical base method, such as PCA or LDA. Using the kernel trick, these analysis models can be made more efficient than they would be on their own. A study by Yang [2002] shows that when kernel methods are used with classical face recognition methods, the algorithms are more resistant to variation in pose, scale, lighting and expression. This greatly increases the quality of the algorithm as recognition can be done with a more variant set of samples and the algorithm isn't dependent on the correctness of the sample images.

The kernel trick is based on *Mercer's kernel* definition. If for any set $\{z_1, z_2, ..., z_n\} \subset \mathbf{X}$ the m by m matrix $K[z] = (K(z_i, z_j))_{ij}$ is positive definite, then $K(x,w)$ is positive definitive. Now, if $K(x,w)$ is continuous, symmetric, i.e. $K(x,w) = K(w,x)$, and positive definitive and the function K is defined as $K : \mathbf{X} \times \mathbf{X} \rightarrow \mathbf{R}$, then $K(x,w)$ is considered a Mercer's kernel [Rodriguez, 2004].

Bach and Jordan [2002], on the other hand, define Mercer's kernel with the help of *Gram matrices*; by their definition a Mercer kernel in $\mathbf{X} = \mathbf{R}^p$ is a function for which the Gram matrix $K_{ij} = K(x_i,x_j)$ is positive semidefinite for any collection $\{x_i\}_{i=1,..,N}$ in **X.**

Now, for any Mercer's kernel K there is a map $\mathbf{\Phi}$ from X to a feature space F, such that

$$K(x, y) = \langle \mathbf{\Phi}(x), \mathbf{\Phi}(y) \rangle. \quad (7)$$

This shows how the kernel can be used to evaluate an inner product in the feature space, and is called the kernel trick [Bach and Jordan, 2002].

4.1. Kernel Independent Component Analysis

Independent component analysis (ICA) is a similar method to LDA and PCA, and it brings the statistical base to a face recognition algorithm. As with LDA, Zhao et al. [2003] classify ICA under PCA.

Bach and Jordan [2002] define ICA to be based on a statistical model

$$\mathbf{y}=A\mathbf{x}\,, \quad (8)$$

where $\mathbf{x}$ is a latent random vector with m independent components, A is an m x m matrix of parameters, and $\mathbf{y}$ is an observed vector with m components. The goal now is to make an estimate of the parameter matrix A. The estimate is based on a set of N independent, identically distributed observations of the vector $\mathbf{y}$. The estimate of A can then be used to estimate the values of $\mathbf{x}$ corresponding to any given observed $\mathbf{y}$; this estimation is done by solving a linear system of equations [Bach and Jordan, 2002].

Bach and Jordan [2002] define $W = A^{-1}$ so that $\mathbf{x} = W\mathbf{y}$. Now, for a given A, the problem of maximizing likelihood with respect to W is equivalent to the problem of minimizing the mutual information between the components $\mathbf{x} = W\mathbf{y}$. ICA problems are a generalized version of PCA problems, and they take advantage of PCA to pre-process data. Using this information, the variable $\mathbf{y}$ can now be *whitened* by multiplying it by a matrix P such that $\check{\mathbf{y}} = P\mathbf{y}$ has an identity covariance matrix. Once the data is whitened, the matrix W is necessarily orthogonal, which reduces the number of parameters that need to be estimated [Bach and Jordan, 2002].

Bach and Jordan [2002] present a KERNEL-ICA algorithm for extracting the matrix W from a data set. The algorithm takes as input the data vectors $y_1, y_2, \ldots, y_n$ extracted from a training set of images, and a kernel $K(x,y)$. The first step in the algorithm is to whiten the data. After that, the contrast function $C(W)$ is minimized with respect to W. The function $C(W)$ is defined as follows: first the centered Gram matrices $K_1, K_2, \ldots, K_m$ of the estimated sources $\{x_1, x_2, \ldots, x_m\}$ are computed where $x_i = Wy_i$. Next, the minimal eigenvalue of the generalized eigenvector equation

$$K_\kappa \alpha = \lambda D_\kappa \alpha \quad (9)$$

is defined as $\lambda_F^K(K_1, K_2, \ldots, K_m)$. Finally, $C(W)$ is defined so that

$$C(W) = \hat{I}_{\lambda F}(K_1, K_2, \ldots, K_m) = -\frac{1}{2}\log \hat{\lambda}_F^K(K_1, K_2, \ldots, K_m). \quad (10)$$

The final output of this algorithm is the matrix *W*.

Martiriggiano et al. [2005] took advantage of Bach and Jordan's [2002] algorithm and used it to recognize faces in the FERET database [Phillips et al., 2000]. They used kernel ICA to find a representation in which the coefficients used to code images are statistically independent. The actual face recognition is evaluated by the nearest neighbour algorithm. Coefficient vectors *b* in each of the test set were given the class label of the coefficient vector in the training set that was the most similar. The similarity of the sets was evaluated by the Euclidian distance measure and the cosine distance measure in order to compare results between these two methods [Martiriggiano et al., 2005].

4.2. Kernel Eigenfaces

The eigenface technique is based on finding the eigenvalues λ of the covariance matrix *C*,

$$\lambda \mathbf{w} = \mathbf{w}C, \quad (11)$$

for eigenvectors $\mathbf{w} \in \mathbf{R}^n$. When implementing kernel methods with the eigenface technique, each vector $\mathbf{x}$ from the training set is projected from the input space to a higher dimensional feature space $\mathbf{R}^f$ by a non-linear mapping function $\mathbf{\Phi} : \mathbf{R}^n \rightarrow \mathbf{R}^f$. The eigenvalue problem (10) in feature space is now

$$\lambda \mathbf{w}^{\Phi} = \mathbf{w}^{\Phi} C^{\Phi}, \quad (12)$$

where *C* is a covariance matrix [Yang, 2002].

Yang [2002] continues to derive the kernel PCA problem as follows: all solutions $\mathbf{w}^{\Phi}$ with $\lambda \neq 0$ lie in the span $\mathbf{\Phi}(\mathbf{x}_1), \ldots, \mathbf{\Phi}(\mathbf{x}_m)$, and there exist coefficients α_i such that

$$\mathbf{w}^{\Phi} = \sum_{i=1}^{m} \alpha_i \mathbf{\Phi}(\mathbf{x}_i) . \quad (13)$$

Denoting an $m \times m$ matrix *K* by

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{\Phi}(\mathbf{x}_i) \cdot \mathbf{\Phi}(\mathbf{x}_j) , \quad (14)$$

the kernel PCA problem becomes

$$m\lambda K\boldsymbol{\alpha} = K^2\boldsymbol{\alpha} \equiv m\lambda\boldsymbol{\alpha} = K\boldsymbol{\alpha} , \quad (15)$$

where $\boldsymbol{\alpha}$ notates a column vector with entries $\alpha_1, \ldots, \alpha_m$. In this derivation Yang [2002] assumes that all projections $\mathbf{\Phi}(\mathbf{x})$ of the samples are centered in $\mathbf{R}^f$.

The samples in $\mathbf{R}^f$ can now be projected to a lower-dimensional space spanned by the eigenvectors $\mathbf{w}^{\Phi}$. If $\mathbf{x}$ is a test sample whose projection is $\mathbf{\Phi}(\mathbf{x})$ in R^f, then the projection of $\mathbf{\Phi}(\mathbf{x})$ onto the eigenvectors $\mathbf{w}^{\Phi}$ is the nonlinear principal components corresponding to $\mathbf{\Phi}$:

$$\mathbf{w}^{\Phi} \cdot \mathbf{\Phi}(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i \, (\mathbf{\Phi}(\mathbf{x}_i) \cdot \mathbf{\Phi}(\mathbf{x})) = \sum_{i=1}^{m} \alpha_i k(\mathbf{x}_i, \mathbf{x}) \, . \tag{16}$$

The principal components, i.e., the eigenvectors $\mathbf{w}^{\Phi}$ can now be extracted with the help of the kernel method without extensive operations [Yang, 2002]. Hence, using the kernel method provides the opportunity to use information available in higher dimensions and the means to calculate the eigenfaces without extensive calculations usually needed when operating in high dimensions.

4.3. Kernel Fisherfaces

Fisherfaces are based on within-class and between-class scatter matrices S_W and S_B. With the same assumption that was made in kernel eigenfaces about the projections $\mathbf{\Phi}(\mathbf{x})$ being centered in $\mathbf{R}^f$, the scatter matrices can be denoted by $S_W{}^{\Phi}$ and $S_B{}^{\Phi}$ [Yang, 2002]. Applying LDA in kernel space, the eigenvalues λ and eigenvectors $\mathbf{w}^{\Phi}$ of

$$\lambda S_W^{\Phi} \; \mathrm{w}^{\Phi} = S_B^{\Phi} \; \mathrm{w}^{\Phi} \tag{17}$$

now need to be calculated. This can be done by

$$J(W^{\Phi}) = \left| \frac{\left(W^{\Phi}\right)^T S_B^{\Phi} W^{\Phi}}{\left(W^{\Phi}\right)^T S_W^{\Phi} W^{\Phi}} \right| = [\mathbf{w}^{\Phi}{}_{\mathbf{1}}, \ldots, \; \mathbf{w}^{\Phi}{}_{\mathbf{m}}] \tag{18}$$

where $\{ \mathbf{w}^{\Phi}{}_i \mid i = 1,2, \ldots, m\}$ is the set of generalized eigenvectors corresponding to the m largest generalized eigenvalues $\{\lambda_i \mid i = 1,2, \ldots, m\}$.

When the problem in hand has c classes, the kernel function can be defined as

$$(k_{rs})_{tu} = k(\mathbf{x}_{tr} \, , \mathbf{x}_{us}) = \mathbf{\Phi}(\mathbf{x}_{tr}) \cdot \mathbf{\Phi}(\mathbf{x}_{us}), \tag{19}$$

where $\mathbf{x}_{tr}$ is the r-th sample of class t which has l_t samples, and $\mathbf{x}_{us}$ is the s-th sample of class u which has l_u samples [Yang, 2002].

Yang [2002] defines K to be an m x m matrix defined by the elements

$$(K_{tu})_{u=1,\ldots,c}^{t=1,\ldots,c} \text{ where } K_{tu} = (k_{rs})_{s=1,\ldots,l_u}^{r=1,\ldots,l_t} \quad . \tag{20}$$

Here K_{tu} is a matrix composed of dot products in feature space $\mathbf{R}^f$. A matrix Z is also defined to be

$$Z = (Z_t)_{t=1,\ldots,c} \, , \tag{21}$$

where Z_t is an l_t x l_t matrix with terms all equal to $1/ l_t$. Continuing towards the final equation, the scatter matrices $S_W{}^{\Phi}$ and $S_B{}^{\Phi}$ in feature space $\mathbf{R}^f$ are defined as

$$S_W^{\Phi} = \sum_{i=1}^{c} l_i \mu_i{}^{\Phi} (\mu_i{}^{\Phi})^{\mathrm{T}} \text{ and } S_B^{\Phi} = \sum_{i=1}^{c} \sum_{j=1}^{l_i} \mathbf{\Phi}(\mathbf{x}_{ij}) \mathbf{\Phi}(\mathbf{x}_{ij})^{\mathrm{T}} \tag{22}$$

where $\mu_i{}^{\Phi}$ is the mean of class i in $\mathbf{R}^f$ and l_i is the number of samples in class i [Yang, 2002].

Using the same logic as in kernel eigenfaces, we have

$$\mathbf{w}^{\Phi} = \sum_{p=1}^{c} \sum_{q=1}^{l_p} \alpha_{pq} \mathbf{\Phi}(\mathbf{x}_{pq}) \quad . \tag{23}$$

The solution for (23) is available by solving

$$\lambda KK\boldsymbol{\alpha} = KZK\boldsymbol{\alpha} \quad . \tag{24}$$

Using the kernel LDA solution $J(W^{\Phi})$ can be defined as

$$J(W^{\Phi}) = \left| \frac{\alpha KZK\alpha}{\alpha KK\alpha} \right| = [\mathbf{w}^{\Phi}{}_{1,\ldots,} \ \mathbf{w}^{\Phi}{}_{\mathbf{m}}] \ . \tag{25}$$

Now $\mathbf{\Phi}(\mathbf{x})$ can be projected to a higher dimensional feature space $\mathbf{R}^f$ spanned by the eigenvectors $\mathbf{w}^{\Phi}$ [Yang, 2002]. The same Fisherface technique as introduced in Chapter 3 can now be applied for these eigenvectors for face recognition.

5. 3D Face Recognition

The depth information in the face represents personal features in detail, and the surface curvatures extracted from the face contain the most important personal facial information [Lee and Han, 2006]. Therefore, recognizing faces from 3D images brings more information available for the algorithm, thus making the results more accurate.

3D recognition can basically be done by one of two methods. The face feature based approach uses feature vectors extracted from within the image as a recognition parameter. The area based approach on the other hand extracts a special area from the face image and recognizes it using the relationship and minimum sum of squared difference.

Normalizing the image also increase the recognition rate, and normalizing 3D image can be done be finding the nose tip and using it put the face shape in a standard spatial position [Lee and Han, 2006].

5.1. 3D Recognition with Eigenfaces

3D recognition with eigenfaces follows quite closely the same procedures as eigenface recognition with normal 2D images. The main difference lies in calculating the training set from 3D images using *surface curvatures*.

For each data point on the facial surface, the principal, Gaussian and mean curvatures are calculated and the signs of those are used to determine the surface type at every point. The z(x, y) image represents a surface where the individual Z-values are surface depth information, x and y being the two spatial coordinates [Lee and Han, 2006]. Any point on the surface can be specified by its position vector

$$\mathbf{R}(x, y) = xi + yj + z(x, y)k \,. \tag{26}$$

Lee and Han [2006] now continue to derive the equations that give the minimum and maximum curvatures which can be used in face recognition. For the outcome, two fundamental forms of the surface need to be calculated.

The first form is the expression for the element of arc length of curves on the surface which pass through the point under consideration. This is given by

$$I = ds^2 = dR \cdot dR = Edx^2 + 2Fdxdy + Gdy^2 \tag{27}$$

where

$$E = 1 + \left(\frac{\partial z}{\partial x}\right)^2, \quad F = \left(\frac{\partial z}{\partial x}\right)\left(\frac{\partial z}{\partial y}\right), \text{ and } G = 1 + \left(\frac{\partial z}{\partial y}\right)^2 . \tag{28}$$

The second form is given by the curvatures of these curves at the point of interest and in the given direction:

$$II = edx^2 + 2fdxdy + gdy^2 \tag{29}$$

where

$$y = \left(\frac{\partial^2 z}{\partial x^2}\right)\Delta, \qquad f = \left(\frac{\partial^2 z}{\partial x \partial y}\right)\Delta, \qquad g = \left(\frac{\partial^2 z}{\partial y^2}\right)\Delta \tag{30}$$

and

$$\Delta = (EG - F^2)^{-1/2} \quad . \tag{31}$$

Defining the matrices

$$V = \begin{pmatrix} dx \\ dy \end{pmatrix}, \; A = \begin{pmatrix} E & F \\ F & G \end{pmatrix}, \text{ and } B = \begin{pmatrix} e & f \\ f & g \end{pmatrix} \quad , \tag{32}$$

the two fundamental forms can be marked as

$$I = V'AV \qquad I = V'BV \quad . \tag{33}$$

Now the curvature of the surface in the direction defined by V is given by

$$k = \left(\frac{V'BV}{V'AV}\right). \tag{34}$$

Extreme values of k are given by the solution to the eigenvalue problem

$$(B - kA)V = 0 \,, \tag{35}$$

which gives k_1 and k_2, as defined by Lee and Han [2006] to be the minimum and maximum curvatures, respectively:

$$k_1 = \{gE - 2Ff + Ge - [(gE + Ge - 2Ff)^2 - 4(eg - f^2)(EG - F^2)]^{1/2}\}/2(EG - F^2) \tag{36}$$

and

$$k_2 = \{gE - 2Ff + Ge + [(gE + Ge - 2Ff)^2 - 4(eg - f^2)(EG - F^2)]^{1/2}\}/2(EG - F^2). \tag{37}$$

The Gaussian curvature K and the mean curvature M can now be defined by

$$K = k_1 k_2 \qquad M = (k_1 k_2)/2 \tag{38}$$

which gives k_1 and k_2, the minimum and maximum curvatures, respectively [Lee and Han, 2006].

The face recognition is done by extracting images around the nose area, and then creating a training set and a testing set. The training images are used to generate an orthogonal basis, as described in deriving the principal curvatures in equations (26) to (36). The 3D images in the training set are then projected into the orthogonal basis with the help of the eigenface technique, described in Chapter 2. The recognition can then be done with similar methods as used with normal eigenfaces, where the images to be recognized belong to the test set.

5.2. 3D Recognition with Other Techniques

Bronstein et al. [2003; 2004] have developed two different face recognition techniques for 3D data. Both of these techniques take advantage of *bending-invariant canonical forms*, but the recognition differs: in one study Bronstein et al. [2004] use *canonical form's moments* to compute geodesic distances, and in another study [2003] they use *eigenforms*.

Calculating bending-invariant canonical forms begins by extracting the distances between points in the surface. If a facial surface S is represented by a set of points p_i (i = 1, ..., n), then the geodesic distance between the points can be marked as

$$\delta (p_i, p_{j)} = \delta_{ij} \ . \tag{39}$$

A strong algorithm is needed for the actual calculation of the distances, and it can by done, e.g., with *fast marching on triangulated domains* [Bronstein et al., 2003]. When the values δ_{ij} are put in matrix form, a matrix of mutual distances between surface points is gained and this can be defined as squared mutual distances

$$(\Delta)_{ij} = \delta_{ij}^2 \quad . \tag{40}$$

Since squared mutual distances can be thought of as dissimilarities, they can be used for *multi-dimensional scaling* (MDS) in order to reduce dimensionality in the data [Bronstein et al., 2003]. MDS allows the surface to be embedded into a low-dimensional Euclidean space $\mathbf{R}^m$. This can be thought of as finding a mapping between two metric spaces

$$\varphi: (S, \delta) \rightarrow (\mathbf{R}^m, d) \ ; \quad \varphi (p_i) = x_i \tag{41}$$

which minimizes the embedding error

$$\varepsilon = f(|\delta_{ij} - d_{ij}|) \quad ; d_{ij} = \| x_i - x_j \|_2 \ . \tag{42}$$

The obtained m-dimensional representation is a set of points $x_i \in \mathbf{R}^m$ (i=1,..., n), corresponding to the surface points p_i [Bronstein et al., 2003]. This set of points x_i is now referred to as the bending-invariant canonical form of the surface and when m = 3, it can be plotted as the surface.

As mentioned, these canonical forms can be used for face recognition in two ways, and the first method is using moments. A canonical form's (p, q, r) – th moment is given by

$$M_{pqr} = \sum_n (x_n^1)^p (x_n^2)^q (x_n^3)^r \quad , \tag{43}$$

where x_n^i is the i^{th} coordinate of the n^{th} point in the canonical surface samples [Bronstein et al., 2003]. To compare between two canonical forms, i.e., to compare two facial surfaces and find out the scale of their similarity, the vector ($M_{p1q1r1,}$ ..., M_{pMqMrM}), termed as the moment's signature, is computed for each surface. The difference between two surfaces can then be measured by the Euclidean distance between the two moments' signatures [Bronstein et al., 2003].

Another way to compute the difference between two surfaces is to use eigenforms [Bronstein et al., 2004]. This technique is based on the correspondence of the pixels a_n in the texture and the canonical surface vertices (x_n^1, x_n^2, x_n^3). Because of the correspondence, the face texture image can be mapped on to the aligned canonical surface in the canonical form space [Bronstein et al., 2004]. The *flattened texture* $\tilde{a}$ and the *canonical image* $\tilde{x}$ can be obtained by interpolating a_n and $x_n{}^3$ onto a Cartesian grid in the X^1X^2 plane. The recognition is done with the help of a training set, which is a set of duplets of the form $\{\tilde{x}_n, \tilde{a}_n\}_{i=1}^N$. Applying eigendecomposition for both ã and ~x, two different sets of eigenspaces can now be produced. These sets of eigenvectors, $e^a{}_n$ and $e^x{}_n$, are termed as eigenforms [Bronstein et al., 2004].

For a new image to be tested that is represented by ($\tilde{x}'$, $\tilde{a}'$), the decomposition coefficients are computed according to

$$\alpha = [e_1^a, ..., e_N^a](\tilde{a}' - \bar{a})$$

and (44)

$$\beta = [e_1^x, ..., e_N^x](\tilde{x}' - \bar{x}),$$

where $\bar{a}$ and $\bar{x}$ denote the average of $\tilde{a}_n$ and $\tilde{x}_n$ in the training set, respectively. The distance between two subjects, represented by ($\tilde{x}_1, \tilde{a}1$) and ($\tilde{x}_2, \tilde{a}_2$) are computed as weighted Euclidean distance between the corresponding decomposition coefficients, (α_1, β_1) and (α_2, β_2) [Bronstein et al., 2004]. As can be seen, this method is also close to the eigenface technique

6. Other Techniques

6.1. Hidden Markov Models

Hidden Markov models (HMMs) are a set of statistical models used to characterize the statistical properties of a signal. HMMs can be divided into two main

processes. Firstly there is a Markov chain with a finite number of states, a state transition probability matrix and an initial state probability distribution, and secondly there is a set of probability density functions associated with each state [Nefian and Hayes, 1998a]. A HMM can be defined as the triplet $\lambda = (A, B, \Pi)$ where A is the state transition probability matrix, B is the observation symbol probability matrix and Π is the initial state distribution [Nefian and Hayes, 1998b].

Nefian and Hayes [1998a] introduce a method for recognizing faces with HMMs. The dominating face regions (eyes, nose, etc.) are assigned to a state in a continuous HMM. This state model is used to initialize the A and Π in the triplet representing the HMM. Each face image is then divided into overlapping blocks, from which observation vectors are calculated. These vectors are then used to obtain initial estimates and to initialize B. To recognise a face, hidden Markov models are calculated from a large set of training data. Then the observation vectors are calculated from the image to be recognized. As Nefian and Hayes [1998a] explain, when $\boldsymbol{O}$ is the observation symbol, face image t is recognized as face k if it holds $P(\boldsymbol{O}^{(t)} \mid \lambda_k) = max_n P(\boldsymbol{O}^{(t)} \mid \lambda_n)$.

Hidden Markov Models can also be used to detect faces from a large image [Nefian and Hayes, 1998b]. The image is divided into rectangular, largely overlapping areas, and observation vectors for these areas are then created. The probabilities of each area are then calculated with the given face model, and when the probability exceeds a given limit, the chances are good that the area indeed contains a face.

6.2. Elastic Bunch Graph Matching

Wiskott et al. [1997] have researched a method to recognize faces using *graphs*. They start by defining a wavelet based *jet* as the set of *Gabor wavelet coefficients* gathered from one image point. The face is then represented by a labeled graph, where nodes are situated at *fiducial points*, such as pupils, corners of the mouth, etc. and are labelled with jets, while the edges of the graph are labelled with two-dimensional distance vectors.

To be able to automatically find a representative graph for a new face, different poses and expressions must be taken into consideration. Wiskott et al. [1997] solve the problem not by making an extremely large database with all the possible variations, but by making a collective stack-like structure of representative model graphs, called the *face bunch graph* (FBG).

The model graphs forming the FBG all have the same kind of grid structure and their nodes refer to identical fiducial points. A set of jets referring to one fiducial point is then called a bunch [Wiskott et al., 1997]. In this new FBG, the nodes are labelled with the corresponding jet bunches, which then offer the

knowledge from all the model graphs in one place. When the actual recognition then happens, the best fitting jet can be selected from the bunch.

To begin face recognition with elastic bunch graph matching, a starter FBG needs to be initialized from an example set of graphs. Then when more comparisons are made, the FBG collects more data and becomes more accurate and detailed. When a new face is matched to the FBG, the matching is based on graph similarities. Similarities between graphs are found by searching the best fitting jet from a bunch in a node, and then when the jets that maximize the similarity between graphs have been found, the corresponding graph is searched from the FBG with the help of a heuristic algorithm [Wiskott et al., 1997]. Since this generates several model graphs to be found from the FBG, on account that the similarity algorithm that gives the model graph relies on the average similarity of the jets, a final comparison between the image graph and the model graphs has to be made. This is relatively simple, as the image graph is simply compared to each of the model graph, and the one which has the highest similarity then gives the desired recognition.

6.3. Bayesian Method with Gabor Features

Wang and Tang [2003] introduce a face recognition technique, where the statistical Bayesian model is combined with graph based *Gabor features* to get more efficient results. The face is first characterized as a graph, where the essential features (fiducial points) are marked as nodes, the Gabor wavelet around is node is marked as node attributes, and the distances between nodes as edge attributes.

In Wang and Tang's study [2003] the graph was designed so that it had 35 nodes. From each node, 40 Gabor features could be obtained. These are then gathered into 35 local vectors, which combined make a large Gabor feature vector that represents the face image.

Face recognition measures whether two images represent the same face or two different faces. In Gabor feature vectors, these are measured with intrapersonal variation $\mathbf{\Omega}_I$ and extrapersonal variation $\mathbf{\Omega}_E$ [Wang and Tang, 2003]. The difference (Δ) between two face vectors is made of three components: intrinsic difference, which differentiates different individuals, transformation difference, which is caused by, e.g., varying expressions, illuminations and all other transformation, and noise, which is randomly distributed in face images. Naturally, the more noise and transformation there is in a face image, the more difficult it is to make a recognition, but fortunately the amount of noise is usually small. Thus the main difficulty comes from transformation [Wang and Tang, 2003]. Gabor features as such don't have the means to extract the effects of transfor-

mation, so combining the Bayesian model here will help with getting the results.

When I_1 and I_2 represent intrinsic differences between two feature vectors, Wang and Tang suggest the use of the Bayesian rule as follows to find the similarity S of two feature vectors:

$S(I_1, I_2) = P(\mathbf{\Omega_I} | \Delta)$

$$= \frac{P(\Delta | \Omega_I) P(\Omega_I)}{P(\Delta | \Omega_I) P(\Omega_I) + P(\Delta | \Omega_E) P(\Omega_E)} \quad . \tag{45}$$

Thus, the similarity between two images, i.e. feature vectors, is the probability given by the Bayesian rule.

7. Conclusions

The fundamental principles of all face recognition algorithms are the same: first, a set of face images is required as a testing set. Then a set of principal vectors are extracted from a test image to signify the unique features of the face with the use of a significant face recognition algorithm. Depending on the algorithm and its output, the data from the test image is then compared to the data available in the training set, and if the similarities are high enough, it is safe to say that the image in question has be recognised as a face in the training set.

The eigenface technique described in Chapter 2 is considered to be the golden rule in the field of face recognition, and this is quite understandable even without the long history behind it. The eigenface method is quite straightforward with moderate calculations and simple outcomes. The steps of the technique are easily adapted and modified to suit a new method in the testing. Results from tests made with the eigenface technique also prove that it is indeed an accurate algorithm.

The Fisherface technique discussed in Chapter 3 is quite similar to the eigenface technique and shows the underlying similarity between most face recognition algorithms. The basic steps in proceeding with the algorithm follow closely to those with eigenfaces: the difference lies in using Fisher's criterion function to find the eigenvectors. The amount of modified methods based on Fisherfaces, such as uncorrelated linear discriminant analysis and orthogonal linear discriminant analysis [Ye, 2005], proves the efficiency behind this algorithm.

Kernel methods explored in Chapter 4 prove the efficiency of making new hybrid algorithms. All the kernel methods introduced were based on an algorithm having a strong statistical base (PCA, LDA or ICA) as well as proven ca-

pability as an individual algorithm. Implementing the kernel trick with these algorithms, the calculations were even more simplified and the efficiency of the algorithms was increased as results became more accurate.

3D face recognition is the new trend and with more data available from the face, naturally the rate of recognition becomes higher when the amount of false recognitions decreases and comparisons between two images become more accurate in detail.

Different ways for recognising faces were introduced in Chapter 6, and these methods prove that although the traditional methods still hold their ground, there is still room for methods approaching face recognition from a completely different angle, and more research on, i.e., elastic bunch graph matching could prove to be quite interesting. All in all, face recognition is a topic in which there are a lot of aspects yet to be researched.

References

[Bach and Jordan, 2002] F.R. Bach and M.I. Jordan, Kernel independent component analysis, *Journal of Machine Learning Research* **3** (2002), 1-48.

[Bartlett et al, 2002] M.S. Bartlett, J.R. Movellana and T.J. Sejnowski, Face recognition by independent component analysis, *IEEE Trans. on Neural Networks* **13**, 6 (Nov. 2002), 1450-1464.

[Belhumeur et al., 1996] Peter N. Belhumeur, Joao P. Hespanha and David J. Kriegman, Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. In *Proc. of the 4th European Conference on Computer Vision, ECCV'96*, (1996), 45-58.

[Bronstein et al., 2003]A. Bronstein, M. Bronstein and R. Kimmel, Expression-invariant 3D face recognition. In Proc. *Audio & Video-based Biometric Person Authentication (AVBPA)*, LNCS **2688** (2003), Springer, 62-69.

[Bronstein et al., 2004] A. Bronstein, M. Bronstein, R. Kimmel and A. Spira, 3D face recognition without facial surface reconstruction. In *Proceedings of ECCV 2004* LNCS **3022** (2004), Springer, 225-238.

[Ding and Fang, 2004] Xiaoqing Ding and Chi Fang, Discussions on some problems in face recognition. In S.Z.Li et al. (eds.), *Sinobiometrics 2004*, LNCS **3338** (2004), Springer, 47-56.

[Lee and Han, 2006] Yeung-Hak Lee and Chang-Wook Han, 3D facial recognition using eigenface and cascade fuzzy neural networks: normalized facial image approach. In D.Grigoriev, J.Harrison and E.A.Hirsch (eds.), *CSR 2006*, LNCS **3967** (2006), Springer, 457-465.

[Martiriggiano et al., 2005] T. Martiriggiano, M. Leo, T. D'Orazio and A. Distante, Face recognition by Kernel independent component analysis. In

M.Ali and F.Esposito (eds.), *IEA/AIE 2005*, LNAI **3533** (2005), Springer, 55-58.

[Moon and Phillips, 2000] H. Moon, P.J. Phillips, Computational and performance aspects of PCA-based face recognition algorithms, *Perception* **30** (2001), 303-321.

[Nefian and Hayes, 1998a] Ara V. Nefian and Monson H. Hayes III, Hidden Markov models for face recognition. In *ICASSP98* **5** (1998), 2721-2724 .

[Nefian and Hayes, 1998b] Ara V. Nefian and Monson H. Hayes III, Face detection and recognition using hidden Markov models. In: *IEEE International Conference on Image Processing* **1** (Oct.1998), 141-145.

[Phillips et al., 2000] P.Jonathon Phillips, Hyenjoon Moon, Syed A. Rizvi and Patrick J. Rauss, The FERET evaluation methodology for face-recognition algorithms. *IEEE Trans. on Pattern Recognition and Machine Intelligence* **22**, 10 (Oct. 2000),1090-1104.

[Rodriguez, 2004] Carlos C. Rodriguez, The Kernel trick, http://omega.albany.edu:8008/machine-learning-dir/notes-dir/ker1/ker1.pdf, visited 29.11.2006.

[Turk and Pentland, 1991a] M.A. Turk and A.P. Pentland, Face recognition using eigenfaces, In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition* (June 1991), 586-591.

[Turk and Pentland, 1991b] M. Turk and A. Pentland, Eigenfaces for recognition, *Journal of Cognitive Neuroscience* **3**, 1(1991), 71-86.

[Wang and Tang, 2003] Xiaogang Wang and Xiaoou Tang, Bayesian face recognition using Gabor features. In: *Proc. of the 2003 ACM SIGMM Workshop on Biometrics Methods and Applications* (Nov. 2003), 70-73.

[Wiskott et al., 1997] L. Wiskott, J.-M. Fellous, N. Krueuger and C. von der Malsburg, Face recognition by elastic bunch graph matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence* **19**, 7 (1997), 776-779.

[Yang, 2002] M.-H. Yang, Kernel eigenfaces vs. Kernel fisherfaces: face recognition using Kernel methods. In: *Proc. of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition* (2002), 215-220.

[Ye, 2005] Jieping Ye, Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems. *The Journal of Machine Learning Research* **6** (Sept.2005), 483–502.

[Yin et al., 2006] Hongtao Yin, Ping Fu and Shengwei Meng, Sampled FLDA for face recognition with single training image per person. *Neurocomputing* **69** (2006), 2443–2445.

[Zhao et al., 2003] W. Zhao, R. Chellappa, P. J. Phillips and A. Rosenfeld, Face recognition: A literature survey. *ACM Computing Surveys* **35,** 4 (Dec. 2003), 399–458.

[Zhou et al., 2006] Dake Zhou, Xin Yang, Ningsong Peng and Yuzhong Wang, Improved-LDA based face recognition using both facial global and local information *Pattern Recognition Letters* **27** (2006), 536–543.

Java-pohjaiset web-ohjelmakehykset ja niiden pyynnönkäsittelijät

Pauli Savolainen

Tiivistelmä.
Ohjelmakehykset ovat tärkeä työkalu ohjelmien tuotannossa. Ne luovat valmiiksi testatun ja toteutetun pohjan, jolla ohjelmien toteuttaminen on helppoa. Java-pohjaiset web-ohjelmakehykset perustuvat Model, View, Controller -suunnittelumallille, missä Controller on selaimen ja palvelimen välisen keskustelun ohjaaja. Java-ohjelmointikieli tarjoaa valmiin Servlet-luokan, joka toimii Java-pohjaisten web-ohjelmien kontrollerina; ohjelmakehykset toteuttavat Servletin kukin omalla tavallaan, mikä johtaa värikkääseen kehystarjontaan. Kehykset jaetaan ryhmiin sen mukaan, miten käyttäjän aktivoima pyyntö käsitellään ja miten pyynnönkäsittelijä sidotaan käyttöliittymässä sijaitseviin komponentteihin. Tutkimus tarkastelee pyynnönkäsittelijän, sen sitomisen ja pyynnönkäsittelyprosessin yksityiskohtia erityyppisissä ohjelmakehyksissä.

Avainsanat ja -sanonnat: web-ohjelmakehys, Java, Struts, WebWork, JavaServer Faces, Shale, pyynnönkäsittelijä, pyyntö, Model 2 -suunnittelumalli, MVC-suunnittelumalli.
CR-luokat: D.2.11, D.2.13, K.6.3

1. Johdanto

Web-ohjelmat ovat valloittamassa maailmaa. Tietokoneet ja web vaikuttavat ihmisten elämään päivä päivältä enemmän, sillä yhä useampi palvelu hoidetaan webin kautta: pankkiasiat on jo pystynyt vuosia hoitamaan selaimen välityksellä, laskut voi paperiversion sijaan saada ja hoitaa selaimella, keskustelufoorumit ja -ohjelmat ovat tärkeitä kommunikaatiokanavia, yritysten sisäiset verkot ovat usein selainpohjaisia, koulutusta tapahtuu verkon välityksellä jne. Palvelut ovat siis siirtymässä verkkoon, mistä seurauksena on tarve luoda entistä enemmän ja entistä parempia web-ohjelmia.

Web-ohjelmia sitoo niille ominaiset yhteiset tekijät. Siinä missä ohjelmat yleensä voivat olla ulkonäöltään millaisia tahansa ja toimia millaisissa laitteissa tahansa, on web-ohjelmat sidottu niiden ympäristöön: ne sijaitsevat palvelimella, niitä käytetään selaimella ja selain ja palvelin keskustelevat tietyllä tavalla. Nämä rajaukset pakottavat web-ohjelmat tiettyihin uomiin, joita niiden on seurattava. Web-ohjelmat koostuvat loogisesti eri osista: on osa, joka näkyy käyttäjälle ja osa, joka toteuttaa ohjelman vaatiman logiikan. Käyttäjät käyttävät näkyvää osaa, josta tekevät pyyntöjä ohjelmalle saadakseen toteutettua halutun toiminnallisuuden. Web-ohjelmissa näkyvän osan ja logiikkaosan välillä tapahtuva tiedon vaihto on rajattu Internetin tarjoamaan ympäristöön. Käytännössä tämä tarkoittaa sitä, että jokainen käyttäjän tekemä toiminto on riippumaton muista toiminnoista, eli ne ovat tilattomia. Tämä puolestaan tekee web-ohjelmien toimintojen käsittelystä suhteellisen moni-

mutkaisen prosessin, sillä ne yrittävät toimia jossain määrin tilallisesti tilattomassa ympäristössä.

Web-ohjelmat siis toimivat niille ominaisella tavalla ja niitä yhdistää webin tarjoama ympäristö. Näille ohjelmille on paljon tarvetta ja niitä luodaan koko ajan lisää. Koska ohjelmointi ja ohjelmien luominen on paljon aikaa ja resursseja vaativaa toimintaa, pyritään ohjelmointiprosessia helpottamaan aina kun mahdollista. Tällöin aloitetaan tutkimalla web-ohjelmia yhdistäviä tekijöitä. Niitä on paljon ja pääpiirteittäin ne toteutetaan eri ohjelmissa samalla tavoin. On turvallisuuteen, ulkonäköön ja tiedon käsittelyyn liittyviä seikkoja, jotka on eri ohjelmissa tehty tuhansia kertoja uudestaan ja uudestaan saman kaavan mukaan. Olisi järkevää yhdistää nämä prosessit ja luoda valmiita osia, jotka toimisivat ohjelmoijien työkaluina. Tällaiseen ratkaisuun onkin päädytty ja on tehty valmiita ohjelman osien joukkoja, ohjelmakehyksiä. Innokkaat ohjelmoijat ovat kuitenkin innostuksissaan luoneet hallitsemattomasti hallitsemattoman määrän ohjelmakehyksiä. Eritasoisia Java-kielisiä ohjelmakehyksiä voi löytää webistä uusia päivittäin. Vaikuttaa siltä, että ohjelmakehyksiä alkaa olla enemmän kuin niitä tarvitsevia ohjelmia. On syntynyt vaikeaselkoinen tilanne, jossa oikean kehyksen valinta voi osoittautua hankalaksi. Tutkimukseni pyrkii selvittämään tätä kehysten ylitarjonnan luomaa sekamelskaa pääasiassa tarkastelemalla kehysten yhteisiä piirteitä. Tällä tavoin saadaan paras kuva web-kehysten yleisestä toiminnallisuudesta, mikä on hyvä lähtökohta niiden tarkemmalle tarkastelulle. Kun yhteiset piirteet on kartoitettu, voi kääntää katseensa mahdollisiin eroavaisuuksiin, jolloin eri kehysten todellinen luonne selkenee, ja on helpompi seuloa runsaasta tarjonnasta eri tilanteisiin parhaiten sopiva kehys. Erityisesti tutkimukseni painottuu pyynnönkäsittelyprosessin tarkasteluun, sillä se on web-ohjelmien tärkeimpiä osa-alueita.

Luvussa 2 tarkastellaan kehyksissä käytetyimpiä tärkeimpiä teknologioita ja käytäntöjä, jotta tiedettäisiin mistä ja miten kehykset koostuvat. Luvussa 3 jatketaan ohjelmakehyksen määrittelyä ja pureudun syvemmälle web-ohjelmakehysten todellisuuteen. Kehykset jaetaan alaryhmiin kunkin luonteensa mukaan. Seuraavaksi määritellään web-ohjelmien moottorin, pyynnönkäsittelijän toimintaa. Pyyntö on web-ohjelmia ajava voima, ja tässä luvussa syvennytään kehyksissä yleisesti pyyntöä yhdistäviin seikkoihin. Lopuksi tarkastellaan pyynnön toiminnan suorittamisen yksityiskohtia.

2. Merkitysten määrittelyä

Ohjelmakehykset ovat monitahoisia kokonaisuuksia. Jotta niitä pystyisi tutkimaan, on ensin käsiteltävä niiden käyttämiä teknologioita. Jotta kehyksiä puolestaan pystyisi vertailemaan keskenään, on tutkittava niitä sitovia tekijöitä: miksi kehyksiä tarvitaan, mitä yhteisiä piirteitä niillä on ja miten ne eroavat toisistaan. Tässä luvussa käydään läpi ohjelmakehyksissä yleisesti esiintyviä asioita alkaen käytettyjen teknologioiden määrittelystä ja malli, näkymä, kontrolleri -suunnittelumallista ja Model 2 -mallista.

2.1. Java EE 5 Platform

Tässä tutkimuksessa tarkastellaan ainoastaan ohjelmakehyksiä, jotka on toteutettu Java Enterprise Editionilla[1] (Java EE). Java EE on Java-ohjelmointikielen laajennus, joka tarjoaa perus-Javaa kattavammat palvelut erityisesti isojen, yleensä yritysten tarvitsemien ohjelmistojen toteuttamiseen [Sun Microsystems, 2006]. Web-rajapintaa varten Java EE tarjoaa JSP[2]- ja Servlet[3] -teknologiat, joiden avulla on mahdollista toteuttaa laajoja dynaamisia web-ohjelmia. Servletit ovat palvelimella sijaitsevia Java-luokkia, jotka käsittelevät palvelimelle esitettyjä pyyntöjä ja niiden päätarkoitus on tarjota turvattu web-rajapinta datan ja HTML-sivujen välille [Sun Developer Network, 2006a]. JavaServer Pages (JSP) on puolestaan Java Servlet -teknologian jatke. Siinä missä servletit tulostavat HTML-koodia Java-luokissa, JSP mahdollistaa Java-koodin upottamisen suoraan HTML-tiedostoihin.

Ohjelmien kehittäminen on monimutkainen prosessi, joka vaatii suunnittelua eritoten arkkitehtuurin suhteen, jotta ohjelman joustavuus ja ylläpidettävyys säilyy. JSP- ja Servlet-teknologiat ovat hyvin perustavanlaatuisia teknologioita tarjoten vain pohjan ohjelmatuotannolle. Pelkästään niitä käyttämällä ohjelmien teko on kuin yrittäisi rakentaa taloa perustan päälle ilman suunnitelmaa tai rakennusten vaatimia tukikehikoita. Tarvitaan arkkitehtuuri ja sen toteuttava runko. Ohjelmakehykset tarjoavat tällaisen rungon valmiiksi suunniteltuna ja toteutettuna. Kehykset auttavat ohjelmien rakenteen ja arkkitehtuurin järkeistämisessä keventäen ohjelmoijan taakkaa. Lisäksi ohjelmakehysten käyttämisen etuna on se, että ne antavat ohjelmoijille yhteisen kielen, ja koska ne ovat yleisesti käytettyjä, niitä on testattu paljon ja huolellisesti. Vaikka eroja kehysten välillä luonnollisesti on, esiintyy myös huomattavan paljon rakenteellisia samankaltaisuuksia, mikä helpottaa millä tahansa kehyksellä luodun yksittäisen ohjelman sisällön ymmärtämistä. Suurin osa olemassa olevista kehyksistä pohjautuu malli, näkymä, kontrolleri -suunnittelumallille. [Raible, 2005].

2.2. Malli, näkymä, kontrolleri

Suunnittelumalli on vakiintunut tapa ratkaista yleisesti esiintyvä ongelma. Suunnittelumallin tavoitteena on antaa sen käyttäjälle, tässä tapauksessa ohjelmoijalle tai ohjelmistoarkkitehdille, valmis kehys tietynlaisen ohjelmiston suunnitteluun liittyvän ongelman ratkaisemiseen. On järkevää tukeutua suunnittelumalleihin, sillä ne ovat testattuja ja toimivia kaavoja, jotka vähentävät suunnitteluun tarvittavaa aikaa ja resursseja. [Sun Developer Network, 2006b].

Malli, näkymä ja kontrolleri (*Model*, *View*, *Controller* eli MVC) -suunnittelumalli on web-ohjelmarunkojen kulmakivi. Sen käyttö on niin vakiintunut, että keskusteltaessa keskikokoisista tai suurista web-ohjelmista voi niiden olettaa noudattavan MVC-suunnittelumallia. MVC-suunnittelumallissa pyritään jakamaan toiminnallisuutta ohjelman eri

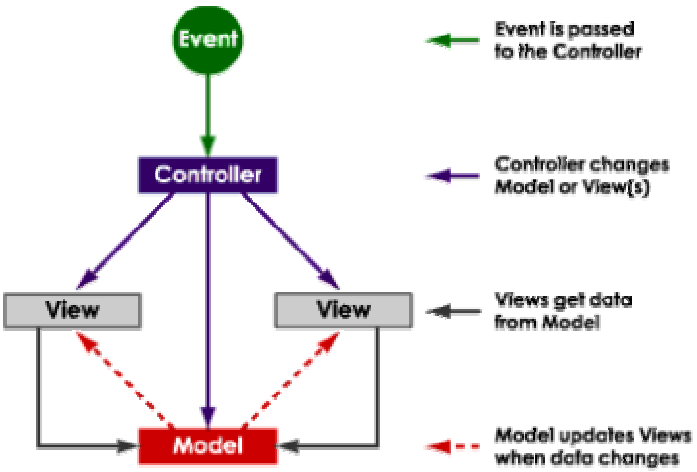
1 Java Platform, Enterprise Edition 5 http://java.sun.com/products/servlet/whitepaper.html
2 Java Server Pages http://java.sun.com/products/jsp/whitepaper.html
3 Java Servlets http://java.sun.com/products/servlet/whitepaper.html

osille, joilla jokaisella on oma tehtävänsä. Tällä tavoin saadaan ohjelman erilaista toiminnallisuutta toteuttavat osat erotettua toisistaan loogisiin kokonaisuuksiin ja ohjelmien ymmärtäminen ja ylläpitäminen on mielekkäämpää. Kuvassa 1 on hyvin yleinen kuvaus MVC-suunnittelumallin toiminnasta.

Malli on ohjelmien informaation käsittelyyn tarkoitettu osa. Se sisältää ohjelmien tarvitseman datan ja keinot muokata ja hakea dataa. Koska malli keskittyy vain omaan tehtäväänsä, ei sen tarvitse ottaa kantaa ohjelmien muiden osien toteutukseen, aivan kuten ohjelmien muiden osien ei tarvitse tietää mallin toteutuksesta. Ihanteellisessa tapauksessa malli on mahdollisimman riippumaton sitä ohjailevista tai sen dataa käyttävistä komponenteista. Siksi malli ja sen edustama data ovat käytettävissä uudelleen tilanteesta riippumatta ja tarvitsematta tehdä muutoksia itse malliin. Mallin muuttuessa se ilmoittaa kiinnostuneita osapuolia muutoksesta, jotka toimivat mallille tuntemattomalla tavalla.

Näkymä on se, mitä ohjelmasta näkyy ulkomaailmalle: näkymä näyttää mallin kuvaaman datan käyttäjälle. MVC-suunnittelumallin tarkoitus on nimenomaan tämä tiedon ja sen esittämisen erottaminen. Sama data voidaan näyttää monessa eri muodossa, ja jaon taustalla oleva logiikka on pilkottu helposti toteutettavaksi ja ylläpidettäväksi. Mallissa ei oteta kantaa sen sisältämän datan esitysmuotoon, vaan se on näkymän tehtävä. Näkymä lähettää mallille pyyntöjä sen tarvitsemasta datasta ja näyttää sen omalla tavallaan. Tällä tavalla esimerkiksi numerotietoa sisältävä ohjelman data voidaan näyttää graafina tai taulukkona muuttamatta itse dataa [Ford, 2006].

Kontrolleri on mallia ja näkymää sitova osa. Se on ohjelman moottori, joka toteuttaa ohjelman toimimiseen tarvittavan logiikan. Se ottaa vastaan näkymän lähettämän syötteen ja sen perusteella toimii siihen ohjelmoidulla ja sille ominaisella tavalla. Kontrolleri tekee käyttäjän syötteen perusteella muutoksia ja päivityksiä malliin käyttämällä mallin tarjoamaa toiminnallisuutta ja valitsee syötteen perusteella käyttäjälle näytettävät näkymät.



Kuva 1: MVC-suunnittelumalli [eNode, 2002].

2.3. Model 2

Model 2 on MVC-suunnittelumallin jatke, joka ei muuta MVC-suunnittelumallin määritelmää, mutta muokkaa sitä web-ohjelmatuotantoon sopivaksi. Model 2 määriteltiin vastaamaan web-ohjelmien suunnittelijoiden tarpeita datan ja sen esittämisen erottamiseksi [Ford, 2006]. JSP-pohjaisille ohjelmille on määritelty kaksi mallia, Model 1 ja Model 2, jotka eroavat toisistaan lähinnä siinä, missä käyttäjän lähettämä pyyntö käsitellään [Seshadri, 1999]. Model 1 -mallissa datan käsittely ja sen näyttäminen on annettu yksittäisten JSP-sivujen tehtäväksi. Tämä johtaa suureen määrään koodia yhdessä tiedostossa, jota on hankala ylläpitää. Model 2:ssa tiedon ja sen kulun hallinta on pilkottu MVC-suunnittelumallin mukaiseksi.

Model 2:lla luodut ohjelmat jakautuvat kolmeen osaan: malli, näkymä ja kontrolleri. Näin ollen myös Model 2 -mallia toteuttavat web-ohjelmakehykset jakautuvat samoihin osiin. Kehykset eivät välttämättä anna tukea jokaisen osan toteuttamiseen, vaan ne voivat keskittyä tiettyyn osa-alueeseen ja usein näin tehdäänkin. On olemassa mallin toteuttamiseen erikoistuneita kehyksiä ja toisaalta näkymään painottuvia kehyksiä. Ohjelmien toiminnan kannalta kontrollerikehykset ovat kuitenkin tärkeimpiä, sillä kontrolleri on yleensä ohjelman erikoistunein osa: malli ja näkymä noudattavat usein standardeja ja yleisiä käytäntöjä, mutta kontrollerin toteutuksessa ohjelmoijilla on eniten tarvetta oman toiminnallisuuden luomiseen. Tarkastellaan seuraavaksi miten MVC-suunnittelumalli taipuu Model 2 -malliin.

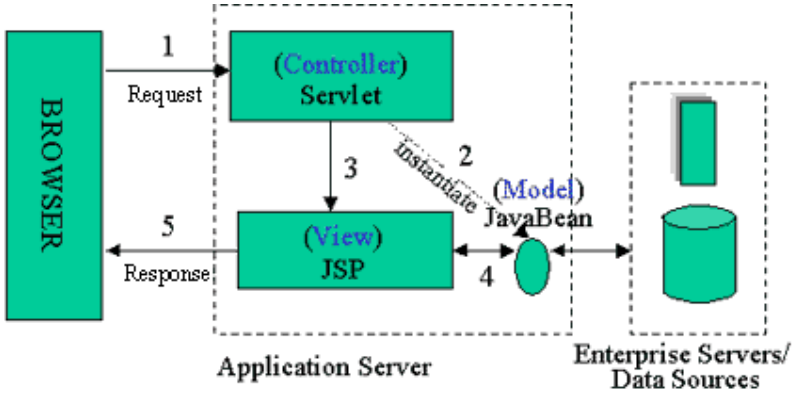
Model 2:ssa malli muodostuu yleensä tietokannasta ja sitä käsittelevistä Java-luokista. Yleinen käytäntö on toteuttaa nämä luokat Java-papuina[4], jotka vastaavat tietokannan tauluja. Papuihin, niiden alaluokkiin tai niitä käsitteleviin luokkiin (*Data Access Objects)*, luodaan funktiot tarvittavia CRUD-operaatioita varten. CRUD-operaatiot ovat luo (c*reate*), hae (*retrieve*), päivitä (*update*) ja poista (*delete*) ja ne ylläpitävät ohjelmassa esitettävää dataa. Jokainen malliin tehty toimenpide käyttää jotain näistä operaatioista tai niiden erikoistuksista.

Näkymä on käytännössä selaimessa näkyvä sivu. Sivuilla on komponentteja, linkkejä ja lomakkeita, joita aktivoimalla käyttäjä lähettää pyyntönsä kontrollerille. Usein ohjelmakehykset tarjoavat työkalut näkymän komponenttien toteuttamiseen. Lisäksi ne yleensä tarjoavat mahdollisuuden käyttää pohjia, joilla sivut voidaan toteuttaa helposti dynaamisesti, mikä vähentää ohjelmoijan työtaakkaa.

Kuten puhtaassa MVC-suunnittelumallissa, Model 2 -mallin mukaisten web-ohjelmien kontrollerit ovat ohjelman logiikan toteuttajia. Kontrollerit hoitavat Model 2 -mallissa myös web-ohjelmille tyypillisiä tehtäviä kuten lokien ja istuntojen *(session)* ylläpito, käsiteltävän datan validointi, käyttäjien todentaminen ja tietoturvasta huolehtiminen. Ohjelmakehykset tarjoavat yleensä rungon näille tehtäville, mutta niiden toteutus saattaa erota eri kehysten välillä. Koska kontrollerin tarkoituksena on käsitellä käyttäjän näkymästä

4 Java Beans http://java.sun.com/developer/onlineTraining/Beans/beans02/

lähettämää pyyntöä, voi sitä kutsua pyynnönkäsittelijäksi. Kuvassa 2 on selvennetty Model 2 -mallin rakennetta.



Kuva 2: Model 2 -malli [Seshadri, 1999]

3. Web-ohjelmakehykset

Ohjelmakehys on valmis kehys ohjelmien tuottamiseen. Se on joukko valmiita komponentteja, jotka on luotu tarkkaan harkitun arkkitehtuurin mukaisesti ja joiden toiminnot on valittu käyttäjien mahdollisten vaatimusten mukaan. Ohjelmakehys tarjoaa rajapintoja, joita toteuttamalla ohjelmoija saa kehyksen tarjoaman toiminnallisuuden käyttöönsä. Niiden käyttäminen säästää ohjelmatuotannossa paljon resursseja, sillä kehykset vähentävät arkkitehtuurin suunnitteluun, ohjelmien testaamiseen ja toiminnallisuuden tuottamiseen kuluvaa aikaa. Se antaa ohjelmoijalle työkalut ratkaista yleisiä usein esiintyviä ongelmia [Raible, 2005]. Web-ohjelmakehykset puolestaan ovat kehyksiä web-ohjelmien tuottamiseen.

Olen tutkimuksessani tarkkaillut kolmea ohjelmakehystä. Valitsemani kehykset ovat Struts, WebWork ja JavaServer Faces ja sitä käyttävät kehykset. Valitsin nämä kehykset kymmenistä, ellei jopa sadoista tarjolla olevista kehyksistä, koska ne ovat yleisimmin käytettyjä, parhaiten dokumentoituja, ne ovat innovatiivisia tai niillä on edessään pitkä tulevaisuus. Viimeksi mainittu onkin mielenkiintoinen aspekti, sillä juuri tulevat kehykset yhdistävät näitä kolmea. Struts on vanhin ja selvästi käytetyin kehys, mutta alkaa olla jo elämänkaarensa loppupuolella. Suurin sitä hengissä pitävä syy ovat ne tuhannet jo olemassa olevat ohjelmat, jotka vaativat sen ylläpitoa. Strutsista on tätä kirjoittaessa tulossa toinen versio, joka pohjautuu niin paljon WebWork -kehykselle, että nykyistä WebWork:iä on hankala erottaa tulevasta Strutsin versiosta. Lisäksi Struts-projektista on kehittynyt alaprojekti Shale, joka on nyttemmin kasvanut omaksi projektikseen. Shale[5] on uusi ja innovatiivinen kehys, joka puolestaan perustuu JavaServer Faces -kehykseen. Kuten edellä on käynyt ilmi, nämä kolme kehystä ovat solmiutuneet erottamattomasti toisiinsa.

Eri ohjelmakehyksillä toteutetut ohjelmat voivat näyttää käyttäjistä identtisiltä. Erot

5 Shale Framework http://shale.apache.org/.

tulevatkin esiin lähinnä ohjelmien arkkitehtuurissa: kuinka sovelluksen eri osat keskustelevat keskenään, miten joustava arkkitehtuuri on tai miten tiukkaan ohjelman kehittäjä on sidottu kehykseen. Nämä ovat tärkeitä kysymyksiä, sillä erikoistuneiden tai suurien ohjelmien tuotanto voi kriittisesti riippua jostakin arkkitehtuurisesta ratkaisusta. Jos esimerkiksi kehyksen tarjoamat toiminnot ovat kattavia ja helposti uudelleen määriteltävissä, mutta niiden toteuttaminen on tiukkaan sidottu kehykseen ja sen tarjoamiin rajapintoihin, saattaa olla hankala tai jopa mahdotonta toteuttaa erikoistunutta ohjelmaa, jonka komponentit tarvitsisivat toimiakseen vapauksia rajapintojen toteuttamisessa. Tästä hyvänä esimerkkinä on Struts-kehyksen 1.2 ja sitä edeltävissä versioissa oleva pyynnönkäsittelijä. Siinä on monia toimintoja, mutta niiden uudelleenmäärittely vaati ohjelmoijan luomaan oman pyynnönkäsittelijän, jonka on perittävä Strutsin pyynnönkäsittelijä-luokka. Tämä sitoo ohjelmoijan hyvin tiukasti Struts-kehykseen. Versiossa 1.3 pyynnön käsittelyn voi toteuttaa rajapintoja käyttäen, mikä löyhentää ohjelmoijaa aikaisemmissa versioissa kahlinneita siteitä. [Apache, 2006].

Web-ohjelmakehykset ovat modulaarisia; ne toimivat toisistaan riippumattomien osien kokonaisuutena, jossa osat pääpiirteissään ovat MVC-suunnittelumallin mukaisia. Yleensä kehykset keskittyvät kontrolleriin tai näkymään, mutta on olemassa joitakin kehyksiä, jotka ovat joko pelkästään mallikehyksiä (Hibernate[6]) tai jotka osittain antavat tukea mallin toteuttamiseen (Spring[7]). Kehykset siis voivat olla hyvin kattavia ja toteuttaa MVC:n kaikki osat tai keskittyä johonkin tiettyyn osa-alueeseen. Niitä voidaan myös käyttää limittäin, esimerkiksi yksi ohjelma voi toteuttaa mallin Hibernatella, kontrollerin Strutsilla ja näkymän luomiseen käyttää JavaServer Faces:iä.

Web-ohjelmakehysten mahdollisen vertailun ongelmana ovatkin juuri niiden siteet toisiinsa. On vaikeaa tarkastella mitä tahansa kehystä yksittäisenä kokonaisuutena, sillä ne eivät ole sellaisia, vaan ne voivat olla sidottu toisiin kehyksiin. Kehysten modulaarisuuden takia ohjelmat voidaan toteuttaa käyttäen useaa eri kehystä ja usein näin tehdäänkin. Esimerkiksi JavaServer Faces on enemmän näkymäkehys, ja monet muut kehykset, kuten Struts, mahdollistavat sen liittämisen itseensä. Tästä syystä kehyksiä tutkittaessa ja vertailtaessa on tärkeää valita tietty osa-alue ja toteuttaa mahdolliset esimerkkiohjelmat mahdollisimman pelkistetyillä kehyksillä.

On olemassa kahdenlaisia web-ohjelmakehyksiä: pyyntö- *(Request based)* ja komponenttikehyksiä *(Component based)*. Nämä kaksi eroavat siinä, miten näkymä saa näytettävän tiedon. Vanhemmat kehykset, kuten Struts[8] ja WebWork[9], ovat pyyntökehyksiä, kun taas JavaServer Faces[10] on komponenttikehys.

6 Hiberante http://www.hibernate.org/

7 Spring http://www.springframework.org/

8 Struts http://struts.apache.org/

9 WebWork http://www.opensymphony.com/webwork/

10 JavaServer Faces http://java.sun.com/javaee/javaserverfaces/

3.1. Pyyntökehykset

Pyyntökehyksillä toteutetut ohjelmat ovat pyyntöihin sidottuja, joissa kontrollerit käsittelevät niille lähetetyt tilattomat (*stateless*) pyynnöt. Pyyntökehykset eroavat toisistaan varsinaisesti vain siinä, miten ohjelmalogiikka sidotaan pyyntöihin ja miten ja missä muodossa ohjelmoijat pääsevät käsittelemään ohjelman dataa [Rife, 2003]. Pyyntökehykset voidaan jakaa kahteen alaryhmään: työntö- (*push*) ja vetokehykset (*pull*). Ero näiden välillä on siinä, miten pyynnönkäsittelijä asettaa mallin esittämän informaation näkymän käytettäväksi. Työntökehyksissä ohjelman pyynnönkäsittelijä tallentaa näytettävän tiedon objektiin, johon myös näkymällä on pääsy. Ohjelmoijien on huolehdittava, että kaikki tarvittava informaatio todella on näkymän saatavilla [Thompson, 2003]. Vetokehyksissä näytettävää tietoa ei varsinaisesti erikseen talleteta mihinkään, vaan niissä annetaan näkymälle mahdollisuus pyytää sitä; näkymät vetävät näytettävän tiedon tarvittaessa itseensä [Thompson, 2003].

Struts on esimerkki työntökehyksestä. Koodissa 1 näkymän tiedostamaan request-objektiin lisätään näytettävä attribuutti. Koodi 2 on osa näkymää, jossa request-objektista haetaan siihen tallennettu tieto. Vetokehyksissä pyynnönkäsittelijä toteuttaa tiedolle aksessorit, joiden avulla näkymän valmisteluvaiheessa näkymä vetää itseensä tarvittavan tiedon niitä käyttäen. WebWork on esimerkki vetokehyksestä. Koodissa 3 listataan yksinkertainen luokka, joka toteuttaa aksessorit attribuutilleen. Koodissa 4 näkymä-elementti hakee informaation käyttäen sille altistetun MyAction-luokan getInformation-metodia.

```
public class MyAction {
  private String information;
  public String getInformation() {
       return this.information;
  }
  public void setInformation(String information) {
       this.information = information;
  }
}
```

Koodi 1: WebWork Action -luokka.

```
<bean:write name="objectName" scope="request" />
```

Koodi 2: Strutsin näkymä

```
request.setAttribute("objectName", "information");
```

Koodi 3: Struts Action -luokan osa.

```
   <ww:text name="information" />
```

Koodi 4: WebWork-näkymä.

Pyyntökehyksen pyynnönkäsittelijä tallettaa siis tavalla tai toisella pyyntöön kaiken informaation, jota näkymä tulee tarvitsemaan. Tällaisissa kehyksissä on ohjelmoijan oltava tarkkana, sillä on etukäteen tiedettävä, mitä tietoa mikäkin näkymä tulee vaatimaan. Varsinkin mikäli pyynnönkäsittelijän ohjelmoija ja näkymän tekijä ovat kaksi eri henkilöä, on tiedon kulun heidän välillään oltava aukotonta. Tuloksena saattaa olla virhe, mikäli näkymä vaatii tietoa, jota pyynnönkäsittelijä ei ole huomannut asettaa näkymän ulottuviin. Tästä syystä pyyntökehyksissä ohjelmakoodin uudelleenkäyttö on hankalampaa kuin komponenttikehyksissä. Toisaalta pyyntökehyksissä tiedon kulku on helposti luettavissa, mikä helpottaa ohjelmien ylläpitoa.

3.2. Komponenttikehykset

Komponenttikehyksissä vastuuta pyynnönkäsittelyprosessin toteutuksesta pyritään siirtämään ohjelmoijilta kehykselle ja sen uudelleenkäytettäviin komponentteihin [Rife, 2006]. Näissä kehyksissä käyttöliittymän osia vastaa tietty kontrollerikomponentti ja näkymää käsiteltäessä sen osat vaativat niitä vastaavilta komponenteilta näkymän toteuttamiseen tarvittavan tiedon. Tällä tavalla teoriassa monet eri käyttöliittymäkomponentit pystyvät käyttämään samoja kontrollerikomponentteja ja koodin uudelleenkäytettävyys paranee. Komponenttikehykset ovat uusia, vanhimmat vain noin neljä vuotta sitten ilmestyneitä (Tapestry[11] maaliskuussa 2002) [Raible, 2005], mistä syystä niiden kehitys ei ole vielä saavuttanut pyyntökehysten kaliiberia. JavaServer Faces itse on vain kaksi vuotta vanha, eikä sen käyttäjäkunta ei ole niin suuri kuin esimerkiksi Strutsin. Komponenttikehyksien tulevaisuuden näkymä on kuitenkin hyvä; ne saavat paljon huomiota osakseen ja niiden kehitys jatkuu kiihtyen. Komponenttinäkökulma web-ohjelmakehyksissä antaa uusia mahdollisuuksia sovellusten kehittämiseen.

Koodissa 5 listataan Shale-kehyksen näkymä. Se on yksinkertainen sisäänkirjautumislomake, jossa on kolme osaa: käyttäjätunnus-kenttä, salasana-kenttä ja sisäänkirjautumisnappi. Jokainen näistä osista liittyy johonkin koodissa 6 esitetyn luokan attribuuttiin tai metodiin.

11 Tapestry http://tapestry.apache.org/

```
<h:form id="loginForm">
  <h:inputText value="#{login.username}">
       <h:outputText value="#{label.username}:" />
  </h:inputText>
  <h:inputSecret value="#{login.password}">
       <h:outputText value="#{label.password}:" />
  </h:inputSecret>
  <h:commandButton action="#{login.login}"
       value="#{label.login}" />
</h:form>
```

Koodi 5: Shale-kehyksen näkymä.

4. Pyynnönkäsittelijä

Pyyntö syntyy, kun käyttäjä (asiakas), aktivoi web-ohjelmassa olevan komponentin: painaa linkkiä, painaa nappia tai lataa sivun uudelleen. Useimmat arkkitehtuurit käyttävät HTTP:tä[12] TCP/IP[13]-yhteyden yli ja oletan yksinkertaisuuden vuoksi kaikkien tekevän näin. HTTP:a käytettäessä asiakkaalta lähtee HTTP Request -pyyntö. Tällaisen pyynnön sisällön muoto on ennalta määrätty ja se sisältää tietynlaista informaatiota, jonka avulla vastaanottaja osaa reagoida siihen. Vastaanottajaa voi myös kutsua pyynnönkäsittelijäksi. Kun vastaanottaja on käsitellyt pyynnön, lähetetään asiakkaalle vastaus, HTTP Response. Esimerkiksi vastaanottaja on palvelin, tietokoneen fyysinen ilmentymä, jolla on yksi tai useita web-säiliöitä *(web container)*, jotka ovat web-ohjelmien ylläpitoon suunniteltuja ohjelmia, web-palvelimia (*web server*). Pyyntö lähetetään palvelimelle tiettyyn säiliöön. Tämän jälkeen säiliö päättää, mitä pyynnöllä tehdään ja miten siihen vastataan.

Java-ohjelmissa web-säiliö käyttää Servlet-luokkaa [Sun Developer Network, 2006c] pyyntöjen käsittelemiseen. Jotta nimenomaan HTTP Request -tyyppiset pyynnöt kyettäisiin käsittelemään oikein, käytetään Servlet-luokasta erikoistunutta luokkaa HttpServlet. Itse HTTP Request -pyynnöstä luodaan sitä vastaava objekti HttpServletRequest, joka annetaan HttpServlet-objektin käsiteltäväksi. Java-pohjaisissa web-ohjelmissa pyynnönkäsittelijä on siis HttpServlet-luokka tai jokin sen alaluokka. Suurin osa ohjelmista toimii tällä tavalla ja yksinkertaisuuden vuoksi oletan aina näin tapahtuvan.

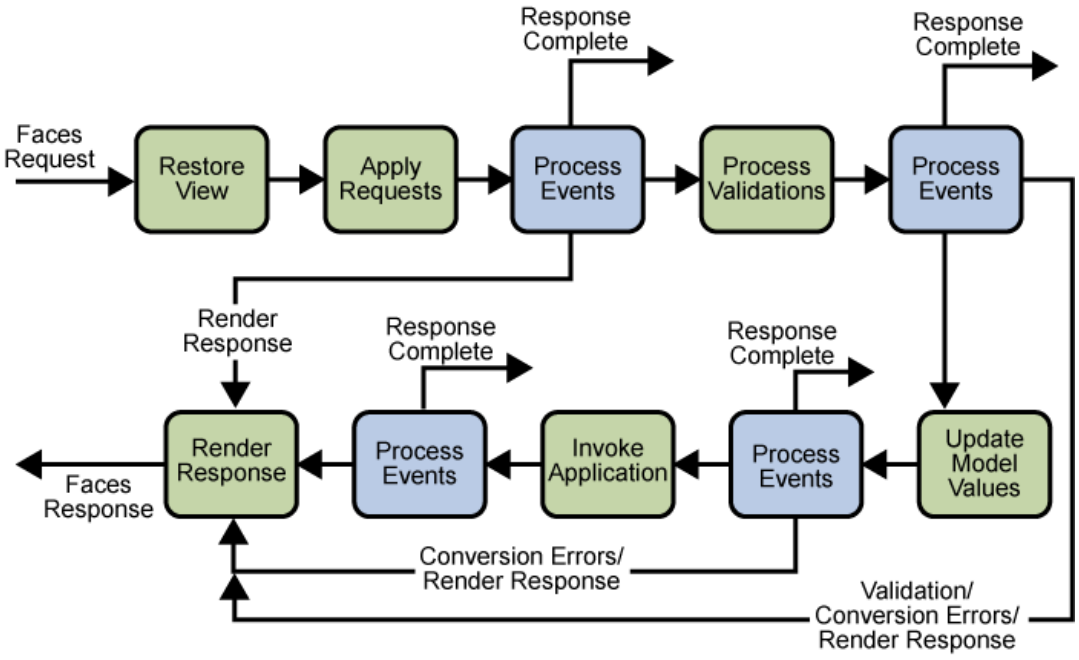
12 Hypertext Transfer Protocol http://www.w3.org/Protocols/rfc2616/rfc2616.html

13 Transmission Control Protocol / Internet Protocol http://computing-dictionary.thefreedictionary.com/TCP/IP

```
@Bean (name="login", scope=Scope.REQUEST)
@View public class Login {

  private String username;
  private String password;

  public String getPassword() {
       return password;
  }
  public void setPassword(String password) {
       this.password = password;
  }

  public String getUsername() {
       return username;
  }

  public void setUsername(String username) {
       this.username = username;
  }

  public String login() {
       Person p = Person.authenticate(username, password);
       if (p == null)
            return FAILURE;
       else {
            return SUCCESS;
       }
  }
}
```

Koodi 6: Shale-kehyksessä näkymän komponentteja vastaava luokka.

Miten pyynnönkäsittelijä sitten käsittelee pyyntöjä ja mitä pyynnönkäsittelyprosessissa tapahtuu? Yksityiskohdissa eri kehysten tarjoamien pyynnönkäsittelijärunkojen välillä on eroja, mutta on myös olemassa universaaleja käytäntöjä siitä, mitä toimenpiteitä käsittelyprosessin aikana suoritetaan. Jokaisessa kehyksessä olevat perustoimenpiteet alkavat pyynnön tietojen ekstraktoimisella: pyynnöstä poimitaan relevantti tieto ja mahdollisesti muutetaan sitä kehyksen tarpeisiin sopivaksi. Toinen tärkeä toimenpide on tiedon varmennus *(validating)*. Tässä vaiheessa pyynnöstä poimittu tieto läpäisee varmennussuodattimen, joka tarkastaa, että kaikki tieto on ohjelman vaatimassa muodossa. Näiden kahden perusvaiheen jälkeen tulee itse pyynnön toiminnon suorittaminen. On tärkeää huomata, että mikäli kahdessa ensimmäisessä vaiheessa on tapahtunut virheitä, prosessi yleensä ohjataan käsittelyvaiheen yli virheilmoituksen palauttamiseen käyttäjälle. Käsittelyvaiheessa pyynnönkäsittelijä toimii pyynnössä tulleen tiedon mukaisesti. Se voi olla esimerkiksi tiedon hakemista tietokannasta, tiedon päivittämistä tai tuhoamista tai pelkästään uudelleenohjauspyyntö johonkin tiettyyn osoitteeseen. Vaihtoehtoja on useita. Lopuksi valmistetaan vastaus (*response*): se on Java-pohjaisissa web-ohjelmissa HttpServletRequest-objektin kaltainen objekti HttpServletResponse ja siihen tallennetaan tietoa tapahtuneen pyynnön käsittelystä. Kuvassa 3 on esimerkki JavaServer Faces -kehyksen pyynnönkäsittelyprosessista. Se on JavaServer Faces -kehyksessä lukittu prosessi, eivätkä ohjelmoijat voi muokata tai erikoistaa sitä omiin tarpeisiinsa sopivaksi. [TheServerSide.com, 2005].



Kuva 3: JavaServer Faces -kehyksen pyynnönkäsittelyprosessi [Sun Developer Network, 2006c].

Web-ohjelmissa pyynnönkäsittelijä siis vastaa MVC-suunnittelumallin kontrolleri-

osaa. Se on ohjelman moottori, se päättää mitä tapahtuu, miten tapahtuu ja miksi tapahtuu. Käsittelijä ei välttämättä ole yksi komponentti tai luokka, vaan se voi muodostua monen luokan ketjusta, jossa tiedon kulku on tarkkaan määritelty prosessi. Edellä mainittujen toimenpiteiden lisäksi ohjelmalta voidaan lisäksi vaatia lukuisia muitakin toimenpiteitä, mistä johtuen käsittelijän toteuttamisesta voi tulla työläs ja monimutkainen prosessi. Tätä varten ohjelmakehyksissä tarjotuissa rungoissa käsittelijän toteuttamiseksi on yleensä paljon kokoonpanomahdollisuuksia, joiden on tarkoitus helpottaa ohjelmoijan työtä.

5. Toiminnon suorittajan sitominen

Kaikki pyynnöt kulkevat aina pyynnönkäsittelijän kautta. Web-ohjelmakehykset tarjoavat oletusarvoisen pyynnönkäsittelijän, joka on usein niin kattavia, että ohjelmoijan on vain erikoistapauksissa turvauduttava tarjotun käsittelijän erikoistamiseen tai oman pyynnönkäsittelijän luomiseen. Pyynnönkäsittelijässä on monia toimenpiteitä, joista yksi on pyynnön toiminnon suorittaminen. Se on pyyntökohtainen toimenpide, johon kehykset eivät ota kantaa, sillä ne eivät voi tietää, mitä pyynnöltä milloinkin odotetaan. Toiminnon suorittamisen toteuttavat ohjelmoijat esimerkiksi käyttäen kehyksen tarjoamia rajapintoja. Pyyntökehyksissä toiminnon suorittaminen tapahtuu joko kehyksen rajapinnat toteuttaneissa tai kehyksen toimintoluokan perineessä Java-luokassa tai kehyksen vaatimassa ennalta määrätyssä muodossa. Komponenttikehyksissä toiminnon suorittaminen tapahtuu toiminnon alkuun panijaan sitoutuneessa Java-luokassa, jota ei välttämättä ole sidottu kehykseen tai mihinkään Java-luokkaan. Kehystyypistä riippumatta toiminnon suoritus tapahtuu Java-luokassa, jota voi kutsua toiminnon suorittajaksi.

Jokaista pyyntöä tulisi vastata pyynnön toiminnon suorittaja. Jotta pyyntö päätyisi oikealle toiminnon suorittajalle, on sen käsittelijä liitettävä käyttöliittymäkomponenttiin tai sen osaan. Tätä kutsutaan sitomiseksi. Eri tapoja sitoa osat toisiinsa lienee yhtä monta kuin on kehyksiä; Java web-ohjelmakehyksissä sitominen tapahtuu yleisimmin XML-kokoonpanotiedostossa, mutta yksityiskohdat kehyksien välillä voivat olla huomattavia. Tällaista tapaa kutsutaan deklaratiiviseksi sitomiseksi.

Deklaratiivisessa sitomisessa sidottavat osat, toiminnon suorittaja ja sen käsittelijä, määritellään kokoonpano- tai ominaisuustiedostoissa (*property*). Näissä tiedostoissa listataan mahdolliset pyynnöt ja määritellään niille sopiva toiminnon suorittaja. Kokoonpanotiedostoon voidaan yleensä määritellä ylimääräisiä muuttujia, esimerkiksi tietoa siitä, miten data tulisi varmentaa, mihin pyyntö ohjataan käsittelyn jälkeen tai mitä tehdään virheen sattuessa. Kun kehys saa pyynnön, se käyttää kokoonpanotiedostoja löytääkseen pyynnölle sopivan toiminnon suorittajan. Deklaratiivisessa sitomisessa haittapuolena on kokoonpanotiedostojen ylläpitämisen haastavuus: tiedostot voivat suurissa projekteissa paisua hallitsemattomiin mittoihin. Toisaalta yksittäisten pyyntöjen räätälöinti tiettyä tarkoitusta varten on huomattavan joustavaa. Koodissa 7 on esimerkki Struts-kehyksen käyttämästä deklaratiivisesta sitomisesta. Siinä listatun kokoonpanotiedoston avulla kehys osaa sisäänkirjautumistoiminnon tapahtuessa kutsua oikean luokan toimintometodia.

```
<action-mappings>
        <action
            path="/login"
            type="org.pauli.example.Login"
            name="loginForm"
            scope="request"
            input="/login.jsp" />
    </action-mappings>
```

Koodi 7: Struts-kehyksen toimintojen sitominen XML-tiedostossa.

On olemassa myös toisenlainen tapa sitoa pyyntö ja sen toiminnon suorittaja toisiinsa: konventionaalinen sitominen. Tällaisessa sitomisessa ei tarvita kokoonpanotiedostoja, vaan sitominen tapahtuu sovittujen sääntöjen mukaan. Tämä tarkoittaa käytännössä sitä, että kehykseen on rakennettu mekanismi, jolla se pyyntöjä lähettävien komponenttien nimien tai tiettyjen attribuuttien perusteella löytää pyynnölle oikean toiminnon suorittajan. Konventionaalisessa sitomisessa kehys vähentää ohjelmoijan työmäärää sitomisen osalta. Tällaisissa kehyksissä on tärkeää kehittää hyvät säännöt ja mekanismi ja tehdä pyyntöjen räätälöinti mahdollisimman helpoksi. Vaarana on tuudittautua kehyksen tarjoamaan sidontamekanismiin, jolloin monimutkaisemman ongelman esiintyessä saattaa olla vaikeaa löytää siihen ratkaisua.

Uusimmissa komponenttikehyksissä on myös kolmas tapa toteuttaa sitominen. Java 5:ssä voi kirjoittaa metainformaatiota (annotaatiot[14]) koodiin, mikä mahdollistaa deklaratiivisen sitomisen ilman kokoonpanotiedostoja. Pyynnön toiminnon suorittajassa ilmoitetaan annotaatioita käyttämällä, minkä pyynnön toimintoja ne ovat valmiita suorittamaan. Näin kehys tietää, mikä toiminnon suorittaja on sidottu mihinkin käyttöliittymäkomponentista suoritettuun pyyntöön. Yksi esimerkki tällaisesta kehyksestä on Shale. Annotaatiositomisessa yhdistyy sekä deklaratiivisen sitomisen, että konventionaalisen sitomisen edut: sitomista voi hallita helposti suoraan toiminnon suorittajasta, eikä siihen tarvita erillisiä kokoonpanotiedostoja. Tällaista sitomista käyttäviä ohjelmia on harvassa, sillä Java 5 ja sitä tukevat web-ohjelmakehykset ovat suhteellisen uusia teknologioita. Koodeissa 5 ja 6 on Shale-kehyksessä toteutettu annotaatiositominen sisäänkirjautumislomakkeen ja sitä vastaavan Java-luokan välillä.

Tutkimuksessa käsiteltävät web-ohjelmakehykset käyttävät kaikki deskriptiivistä sidontaa. Syy tähän on se, että erittäin harvat[15] Java web-ohjelmakehykset käyttävät konseptuaalista sidontaa. Kehykset käyttävät joko deklaratiivista sidontaa tai joissain tapauksissa annotaatiosidontaa. Valituista kehyksistä ainoa komponenttikehys on JavaSever Faces,

14 Annotations http://java.sun.com/j2se/1.5.0/docs/guide/language/annotations.html

15 Kirjoittajan tiedossa ei ole ainuttakaan Java web-ohjelmakehystä, joka käyttäisi puhtaasti konseptuaalista sidontaa.

mutta se ei perusominaisuuksillaan tue annotaatiosidontaa [Sun Developer Network, 2006c].

6. Yhteenveto

Web-ohjelmat mukailevat niille ominaisia kaavoja. Tästä syystä niitä varten on mahdollista luoda kehyksiä, jotka tarjoavat kattavia palveluita. Kehykset ovat ohjelmien raakaversioita; ne ovat itsenäisiä ohjelmia jo sellaisinaan, mutta ilman toiminnallisuutta. Kehykset kuitenkin ohjaavat niillä luotujen ohjelmien tuotannon tiettyyn suuntaan: ne rohkaisevat käyttämään tiettyjä teknologioita, ne pakottavat ohjelmoijat tiettyihin käytäntöihin ja ne tarjoavat erilaisia palveluita, mikä johtaa yksilöllisiin ratkaisuihin ohjelmoijien osalta tietyissä ongelmissa. Lisäksi aloitteleville käyttäjille eri kehysten luoma ympäristö vaikuttaa heidän oppimiseensa: jotkut kehykset on helpompi sisäistää kuin toiset. Pohjimmiltaan kehyksissä heijastuvat samat perusominaisuudet: jako MVC-suunnittelumallin mukaisesti ja pyynnönkäsittelijän perusvaiheet. Eniten ohjelmistotuotantoon vaikuttavat erot tulevat esiin siinä miten ja kuinka tiukkaan ohjelmoija on sidottu kehykseen ja miten vaivattomasti kehysten erikoistaminen käy. Lisäksi pyyntö- ja komponenttikehysten välinen ero voi olla huomattava ohjelmista ja käytetyistä kehyksistä riippuen.

Ohjelmien erilaisuuden ja ihmisen tuoman arvaamattomuuden takia on lukemattomia eri tapoja toteuttaa ohjelmia. Vaikka web-ohjelmat noudattavatkin tiettyä kaavaa ja säännönmukaisuutta, on niiden välillä huomattavia eroja. On vaikeaa ellei mahdotonta luoda ohjelmakehystä, joka sopisi saumattomasti kaikkien ohjelmien toteuttamiseen. Siksi on tärkeää, että ohjelmakehykset antavat vapauksia yksittäisten ohjelmien luomiseen. Ohjelmakehyksien on tarjottava ohjelmoijille mahdollisuuden laajentaa tai uudelleen määrittää niiden toiminnallisuutta, myös pyynnönkäsittelijää.

On siis ensiarvoisen tärkeää, että kehys tarjoaa selkeät linjat joustavaan ohjelmatuotantoon. Ylläpidettävyyden edellytyksenä on ohjelman eri osien minimalisointi: luodaan mahdollisimman vähän ylläpidettäviä komponentteja, luodaan niiden ylläpitoa varten yhtenäinen rakenne ja tehdään osien välisestä tiedon kulusta mahdollisimman mutkatonta. Näkymää, mallia ja kontrolleria sitovat kokoonpanotiedostot eivät myöskään saisi paisua hallitsemattoman suuriksi ja niiden rakenteen on oltava selkeä. Tarkasteltujen kehysten vaatimien osien määrässä ei ole havaittavissa suuria eroja ja osien sitominen on hyvin samankaltaista.

Ohjelmien muokattavuudessa ja erikoistamisessa toisaalta löytyy eroja. Esimerkiksi JavaServer Faces -kehyksessä pyynnönkäsittelyprosessia ei voi varsinaisesti muokata, mikä pakottaa ohjelmoijan tiettyyn muottiin, mutta toisaalta siinä ei ohjelmoijaa sidota kehykseen millään lailla. Struts-kehyksessä ohjelmoija sidotaan kehykseen pakottamalla ohjelmoijaa toteuttamaan kehyksen tarjoamat rajapinnat, mutta pyynnönkäsittelyprosessin muokkaaminen on joustavampaa kuin JavaServer Facessa. WebWork kehyksessä pyynnön käsittelyn erikoistaminen ja muokkaaminen on erittäin vapaata, eikä ohjelmoijaa sidota kehykseen. WebWork tarjoaa valmiita palveluita, joilla se lupaa ohjelmoijalle kat-

tavaa toiminnallisuutta, mutta ei pakota palveluitaan käyttämään. Tämä kuulostaa hyvältä ja sitä se onkin, mutta toisaalta se antaa ohjelmoijalle lisää vastuuta, ja tämän vastuun myötä myös virheiden riski kasvaa.

Kehystä valittaessa on tarkasteltava kehysten ominaisuuksia ja käytettävissä olevia resursseja, mistä yksi tärkeä tekijä on ohjelmoijien taustatieto. Mikäli ohjelmoijilla on paljon kokemusta tietystä kehyksestä, joka ei välttämättä ole paras vaihtoehto jonkin ohjelman tuottamiseen, on ajan ja resurssien säästämiseksi silti yleensä järkevämpää valita kehys, mistä on eniten kokemusta. Oppimisprosessi voi pitkittyä paljonkin uusia kehyksiä opeteltaessa ja jotain ilmiselvän järkevää toiminnallisuutta voi kokemattomuuden takia jäädä käyttämättä. Tähän liittyen yksi tärkeimmistä tekijöistä kehyksiä valittaessa on niiden käyttöaste: kuinka paljon ihmiset käyttävät niitä. Käyttöaste korreloi sen kanssa, miten paljon informaatiota kehyksistä löytyy. Web-ohjelmakehysten suurin informaation lähden on web ja sieltä löytyvät esimerkit, artikkelit, valmisohjelmat ja keskustelufoorumit, joita aloittelevat kehysten käyttäjät joutuvat käymään läpi oppimisprosessin aikana kyllästymiseen asti. He tarvitsevat kaiken mahdollisen saatavilla olevan avun, ja mitä enemmän apua on tarjolla, sitä helpommin sitä löytää.

Web-ohjelmakehykset ovat siis työkaluja web-ohjelmien tuottamiseen. Ne ovat joukko valmiiksi toteutettuja käytäntöjä, joita ohjelmoijat käyttävät ohjelmia tehdessään. Ne ovat hyödyllisiä, sillä ne voivat vähentää suunnitteluun, toteutukseen ja testaukseen käytettävää aikaa. Kehyksissä on tärkeää niiden ymmärrettävyys, ylläpitoa ja muokattavuutta tukeva toiminnallisuus, sekä niistä löytyvän informaation määrä. Kehyksiä tutkittaessa joutuu niitä vääjäämättä jossain määrin vertailemaan keskenään, ja vertailun tuloksena pitäisi tietenkin pystyä sanomaan, mikä on paras ohjelmakehys. Sitä on kuitenkin mahdotonta tällä hetkellä määrittää, sillä sellaista kehystä ei ole. Ei ole parasta web-ohjelmakehystä, on parhaiten tilanteeseen ja olosuhteisiin sopivia kehyksiä. Parhaiten tilanteeseen ja olosuhteisiin sopiva kehys on sellainen, jota oikeissa työtehtävissä olevat ihmiset osaavat tuotteliaasti käyttää. Se on sellainen, jolla annetun tehtävän saa helpoiten tehtyä.

Viiteluettelo

[Apache, 2006] The Apache Software Foundation, Struts. http://struts.apache.org/. Checked 23.10.2006.

[eNode, 2002] eNode Inc., Model-View-Controller Pattern. http://www.enode.com/x/markup/tutorial/mvc.html. Checked 23.11.2006

[Ford, 2006] Neal Ford, Comparison of Java Web Frameworks. http://bdn1.borland.com/article/borcon/files/6000/paper/6000.html. Checked 5.10.2006.

[Raible, 2005] Matt Raible, Java Web Frameworks. http://www.virtuas.com/osl-jwf-01.pdf. Checked 5.10.2006.

[Rife, 2006] Rife, Full-stack open-source component framework to quickly and consistently develop and maintain Java web applications. http://rifers.org/. Checked 19.11.2006.

[Seshadri, 1999] Govind Seshadri, Understanding JavaServer Pages Model 2 architecture.

http://www.javaworld.com/javaworld/jw-12-1999/jw-12-ssj-jspmvc.html. Checked 16. 10.2006.

[Sun Developer Network, 2006a] Sun Developer Network, The Java Servlet API White Paper. http://java.sun.com/products/servlet/whitepaper.html. Checked 31.10.2006.

[Sun Developer Network, 2006b] Sun Developer Network, The Java EE 5 Tutorial http://java.sun.com/blueprints/patterns/. Checked 30.10.2006.

[Sun Developer Network, 2006c] Sun Developer Network, The Java EE 5 Tutorial. http://java.sun.com/javaee/5/docs/tutorial/doc/. Checked 23.10.2006.

[Sun Microsystems, 2006] Sun Microsystems, Inc., Simplified Guide to the Java™ 2 Platform Enterprise Edition. http://java.sun.com/j2ee/reference/whitepapers/j2ee_guide.pdf. Checked 31.10.2006.

[TheServerSide.com, 2005] TheServerSide.com, JavaSeverFaces vs Tapestry. http://www.theserverside.com/tt/articles/article.tss?l=JSFTapestry. Checked 19.11. 2006.

[Thompson, 2003] Kris Thompson, Clarification on MVC Pull and MVC Push. http://www.theserverside.com/patterns/thread.tss?thread_id=22143, Checked 17.11. 2006.

Aivokäyttöliittymistä

Outi Tuisku

Tiivistelmä.
Tässä tutkielmassa käsitellään aivokäyttöliittymiä. Aivokäyttöliittymät voidaan jakaa kolmeen eri kategoriaan: kallon sisäinen, osittain kallon sisäinen ja kallon ulkopuolinen aivokäyttöliittymä. Kallon ulkopuolisia aivokäyttöliittymiä voidaan käyttää mm. EEG-, MEG-, PET-tekniikoilla ja kallon sisäistä aivokäyttöliittymää voidaan käyttää esim. ECoG-tekniikalla. Näiden tekniikoiden avulla on kehitetty erilaisia ratkaisuja aivokäyttöliittymän toteuttamiseksi, esimerkkinä mainittakoon virtuaaliset näppäimistöt.

Avainsanat ja -sanonnat: Aivokäyttöliittymä, Brain-Computer Interface, BCI
CR-luokat: B.4.2, C.3, J.7

1. Johdanto

Aivokäyttöliittymien tutkiminen on aloitettu jo 1970-luvulla [Surakka et al., 2004]. Aivokäyttöliittymätutkimus kasvanut kuitenkin vasta 15 viime vuoden aikana [Arenas et al., 2005]. Aivokäyttöliittymien kehittämisestä selkeästi eniten hyötyvät vammautuneet henkilöt, jotka eivät mahdollisesti muuten pysty kommunikoimaan muiden ihmisten kanssa. Patnaik [2005] toteaa, että aivokäyttöliittymät tarjoavat käyttäjille, jotka ovat kokonaan halvaantuneita (esim. aivohalvaus), mahdollisuuden ilmaista tarpeensa heitä hoitaville henkilöille. Aivokäyttöliittymätutkimus auttaa myös henkilöitä, jotka ovat menettäneet kokonaan lihaskontrollinsa (locked-in) [Surakka et al., 2004].

Vammaisten henkilöiden avuksi on ennen aivokäyttöliittymiä kehitetty erilaisia katseen avulla ohjattavia käyttöliittymiä. Aivokäyttöliittymää voidaan pitää äärimmäisenä tapana auttaa ihmisiä kommunikoimaan [Surakka et al., 2004], mutta siitä on myös selkeää hyötyä esimerkiksi henkilöille, jotka kärsivät aivovauriosta.

Tämän tutkimuksen luvussa 2 esitellään erilaisia käsitteitä ja tekniikoita, jotka ovat relevantteja tämän tutkimuksen kannalta. Luvussa 3 käsitellään erityyppisiä aivokäyttöliittymiä. Luvussa 4 luodaan lyhyt katsaus toteutettuihin aivokäyttöliittymiin.

2. Käsitteistä ja tekniikoista

Aivokäyttöliittymä voidaan toteuttaa erilaisilla tekniikoilla, jotka muuttavat aivojen sähköiset tai magneettiset impulssit käyttöliittymän ymmärtämään muotoon. Tässä luvussa esitellään eri tekniikoita, joilla aivokäyttöliittymä voidaan toteuttaa, sekä esitellään lyhyesti tarvittavia käsitteitä.

Aivot on elin, joka hallitsee ihmisen keskushermostoa. Niiden päätehtävä on säädellä aistien tuomaa informaatiota ja säilyttää tasapaino ihmisen ja ympäristön välillä. Aivot jakaantuvat kahteen puoliskoon, oikeaan - ja vasempaan aivopuoliskoon. Aivopuoliskot jakaantuvat neljään osaan, jotka ovat otsalohko, ohimolohko, päälakilohko ja takaraivolohko. Aivojen oikea ja vasen puolisko säätelevät eri toimintoja ja käsittelevät informaatiota eri tavoin. Vasenta aivopuoliskoa sanotaan järjen ja oikeaa aivopuoliskoa tunteen puoliskoksi. Aivot muodostuvat isoista aivoista, pikkuaivoista ja aivorungosta. Aivorunko puolestaan koostuu ydinjatkoksesta, aivosillasta, keskiaivoista ja väliaivoista. [Himberg et al., 2001]

Karkeasti kuvailtuna aivokäyttöliittymän avulla pystytään käyttämään tietokonesovelluksia pelkästään ”ajatuksen voimalla” käyttäen hyväksi aivotoiminnan synnyttämiä sähköisiä tai magneettisia signaaleja. Jokaista liikettä, jonka ihminen tekee, vastaa aivoissa tietty sähköinen tai magneettinen signaali. Kun ihminen ajattelee vasemman käden liikuttamista, oikeanpuoleinen aivolohko aktivoituu ja näkyy mitattavana signaalina, jolloin käyttäjä pystyy esimerkiksi liikuttamaan hiiren kursoria. Aivokäyttöliittymä muuntaa nämä signaalit tietokoneen ymmärtämään muotoon tehtävään soveltuvan algoritmin avulla [Arenas et al., 2005; Birbaumer et al., 2005; Sun and Zhang, 2005]. Näitä algoritmeja ei tässä tutkimuksessa kuitenkaan käsitellä sen tarkemmin.

EEG eli elektroenkefalogrammi mittaa pään pinnalle kiinnitettyjen pienten metallielektrodien avulla aivokuoressa olevien hermoverkkojen sähköisen toiminnan vaihtelua [Sun and Zhang, 2005]. EEG:n avulla voidaan mitata myös muutoksia aivojen hermosolujen toiminnassa [del R. Millan, 2003]. Sen avulla voidaan siis mitata laajempia kokonaisuuksia sekä pienintäkin yksityiskohtaa. Käytettäessä EEG:tä havaitaan pienimmätkin muutokset jotka esiintyvät aktivoitaessa aivojen eri osia [Arenas et al., 2005].

EEG:n hyvänä puolena pidetään sen nopeaa vasteaikaa, ja sitä, että sitä voidaan käyttää monissa eri ympäristöissä, eikä ainoastaan laboratorio-olosuhteissa. Lisäksi EEG:n käyttöä varten tarvitaan suhteellisen yksinkertaisia ja halpoja laitteita [Birbaumer et al., 2002]. Ongelmia, jotka tulevat esille EEG:n avulla mittaamisesta, ovat EEG:stä lähtevä mahdollisesti häiritsevä ääni, sekä inhimilliset tekijät. Näitä tekijöitä ovat mm. käyttäjän motivaatio, keskittyminen ja turhautuminen, jotka voivat vaikuttaa merkittävästi tutkimukseen [Sun and

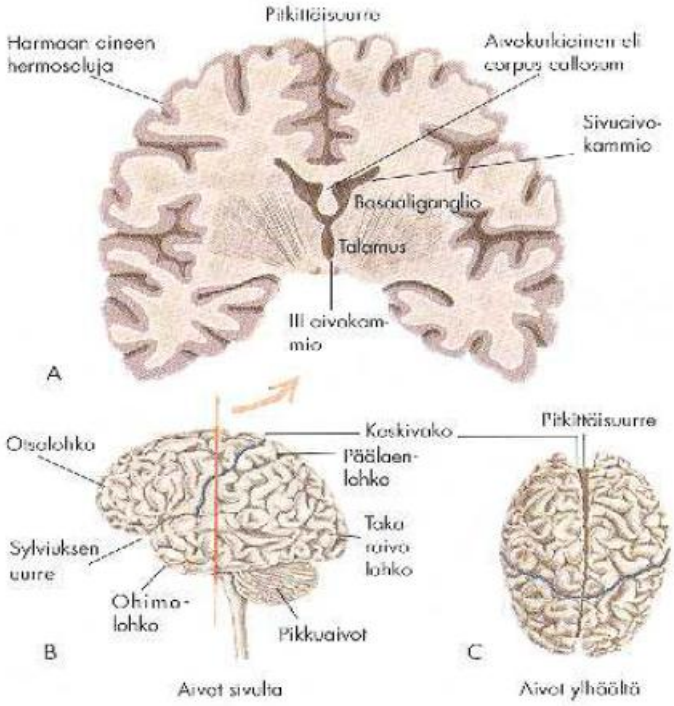
Zhang, 2005] ja ennen kaikkea siitä saatavaan tulokseen. Jos koehenkilön keskittyminen herpaantuu, ei EEG:n avulla saada relevanttia tietoa siitä, mikä aivojen osa tuottaa mitäkin aktiivisuutta.

MEG eli magneettienkelografia mittaa pään ulkopuolelta magneettikenttää, joka aiheutuu hermosolujen liikkumisesta aivoissa. MEG:llä pystytään EEG:tä paremmin paikallistamaan, mistä kohtaa aivoista tietty signaali tulee. Tämä johtuu siitä, että MEG mittaa magneettikenttää, eikä se häiriinny EEG:n tavoin ympäristön ”hälyäänistä” [Birbaumer et al., 2005]. Birbaumerin ja muiden [2005] mukaan MEG:n signaalit ovat EEG:tä parempilaatuisia, ja siksi MEG:llä on saatu lupaavia oppimistuloksia aivokäyttöliittymien käyttämisestä.

PET eli positroniemissiontomografia mittaa aivojen verenkierron muutoksia. Tämä tapahtuu tuomalla elimistöön radioaktiivista ainetta, joka elimistöön jakautuessaan tekee aivojen verenkierron havaittavaksi. PET mittaa verenkierron muutoksia potilaan eri tehtävien aikana, esimerkiksi puhumisen tai katselun aikana.

ECoG (electrocorticography) on menetelmä, jonka avulla mitataan aivojen sähköistä aktiivisuutta suoraan aivokuoresta. Tämä tapahtuu siten, että mittauselektrodit asetetaan suoraan aivokuorelle, harmaaseen aineeseen (ks. kuva 1). Kun elektrodit asetetaan suoraan harmaaseen aineeseen, saadaan aivoista paljon tarkempaa tietoa ja paljon tehokkaammin kuin esimerkiksi EEG:n avulla.

ECoG:n haittapuolena pidetään kuitenkin sitä, että joudutaan tekemään suuri ja riskialtis neurologinen leikkaus, jotta saadaan asetettua elektrodit suoraan aivokuorelle.



Kuva 1. Harmaa-aineen sijainti aivoissa [Himberg et al., 2001].

3. Erityyppisiä aivokäyttöliittymiä

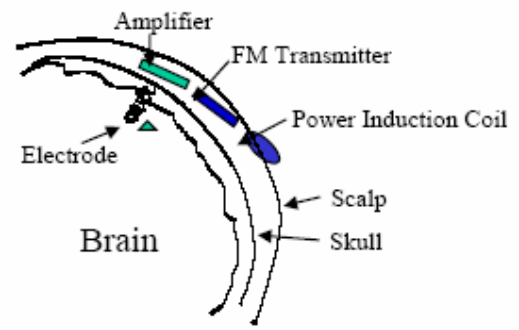
Aivokäyttöliittymät voidaan jakaa kahteen eri kategoriaan, *invasive-* ja *noninvasive-aivokäyttöliittymä* [MacFarland and Wolpaw, 2004]. Tämän lisäksi on olemassa kolmaskin kategoria, *partially-invasive-aivokäyttöliittymä* [Wikipedia], mutta sitä ei ole tutkittu kovinkaan paljon. Näille sanoille ei ole suomenkielistä vastinetta, mutta karkeasti suomennettuna invasive tarkoittaa ihon alaista. Käytänkin tässä tutkimuksessa invasive-sanalle suomenkielistä vastinetta kallon sisäinen. Vastaavasti partially-invasive-sanalle käytän suomennosta osittain kallon sisäinen ja noninvasive-sanalle suomennosta kallon ulkopuolinen.

Tässä luvussa esitellään perusteet jokaisesta erityyppisestä aivokäyttöliittymästä.

3.1. Kallon sisäinen aivokäyttöliittymä

Kallon sisäinen aivokäyttöliittymä asennetaan suoraan aivojen harmaaseen aineeseen neurokirurgisessa operaatiossa [Wikipedia]. Kuvassa 1 on havainnollistettu, missä kohdin harmaa aine sijaitsee aivoissa. Aivojen harmaaseen aineeseen asennetaan mikrosiru, joka valvoo käyttäjän aivojen toimintoja ja muuttaa käyttäjän aikomukset tietokoneen ymmärtämään muotoon. Mikrosiru kiinnitetään kuvan 2 osoittamalla tavalla. Kallon sisäisten aivoliittymien käyttö perustuu yksittäisten aivosolujen ’ryhmistä’ (ensembles of single brain cells) tai mo-

ninkertaisten neuronien aktiivisuudesta (activity of multiple neurons) saataviin havaintoihin [Lebedev and Nicolelis, 2006]. Tällä aivokäyttöliittymätyypillä saadaan kaikkein tarkinta ja reaaliaikaisinta tietoa aivojen toiminnasta [Lebedev and Nicolelis, 2006]. Kallon sisäisen aivokäyttöliittymän tutkimisen tarkoituksena on saada aikaan täysin langaton laite, mutta ainakin vielä tällä hetkellä kallon sisäiset aivokäyttöliittymät on kytketty johdoilla ulkoiseen laitteeseen.



Kuva 2. Kallon sisäinen aivokäyttöliittymä [Kennedy and Moore, 2004].

Kallon sisäisissä aivokäyttöliittymissä pidetään ongelmana sitä, että saattaa syntyä mahdollisia teknisiä ongelmia sekä infektioriskejä potilaalle. Koska kallon sisäinen aivokäyttöliittymä on asennettu harmaaseen aineeseen, voi siitä seurata potilaalle vakavakin infektio. Kallon sisäisen aivokäyttöliittymän odotetaan toimivan hyvin pitkiä aikoja kerrallaan [McFarland and Wolpaw, 2004]. Tekniikka ei ole vielä kuitenkaan kehittynyt tarpeeksi, jotta voitaisiin tehdä lupauksia pitkäkestoisista aivon sisäisistä käyttöliittymistä. Koska kallon sisäisten aivokäyttöliittymien asennus vaatii pitkän neurologisen leikkauksen, ei haluta ottaa turhia riskejä, jos aivokäyttöliittymä toimii vain lyhyen aikaa. MacFarlandin ja Wolpawin [2004] mukaan kallon sisäisten aivokäyttöliittymien tutkiminen perustuu siihen uskomukseen, että ainoastaan kallon sisäiset aivokäyttöliittymät mahdollistavat reaaliaikaista ja moniulotteista tietoa aivoista. Tämä mahdollistaa esimerkiksi robottikäsien käyttämisen.

Kallon sisäisiä aivokäyttöliittymiä on tutkittu lähinnä eläimillä, varsinkin rhesus-apinoilla [Lebedev and Nicolelis, 2006] ja rotilla. Tutkijat ovat muun muassa tutkineet rottien ja apinoiden avulla aivokuoren eri osia, jotka liittyvät liikkeiden suunnitteluun ja suorittamiseen [Gerstner et al., 2004]. Nämä aivokuoren osat ovat motorinen, esimotorinen ja taaempi aivokuori (posterior parietal cortex). Gerstnerin ja muiden [2004] mukaan näiden kokeiden avulla on ollut mahdollista estää eläimen liikkumisaikeet, ennustaa apinan käden tuleva liikerata ja liikuttaa tietokoneen kursori haluttuun kohtaan. Nämä kokeet ovat hyvä esimerkki siitä, mitä mahdollisuuksia kallon sisäisillä aivokäyttöliittymillä on tulevaisuudessa.

Ihmisten on vaikea hyväksyä kallon sisäisten aivokäyttöliittymien käyttämistä siihen liittyvien riskien vuoksi. Lebedevin ja Nicoleliksen [2004] mukaan kallon sisäisen aivokäyttöliittymän hyväksyttävyyden edellytyksenä on, että se tuntuu oikealta raajalta. Lisäksi Lebedev ja Nicolis [2004] ovat sitä mieltä, että on toteutettava täysin langaton aivokäyttöliittymä, koska se vähentää infektioriskiä. Infektioriski johtuu heidän mukaansa niistä kaapeleista, jotka yhdistävät kallon sisällä olevan aivokäyttöliittymän ulkoiseen laitteeseen. On kuitenkin muistettava, että aivot ovat jokaisella ihmisellä yksilölliset, joten riippuu hyvin paljon ihmisestä, millainen kallon sisäinen aivokäyttöliittymä hänelle sopii.

3.2. Osittain kallon sisäinen aivokäyttöliittymä

Tästä aivokäyttöliittymätyypistä on selvästi vähiten tietoa, koska sitä ole juurikaan tutkittu. Toinen mahdollisuus on, että osittain kallon sisäiset aivokäyttöliittymät mielletään tutkimuksissa useimmiten kallon sisäisiksi aivokäyttöliittymiksi.

Osittain kallon sisäisessä aivokäyttöliittymässä yksi osa laitteesta sijoitetaan kallon sisälle, mutta loput laitteesta jää kallon ulkopuolelle. Laitetta ei asenneta harmaaseen aineeseen, minkä takia tällä aivokäyttöliittymätyypillä ei ole potilaalle niin suurta infektioriskiä kuin kallon sisäisellä aivokäyttöliittymällä [Wikipedia].

3.3. Kallon ulkopuolinen aivokäyttöliittymä

Kallon ulkopuolinen aivokäyttöliittymän toimintaperiaate perustuu kallon ulkopuolisiin laitteisiin (esim. EEG ja MEG), joilla mitataan sähköisiä ja magneettisia muutoksia pään pinnalta [MacFarland and Wolpaw, 2004]. EEG-laite muokkaa aivojen sähköiset muutokset tietokoneen ymmärtämään muotoon, jolloin voidaan esimerkiksi liikuttaa tietokoneen kursoria. Kallon ulkopuolisia aivokäyttöliittymiä on jo käytössä peruskommunikointia varten [MacFarland and Wolpaw, 2004]. Kallon ulkopuoliset aivokäyttöliittymät ovat potilaalle riskittömiä, koska potilaille ei tarvitse tehdä kirurgisia operaatioita.

Koska kallon ulkopuoliset aivokäyttöliittymät mittaavat aivojen aktiviteettia ulkopuolelta, saadaan signaalit aivokudosten, luun ja ihon läpi, ja ne voivat olla erittäin epätarkkoja. Tämän takia menetetään osa aivojen signaaleista [Lebedev and Nicolelis, 2006].

Kallon ulkopuoliset aivokäyttöliittymät ovat auttaneet esimerkiksi halvautuneita (locked-in) potilaita kommunikoimaan ulkomaailman kanssa [Lebedev and Nicolelis, 2006].

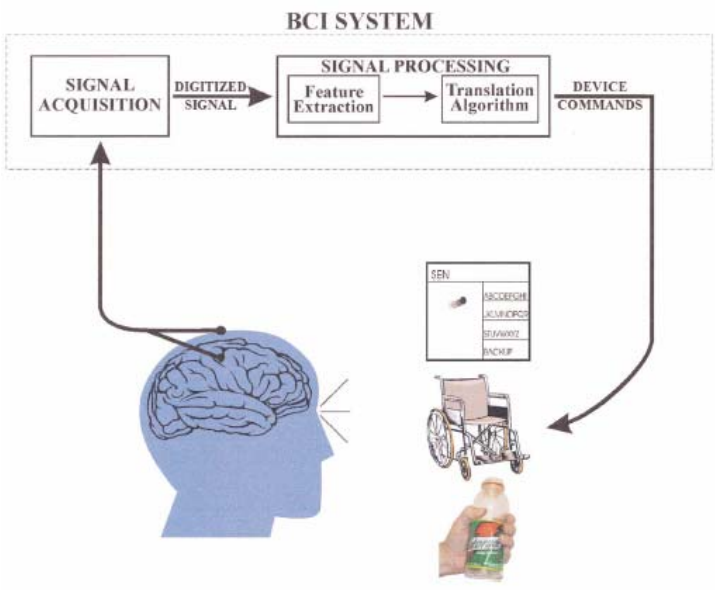
3.4. Kallon sisäisen ja kallon ulkopuolisen aivokäyttöliittymän vertailua

Kallon ulkopuolisia aivokäyttöliittymiä pidetään turvallisempina kuin kallon sisäisiä aivokäyttöliittymiä, koska siinä ei ole minkäänlaista infektioriskiä. Kallon ulkopuolisia aivokäyttöliittymiä ei kuitenkaan voida käyttää jatkuvasti juuri sen takia, että laitteet ovat kallon ulkopuolella ja pois otettavissa. Gerstner ja muut [2004] pitävät kallon ulkopuolisia aivokäyttöliittymiä hitaina, koska vuoteen 2004 mennessä niillä on pystynyt tekemään vain pieniä perustehtäviä, kuten liikuttamaan tietokoneen kursoria näytöllä tai avaamaan ja sulkemaan käsiproteesia. Lisäksi mm. Birbaumer [2006] on sitä mieltä, että kallon ulkopuolista aivokäyttöliittymää hallitsemaan oppimiseen menee kauan aikaa. Potilaat tarvitsevat hänen mukaansa aikaa jopa viikkoja pystyäkseen hallitsemaan kunnolla kallon ulkopuolisen aivokäyttöliittymän käytön, ja silloinkin virheprosentti on suuri.

4. Erilaisia käyttöliittymiä

Perusperiaatteeltaan kaikki aivokäyttöliittymät toimivat samalla tavalla. Kuvassa 3 nähdään, kuinka aivokäyttöliittymä toimii. Jokaisella erityyppisellä aivokäyttöliittymällä liitetään aivoihin tietynlainen laite, joko kallon sisäpuolelle tai vaihtoehtoisesti kallon ulkopuolelle. Tämä laite lukee aivojen sähköisiä tai magneettisia muutoksia, ja muuttaa tämän signaalin digitaaliseen muotoon. Kyseiseen tehtävään soveltuva algoritmi muuttaa tämän signaalin aivokäyttöliittymän ymmärtämään muotoon, jolloin käyttäjä voi esimerkiksi liikuttaa kursoria tietokoneen näytöllä [Birbaumer et al., 2002].

Tässä luvussa esitellään, millaisia aivokäyttöliittymiä eri tutkijat ovat tähän mennessä kehitelleet, ja luodaan katsaus tulevaisuuteen.



Kuva 3. Aivokäyttöliittymän toiminnan perusperiaate [Birbaumer et al., 2002].

4.1. Virtuaaliset näppäimistöt

Monet tutkijat ovat suunnitelleet erilaisia virtuaalisia näppäimistöjä. Gerstner ja muut [2004] ovat suunnitelleet kuvan 4 mukaisen virtuaalisen näppäimistön, joka on tyypiltään kallon ulkopuolinen aivokäyttöliittymä. Sen toimintaperiaate on yksinkertainen: Koko näppäimistö on jaettu kolmeen osaan kuvan 4 esittämällä tavalla. Jokaista eri lohkoa vastaa eri psyykkinen tehtävä (mental task). Psyykkinen tehtävä voi olla esimerkiksi vasemman käden liikuttaminen. Kun käyttäjä ajattelee vasemman käden liikuttamista, aivokäyttöliittymä poimii siitä saatavat signaalit ja muuttaa ne tietokoneen ymmärtämään muotoon. Käyttäjän saatua valittua lohko ensimmäisestä vaiheesta (kuva 4), jakaantuu valittu lohko kolmeen osaan. Jälleen ajattelemalla tiettyä psyykkistä tehtävää saadaan valittua oikea lohko, ja lopulta käyttäjä pystyy valitsemaan haluamansa kirjaimen.

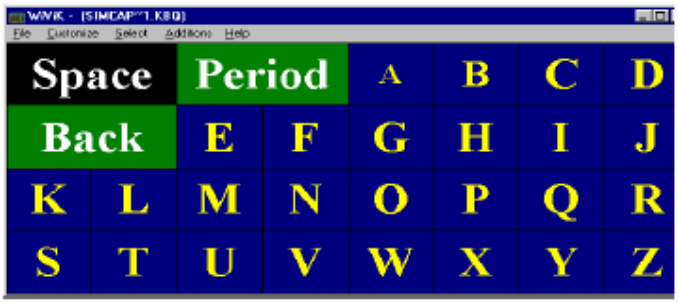
Kuva 4. Virtuaalinen näppäimistö, joka on jaettu kolmeen osaan [Gerstner et al., 2004].

Käyttöliittymässä on varauduttu myös virheisiin. Käyttöliittymässä siirrytään aina pienempään lohkoon, mikäli edellistä lohkoa vastaava psyykkinen tehtävä on tunnistettu kolme kertaa peräkkäin. Jos käyttäjä tekee virheen, siitä voidaan toipua siten, että käyttäjä ajattelee mitä tahansa muuta psyykkistä tehtävää. Kun käyttöliittymä huomaa virheen, palataan toiminnoissa edelliseen kohtaan.

Kirjoittaminen on tämän tapaisella virtuaalisella näppäimistöllä hyvin hidasta. Gerstner ja muut [2004] ovat testanneet virtuaalisen näppäimistön toimintaa, ja nopein koehenkilö sai kirjoitettua yhden kirjaimen keskimäärin 22 sekunnissa. Tämä aika sisältää myös ajan, joka on kulunut virheistä toipumiseen. Gerstnerin ja muiden [2004] käyttämät koehenkilöt olivat kuitenkin ”normaaleja” henkilöitä ja heilläkin meni virtuaalisen näppäimistön käytön opetteluun muutama päivä. Tästä herääkin kysymys, kuinka nopeasti esimerkiksi halvaantuneet ihmiset pystyvät oppimaan käyttämään virtuaalisia näppäimistöjä, koska heille se tulisi kuitenkin olemaan varsinainen väline kommunikointiin.

Myös Kennedy ja Moore [2004] ovat suunnitelleet virtuaalisen näppäimistön. He kutsuvat näppäimistöään nimellä WiViK (The Windows Visual Keyboard). WiVik käyttöliittymä eroaa selkeästi Gerstnerin ja muiden [2004] suunnittelemasta virtuaalisesta näppäimistöstä, kuten kuvasta 5 selviää.

WiVik:a käytetään kahden eri apuvälineen kanssa. Ensimmäinen niistä on Parmouse-hiiri, jota ohjataan aivosignaalien avulla. Parmousen avulla käyttäjä voi liikuttaa hiirtä tietokoneen ruudulla. Toinen apuväline on TalkAssist-puhesyntetisaattori. Se lukee kirjoitetun kirjaimen näytöltä. WiVik:lla voi kirjoittaa ainoastaan NotePad-ohjelmaan, ja sitä pidetäänkin sen heikkona puolena.



Kuva 5. WiVik

Gerstnerin ja muiden [2004] ja Kennedyn ja Mooren [2004] virtuaalinäppäimistöt eroavat toisistaan. Tämä on hyvä esimerkki siitä, kuinka samaan tehtävään, tässä tapauksessa kirjoittamiseen, on kehitetty kaksi täysin erilaista rat-

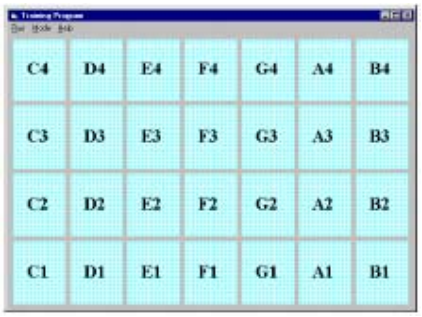
kaisua. Toisessa käytetään pelkästään virtuaalista näppäimistöä ja toisessa käytetään erilaisia ”apuohjelmia”.

4.2. Virtuaalinen piano

Virtuaalinen piano on tyypiltään kallon sisäinen aivokäyttöliittymä. Sen ovat kehittäneet Kennedy ja Moore [2004] mittatilaustyönä eräälle potilaalle, jota he kutsuvat nimellä JR. Hänen aivoihinsa on asennettu elektrodi, jonka avulla hän pystyy ohjaamaan tietokonetta. JR on aikaisemmin käyttänyt edellä kuvatun kaltaista virtuaalista näppäimistöä (WiViK), mutta koska hän on menettänyt myöhemmin näkönsä, hän ei näe tietokoneen ruutua. Tämän takia hän ei näe visuaalista palautetta, joka syntyy ruudulle.

Kuvassa 6 on esitelty virtuaalisen pianon käyttöliittymä. Käyttöliittymässä on neljä oktaavia nuotista C alkaen. Jokaista oktaavia vastaa käyttöliittymässä yksi vaakarivi. Jokainen nuotti on myös nimetty käyttöliittymässä kuvan 6 mukaisella tavalla. Käyttäjän liikkuessa virtuaalisessa pianossa vaakasuoraan, soittaa se C-skaalan, ja vastaavasti pystysuoraan.

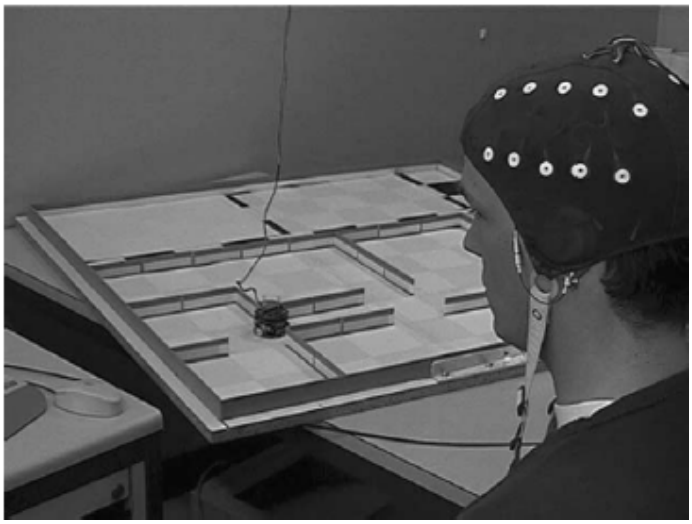
Virtuaalinen piano on kehitetty JR:lle, jotta hän pystyisi harjoittelemaan aivoimpulssiensa hallintaa. Tämä harjoittelu tapahtuu siten, että JR:lle annetaan tavoitenuotti, johon hänen tulee navigoida. Näin hän pystyy opettelemaan aivoimpulssiensa hallintaa, jotta hän pystyy myöhemmin käyttämään parempia aivokäyttöliittymiä apuna jokapäiväisessä elämässään. Tämän tapaisesta aivokäyttöliittymästä on varmasti paljon hyötyä JR:lle, joka on ennen sairastumistaan ollut ammattimuusikko. Kuitenkin henkilö, joka ei ole koskaan aikaisemmin soittanut pianoa, ei saa vastaavaa hyötyä virtuaalisesta pianosta. Ongelma aivosignaalien käytön harjoittelemisessa onkin se, että koska jokainen ihminen on erilainen, ei kaikille sovi samanlainen aivokäyttöliittymä. Tämän takia melkein jokaiselle pitäisi kehittää juuri hänelle sopiva aivokäyttöliittymä, mikä taas on erittäin hankalaa jo rahoituksenkin kannalta.



Kuva 6. Virtuaalinen piano [Kennedy and Moore, 2004].

4.3. Robotit

On myös kehitetty aivokäyttöliittymillä ohjattavia robotteja [Gerstner et al., 2004; del R. Millan, 2003]. Tyypiltään nämä robotit ovat kallon ulkopuolisia aivokäyttöliittymiä. Tyypillisesti tällaisten robottien ohjaamiseen käytetään EEG-tekniikkaa, kuten kuvasta 7 voidaan havaita. Näitä robotteja ohjataan psyykkisten tehtävien eli eri komentojen avulla [Gerstner et al., 2004]. Tällaisten robottien tarkoituksena on auttaa kehittämään aivokäyttöliittymällä ohjattavia pyörätuoleja, jotta halvaantuneet ihmiset voisivat liikkua paikasta toiseen [del R. Millan, 2003].



Kuva 7. Käyttäjä ohjaa robottia huoneesta toiseen aivokäyttöliittymän avulla [Gerstner et al., 2004].

Tyypillisesti näitä robotteja ohjataan muutamilla komennoilla, kuten esimerkiksi: liiku eteenpäin (mikäli edessä on oviaukko, mene toiseen huoneeseen), pysähdy, käänny vasemmalle ja käänny oikealle [del R. Millan, 2003]. Tavallisesti robotti jatkaa liikettään käyttäjän haluamaan suuntaan, kunnes se saa uuden psyykkisen komennon käyttäjältä [Gerstner et al., 2004]. del R. Millanin [2003] tutkimuksen jälkeen robotin kontrolloimiseen on lisätty sensorit, joilla robotti tunnistaa ympärillään olevat esteet sekä seinät [Gerstner et al., 2004]. Robotti on tämän takia jo sen verran kehittynyt, että jos käyttäjä on antanut robotille käskyn ”eteenpäin”, niin robotin mahdollisesti törmätessä seinään se jatkaa matkaa seuraten seinää eikä pysähtyen kuten ensimmäinen versio.

Tällaisen robotin ongelmana voi pitää sitä, että käyttäjän on tarvinnut ensin opetella hallitsemaan psyykkisten komentojen käyttämistä. Opettelun tulisi tapahtua esimerkiksi edellä mainittujen virtuaalisen näppäimistön tai pianon avulla.

4.4. Muita aivokäyttöliittymiä

Tutkijat ovat myös kehittäneet erilaisia aivokäyttöliittymillä toimivia pelejä [del R. Millan, 2003]. Esimerkkinä del R. Millan [2003] käyttää Pacman-peliä, mutta myös muunlaisia, opettavaisempia pelejä, on hänen mielestään mahdollista käyttää. Pacmania ohjataan kahdella psyykkisellä tehtävällä, käänny oikealle ja – vasemmalle. Pacman liikkuu haluttuun suuntaan käyttöliittymän tunnistaessa saman käskyn kahteen kertaan peräkkäin. Niin kauan kunnes se ei saa uutta käskyä, jatkaa se kulkuaan edellä määrättyyn suuntaan.

Hiiren liikuttaminen näytöllä aivokäyttöliittymän avulla tapahtuu edellä kuvatun Parmouse-hiiren avulla.

4.5. Tulevaisuus

Tulevaisuudessa aivokäyttöliittymien tutkimus tulee jatkumaan ja aivokäyttöliittymät kehittynevät entistä nopeammiksi ja tarkemmiksi. Nykyään aivokäyttöliittymät toimivat vielä hitailla vasteajoilla, mutta niiden käyttö tulee nopeutumaan tulevaisuudessa, tekniikan kehittyessä. Tällä hetkellä suosituin aivokäyttöliittymätyyppi on kallon ulkopuolinen aivokäyttöliittymä. Uskon kuitenkin, että kehittyessään kallon sisäinen aivokäyttöliittymä nousee nykyistä suurempaan suosioon. Kallon sisäpuolinen aivokäyttöliittymä on kuitenkin nopeampi ja tarkempi verrattuna kallon ulkopuoliseen käyttöliittymään.

Uskon, että tulevaisuudessa kaikki kallon sisäisiä aivokäyttöliittymiä koskevat ennakkoluulot hälvenevät. Yhteistyössä esimerkiksi kirurgien kanssa varmasti saadaan ratkaistuksi infektioriskin ongelma. Lisäksi aivokäyttöliittymät tulevat varmasti kehittymään niin, että niiden avulla voidaan esimerkiksi liikuttaa pyörätuolia. Nykyisellään kaikki aivokäyttöliittymät pystyvät suorittamaan ainoastaan yhtä tehtävää kerrallaan. Tulevaisuudessa olisi tarkoituksenmukaista, että aivokäyttöliittymällä pystyisi suorittamaan monia eri tehtäviä.

5. Yhteenveto

Aivokäyttöliittymät voidaan jakaa kolmeen eri kategoriaan, jotka ovat kallon sisäinen, osittain kallon sisäinen ja kallon ulkopuolinen aivokäyttöliittymä. Kallon ulkopuolisen aivokäyttöliittymän toteuttamisessa käytetään tunnetumpia tekniikoita, joita ovat muun muassa EEG- ja MEG-tekniikat. Kallon sisäisen aivokäyttöliittymän voi toteuttaa muun muassa vähemmän tunnetulla ECoG-tekniikalla. Kallon sisäisiä aivokäyttöliittymiä pidetään vaarallisina, koska niissä kiinnitetään elektrodi suoraan aivokuorelle, harmaaseen aineeseen. Tämä voi aiheuttaa potilaalla infektiovaaran. Lisäksi kallon sisäisten aivokäyttöliittymien tulisi kestää pitkiä aikoja kerrallaan, mikä ei ainakaan vielä ole mahdollista. Kallon ulkopuoliset aivokäyttöliittymät ovat turvallisempia kuin kallon sisäiset

aivokäyttöliittymät, mutta niistä saatava signaali on huomattavasti kallon sisäistä aivokäyttöliittymää heikompi ja tämän takia ne eivät ole läheskään niin tarkkoja. Hyvä puoli kallon ulkopuolisissa aivokäyttöliittymissä on se, että niitä on jo pystytty kehittämään, esimerkkinä virtuaaliset näppäimistöt.

Suurimman hyödyn aivokäyttöliittymistä saavat halvaantuneet henkilöt, jotka voivat kommunikoida aivokäyttöliittymien avulla toisten ihmisten kanssa. Aivokäyttöliittymätutkimuksen tavoite on, että halvaantuneet ihmiset pystyisivät kommunikoinnin lisäksi liikkumaan pyörätuolin kanssa aivokäyttöliittymän avulla. Kuitenkin liikkumista varten on toteutettu vasta pieniä robotteja (kohta 4.3), eikä aivokäyttöliittymää ole vielä toteutettu esimerkiksi pyörätuolin liikuttamista varten.

Aivokäyttöliittymän toteutus on kuitenkin melko hankalaa ja varsinkin sen käyttämään opettelu on usein varsin aikaa vievää. Tämä johtuu siitä, että käyttäjän tulee oppia hallitsemaan aivosignaaliensa käyttöä eri tehtävien avulla. Tällä hetkellä eri tutkimuksien mukaan jopa terveillä henkilöillä menee oppimiseen aikaa. Hyviä oppimistuloksia on saatu myös vammautuneiden henkilöiden oppimisesta, mutta heille aivokäyttöliittymän oppimiseen menee luonnollisesti enemmän aikaa. Esimerkiksi terveellä ihmisellä menee yhden kirjaimen kirjoittamiseen virtuaalisella näppäimistöllä aikaa jopa 22 sekuntia. Tämä johtuu siitä, että käytettyjen tekniikoiden vasteajat ovat vielä pitkiä, ja sama valinta on tehtävä jopa kolme kertaa peräkkäin, jotta tietokone saa tarvitsemansa vahvistuksen. Tulevaisuudessa on kuitenkin lupa odottaa entistä tehokkaampia ja nopeampia aivokäyttöliittymiä.

Viiteluettelo

[Arenas et al., 2005] M.G. Arenas, P. A. Castillo and J.J. Merelo, Evolutionary design of a brain-computer interface. In: *Lecture Notes in Computer Science: Computational Intelligence and Bioinspired Systems* **3512**, 669-676, 2005.

[Birbaumer et al., 2002] N. Birbaumer, D.J. MacFarland, G. Pfurtscheller, T.M. Vaughan and R.J. Wolpaw, Brain-computer interfaces for communication and control. *Clinical Neurophysiology* **113** (2002), 767-791.

[Birbaumer et al., 2005] Niels Birbaumer, Martin Bogdan, N. Jeremy Hill, Thilo Hinterberger, Thomas Hofmann, Jürgen Mellinger, Thomas Navin Lal, Hubert Preissl, Wolfgang Rosenstiel, Michael Schröder and Bernhard Schölkopf, A brain computer interface with online feedback based on magnetoencephalography. In: *Proceedings of the 22nd International Conference on Machine Learning ICML '05*, ACM Press, 465-472, 2005.

[Birbaumer, 2006] Niels Birbaumer, Brain–computer-interface research: coming of age. *Clinical Neurophysiology* **117**, 3 (2006), 479-483.

[Gerstner et al., 2004] Wulfram Gerstner, Josep Mourino, Jose del R. Millan and Frederic Renkens, Brain-actuated interaction. *Artificial Intelligence* **159** (2004), 241-259.

[Himberg et al., 2001] Lea Himberg, Vesa Laine ja Heikki Lyytinen, *Psykologia 4 Ihmisen toiminnan neuropsykologia*. WSOY, Porvoo 2001.

[Kennedy and Moore, 2004] Philip R. Kennedy and Melody M. Moore, Human factors issues in the neural signals direct brain-computer interfaces. In: *Proceedings of the Fourth International ACM SIGACCESS Conference on Assistive Technologies*, 114 -120.

[Lebedev and Nicolelis, 2006] Mikhail A. Lebedev and Miguel A.L. Nicolelis, Brain–machine interfaces: past, present and future. *TRENDS in Neurosciences* **29**, 9 (2006), 536-546.

[MacFarland and Wolpaw, 2004] Dennis J. McFarland and Jonathan R. Wolpaw, Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans. In: *Proceedings of the National Academy of Sciences of the United States of America* **101**, 51 (2004), 17849-17854.

[Patnaik, 2005] L.M. Patnaik, The Brain, complex networks, and beyond. In: *Lecture Notes in Computer Science: Distributed Computing – IWDC 2005* **3741**, 111-116, 2005.

[del R. Millan, 2003] Jose del R. Millan, Adaptive brain interfaces. *Communications of the ACM* **46**, 3 (2003), 74-80.

[Sun and Zhang, 2005] Shiliang Sun and Changshui Zhang, Learning on-line classification via decorrelated LMS algorithm: application to brain-

computer interfaces. In: *Lecture Notes in Computer Science: Discovery Science* **3735**, 215-226, 2005.

[Surakka et al., 2004] Veikko Surakka, Marko Illi and Poika Isokoski, Gazing and frowning as a new human-computer interaction technique. *ACM Transactions on Applied Perception* **1**, 1 (2004), 40-56.

[Wikipedia] Wikipedia, Brain-computer interface. Available: http://en.wikipedia.org/wiki/Brain-computer_interface (checked 23.10.2006).

Objekti-rooli -mallin perusteet

Petra Vyyryläinen

Tiivistelmä

Objekti-rooli -malli on käsitteelliseen mallinnukseen kehitetty kieli, joka kasvattaa koko ajan suosiotaan. Esittelen objekti-rooli -mallia, sen notaation ja perusteet merkintöjen käytöstä. Käyn myös esimerkin kautta läpi objekti-rooli -mallin mukaisen käsitekaavion muuntamisen relaatiokaavioksi. Lisäksi tarkastellaan objekti-rooli -mallia suhteessa tunnetumpaan mallinnuskieleen eli ER-malliin. Objekti-rooli -mallia varten on kehitetty oma käsitteellinen kyselykieli, jota esittelen myös lyhyesti.

Avainsanat ja -sanonnat: Objekti-rooli -malli, käsitteellinen mallinnus, käsitekaavio.
CR-luokat: H.2

1. Johdanto

Hyvän tietojärjestelmän suunnittelu vaatii kohdealueen hyvää tuntemusta ja käsitteiden sisällön hallintaa. Koska käsitteitä saattaa olla jopa yli 1500, niiden esittäminen ja hallinta luonnollisella kielellä olisi erittäin työlästä. Käsitekaavioilla tarkoitetaan toteutuksesta riippumatonta kohdesisällön kuvausta. Siitä nähdään mitä tietoja tarvitaan, eikä keskitytä pelkästään toteutukseen kuten usein valitettavasti käy. [Kangassalo, 2005]

Tietojärjestelmät kannattaa siis aina mallintaa ensin käsitteellisellä tasolla, näin siitä saadaan paremmin kohdealuetta kuvaava, selkeämpi ja helpommin muunneltava. [Halpin, 1998] Käsitteellisen tason mallinnukseen on kehitetty useita kieliä, joista tunnetuimpia ovat ER-malli ja UML. Objekti-rooli -malli (lyhennetään yleisesti ORM, tässä työssä ei käytetä lyhennettä) on kehitetty 1970-luvulla ja sen avulla on helppo toteuttaa ilmaisuvoimaisempi ja helppolukuisempi käsitekaavio kuin ER-mallilla. [Halpin, 1998]

Tässä työssä esittelen objekti-rooli -mallin perusmerkistön, joka on helppo oppia ja ottaa käyttöön. Objekti-rooli -mallin mukaisen käsitekaavion muuntaminen relaatiokaavioiksi ei ole ollenkaan niin vaikeaa kuin ER-mallin mukaisen. Tätä käyn läpi esimerkin avulla. Esitän myös muita esimerkkejä ER- ja objekti-rooli -mallien eroista.

Lukijan ei tarvitse tietää etukäteen mitään objekti-rooli -mallista, mutta jonkinlainen tietämys ER-mallista ja tietokannoista helpottaa lukemista.

2. Mikä objekti-rooli -malli?

Objekti-rooli -malli on käsitteellisellä tasolla tapahtuvaan mallinnukseen kehitetty mallinnuskieli. Sen on kehittänyt alun perin 1970-luvulla Eckhard Falkenberg, joka myös otti käyttöön nimen objekti-rooli -malli. Hän kutsui kieltä nimellä ENALIM (Evolving NAtural Language Information Model). Vuoteen 1982 mennessä kieltä oli kehitetty niin paljon, että ENALIM muutettiin muotoon ”An Information Analysis Method” (NIAM), nykyään NIAM tarkoittaa ”Natural language Information Analysis Method”. NIAMista kehitettiin objekti-rooli -malli (object-role model), jolle Terry Halpin loi ensimmäisen formaalin esitystavan. [Halpin, 1998]

Mallintamiskielenä objekti-rooli -malli helpottaa kohdealueen mallintamista, koska mallintamisprosessissa käytetään luonnollista kieltä. Verrattuna ER-malliin tilanne on seuraavanlainen: ER-mallissa syntymäPaikka on Henkilön attribuutti, mutta objekti-rooli mallissa voidaan käyttää ilmaisua, Henkilö on syntynyt paikassa, tällöin käsitekaavion ymmärtäminen on helpompaa. Lisäksi käsitekaavioon voidaan lisätä esimerkkejä kohdealueen todellisista ilmentymistä. [Halpin, 1998]

3. Objekti-rooli -mallin perusteet

Objekti-rooli -mallin kanssa on helppo päästä käsitteellisen mallintamisen alkuun. Helpoin tapa aloittaa objekti-rooli -mallin mukaisen käsitekaavion tekeminen on kirjoittaa ylös kohdealueelta löydettäviä *faktalauseita* eli tosiasioita. Kun suunnitellaan yrityksen tietojärjestelmää, puhutaan liiketoimintasäännöistä.

Otetaan esimerkkikohdealueeksi pieni levy-yhtiö ja tehdään sen yhtyeitä kuvaava käsitekaavio. Mietitään aluksi faktoja yhtyeestä. Usein näitä kerätään haastattelemalla kohdealueen työntekijöitä, tarkkailemalla itse kohdealuetta tai antamalla työntekijöiden itse koota faktoja tai pieniä käsitekaavioita.

Faktat koostuvat yleensä kahdesta termistä, jos käsitteellinen mallintaminen haluttaisiin tehdä perusteellisesti, jokainen termi tulisi määritellä erikseen, mutta me menemme suoraan faktalauseisiin.

- Yhtyeen jäsen soittaa yhtyeessä.
- Yhtyeen jäsenellä on nimi.
- Yhtyeen jäsen soittaa vain yhtä soitinta.
- Yhtyeen jäsen saa kuukausipalkkaa.
- Yhtye tekee albumin.
- Albumilla on nimi.
- Albumi sisältää kappaleita.
- Yhtyeellä on fan club.

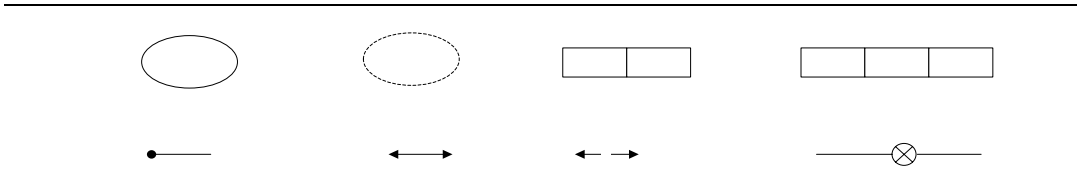
- Fan clubilla on jäseniä (ei ole pakko olla), mutta yhtyeen jäsen ei voi olla fan clubin jäsen.
- Fan clubilla on puheenjohtaja, joka on myös fan clubin jäsen.
- Fan clubilla on yksi tai useampi sihteeri.
- ….

Huomataan, että yksinkertaisestakin kohdealueesta tulee helposti paljon faktalauseita, ne kannattaa miettiä hyvin, koska sen jälkeen käsitekaavion teko on helppoa.

Ennen käsitekaavion tekemistä on kuitenkin tiedettävä, mitä merkintöjä objekti-rooli -mallin notaatioon kuuluu.

3.1. Notaatio

Objekti-rooli -mallin notaation esitti ensimmäisen kerran Shir Nijssen vuonna 1976. Silloin otettiin käyttöön nykyisin käytössä olevat soikiot ja suorakulmiot. Kuvassa 1 on esitetty objekti-rooli -mallin perusmerkinnät.



Kuva 1. Objekti-rooli -mallin notaation perusosat. [Halpin, 1998]

Soikioilla (kuva1A) kuvataan kohdetyyppejä eli entiteettejä eli *olioita*. Oliolla tarkoitetaan olemassa olevaa käsitettä tai termiä, meidän esimerkkikohdealueessamme yhtyettä, yhtyeen jäsentä ja vaikkapa yhtyeen tekemää albumia.

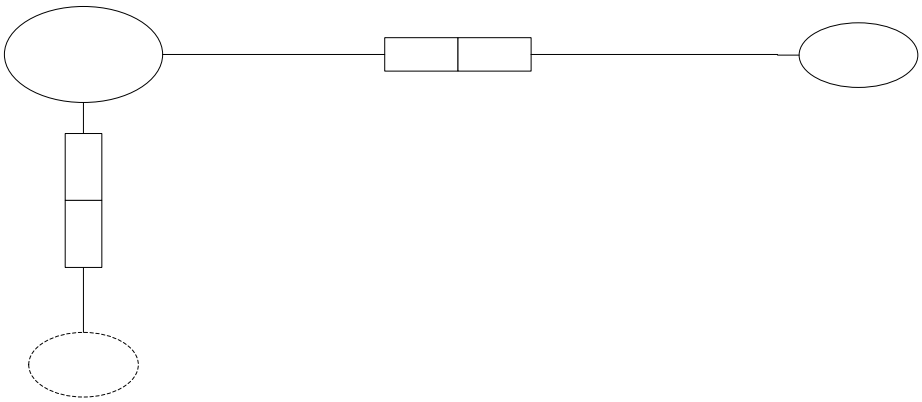
Yhtyeen jäsenen nimeä tai albumin nimeä kutsutaan *arvotyypiksi* eli se on jonkin olion saama arvo, arvotyyppiä kuvataan katkoviivalla piirretyllä soikiolla (kuva 1B). Arvotyyppi saa normaalisti arvoikseen lukuja tai merkkijonoja.

Kahdella vierekkäisellä suorakulmiolla (kuva 1C) kuvataan olioiden välillä olevaa *suhdetta* ja suorakulmioiden alla olevassa tekstissä kerrotaan olioiden *roolit* tässä suhteessa, käytettävästä ohjelmasta riippuen roolien nimet voidaan kirjoittaa myös suorakulmioiden sisään. Suhteeseen voi osallistua myös useampia olioita, tällöin suhdemerkintään lisätään suorakulmioita niin monta kuin tarvitaan, kuvassa 1D on kolmen olion välinen suhde.

Jos halutaan, että olion täytyy osallistua aina tiettyyn suhteeseen, merkitään musta täplä olioon (kuva 1E). Kuva 1F liittyy suhteen kardinaliteettiin ja se on myös yksilöintirajoitus samoin kuin nuolen sisällä oleva P (kuva 1G), jolla merkitään ensisijaista yksilöintirajoitusta. Ympyrän sisällä olevalla ruksilla (kuva

1H) kuvataan sitä, että kahden olion ilmentymät eivät saa sisältää samoja yksilöitä, tätä sanotaan poissulkevuusrajoitukseksi.

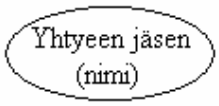
Objekti-rooli -malli sisältää paljon rajoituksia, jotka tekevät siitä erittäin monipuolisen ja ilmaisuvoimaisen kielen verrattuna moniin muihin mallinnuskieliin. Rajoituksista on lisää esimerkkejä myöhemmin.



Kuva 2. Yksinkertainen objekti-rooli -mallin avulla tehty käsitekaavio.

Kuvassa 2 on yksinkertainen käsitekaavio, joka on tehty objekti-rooli -mallin notaation mukaan. Oliot liitetään viivoilla suhdemerkkiin ja roolien nimet luetaan aina siten, että vasemmalla olevan olion rooli on vasemmalla ja päinvastoin eli kuvassa 2 yhtyeen jäsen soittaa yhtyeessä ja yhtye koostuu yhtyeen jäsenistä. Käsitekaavio on siis luotu suoraan niistä faktalauseista, joita aluksi määrittelimme. Yksinkertaisen kaavion laadinta ei siis ole vaikeaa.

Aina ei tarvitse luoda uutta suhdetta olion arvotyypin ilmoittamiseksi, vaan voimme merkitä sen suluissa oliotyypin nimen alle kuten kuvassa 3. Valittava tapa riippuu siitä, mihin muihin suhteisiin olio osallistuu ja millaisia rajoituksia sille on asetettu.[Halpin, 1998]



Yhtyeen jäsen

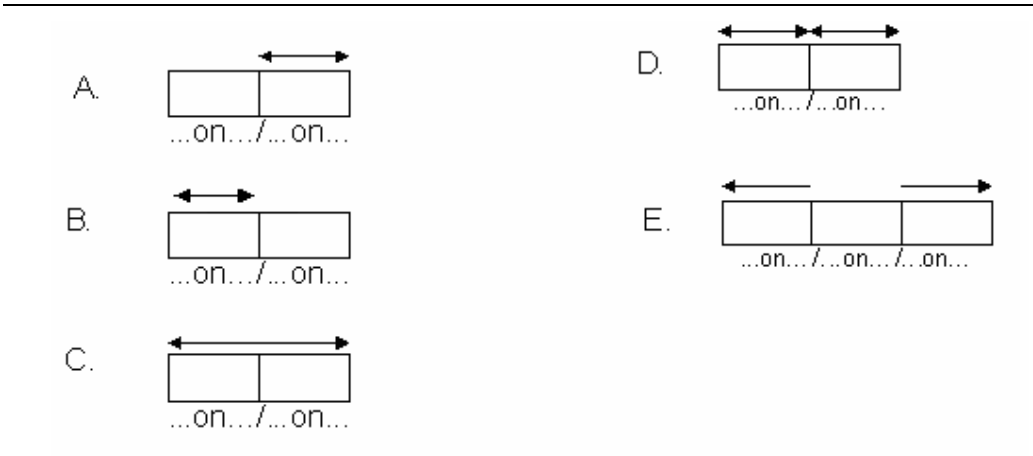
Kuva 3. Vaihtoehtoinen tapa esittää se, mitä oliosta halutaan kertoa.

3.2. Suhteen lukumäärä- ja yksilöintirajoitukset

Kaikissa mallinnuskielissä on tarpeen olla keino esittää, voiko olion ilmentymiä, eli todellisia reaalimaailman kohteita, esiintyä suhteessa enemmän kuin yksi. Tämä vaikuttaa ratkaisevasti tietokannan tekemiseen. Tätä esitystapaa kutsutaan suhteen *kardinaliteetiksi* eli lukumääräksi. Objekti-rooli -mallin notaa-

viittaa / on

tiossa kardinaliteetti esitetään nuolilla, jotka piirretään suhde-merkin yläpuolelle. Kuvassa 4 on esitetty eri vaihtoehdot.



Kuva 4. Suhteen kardinaliteetti

Objekti-rooli -mallissa on käytössä myös muista mallinnuskielistä tutut 1:1, 1:N, N:1 ja N:M rajoitukset, ne vain merkitään eritavalla. Kuvassa 4 kohta

- A kuvaa 1:N-suhdetta
- B kuvaa N:1-suhdetta
- C kuvaa N:M-suhdetta
- D kuvaa 1:1-suhdetta

  (myös ilman nuolia oleva suhde tulkitaan 1:1-suhteeksi).

Nuolilla on myös toinen tarkoitus: ne ilmoittavat kumpi suhteeseen osallistuva olio yksilöi suhteen, tätä kutsutaan *yksilöintirajoitukseksi*. Tätä ei tarvitse miettiä sen enempää, koska suhteen kardinaliteetin myötä yksilöintirajoitus selviää. Ainoastaan useamman olion välisissä suhteissa joudutaan miettimään, mikä tai mitkä oliot yksilöivät suhteen (kuva 4E). Tässä tapauksessa ensimmäinen ja kolmas olio yhdessä yksilöisivät suhteen, tätä kutsutaan *yhdistetyksi yksilöintirajoitukseksi.* Relaatiokaavio, jonka tekemiseen kardinaliteettiä ja yksilöintirajoituksia tarvitaan, tehdään objekti-rooli -maalissa hieman eri tavalla kuin ER-mallissa, mutta tähän palaamme myöhemmin.

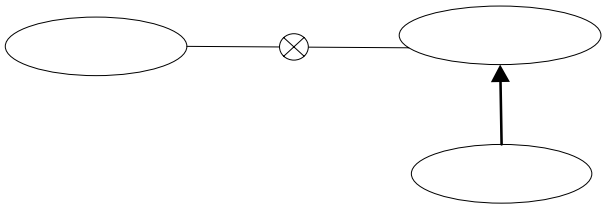
3.3. Joukkorajoitukset

Objekti-rooli -malliin saadaan paljon ilmaisuvoimaa, kun otetaan käyttöön erilaiset *joukkorajoitukset* eli tiedot siitä, millaiset populaatiot olioiden ilmentymillä täytyy olla, jotta ne kuvaavat halutulla tavalla kohdealuetta. Tärkeimmät näistä rajoituksista ovat

- Osajoukkorajoitus, jolla kuvataan sitä, että jonkin olion populaatio sisältyy kokonaan toisen olion populaatioon. Esimerkissämme fan clubin puheenjohtaja on myös fan clubin jäsen, joten puheenjohtajan ja fan clu-

bin jäsenen välille voidaan piirtää paksu musta nuoli, jolla merkitään osajoukkorelaatiota. Nuoli osoittaa aina siihen joukkoon, joka sisältää toisen joukon. Katso esimerkki kuvasta 5.

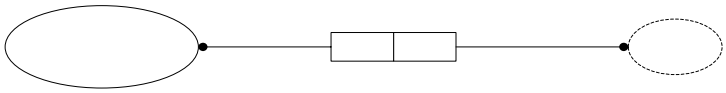
- Ekvivalenssirajoitus, jolla kuvataan kahdella joukolla olevan samat populaatiot eli kaksi joukkoa on täsmälleen samat. Esimerkissämme yhtyeen kotisivua ylläpitää fan clubin sihteeri, joten kotisivun ylläpitäjän ja fan clubin sihteerin välille voidaan piirtää ympyrän sisällä oleva yhtäsuuruusmerkki, jolla merkitään ekvivalenssirajoitusta.
- Poissulkevuusrajoitus, jolla kuvataan sitä, että kahteen joukkoon ei saa sisältyä yhtään samaa ilmentymää. Esimerkissämme yhtyeen jäsen ei saa kuulua fan clubiin, joten voimme piirtää fan clubin jäsenen ja yhtyeen jäsenen välille ympyrän sisällä olevan ruksin.



Kuva 5. Esimerkki poissulkevuusrajoituksesta ja osajoukkorelaatiosta.

3.4. Populaatiorajoitukset

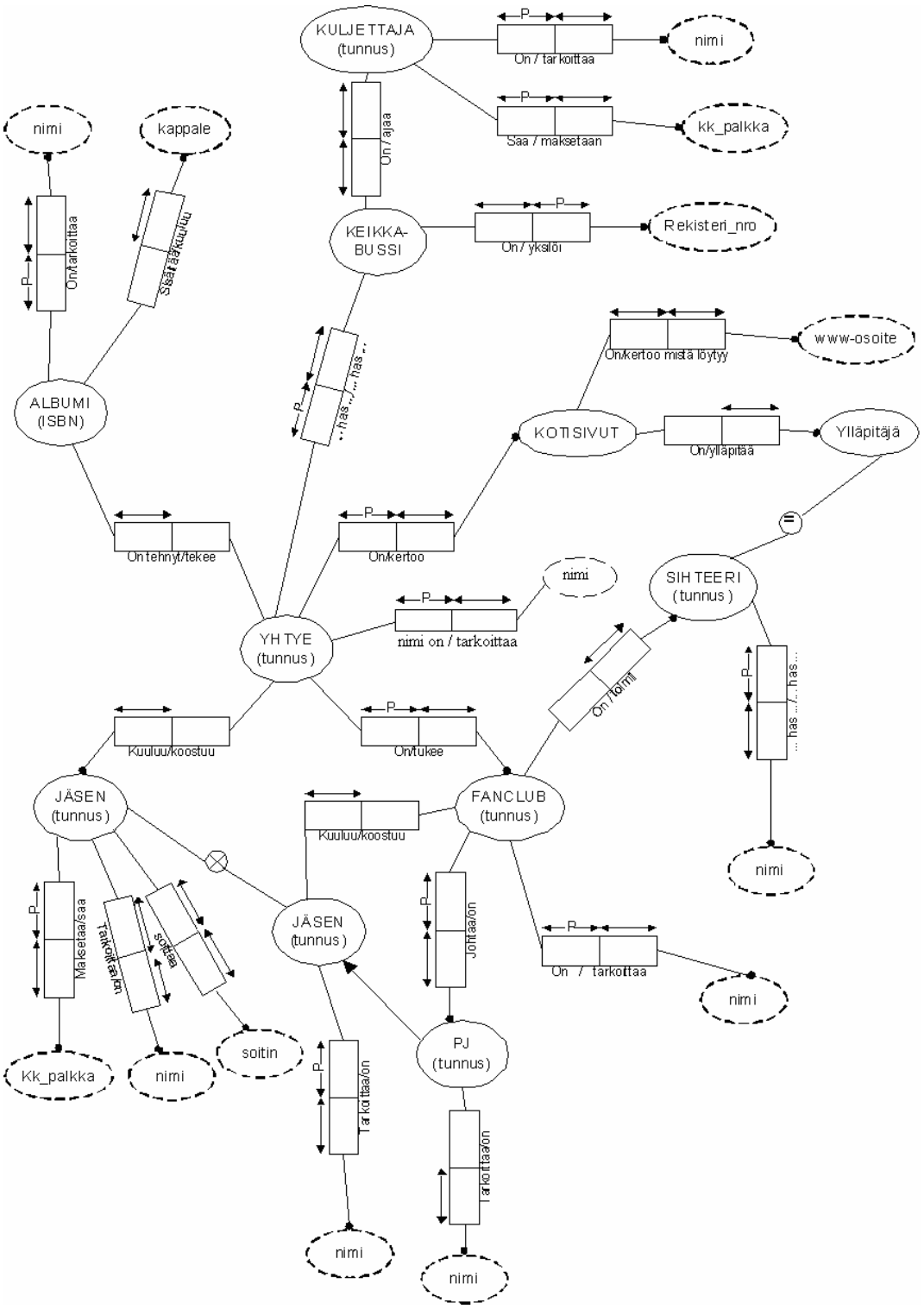
Objekti-rooli -mallissa on mahdollista laittaa näkyviin ne arvot, jotka ovat sallittuja. Näin tietotekniikkaa ja käsitteellistä mallinnusta ennestään tuntematonkin osaa tulkita käsitekaaviota paremmin ja tietokantaa käyttävä tietää mitkä arvot ovat mahdollisia lisätä. Esimerkki esitetään kuvassa 6.



Kuva 6. Populaatiorajoituksen esittäminen. Rajoitus tulisi esittää oliotyypin välittömässä läheisyydessä, jotta lukija tietää mitä oliota se koskee.

3.5. Lopullinen käsitekaavio

Kun sovelletaan kaikkia edellä mainittuja merkintöjä faktalauseisiimme, saamme kuvassa 7 esitetyn käsitekaavion.



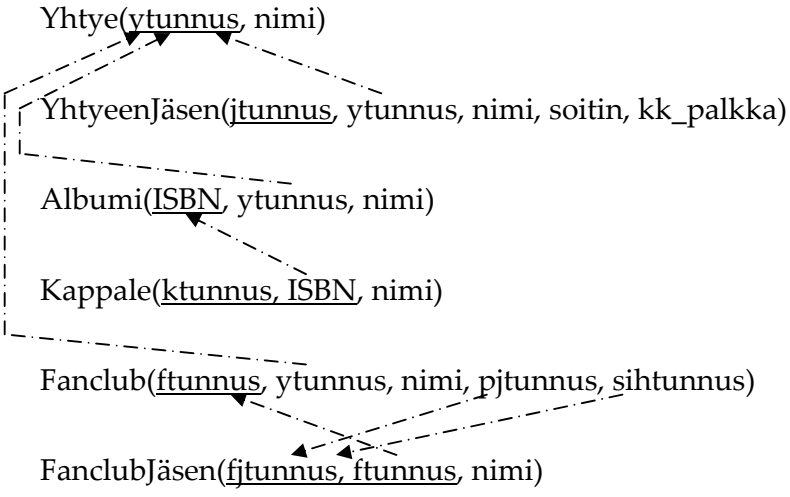
Kuva 7. Yhtyettä kuvaava käsitekaavio. Käsitekaavion asettelu ei ole aivan onnistunut, yleensä pyritään selkeämpään esitystapaan ja suoriin viivoihin.

4. Käsitekaaviosta relaatiokaavioksi

Objekti-rooli -mallin mukaisesta käsitekaaviosta saadaan helposti muodostettua tietokannan tekoon tarvittavat relaatiokaaviot. Relaatiokaavion odotetaan yleisesti olevan paikkansapitävä eli vastaavan hyvin käsitekaaviota, tehokas eli nopeasti kyselyihin vastaava ja nopeasti päivitettävä sekä selkeä eli suhteellisen ymmärrettävä ja sellainen, että työnteko relaatiokaavion kanssa on mahdollisimman helppoa. [Halpin, 2001a]

Objekti-rooli -mallin mukaisen käsitekaavion muuttamisessa relaatiokaavioksi voidaan käyttää Rmap (relatinal mapping) algoritmia. Rmap-algoritmin notaatiossa on muutama eroavaisuus tavalliseen relaatiokaavioon. Esimerkiksi notaatiossa A(a, b, [c]) A on taulun nimi, a on avain, b on pakollinen ominaisuus, eli siinä on ollut musta täplä, ja c voi saada myös arvon null. Vierasavaimesta piirretään katkoviivalla nuoli siihen pääavaimeen, johon se viittaa.[Halpin, 2001a]

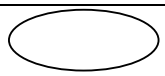
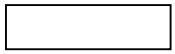
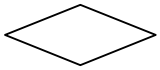
Tehdään esimerkkinä osa kuvan 7 käsitekaavion relaatiokaaviosta. Yhtye on keskusolio, joten aloitetaan siitä. Emme laita yhtye-relaatioon muuta kuin tunnuksen ja nimen, koska kaikki muut siihen suhteessa olevat, ovat suhteessa myös johonkin muuhun olioon. Yhtyeen jäsen -relaatioon taas laitamme kaikki, joihin se on suhteessa, koska nämä eivät ole enää suhteessa muihin olioihin, ja vierasavaimeksi yhtyeen tunnuksen. Vierasavain löytyy helposti siitä ”isommasta” oliosta johon kyseinen olio on suhteessa. [Halpin, 2001a]



5. Objekti-rooli -malli verrattuna ER-malliin

5.1. Tietomallin vertailu

Tietomallilla tarkoitetaan sitä millaisia käsitteitä mallinnuksessa käytetään ja kuinka niitä käytetään. Objekti-rooli -mallin ja ER-mallin tietomallin ovat yllättävän erilaiset, vaikka molemmilla mallinnuskielillä pyritään samaan asiaan, mahdollisimman hyvään malliin kohdealueesta. Merkinnät ovat hyvin erilaisia, ER-mallissa kohdetyypit eli oliot mallinnetaan suorakulmioilla, jotka objekti-rooli -mallissa tarkoittavat suhteita. Objekti-rooli -mallissa kohdetyypit mallinnetaan soikiolla, joka taas ER-mallissa tarkoittaa attribuuttia eli kohdetyypin ominaisuutta. Kuvaan 8 on kerätty suurimmat merkinnälliset erot.

merkintä	ORM	ER-malli
	kohdetyyppi eli oliotyyppi	attribuutti eli ominaisuus
	arvotyyppi	johdettu attribuutti
	suhde	kohdetyyppi eli oliotyyppi
		suhde

Kuva 8. Merkinnät tarkoittavat aivan eri asioita objekti-rooli -mallissa ja ER-mallissa. Kummassakin mallissa on lisäksi myös muita merkintöjä.

Suurin ero objekti-rooli -mallin ja ER-mallin välillä on attribuuttien käyttö. ER-mallissa ne ovat suuressa osassa kun taas objekti-rooli -mallissa ei käytetä ollenkaan attribuutteja, kaikkien olioiden välillä on jokin suhde. Tämä tuo suuria eroja tietysti myös ajattelutapaan, jolla kohdealuetta tulee lähestyä.

5.2. Relaatiokaavioiden normalisointi

Relaatiokaavioiden *normalisoinnilla* tarkoitetaan relaatiokaavion muokkaamista niin, että se tukee tietojen eheyttä ja tiedon tehokasta saatavuutta. Erityisesti normalisoinnilla vähennetään tiedon redundanssia eli saman tiedon tallennusta useaan paikkaan. Normaalimuotoja on seitsemän erilaista ja tietokannan taulun sanotaan olevan tietyssä normaalimuodossa, jos se täyttää normaalimuodon ehdot. [Elmasri, Navathe, 2004]

ER-mallia käytettäessä normalisointi saattaa välillä olla monimutkaista, sillä on mietittävä funktionaalisia riippuvuuksia ja redundanssin välttämistä, mutta objekti-rooli -mallissa normalisointi ei tarvitse erikseen tehdä ollenkaan. Rmap-algoritmia oikein käyttämällä hyvin tehtyyn käsitekaavion, relaatiokaaviosta

tulee automaattisesti normalisoitu eikä redundanssia ole.[Halpin, 2001b] Objekti-rooli -mallissa ei näin ollen tarvitse myöskään miettiä funktionaalisia riippuvuuksia.[Becker, 1998]

On myös olemassa työkaluja, jotka tekevät käsitekaaviosta suoraan relaatiokaaviot. Niitä löytyy sekä ER- että objekti-rooli -mallille. Esimerkkejä tällaisista työkaluista objekti-rooli -mallille ovat Microsoftin Visio Enterprise 2000 ja uusi tuote Visual Studio Enterprise Architect (VSEA). [Halpin, 2001b]

6. Käsitteellinen kyselykieli

Objekti-rooli -malliin perustuva, tällä hetkellä paras, käsitteellinen kyselykieli on nimeltään ConQuer. Se mahdollistaa monimutkaisten kyselyjen tekemisen tavalla, jonka kokematonkin käyttäjä ymmärtää. Käyttäjän ei myöskään tarvitse tietää, miten tiedot ovat tallennettu tietokantaan. [Halpin, 1996]

Objekti-rooli -mallin perusteella tehtyjen käsitekaavioiden kyselykieliä on kehitetty aiemminkin. Tunnetuin niistä on RIDL, joka on hyvin voimakas kyselykieli, mutta melko vaikea oppia. Toinen kyselykieli on LISA-D, joka on myös ilmaisuvoimainen, mutta teknisesti haastava, eikä sitä tue tällä hetkellä mikään työkalu. [Halpin, 1996]

ConQuer-kyselykielen etuja muihin objekti-rooli -mallin kyselykieliin verrattuna ovat suurempi ilmaisuvoima, uusien käyttäjien on helppo oppia kielen perusteet sekä ConQuer-kielelle on olemassa kaupallisia työkaluja, jotka muuttavat ConQuer-kyselyn SQL-kyselyksi. Paras puoli tässä kielessä on se, että käyttäjän ei tarvitse tuntea objekti-rooli -mallin notaatiota, eikä osata lukea edes käsitekaaviota. [Halpin, 1996]

Käyttäjiä helpottaa myös se, että eräs ohjelmaversio, jolla kyselyjä voi tehdä, on saatavilla Windows-sovelluksiin ja käyttöliittymä on siten tuttu useimmille. Lisäksi käyttöliittymä on hyvin helppokäyttöinen, kyselyjä voi tehdä klikkailemalla objekteja ja antamalla ehtoja. [Halpin, 1996]

ConQuerin käyttö ei ole kovin yleistä, mikä johtuu työkalun huonosta saatavuudesta ja tietämättömyydestä koko kyselykielen olemassa olosta. Sitä kuitenkin käytetään jonkin verran. [Kangassalo, 2005]

Hallock[2000] esittää, että ConQuer -työkalua voitaisiin käyttää myös objekti-rooli -mallin opettelussa helposti avuksi vastaamaan käyttäjän kysymyksiin.

7. Yhteenveto

Objekti-rooli -malli on käsitteellinen mallinnuskieli, jonka avulla on helppo tehdä monipuolisia ja ilmaisuvoimaisia käsitekaavioita valitusta kohdealueesta.

Perusmerkinnät ovat yksinkertaisia, kohdetyyppi eli olio kuvataan soikiolla ja

olion saamaa arvoa eli arvotyyppiä katkoviivalla piirretyllä soikiolla. Kahden olion välinen suhde kuvataan suorakulmiolla, jonka alapuolelle kirjoitetaan olioiden roolit suhteessa. Suhdemerkin yläpuolelle merkitään nuolilla suhteen kardinaliteetti ja yksilöintirajoitukset.

Kohdealuetta mietitään luonnollisen kielen lauseilla, joista saadaan faktoja, jotka muodostuvat olioista ja niiden välistä suhteista.

Objekti-rooli -mallin käytöstä on muitakin etuja kuin käsitekaavion helppo tekeminen, myös relaatiokaavion tekeminen ja sen normalisointi on helppoa.

On olemassa helppo käsitteellinen kyselykieli ConQuer, joka on kehitetty objekti-rooli -mallille. Se ei ole kovin yleisessä käytössä, mutta sitä pidetään hyvänä kyselykielenä ja toivotaan sen käytön yleistyvän. [Hallock, 2000]

Viiteluettelo

[Becker, 1998] Scot A. Becker, Normalization and ORM. *Journal of Conceptual Modeling* **4** (Aug. 1998).

[Elmasri, Navathe, 2004] Ramez Elmasri, Shamkant B. Navathe, *Fundamentals of Database Systems.* Addison-Wesley, 2004.

[Evans, 2005] Ken Evans, Requirements engineering with ORM. In: *On the Move to Meaningful Internet Systems 2005: OTM Workshops*. (2005), Lecture Notes in Computer Science **3762**, 646-655.

[Hallock, 2000] Patrick Hallock, Object-role modeling wish list (Part one). *Journal of Conceptual Modeling* **17** (Dec.2000).

[Halpin, 1996] Terry Halpin, ConQuer: A Conceptual Query Language. (available on www.orm.net)

[Halpin, 1998] Terry Halpin, Object-Role Modeling (ORM/NIAM). In: P. Bernus, K. Mertins, G. Schmidt (eds.), *Handbook on Architectures of Information Systems.* Springer-Verlag, 1998. (available on www.orm.net)

[Halpin, 2001a] Terry Halpin, Information Modeling and Relational Databases. MKP, 2001.

[Halpin, 2001b] Terry Halpin, Microsoft's new database modelling tool (Part 1). *Journal of Conceptual Modeling* **20** (Jun. 2001).

[Kangassalo, 2005] Hannu Kangassalo, Tietojärjestelmien ja tietokantojen suunnittelu -kurssin luentomateriaali, kevät 2005.