Käyttöliittymäteoriat ja -mallit

Roope Raisamo (toim.)

Tampereen yliopisto
Tietojenkäsittelytieteiden laitos
Joulukuu 2001

Käyttöliittymäteoriat ja -mallit

Käsissäsi on ensimmäinen laajamittainen suomen kielellä kirjoitettu teos niistä teorioista ja malleista, joita voidaan soveltaa ihmisen ja tietokoneen vuorovaikutuksessa. Julkaisu perustuu samannimiseen seminaariin, jonka julkaisun toimittaja organisoi keväällä 2001 Tampereen yliopiston Tietojenkäsittelytieteiden laitoksella. Seminaarissa kartoitettiin mahdollisimman laajasti ja monitieteisesti ihmisen ja tietokoneen vuorovaikutuksen ymmärtämiseen, suunnitteluun ja toteutukseen liittyviä teorioita ja malleja.

Alan nuoruudesta johtuen suuri osa esiteltävistä teorioista ja malleista on peräisin tietojenkäsittelytieteiden ulkopuolelta, kuten psykologiasta, kielitieteistä ja sosiologiasta. Viime vuosina ovat erityisesti vuorovaikutuksen mallintamiseen liittyvät ennustavat mallit nousseet keskeisiksi alan konferenssiseissa ja tieteellisissä lehdissä. Nämä mallit ovat käyttöliittymätutkijoiden itse kehittämiä, päinvastoin kuin aiemmat teoriat, joita on sovellettu vuorovaikutuksen mallintamisessa. On odotettavaa, että teoreettinen tarkastelu on jatkossa entistä keskeisempää myös käytännöllisesti suuntautuneessa käyttöliittymätutkimuksessa. Niinpä nyt olikin jo aika koota yhteen, millä tasolla käyttöliittymien teoreettinen tarkastelu on nykyisin.

Tätä teosta voidaan käyttää alan kurssien materiaalina. Kannattaa kuitenkin ottaa huomioon, että kyseessä on teoksen ensimmäinen painos, joka on seminaaritöiden kooste. Lukujen kirjoittajat käsittelevät seminaarin aihepiiriä kukin omalla tavallaan, mutta kauttaaltaan asiantuntevasti. Olemme myös pyrkineet tarjoamaan kunkin luvun lopussa mahdollisimman kattavan luettelon lähteitä, joista lukija saa tarkempaa tietoa esitetyistä asioista. Lähes kaikki seminaarin osallistujat olivat tieteellisen jatkotutkinnon suorittaneita tai jatko-opiskelijoita, jotka ovat perehtyneet alaansa usean vuoden ajan.

Tietojenkäsittelytieteiden laitoksen TAUCHI-yksikkö (Tampere Unit for Computer-Human Interaction) pyrkii jatkossakin edistämään ihmisen ja tietokoneen vuorovaikutuksen opetusta ja tutkimusta Suomessa. Alan toimijoita kokoaa yhteen ACM SIGCHI:n Suomen paikallisyhdistys SIGCHI Finland, jossa toimimme aktiivisesti edistäen tieteiden ja paikkakuntien välistä yhteistyötä. Otan mielellään vastaan tämän teoksen sisältöön liittyviä kommentteja, koska pyrimme kehittämään teosta tulevina vuosina entistä täydellisemmäksi.

Tampereella joulukuussa 2001

Roope Raisamo

Sisällys:

Fittsin laki ja johdannaiset

Poika Isokoski

Fittsin laki on käyteyimpiä malleja ihmisen ja tietokoneen vuorovaikutuksen kuvaamisessa. Osoittamiseen keskimäärin kuluva aika voidaan lain avulla laskea tarkasti kun tiedetään osoittimen kuljettavana oleva matka osoitettavan kohteen koko ja ihmisen motorisen järjestelmän informaationvälityskapasiteetti tiettyä osoitinlaitetta käytettäessä. Fittsin lakiin pohjautuvat mallit toimivat, eli kuvaavat tarkasti ihmisen toimintaa, mutta ei ole aivan selvää että miksi. Koska täsmällistä selitystä Fittsin lain paikkansapitävyydelle ei ole, on myös lain pätevyysalueen määrittely siis kesken. Esimerkiksi sen sopivuudesta katseenseurantalaitteella osoittamisen mallintamiseen on saatu ristiriitaisia tuloksia. Tässä kirjoitelmassa kuvaan ensiksi Fittsin lain vakiintuneita käyttötapoja ja sitten uusia soveltamistapoja ja -alueita avoimine kysymyksineen. Erityisesti viittaan usein tekstinsyötön mallintamiseen. Syynä tähän on ensinnäkin se, että itse tutkin juuri tekstinsyöttöä ja toiseksi se, että tekstinsyöttö on motorisen työn suuren osuuden vuoksi hedelmällinen sovellusalue Fittsin lakiin pohjautuville malleille.

1. Johdanto

Malleilla ja teorioilla on kaksi tärkeää tehtävää ihmisen ja tietokoneen välisen vuorovaikutuksen tutkimuksessa. Ensinnäkin mallit kuvaavat todellisuutta yksinkertaistetussa muodossa. Hyödylliset mallit ovat yksinkertaisuudestaan huolimatta kuitenkin niin täsmällisiä, että niiden avulla voidaan selittää erilaisia vuorovaikutuksessa havaittavia ilmiöitä. Kyky ymmärtää ja selittää auttaa muuttamaan vuorovaikutusta niin että esimerkiksi tehokkuus paranee kun epäoleellisia ilmiöitä aiheuttavia ominaisuuksia voidaan karsia järjestelmistä. Toiseksi teoriat ja mallit auttavat ennustamaan sunniteltujen systeemien toimintaa ennen kuin niitä on rakennettu. Tämä on tärkeää koska joissain tapauksissa kokeellinen testaaminen on erittäin työlästä ja kallista.

Tekstinsyöttömenetelmien testaaminen on hyvä esimerkki hedelmällisestä maaperästä mallien käytölle. Kirjoittaminen on nimittäin toimi, jossa suorituskyky kehittyy yleensä hyvin pitkälle. Jatkuva harjoitus hioo kirjoitustaitoa vuosien ajan kunnes saavutetaan kirjoittajalle tyypillinen suoritustaso. Yleensä käyttöliittymää suunniteltaessa ei ole mahdollista harjoituttaa laajaa koehenkilöjoukkoa vuosikausia. Niinpä, jos halutaan tietää suunniteltuun käyttöliittymään kuuluvan kirjoitusmenetelmän suorituskyky, on mallinnus tehokas keino olettaen että sopivia malleja on käytettävissä.

Fittsin lain pohjalta voidaan rakentaa malleja, jotka kuvaavat juuri harjaantuneen käyttäjän suorituskykyä tietynlaisten kirjoitusmenetelmien yhteydessä. Nämä mallit eivät sinänsä ole tämän kirjoitelman ydinasiaa ja niihin viitataan vain lyhyesti. Keskeistä sen sijaan on esittää niin kattavat tiedot Fittsin laista, että lukija pystyy niiden perusteella itse konstruoimaan tarvittavia malleja.

2. Fittsin lain alkutaival 1954-

Vuonna 1954 tilanne oli se, että Claude Shannon oli vastikään julkaissut tuloksensa, joita nykyisin pidetään infomaation välittämisen perusreunaehtoina [Shannon ja Weaver 1949]. Yksi Shannonin perustavanlaatuisiksi osoittautuneista tuloksista oli se, että signaalin, jonka teho on *P*, kyky kuljettaa informaatiota kohinan (teho *N*) vallitessa voidaan kuvata kaavalla:

$$C = W \log_2 \frac{P+N}{N} \qquad (1)$$

jossa *C* on kiinnostuksen kohteena oleva kapasiteetti ja *W* on käytetyn kaistan leveys (Hz). Kapasiteetin *C* yksikkö on bittiä sekunnissa. Perustelu bitin

käytölle informaation määrän yksikkönä ja kaksikantaisen logaritmin käyttö yllä olevassa kaavassa menee jokseenkin niin, että jos näin tehdään, niin säästytään monelta murheelta varsinkin näiden asioiden matemaattisessa käsittelyssä [Shannon ja Weaver 1949].

Ei ehkä ole ensikatsomalta aivan selvää, että miten kaavaa 1 voidaan soveltaa ihmisen liikkeisiin. Fittsin [1954] selitys kulkee jokseenkin seuraavalla tavalla.

Tietedään, että pitemmät liikkeet vievät enemmän aikaa. Samoin pienee kohteeseen on vaikeampi osua kuin suurempaan jos etäisyys on sama. Tälle tiedolle olisi hyvä löytää formaalimpi esitystapa. Lisäksi, jos näistä tiedoista voidaan perustella yhteys liikkeen välittämän informaation määrään, niin kanavan kapasiteetti voidaan mitata kokeellisesti. Mikäli kanavan kapasiteetilla on tietty maksimiarvo, sen pitäisi näkyä selvästi mittaustuloksista.

Tehtäväksi jää siis liikkeiden ja signaali- ja kohinatehon yhteyksien miettiminen ja mainitun kokeen suorittaminen.

Kohina on ehkä helpompi ymmärtää, joten aloitamme siitä. Tähdätty liike vääristyy kohinan vaikutuksesta siten, että se ei osukaan täsmälleen tähdättyyn pisteeseen. Jos kohina on satunnaista (white noise), niin riittävän suuri joukko toistoja tuottaa todennäköisesti osumapilven, joka on normaalisti jakautunut ja jonka keskipiste on tähdätyssä kohdassa. Jos oletetaan, että ihminen tähtää aina kohteen keskelle, niin huomataan että suurempi kohde sallii enemmän kohinaa ja osoitus osuu silti kohteen aluelle. Niinpä, kun toimitaan kanavan välityskyvyn ylärajalla, niin kohteen halkaisija edustaa sitä kohinan tehoa, joka voi vallita ilman että signaaliin tulee virheitä.

Se, että mikä ja millä perusteella edustaa signaalin tehoa onkin sitten hankalampi kysymys. Perustelu mennee jokseenkin niin, että jos valitsemme liikkeen pituuden signaalin tehon tilalle Shannonin kaavaan, niin se tuntuu pitävän paikkansa mitattujen liikkeiden suhteen. Muunkinlaisia perusteluja voi kehittää, mutta analogia signaalin tehon kanssa ontuu kaikissa keksimissäni perusteluissa jollain tavalla.

Kuva 1 on kaavakuva Fittsin käyttämästä osoitustehtävästä. Koehenkilö piti kädessään kynän muotoista sähköjohtimen päätä ja kosketti sillä vuorotellen kahta johtavaa pintaa. Pintojen leveyttä ja etäisyyttä muuteltiin pyrittäessä mittaamaan käden liikkeitä ohjaavan kanavan kapasiteettia.

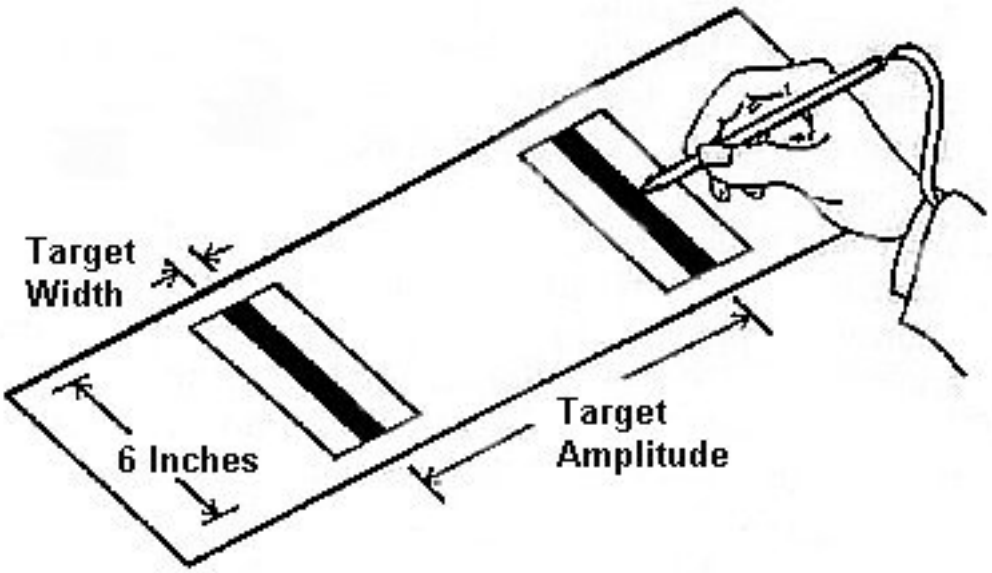
Fitts muotoili kuvan 1 osoitustehtävälle vaikeusindeksin

$$I_d = \log_2 \frac{2A}{W} \textit{bittiä / suoritus} \qquad (2)$$

jossa *A* on kohteiden välimatka ja *W* on kohteiden leveys. Jotta voitaisiin laskea arvoja kanavan kapasiteetille, täytyy aika ottaa mukaan laskuihin. Fittsin suorituskykyindeksi (I_p) on muotoa

$$I_p = \frac{1}{t}\log_2 \frac{2A}{W} \text{ bittiä / sekunti} \qquad (3)$$

jossa *t* on valituilla *A*:n ja *W*:n arvoilla mitattu keskimääräinen suoritusaika. I_p on siis tuo kaivattu ihmisen motorisen kanavan kapasiteetti.



Kuva 1. Fittsin osoitustehtävä [MacKenzie 1991].

Fittsin kaava I_p:n laskemiseksi ei kuitenkaan ole sama kuin aluksi esitelty Shannonin kaava (1) kanavan kapasiteetille. Fitts näyttääkin käyttäneen Shannonin työtä lähinnä vihjeenä siitä, että tällainen informaation välityskyvyn maksimiarvo voi olla olemassa. Fitts huomauttaa että hänen I_p määritelmänsä muistuttaa kovasti kaavaa, joka approksimoi Shannonin tarkoittamaa informaationvälityskapasiteettia tilanteessa jossa kohinan teho on vähäinen signaalin tehoon verrattuna [MacKenzie 1989].

3. Soveltaminen käytännössä

3.1 Yleistä

Fitts osoitti kokeellisesti, että em. tavalla laskettu vaikeusindeksi I_d korreloi hyvin tarkasti keskimääräisen liikkeisiin kuluneen ajan kanssa. Kokeisiin sisältyi Kuvan 1 toistuva osoitustehtävä kahdella eri painoisella kynällä, tikkujen siirto laudassa olevasta reikärivistä toiseen, ja pyöreiden,

reiäillisten, levyjen siirtäminen tikusta toiseen. Kaikissa tapauksissa laskettu I_d:n ja suoritusajan väliseksi korrelaatioksi saatiin yli 0,91.

Kokeen perusteella lasketut kapasiteetin I_p arvot eivät olleet kaikissa tilanteissa samat. Vaihteluväli oli 5,49-12,57. Fitts kuitenkin katsoo että tulokset vahvistavat arvelut siitä, että ihmisen liikkeiden informaationvälityskapasiteetilla on selvä yläraja. Tämä raja voi vaihdella erilaisissa tehtävissä, mm. koska käytetään eri lihaksia ja raajoja. Jos kuitenkin tehtävän tyyppi ei muutu paljon, niin Fittsin kaavoista voidaan johtaa käyttökelpoinen tapa laskea tehtäviin vähintään kuluvaa aikaa.

Jotta kaava olisi käyttökelpoinen kulloinkin mielenkiinnon kohteena olevassa tehtävässä, pitää käytetyn liikejärjestelmän (ihminen, raaja ja käytetty väline) kapasiteetti mitata ko. tehtävässä tai sitä läheisesti muistuttavassa tehtävässä. Tämä tapahtuu teettämällä käyttäjillä tehtäviä, joilla on laaja skaala eri I_d arvoja. Jokaisen tehtävän tekemiseen kuluva aika mitataan. Kapasiteetin käänteisluku b saadaan I_d-aika koordinaatistoon piirrettyyn pisteparveen pienimmän neliösumman menetelmällä sovitetun suoran kulmakertoimena (ts. lineaarisella regressioanalyysillä). Kuvassa 2 on esimerkki tällaisesta suoran sovituksesta. Edellä saadusta kapasiteetin käänteisluvusta saadaan tutkitun liikejärjestelmän informaationvälityskapasiteetti, jota voidaan kohtalaisen luotettavasti käyttää suoritusaikojen arvioimiseen myös mitattujen pisteiden välillä.

Suoritusaikojen kuvaamiseen Fittsin laki saadaan sovitettua lähtien I_p:n määrittelevästä kaavasta muotoon

$$t = a + b\log_2 \frac{2A}{W} \qquad (4)$$

jossa t on A:n ja W:n määrittämään liikkeeseen keskimäärin kuluva aika. Kaavan a ja b saadaan edellä mainitulla regressioanalyysillä. Se seikka, että suoran sovituksen tuottama vakio a otetaan mukaan kaavaan voi näyttää Fittsin lain kannalta jokseenkin huolestuttavalta. Tämä a nimittäin merkitsee sitä, että mittauksissa saadaan tuloksia, jotka johtavat suoraan, joka ei kulje kovin läheltä koordinaatiston origoa. Toisin sanoen kun I_d on 0, niin tehtävän suorittamiseen kuluva aika ei olekaan 0, vaan jotain muuta (yleensä >0). Selitys on se, että tehtävän suoritusaika muodostuu Fittsin lain kuvaamalla tavalla kohteiden koon ja etäisyyden suhteesta ja sen lisäksi joistain muista komponenteista. Näitä muita komponentteja kuvataan paremman puutteessa tällä vakiolla a.

Esimerkiksi Fittsin kokeessaan käyttämälle tehtävälle voidaan nyt MacKenzien [1989] valmiiksi laskemia a:ta ja b:tä käyttäen suoritusaikoja

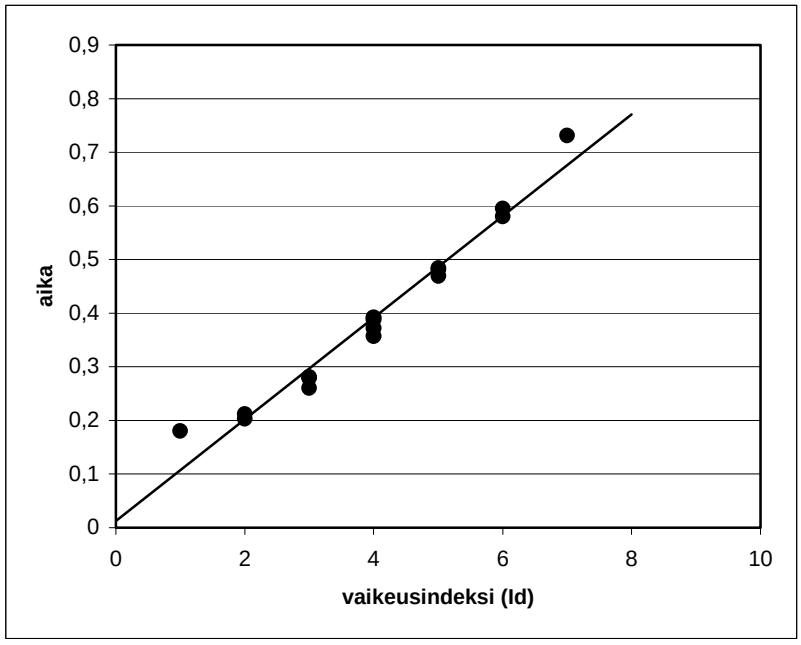
sellaisillekin tehtäville, joita ei kokeessa käytetty. Esimerkiksi 0,75 tuuman osoitettavat raidat 10 tuuman päässä toisistaan tuottavat suoritusajaksi

$$t = 0{,}0128 + 0{,}0947 \log_2 \frac{2 \times 10}{0{,}75} = 0{,}461s \qquad (5)$$

Voimme olla kohtalaisen varmoja siitä, että kyseinen laskelma pitäisi paikkansa mikäli tällainen koe järjestettäisiin ja keskimääräinen suoritusaika mitattaisiin.

3.2 Tarkennuksia

Fittsin tulokset herättivät runsaasti mielenkiintoa, mutta pian myös huomattiin että esitetty malli ei hyvästä korrelaatiosta huolimatta kuvaa suoritusajan ja tehtävän suhdetta täydellisesti. Erityisen ongelmallista on se, että pienillä I_d arvoilla Fittsin kaava tuottaa johdonmukaisesti liian lyhyitä suoritusaikoja. Kuvassa 2 nähdään Fittsin tulokset kevyellä kynällä suoritetuista kokeista. Pisteparveen sovitettu suora on alapäästä selvästi erillään pisteparvesta.



Kuva 2. Fittsin tulokset kevyellä kynällä.

Fitts paikkasi alkuperäistä kaavaansa lisäämällä kertoimen 2 *A*:n eteen I_d:n logaritmitermissä. Perusteluna oli se, että näin I_d ei pääse negatiiviseksi (elleivät osoituksen kohteet ole osin päällekkäin) ja se että näin toimien saatava I_d vastaa sitä kahdella jakamisten määrää, joka tarvitaan että 2*A*:sta saadaan *W*. Tämä jälkimmäinen argumentti on jokseenkin outo koska jos kakkonen jätetään pois, niin tällöin I_d pienenee yhdellä, eli vastaa sitä kahdella jakamisten määrää, joka tarvitaan että *A*:sta saadaan *W*. Näyttää siis siltä, että Fittsin lisäämä kakkonen

ei suinkaan selvennä tilannetta vaan pikemminkin hämmentää. Kahdella jakamisten määrällä sinänsä sen sijaan on järkevä yhteys Shannonin teorioihin. Shannon nimittäin määrittelee informaation määrän mahdollisten vaihtoehtojen määrän kaksikantaiseksi logaritmiksi, jonka yksikkönä on bitti. Fittsin osoitustehtävässä voidaan ajatella olevan 2^{Id} kappaletta *W*:n levyisiä kohteita vierekkäin. Näistä valitaan tehtävää suoritettaessa yksi. Tämän operaation välittämä informaatio on siis maksimissaan I_d. Koko kohdejoukon leveys on siis joko *A* tai *2A* riippuen siitä onko kaavassa 4 kakkonen viivan päällä vai ei.

MacKenzie [1989] on esittänyt, että käyttämällä tarkemmin Shannonin kaavaa (1) noudattelevaa muotoa

$$t = a + b\log_2 \frac{A+W}{W} \tag{6}$$

eli

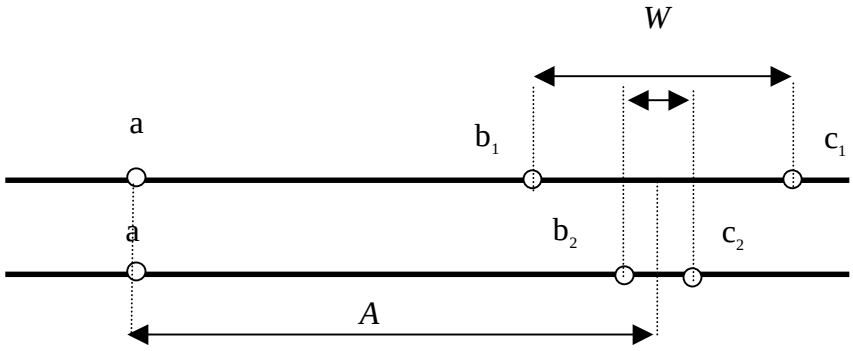
$$t = a + b\log_2 (\frac{A}{W} + 1) \tag{7}$$

Fittsin laista, vältetään negatiiviset vaikeusindeksit myös ääritilanteissa ja saadaan lisäksi parempi korrelaatio vaikeusindeksien ja keskimääräisten suoritusaikojen välille. Näin Fittsin lain teoreettinen pohja on paremmassa kunnossa. Shannonin kaava pätee nimittäin myös jos kohinan teho on kohtalaisen suuri signaalin tehoon nähden toisin kuin Fittsin käyttämä approksimaatio [MacKenzie 1989].

3.3 Käsitehygieniaa

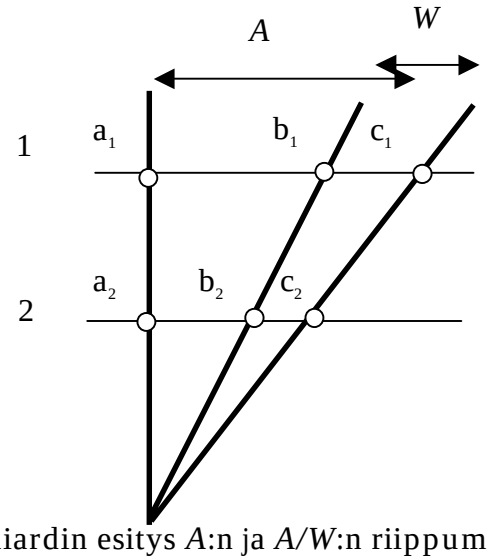
Käytännössä Fittsin lakiin liittyviä kokeita suoritettaessa manipuloidaan kahta riippumatonta muuttujaa. Nämä ovat *A*, eli kohteen etäisyys lähtöpaikasta ja *W*, eli kohteen leveys lähestymissuunnassa. Voidaan kuitenkin väittää, että näitä kahta muuttujaa ei voi todellisuudessa manipuloida toisistaan riippumatta, koska Fittsin laissa oleva termi *A/W* sitoo ne toisiinsa. Toisin sanoen Fittsin lain pätevyyden tutkimisessa joudutaan turvautumaan osin Fittsin lakiin. Jos ollaan tarkkoja, niin muuttujien vaikutusta pitäisi tarvittaessa pystyä tutkimaan erikseen. Jos näin ei voida tehdä, niin kokeiden yhteydessä pitäisi rehellisyyden nimissä puhua niistä muuttujista joita todellisuudessa manipuloidaan. Yves Guiard on äskettäin esittänyt, että Fittsin kokeen kaltaisissa kokeissa on kaksi muuttujaa, joita voidaan tutkia vähemmin kehäpäätelmin. Nämä ovat absoluuttinen etäisyys (*A*) ja liikkeen vaikeus (*A/W*). Liikkeen vaikeuden manipulointia voidaan kuvata graafisesti kuten kuvassa 3 on tehty [Guiard 2001].

Kuvassa 3 on kaksi yksiulotteista osoitustehtävää. Kummallakin vaakasuoralla janalla on kolme pistettä: a, b ja c. Pisteiden b ja c väli on *W*, eli kohteen leveys. Etäisyys pisteestä a pisteiden b ja c puoliväliin on etäisyys *A*. Kuvasta huomataan, että *A* voidaan pitää vakiona kun *W*:tä muutetaan. Samalla kun *W* muuttuu, muuttuu myös suhde *A/W*. *A/W* on Guiardin tarkoittama tehtävän vaikeus eri osoitustehtävän pituus suhteessa kohteen kokoon. Koska *W*:n manipulointi vaikuttaa itse asiassa suhteeseen *A/W*, kannattaa kokeessa ajatella manipuloitavan tehtävän vaikeutta eli *A/W*:tä.



Kuva 3. Guiardin esitys *W*:n ja *A/W*:n riippuvuudesta.

W:n muuttaminen pitäen *A* vakiona johtaa siis väistämättä *A/W*:n muuttumiseen. Sen sijaan *A*:ta voidaan muuttaa pitäen *A/W* vakiona. Tätä havainnollistaa Kuva 4.



Kuva 4. Guiardin esitys *A*:n ja *A/W*:n riippumattomuudesta.

Kuvassa 4 suorille 1 ja 2 muodostuu pisteiden a, b ja c jakamana samanlaiset yksiulotteiset osoitustehtävät kuin kuvassa 3. Näissä tapauksissa suhde *A/W* on sama. *A*, eli etäisyys pisteestä a pisteiden b ja c puoliväliin on suorilla 1 ja 2 eri

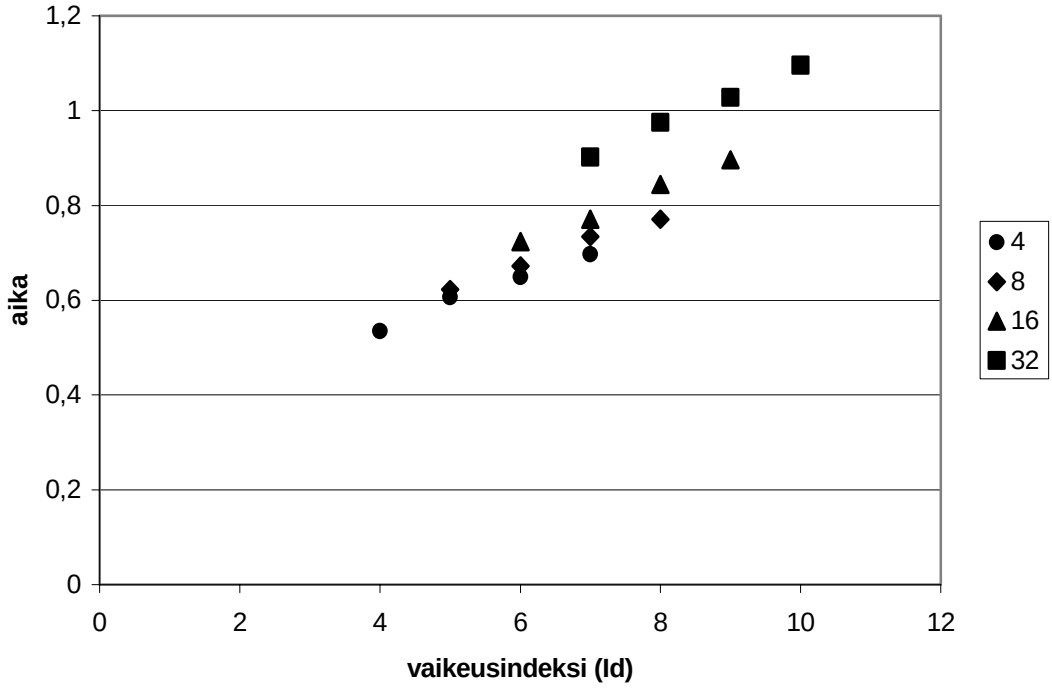
pituinen. Yleisesti ottaen *A*:ta eli osoitustehtävän kokoa voidaan muuttaa vapaasti koskematta suhteeseen *A/W*.

Kuviin 3 ja 4 liittyvän selityksen tuloksena voidaan siis todeta, että *A* ja *A/W* ovat kokeissa manipuloitavissa toisistaan riippumatta. Yksi hyvä syy sille että miksi näin pitäisi tehdä on se, että osoitustehtävän koolla tiedetään olevan vaikutus suoritusaikaan. Vaikutus on selvästi pienempi kuin Fittsin lain vakuuttavasti demonstroima tehtävän vaikeuden (*A/W*) tehtävän koosta riippumaton vaikutus, mutta se on silti saatu tilastollisesti luotettavalla tavalla esiin useissa kokeissa. Esimerkiksi Accot ja Zhai mittasivat 17% vaikutuksen I_d:ssä kun liikkeen kokoa (*A*) muutettiin 16-kertaisesti [Accot ja Zhai 2001].

Jos siis osoitustehtäviä mallinnettaessa käytetään perusteena sekä koosta riippumatonta perinteistä Fittsin lain tulkintaa että tehtävän koon vaikutusta, niin voidaan saada aikaan tarkempia malleja. Yleensä Fittsin lain I_d:n ja kunkin tehtävän keskimääräisten suoritusaikojen korrelaatiot ovat erittäin korkeita ($r>0.95$). Tämä ei kuitenkaan kerro koko totuutta, sillä käytännössä käy usein niin, että koejärjestelyt eivät tuota täydellistä tehtävän koon ja vaikeuden välistä ristiintaulukointia. Suurimmat vaikeusarvot (I_d) saadaan pitkillä etäisyyksillä ja pienillä kohteilla. Pienimmät taas lyhyillä etäisyyksillä ja suurilla kohteilla. Näin tehtävän koon vaikutus on saman suuntainen kuin tehtävän vaikeuden vaikutus. Vaikka korrelaatio onkin korkea, niin käytetty muuttuja selittävä muuttuja (*A/W*) ei todellisuudessa selitäkään korrelaatiota kokonaan. Käytännössä tämä käsitteellinen tulkintavirhe on merkityksetön, tai ainakin hyvin vähämerkityksinen koska käytännön tilanteet ovat samanlaisia kuin koetilanteet. Tämä tarkoittaa sitä, että esimerkiksi hiirellä ei useinkaan tarvitse osoittaa valtavia kohteita hyvin kaukana toisistaan, koska näytön koko ei mahdollista tätä. Joskus tosin yritetään osoittaa todella pieniä kohteita hyvin lähellä toisiaan, mutta sekään ei ole järkevää. Parempi on suurentaa näkymä helpommin käsiteltävään kokoon.

Kuva 5 havainnollistaa Guiardin kritiikkiä perinteistä Fittsin lain koetta kohtaan. Kuvan data on Fittsin reikälevyjensiirtotehtävästä, jossa reiän halkaisijat olivat 0,0625, 0,125, 0,25 ja 0,5 tuumaa ja tappien etäisyydet 4, 8, 16 ja 32 tuumaa. Kuvassa 5 nähdään neljä sarjaa pisteitä. Parhaiten erottuu ylin neljän sarja (neliöt). Tämä on pisimmän etäisyyden (32 tuumaa) sarja. Seuraavana ylhäältä on 16 tuuman sarja (kolmiot), sitten 8 tuuman ja alimpana 4 tuuman. Liikkeen koko *A* vaikutti siis selvästi suoritusaikoihin, mutta silti suoritusaikojen selittämiseen käytettiin vain *A/W*:tä (tarkalleen ottaen *A/W*:stä laskettua I_d:tä). Lisäksi nähdään kuinka suurikokoisten tehtävien tuottama pitempi suoritusaika johtaa Fittsin valitsemilla *A*:n ja *W*:n arvoilla samansuuntaiseen vaikutukseen kuin *A/W*:llä selitettävissä oleva. Jos kaikki sarjat jatkuisivat koko I_d-alueelle (4-10), olisi saavutettu korrelaatio

suoritusaikojen ja I_d arvojen välillä selvästi matalampi. Kuten kuitenkin jo edellä mainittiin, tämä koeasettelun puute vastaa useinmiten tietokoneen käyttötilanteita, joten se ei ehkä ole käytännön kannalta kovin vakava.



Kuva 5. Fittsin pyöreiden reiällisten levyjen siirtotehtävän tulokset.

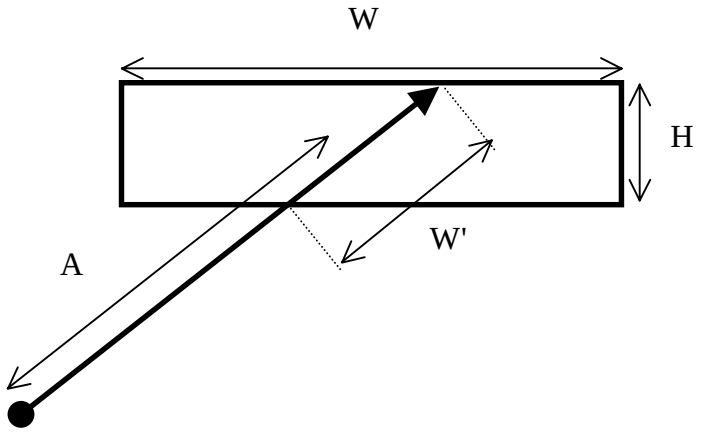
3.4 Hiiristä ja kaksiulotteisessa osoittamisessa

Fitts perusteli lakinsa pätevyyttä erilaisilla suoran osoittamisen ja siirtämisen tehtävillä. Käytössä oli työkaluja (kynänomainen laite, tappeja ja pyöreitä reiällisiä levyjä), mutta niitä käsiteltiin suoraan sormin. Ei ole tällä perusteella täysin selvää, että laki pätee myös epäsuoraan osoittamiseen kuten hiiren käyttöön vaikkapa graafisen käyttöliittymän kursoria ohjattaessa. Asiaa on selvitelty kokeellisesti, ja on käynyt ilmi, että laki toimii hyvin myös hiirtä käytettäessä. Myös ohjauspallojen ja ohjaustikkujen käytön on todettu noudattavan Fittsin lakia ainakin jos käyttöä ohjaa näytöltä saatava näköpalaute.

Fittsin lain soveltamiseen graafisissa käyttöliittymissä tapahtuvan osoittamsen kuvaamiseen liittyy kuitenkin toinenkin epävarmuustekijä. Fittsin koe oli nimittäin tavallaan yksiulotteinen koska osoitettavien kohteiden kynän kulkusuunnan mukaista leveyttä kontrolloitiin, mutta niiden pituus pidettiin jatkuvasti niin suurena, että se ei vaikuttanut kokeen kulkuun. Fittsin reikälevyn- ja tapinsiirtotehtävissä, joissa tehtävä ei ole samalla lailla yksiulotteinen, suoritusaikojen ja I_d:n välinen korrelaatio oli matalampi. Ei ole itsestään selvää, että Fittsin lakia voidaan sellaisenaan soveltaa osoittamiseen

jos kappale voi olla merkitsevän kapea kahdessa ulottuvuudessa. Tällöin ei myöskään ole aivan selvää miten mallissa käytetty *W* pitäisi mitata.

Kokeet ovat osoittaneet, että nämä huolet ovat sikäli turhia, että Fittsin laki toimii hyvin, kunhan *W* määritellään järkevästi. Ehkä intuitiivisesti järkevin *W*:n määritelmä on määritellä se kohteen leveydeksi lähestymissuunnassa (kuvan 4 *W*'). MacKenzie ja Buxton [1992] totesivatkin, että näin määritelty *W* toimii ihan hyvin, eli malli kuvaa todellisuutta tarkasti. Toinen MacKenzien ja Buxtonin vertailussa hyvin menestynyt *W*:n määritelmä oli kohteen korkeuden ja leveyden minimi (kuvan 4 *H*). Korkeuden ja leveyden minimi tuotti hieman korkeamman korrelaation, mutta ero *W*' -malliin ei ollut tilastollisesti merkitsevä.



Kuva 4. A ja W 2-ulotteisessa tilanteessa.

3.5 Virallisesti ottaen ISO-9241 osa 9

Fittsin lakiin perustuva menetelmä on standardoitu tapa tutkia osoituslaitteen suorituskykyä. ISO-9241 (Ergonomic design for office work with visual display terminals (VDTs)) standardin osa 9 (Requirements for non-keyboard input devices) käsittelee osoitinlaitteita [Douglas et al. 1999]. Tämän osion viimeisimmässä luonnoksessa kuvattu laitteen suorituskykyä osoituksessa tutkiva testausmenetelmä on perinteinen Fittsin lain koe. Standardoidun testausmentelmän etuna on se, että tulosten vertailtavuus paranee kun kaikki testaajat noudattavat samaa menetelmää. Tätä kirjoitettaessa osa 9 on ISO:n organisaatiossa viimeisessä vaiheessa ennen hyväksymistä osaksi virallista standardia.

Standardiehdotus määrittelee käsitteen *throughput*, joka on sama kuin aiemmin esitelty informaationvälityskapasiteetti I_p eli suoritusaika jaettuna vaikeusindeksillä I_d. Osoittamisessa tapahtuvia virheitä ei lasketa erikseen, vaan vaikeusindeksin (ja siten myös informaationvälityskapasiteetin)

laskennassa käytetään kohteen todellisen leveyden (*W*) sijasta osumapilvestä laskettua leveyttä W_e, joka lasketaan osumakoordinaattien keskihajonnasta kertomalla se luvulla 4,133. Näin toimittaessa osoituslaitteen virheherkkyys muodostaa osan *throughput*ista. Kun osuminen on vaikeampaa, osumapilvi kasvaa, jolloin vaikeusindeksi I_d pienenee, jolloin t*hroughput* pienenee. Suurempi *throughput* kertoo paremmasta osoitinlaitteesta.

3.6 Esimerkki

Fittsin lain avulla voidaan mallintaa liikkeisiin keskimäärin kuluvia aikoja kun liike on kognitiivisesti helppo. Kognitiivinen helppous tässä tarkoittaa sitä, että liikkeen suorittaminen ei vaadi tietoista suunnittelua tai päätöksentekoa. Monet osoitustehtävät ovat kognitiivisesti helppoja kunhan niitä on harjoiteltu tarpeeksi. Tämä pätee myös tietokoneen käytössä esiintyviin osoitustehtäviin. Hiiren käyttö ei vaadi erityistä vaivannäköä ainakaan muutaman vuoden harjoittelun jälkeen.

Jos tehtävä on selvästi yksittäisen kohteen osoittamista mutkikkaampi, niin Fittsin lain mukaisilla malleilla saadaan mallinnettua vain liikkeisiin tarvittavaa aikaa. Tämä vastaa sellaisen käyttäjän suorituskykyä, jonka ei enää tarvitse tietoisesti ohjata liikkeitään. Seuraavaksi esiteltävä malli mallintaa näppäimistön käyttöä yhdellä sormella juuri tällaisessa tapauksessa. Kysymys, johon malli vastaa koskee tietyllä näppäiten asettelulla täysin oppineen käyttäjän saavutettavissa olevaa maksimaalista kirjoitusnopeutta.

Yhdellä sormella näppäiltäessä edellinen näppäin on seuraavaan näppäimeen kohdistuvan osoituksen lähtöpaikka. Kun lähtöpaikka ja kohde tunnetaan, voidaan osoitustehtävä mallintaa Fittsin lakia käyttäen. Jonkin tietyn tekstin kirjoittamiseen kuluva aika voidaan simuloida laskemalla sormen kulkeman reittiin kuluva aika. Jos halutaan yleisempiä tuloksia jotain tiettyä kieltä ja näppäimistöä käytettäessä, on kaksi vaihtoehtoa: joko simuloidaan kaikki kielellä kirjoitetut tekstit, tai käytetään jotain hieman vähätöisempää mallinnustapaa. MacKenzie ja Zhang [1995] esittävät tällaisen vähätöisemmän laskentakeinon. Se voidaan toteuttaa tavallisella taulukkolaskentaohjelmalla. Näppäimeltä toiselle siirtymiseen kuluva aika lasketaan ko. näppäimistölle mitattuja Fittsin lain vakioita *a* ja *b* käyttäen aivan tavalliseen tapaan jokaiselle näppäinparille (englannin kielessä pareja on 27x27 jos välilyönti lasketaan mukaan). Tämän jälkeen jokainen tällainen aika kerrotaan ko. näppäinparin keskimääräisellä esiintymistiheydellä. Kun näin saadut 27x27 lukua lasketaan yhteen, saadaan mallinetulle kieli-näppäimistö yhdistelmälle tunnusluku, joka kertoo yhteen näppäimen painallukseen keskimäärin kuluvan ajan. Tämän ajan käänteisluku kertoo montako merkkiä täysin oppinut käyttäjä voisi hyvissä olosuhteissa syöttää sekunnissa.

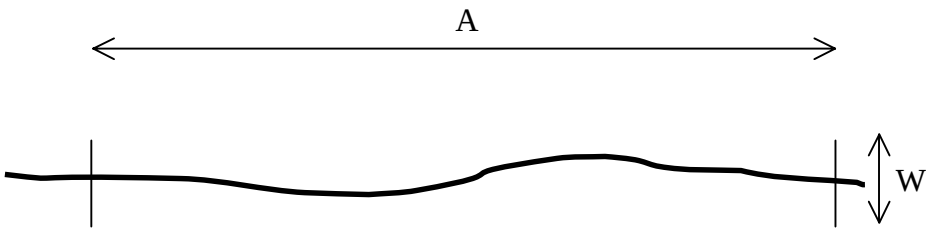
MacKenzien ja Zhangin mallin perusteella voidaan yrityksen ja erehdyksen menetelmällä suunnitella nopeampia näppäinasetteluita [MacKenzie et al. 1999]. Voidaan myös suunnitella algoritmi, joka yrittää optimoida näppäinasettelun niin että valittua merkkiparijakaumaa voidaan syöttää mahdollisimman nopeasti. Tämä optimointitehtävä kuuluu niihin, joiden täydellinen ratkaiseminen on liian työlästä nykyisille tietokoneille (ja nykyisen tiedon mukaan myös tulevaisuuden tietokoneille). Lähes optimaalisia ratkaisuja voidaan kuitenkin tuottaa erilaisilla approksimointialgoritmeilla. Hunter et al. [2000] ovat esittäneet yhden tällaisen algoritmin tuottamia tuloksia. Tällä tavalla automaattisesti tuotettujen näppäimistöjen optimaalisuudesta ei ole muita takeita kuin se, että ne ovat käytetyn mallin mukaan tehokkaampia kuin muut kokeillut vaihtoehdot. Vaikka todistettavasti optimaalisin näppäinasettelu joskus löydettäisiinkin, se tuskin olisi juurikaan tehokkaampi kuin nyt tiedossa olevat. Itse asiassa näyttää siltä, että Fittsin lain kannalta optimaalisin näppäimistö ei ole käytännössä paras. Täysin oppineen käyttäjän suorituskykyä tärkeämpää on nimittäin usein se mitä tapahtuu kun näppäinasettelun käyttöä opetellaan. Maksimaalisesta suorituskyvystä kannattaa luovuttaa melko paljon, jos tällä keinolla saadaan näppäinasettelun oppiminen helpommin alulle. Zhai ja Smith [2001] ovat tutkineet tätä vaihtoehtoa ja yhdistäneet edellä mainittuun approksimointialgoritmiin taipumuksen tuottaa aakkosjärjestykseen pyrkiviä näppäinasetteluita. Zhain ja Smithin [2001] kokeissaan saamat tulokset viittaavat siihen, että uuden näppäinasettelun käyttöä aloitettaessa näin saavutetaan merkittävää etua.

4. Ohjauslaki

4.1 Yleistä

Fittsin laki on siinä mielessä yksiulotteinen, että se ei ota kantaa muuhun kuin etäisyyksiin suunnassa, joka johtaa suoraan liikkeen lähtöpisteestä kohteeseen. Usein se, mitä matkalla lähtöpisteestä kohteeseen tapahtuu, on hyvinkin oleellista. Accotin ja Zhain [1997] kehittämä muoto Fittsin laista kuvaa liikkeeseen kuluvan ajan kuljetun reitin ja sallitun poikkeaman funktiona. Tämä muoto nopeuden ja tarkkuuden väliselle suhteelle tunnetaan nimellä *steering law*. Suorahko suomennos lain nimelle lienee ohjauslaki. Paremman käännöksen puutteessa käytän tätä termiä.

Accotin ja Zhain perustelu lailleen on jotakuinkin seuraavanlainen. Kokeellisesti voidaan todentaa, että kuvan 5 mukaisen tehtävän suoritusaika noudattaa MacKenzien muotoilemaa Fittsin lain varianttia (7), joka on siis se joka noudattaa lähimmin Shannonin alkuperäistä kaavaa (1).



Kuva 5. Kursorin jälki läpi kahden kohteen.

Tämä Fittsin lain variantti on muotoa:

$$t = a + b\log_2(\frac{A}{W} + 1) \quad (8)$$

Käsiteltävien kaavojen yksinkertaistamiseksi tarkastellaan vain lain I_d termiä, eli pudotetaan pois kertoimet *a* ja *b*

$$I_d = \log_2(\frac{A}{W} + 1) \quad (9)$$

Jos em. kahden kohteen tehtäviä asetetaan samalle matkalle kaksi peräkkäin, niin tehtävän kokonaisvaikeus muodostuu kahdesta osasta, joiden pituus on puolet alkuperäisestä, eli

$$I_{d2} = 2\log_2(\frac{A}{2W} + 1) \quad (10)$$

Edelleen jos tehtäviä on peräkkäin N kappaletta, niin

$$I_{dN} = N\log_2(\frac{A}{NW} + 1) \quad (11)$$

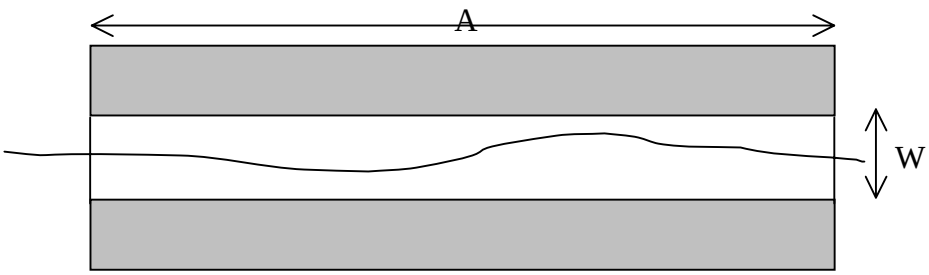
Kun N lähestyy ääretöntä, niin vaikeusindeksi lähestyy arvoa

$$\frac{A}{W\ln 2} \quad (12)$$

Tästä *1/ln2* voidaan vakiona unohtaa (tai sen voidaan ajatella siirtyvän osaksi regressioanalyysillä saatavaa kerrointe *b*) ja näin saadaan kuvan 6 tyyppiselle suoralle tunnelinläpäisylle suoritusaika kaavalla

$$t = a + b\frac{A}{W} \quad (13)$$

Accot ja Zhai suorittivat myös kuvan 6 mukaisen kokeen käyttäen *A* ja *W* avoja, jotka johtavat I_d arvoihin väliltä 2,8 - 50 ja saivat tulokseksi korkean korrelaation (r^2=0,97) I_d arvojen ja mitattujen suoritusaikojen välille.



Kuva 6. Kursorin jälki läpi tunnelin.

Ohjauslain yleinen muoto käyrälle K on

$$T_k = a + b\int_K \frac{ds}{W(s)} \qquad (14)$$

Tästä voidaan johtaa lakeja eri muotoisille käyrille, mutta edellä esitetty kaava suoralle tunnelille riittää varsin pitkälle, sillä monet osoitustehtävät (esim valikkovalinnat) muodostuvat suorista tunneleista. Todellisia (mitattuja) käyriä käytettäessä toimitaan yleensä tietokonetta apuna käyttäen. Tällöin voidaan käyrä jakaa peräkkäisiksi kahden kohteen tehtäviksi. Kun jakoväli on riittävän tiheä, niin saadaan laskutarkkuuden rajoissa sama tulos kuin matemaattisemmalla lähestymistavalla.

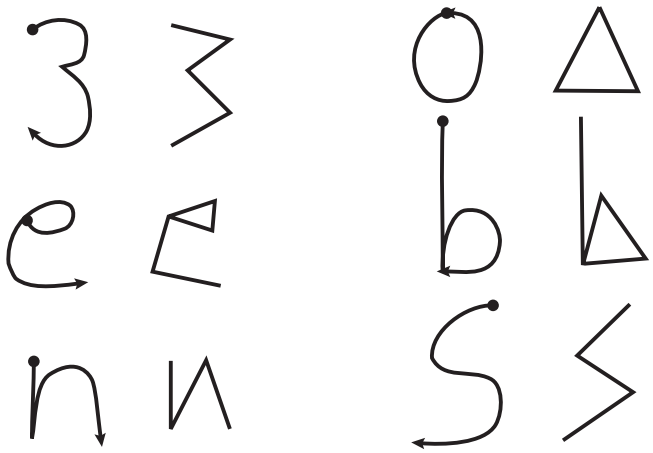
4.2 Esimerkki

Edellä viittasin tietokoneen käyttöön ohjauslain soveltamisessa ja totesin, että kun käyriä käsitellään riittävän pienissä paloissa, niin saadaan tarkkoja tuloksia ilman että jokaiselle käyrälle tarvii löytää matemaattisesti oikea funktio, jota pitkin ohjauslain integraali voidaan laskea. Suoran tunnelin ohjauslaki on kuitenkin niin yksinkertainen, että joskus voi tuntua turhan vaivalloiselta käydä ohjelmoimaan tietokonetta sen käyttämiseksi. Esitänkin nyt esimerkin, jossa ohjauslakia käytetään käyriä käsin piirrettäessä kuluvan ajan arvioimiseen ilman tietokonetta.

Tavoitteena on käyrää vilkaisemalla pystyä toteamaan kauanko sen piirtämiseen keskimäärin kuluu aikaa. Intuitiivisesti voisi olettaa että mitä sykkyräisempi käyrä on sen kauemmin sen piirtäminen kestää. Niinpä "I" on aika nopea piirtää (suora viiva) sen sijaan "W" on jonkin verran hitaampi, ehkä noin neljä kertaa hitaampi, koska se muodostuu neljästä suorasta viivasta. Itse asiassa tämä havainto on sama kuin sanoisi, että valitaan jokin ohjauslain tunneli, jolla on jotkut tietyt arvot *A* ja *W* ja asetetaan näitä tunneleita "I":n ja "W":n viivojen päälle. Kun kaikki viivat on peitetty, lasketaan tunneleiden määrä ja näin saadaan arvio piirtoajalle. Toisin sanoen ohjauslaki, kuten monet

muutkin tieteelliset saavutukset, on vain täsmällisempi muoto ihan jokapäiväiselle ja selvälle asialle.

Esimerkkiä täytyy vielä jatkaa, sillä kaikki merkit eivät muodostu suorista viivoista. Jos vaaditaan, että pyöreämuotoiset käyrät piirretään uudelleen käyttäen mahdollisimman pientä määrää suoria viivoja niin, että käyrä on edelleen tunnistettavissa, päästään yksinkertaiseen menetelmään, jolla voidaan arvioida kynää nostamatta piirrettävien käsinkirjoitusmerkkien piirtoaikoja. Esimerkkejä tällaisesta käyrien kuvaamisesta joukolla suoria viivoja on Kuvassa 7.



Kuva 7. Käyriä ja niiden minimaalisia suorin viivoin piirrettyjä versioita.

Käytetyttyjen suorien viivojen voidaan siis ajatella olevan ohjauslain tunneleita. Näillä tunneleilla on kaikilla sama *A/W* suhde. Tiedämme, että täsmällisesti ottaen ohjauslaki edellyttäisi että *W* on tunnettu koko käyrän matkalta. Koska *W*:tä ei tunneta, niin tämän yksinkertaisen mallin antama piirtoaika-arvio on varsin karkea. Järjestämässäni kokeessa, jossa arvoin neljä 12 henkilön piirtämää käsinkirjoitusmerkistöä, mallin antamat arviot merkkien vaikeusindeksistä ja mitatut piirtoajat korreloivat voimakkaasti. Korrelaatiokertoimien neliö (r^2) oli keskimäärin 0,7:n paikkeilla [2001]. Tämä siis kullekin merkistölle ja henkilölle erikseen. Jos luokitellaan kaikki 12 eri henkilöltä mitatut piirtoajat mallin antamien vaikeusindeksiarvioiden mukaan ja lasketaan kullekin luokalle piirtoaikojen keskiarvo, niin saatu korrelaatiokerroin on selvästi korkeampi (r^2>0.91).

Voidaan siis sanoa, että yleisesti ottaen malli kuvaa hyvin populaation keskimääräisiä piirtoaikoja, mutta ei pysty kovin tarkasti ennustamaan yksittäisen henkilön keskimääräisiä piirtoaikoja yksittäisen merkin piirtoajasta puhumattakaan.

5. Kritiikkiä

5.1 Fittsin Lain Ekstensio

Fittsin laki operoi varsin matalalla tasolla. Kaikki kognitiivinen prosessointi on Fittsin lain kokeissa kauhistus. Fittsin lakiin pohjautuvat mallit mallintavat vain motorisen järjestelmän informaationvälityskapasiteettia ja kaikki muu mikä vaikuttaa liikkeisiin jää sen ulkopuolelle. Ihmisen toiminnan kuvaamisessa Fittsin lailla on siis varsin kapea sovellusalue. Ihminen tyypillisesti miettii ja tekee päätöksiä. Näihin toimiin kuluvasta ajasta ei saada Fittsin lain avulla minkäänlaista kuvaa.

Fittsin laki edellyttää näköpalautteen käyttöä liikkeen ohjaamisessa. On onnistuttu kehittämään koejärjestelyjä [Schmidt et al 1979] joissa nopeita liikkeitä voidaan suorittaa ilman liikettä ohjaavaa näköpalautetta ja näissä tapauksissa Fittsin laki ei kuvaa suoritusaikoja kovinkaan tarkasti. Vaikka kaikki Schmidtin ja kumppaneiden [1979] tästä vetämät johtopäätökset eivät ehkä pidäkään paikkaansa [Meyer ja Smith 1982], niin Fittsin lain riippuvuus jonkinlaisesta palautteesta näyttää ilmeiseltä. Esimerkiksi lain sopivuus silmän liikkeiden mallintamiseen on epäselvää siksi, että jos käyttäjä ei kokeessa saa jatkuvaa näköpalautetta katseensa paikasta, niin liikkeisiin kuluva aika näyttäisi olevan ennemminkin lineaarisessa suhteessa matkaan kuin Fittsin lain kuvaamassa logaritmisessa suhteessa [Silbert ja Jacob 2000, Abrams et al. 1989, Partala et al. 2001]. Tällöin silmät liikkuvat kuten tavallisestikin ympäristöä katsellessa käyttäen pitkiä etukäteen ohjelmoituja sakkadisia liikkeitä. Jos sen sijaan katseen paikka tehdään näkyväksi, niin käyttäjät turvautuvat osin muihin silmän liikuttelutapoihin (seuraaminen, usempi lyhempi sakkadi) ja tuloksena näyttäisi olevan Fittsin lain mukainen liikkeen yleiskuva [Ware ja Mikaelian 1987, Miniotas 2000].

5.2 Taustaoletuksista

Keskeisin oletus Fittsin laissa on se, että osoittimen (sormi, kursori, tms.) ohjaus voidaan rinnastaa Shannonin tarkoittamaan tiedonsiirtokanavaan. Tämän oletuksen paikkansapitävyys saattaa näyttää varsin epätodennäköiseltä kun perehtyy tehtyjen kokeiden tuloksiin. Jos tällainen selkeä maksimikapasiteetti on olemassa, luulisi, että se ilmenee jokaisessa kokeessa. Erityisesti odottaisi, että eri kokeissa kapasiteetille saataisiin sama arvo. Näin ei kuitenkaan ole.

Esimerkiksi MacKenzie [1992] listaa kuuden eri kokeen tulokset. I_p vaihtelee näissä tuloksissa välillä 2,3-12bps. Tältä pohjalta on aika vaikea sanoa mitään kovin varmaa hiirellä osoittamisen informaationvälityskapasiteetista. I_p:n vaihtelevuus kokeesta toiseen on siis Fittsin lain soveltamiseen liittyvä tunnettu ongelma. Osa vaihtelevuudesta voidaan selittää I_p:n laskemiseen käytetyillä

menetelmillä, koejärjestelyllä, koehenkilöillä, virheiden käsittelyllä ja muilla syillä, mutta tämä ei poista sitä ongelmaa, että julkaistujen tutkimustulosten vertailu on vaikeaa. Tekeillä oleva ISO-9241 standardi voi ehkä osaltaan ohjata tutkijoita käyttämään menetelmiä, jotka johtavat vertailukelpoisiin tuloksiin. Itse lain pätevyyden kannalta tämäkään ei kuitenkaan ole mikään pelastus. Lienee syytä hyväksyä se tosiasia, että ihmisen toimintaa ei voi kuvata samankaltaisella tarkkuudella kuin monien mekaanisten systeemien. I_p pitää mitata mallinnettavalle tehtävälle erikseen, mikäli tahtoo saada luotettavia tuloksia.

Fittsin lakia on yritetty selittää vielä alemman tason tapahtumilla. Näin vältettäisiin edellä mainittu ongelma teoreettisen perustelun ja todellisuuden jonkinasteisen epäyteensopivuuden kanssa. Tällaiset perustelut yrittävät etsiä osoitusliikkeestä osia, joiden suhteilla voitaisiin perustella Fittsin lain logaritminen suhde ajan ja osoitustehtävän mittojen välillä [Meyer et al. 1982]. Tällaisten mallien perustelut kuitenkin pettävät joissain tilanteissa ja siksi ne eivät ole yleisesti käytössä. Ei siis ole aivan selvää miksi Fittsin laki pitää paikkansa niin monissa eri tilanteissa. Jos tähän saataisiin jokin selvyys, niin olisi paljon helpompi päättää milloin tehtävän mallintamiseen on järkevää käyttää Fittsin lakia ja milloin ei.

Viiteluettelo

[Abrams et al. 1989] Richard A. Abrams, David E. Meyer ja Sylvan Kornblum, Speed and Accuracy of Saccadic Eye Movements: Characteristics of Impulse Variability in the Oculomotor System, *Journal of Experimental Psychology: Human Perception and Performance*, 1989, **15**, 3, 529-543.

[Accot ja Zhai 1997] Johnny Accot ja Shumin Zhai, Beyond Fitts' Law: Models for Trajectory-Based HCI-Tasks, *Proceedings of the CHI1997*, ACM, New York, 295-302, 1997.

[Accot ja Zhai 1999] Johnny Accot ja Shumin Zhai, Performance Evaluation of Input Devices in Trajectory-based Tasks: An application of The Steering Law, *Proceedings of the CHI 1999,* ACM, 466-472, 1999.

[Accot ja Zhai 2001] Johnny Accot ja Shumin Zhai, Scale Effects in Steering Law Tasks. *CHI Letters: Human Factors in Computing Systems, CHI 2001*, 2001, **3**, 1, ACM,1-8.

[Douglas et al. 1999] Sarah A. Douglas, Arthur E. Kirkpatrick ja Scott MacKenzie, Testing Pointing Device Performance and User Assessment with the ISO 9241 Part 9 Standard, *Proceedings of the CHI 1999*, 215-222, ACM, 1999.

[Fitts 1954] Paul M. Fitts, The Information Capacity of the Human Motor System in Controlling the Ampliture of Movement. *Journal of Experimental Psychology*, 1954, **47**, 6, 381-391.

[Guiard 2001] Yves Guiard, Disentangling Relative from Absolute Amplitude in Fitts' Law Experiments. *CHI2001 Extended Abstracts*, ACM, 2001, 315-317.

[Hunter et al. 2000] Michael Hunter, Shumin Zhai ja Barton A. Smith, Physics-based Graphical Keyboard Design, *CHI2000 Extended Abstracts*, ACM, 2000, 157-156.

[Isokoski 2001] Poika Isokoski, Model for Unistroke Writing Time, *CHI Letters: Human Factors in Computing Systems, CHI2001*, 2001, **3**,1, 357-364, ACM.

[MacKenzie 1992] Scott MacKenzie, Movement time prediction in human-computer interfaces, *Proceedings of Graphics Interface 1992*, Canadian Information Processing Society, 1992, 140-150.

[MacKenzie 1991] Scott MacKenzie, *Fitts' Law as a Performance Model in Human Computer Interaction*. Väitöskirja, University of Toronto, Kanada, 1991.

[MacKenzie 1989] Scott MacKenzie, A note on the information theoretic basis for Fitts' law. *Journal of Motor Behavior*, 1989, **21**, 323-330.

[MacKenzie ja Buxton 1992] Scott MacKenzie ja William Buxton, Extending Fitts' Law to Two-Dimensional Tasks, *Proceedings of the CHI1992*, 1992, ACM, 219-226.

[Meyer et al. 1982] David E. Meyer, J. E. Keith Smith ja Charles E. Wright, Models for the Speed and Accuracy of Aimed Movements, *Psychological Review*, 1982, **89,** 5, 449-482.

[Miniotas 2000] Darius Miniotas. Application of fitts' law to eye gaze interaction. *CHI 2000 Extended Abstracts*, ACM, 2000, 339-340.

[Soukoreff ja MacKenzie 1995] William Soukoreff ja Scott MacKenzie, Theoretical Upper and Lower Bounds on Typing Speeds Using a Stylus and Keyboard, *Behaviour and Information Technology*, 1995, **14**, 370-379.

[Schmidt et al. 1979] Richard A. Schmidt, Howard Zelaznik, Brian Hawkins, James S. Frank ja John T. Quinn Jr., Motor Output Variability: A Tehory for the Accuracy of Rapid Motor Acts, *Psychological Review*, 1979, **86,** 5, 415-451.

[MacKenzie et al. 1999] MacKenzie, I. S., Zhang, S. X., & Soukoreff, R. W. Text entry using soft keyboards. *Behaviour & Information Technology*, 1999, **18**, 235-244.

[Partala et al. 2001] Timo Partala, Anne Aula ja Veikko Surakka, Combined Voluntary Gaze Direction and Facial Muscle Activity as a New Pointing Technique, (to appear) *in the 8th IFIP TC.13 Conference onHuman-Computer Interaction (INTERACT2001)*, Tokyo, Japan, July 9-13, 2001.

[Shannon ja Weaver 1949] Claude Shannon ja Warren Weaver, *The mathematical theory of communication*. University of Illinoiss Press, Urbana, Illinois, USA, 1949.

[Sibert ja Jacob 2000] Linda E. Sibert ja Robert J. K. Jacob. Evaluation of eye gaze interaction. *CHI Letters: Human Factors in Computing Systems, CHI 2000,* 2000, ACM, **2**, 1, 281 - 288.

[Ware ja Mikaelian 1987] Colin Ware ja Harutune H. Mikaelian, An Evaluation of an Eye Tracker as a Device for Computer Input, *Proceedings of the CHI +GI 1987*, ACM, 1987, 183-188.

[Zhai ja Smith 2001] Shumin Zhai ja Barton A. Smigh, Alphabetically Biased Virtual Keyboars Are Easier to Use - Layout Does Matter, *CHI2001 Extended Abstracts,* ACM, 2001, 321-322.

GOMS-malli

Antti Aaltonen

GOMS on lyhenne sanoista GOALS, OPERATORS, METHODS, and SELECTION RULES. GOMS-mallia on käytetty vuodesta 1983 lähtien, jolloin Card, Moran ja Newell esittelivät sen, ihmisen ja tietokoneen välisen vuorovaikutuksen mallintamiseen sekä tehtävien analysointiin. Yleinen GOMS-malli muodostuu menetelmistä (methods), joiden avulla saavutetaan tietty tavoite (goal). Menetelmät muodostuvat operaattoreista (operators), jotka ovat yksinkertaisia toimintosarjoja joita käyttäjä tai tietokone suorittaa. Valintasääntöä (selection rule) tarvitaan käytettävän menetelmän valintaan jos jokin tavoite voidaan saavuttaa useamman menetelmän avulla. Yksinkertaisin GOMS-malli on ns. Keystroke-Level Model (KLM), jossa ei käytetä menetelmiä tai valintasääntöjä. Kaikki käyttäjän toimet kuvataan vain tiettyjen perusoperaatioden avulla ja sopivan operaation valitseminen jää analyysin tekijän hartijoille. Kun jokaiselle perusoperaatiolle on määritetty suoritusaika niin analyysin tuloksena saadaan arvio siitä kuinka kauan tehtävän suorittamiseen kuluu aikaa. GOMS:n etuna on että, sillä voidaan mallintaa tietyn tehtävän suorittamista ja siihen kuluvaa aikaa. Toisaalta sen heikkoksia on muunmuassa se, ettei se huomioi käyttäjien välisiä eroja, oppimisen tai virheiden vaikutusta. Lisäksi GOMS soveltuu parhaiten sellaisten tehtävien mallintamiseen, jotka ovat yksinkertaisia sekä koostuvat peräkkäisistä suoritusvaiheista.

1. Johdanto

Suunniteltaessa käyttöliittymää on keskeistä löytää käyttäjän tehtäviin liittyvät tavoitteet, toiminnot ja rajoitteet. Nämä asiat pyritään löytämään tehtäväanalyysissa ja niiden avulla voidaan muodostaa malli käyttäjän toiminnasta, jonka hän tekee tehtävän suorittamiseksi. Eräs tällainen malli on Cardin, Moranin ja Newel'n [1983] esittämä GOMS (Goals, Operators, Methods and Selection rules), jonka avulla voidaan kuvata sekä arvioida ihmisen vuorovaikutusta tietokoneen kanssa.

Tässä raportissa malleista käsitellään tarkemmin kaksi varhaisinta GOMS-mallia. Ensiksi esitellään KLM (Keystroke-Level Model), joka on toiminut lähtökohtana CMN-GOMS-mallille, joka esitellään KLM:n jälkeen. GOMS-mallista on useita muitakin versioita, joita John ja Kieras [1996a] ovat esitelleet ansiokkaasti artikkelissaan.

Vaikka Cardin ja kumppanien esittämänsä niin sanottu *yleinen* (common tai CMN eli Card-Moran-Newell) GOMS onkin jo pari vuosikymmentä vanha niin se on edelleenkin käyttökelpoinen tapa esimerkiksi pienten mobiililaitteiden – kuten esimerkiksi kämmentietokoneiden ja matkapuhelimien, sovellusten käytön tutkimiseen. GOMS-mallia käytettiin 80-luvun alkupuolen tutkittaessa päätteiden valikkopohjaisia käyttöliittymiä, jotka ovat nykyään hyvin yleisiä mobiililaiteissa. 80-luvn alun päätteillä ja pienillä laitteilla on useita muitakin samankaltaisuuksia. Molemmissa ympäristöissä vuorovaikutus on pohjimmiltaan melko yksinkertaista, koska käyttöliittymä on valikkopohjainen. Syötelaitteena käytetään yleensä ainoastaan näppäimistöä ja pieniresoluutioisella näytöllä ei voida esittää kuin tekstiä tai yksinkertaista grafiikkaa. Lisäksi käyttäjä tekee yleensä vain yhtä toimintoa kerrallaan, koska pieni näyttötila, muistimäärä ja laskentateho ei mahdollista monimutkaisten sovellusten vaatiman käyttöliittymän esittämistä.

2. CMN-GOMS

CMN-GOMS:ksi nimitetään teoksessa "The Psychology of Human-Computer Interaction" [Card *et al.*, 1983] esitettyä mallia. Mallin lähtökohtana on *rationaalisuusperiaate* (rational principle), mikä on tehtäväanalyysin keskeinen toiminta-ajatus. Tämä sääntö perustuu ajatukseen, että käyttäjä toimii järkevästi saavuttaakseen tavoitteensa [Card *et al.*, 1983, s. 27]. Siksi tehtäväanalyysissä on analysoitava käyttäjän toimintaan vaikuttavat asiat, kuten tehtävän rakenne ja käyttäjän antama syöte sekä sitä rajoittavat asiat kuten käyttäjän tiedot ja prosessointikyky.

Käyttäjän kognitiivinen toiminta muodostuu neljästä osasta: *tavoitteista* (Goals), *toiminnoista* (Operators), *menetelmistä* (Methods) sekä *valintasäännöistä* (Selection rules), jotka muodostavat GOMS-mallin [Card *et al.*, 1983].

Yleisesti ottaen *tavoite* on tila, minkä käyttäjä haluaa toimillaan saavuttaa. Tavoite voidaan jakaa (ja yleensä se jaetaan) sarjaksi pienempiä *osatavoitteita* (subgoals), jotka voidaan myös tarvittaessa jakaa edelleenkin pienemmiksi tavoitteiksi.

Tavoitteen tai osatavoitteiden saavuttamiseksi määritellään joukko *menetelmiä*, jotka ovat muodostuvat sarjoista peräkkäin suoritettavia operaattoreita ja osatavoitteita.

Operaattorit ovat toimintoja, jotka liittyvät havainnointiin, ajatteluun, kognitiivisiin tai motorisiin toimintoihin tai edellä mainittujen toimintojen yhdistelmiin, jotka suoritetaan tavoitteiden saavuttamiseksi. Operaattoriin liittyvät ominaisuudet – kuten esimerkiksi suoritusaika, määritellään siten että ne eivät riipu aikaisemmin suoritetuista toiminnoista. Halutusta tarkkuudesta ja valitusta mallista riippuen operaattorit voivat olla hyvin yksinkertaisia, kuten näppäimistön näppäimen painalluksia, tai monimutkaisempia, kuten esimerkiksi valikkovalinnan tekeminen. Esimerkiksi osatavoitteeseen *paina_painiketta* voisi liittyä menetelmä, joka muodostuu operaattorisarjasta *siirrä_hiiri* painikkeen päälle, *paina_vasen_hiirinäppäin_alas* ja *päästä_vasen_hiiri_näppäin_ylös*.

Tavoitteiden saavuttamiseksi voi olla monia vaihtoehtoisia menetelmiä kuten esimerkiksi komennon suorittaminen painikkeen painamisella (kuten edellä) tai pikanäppäimellä (esim. *paina_näppäintä-CTRL* ja *paina_näppäintä-S*) ja tällöin käytetään *valintasääntöä* määrittämään mikä menetelmistä valitaan. Käytetyn menetelmien valinta riippuu monesta asiasta kuten esimerkiksi käyttäjän kokemuksesta.

GOMS-mallin kuvaamiseksi on kehitetty useita erilaisia menetelmiä kuten esimerkiksi Kieraksen [1996] kehittämä NGOMSL (Natural GOMS Language), jonka avulla voidaan esittää tehtävä ohjelmointikielen kaltaisesti.

GOMS-mallin muodostamista ja käyttöä onkin helpoin ymmärtää seuraavan esimerkin avulla, joka on mukailtu Kieras [1999, s. 10] esimerkistä. Siinä kuvataan tehtävä *perform disk maintenance* kovalevyn ylläpidossa. Tehtävään liittyy kaksi tavoitetta; tiedoston tai hakemiston poistamisen MS-DOS –käyttöjärjestelmässä.

```
method_for_goal: perform disk maintenance
1:   accomplish_goal: perform delete_file "C:\" "LOG.TXT"
2:   accomplish_goal: perform delete_directory "TEMP"
```

Lisäksi tarvitaan valintasäännöt joiden avulla valitaan tehtävän tavoitteeseen sopiva menetelmä.

```
selection_rules_for_goal: perform <task>
if type of <task> is delete_file
     then accomplish_goal: delete file using
          <filename>
if type of <task> is delete_directory
     then accomplish_goal: delete directory using
          <directory>
```

Ennen kuin varsinaiset menetelmät voidaan määritellä, luodaan kaksi *apumenetelmää* (submethod) *perform command* ja *enter_file_info* joiden avulla voidaan toteuttaa MS-DOS –komennot.

```
method_for_goal: perform command using <command>,
        <directory> and <filename>
1:   type_in <command>
2:   accomplish_goal: enter_file_info using
        <directory> and <filename>
3:   verify "command is entered correctly"
4:   keystroke "\n"
5:   return_with_goal_accomplished

method_for_goal: enter_file_info using <directory> and <filename>
1:   keystroke " "
2:   decide: if <directory> is NULL
        then goto 5
3:   type_in <directory>
4:   decide: if <directory> is_not NULL
        then keystoroke "\"
5:   decide: if <filename> is NULL
        then goto 7
6:   type_in <filename>
7:   return_with_goal_accomplished
```

Yksinkertaisin menetelmä on tiedoston poistaminen (*delete file)*, jossa käytetään vain yhtä apumenetelmää.

```
method_for_goal: delete file
```

```
1:   accomplish_goal: perform command using DEL
        directory_name of <task>, filename of <task>
2:   return_with_goal_accomplished
```

Sen sijaan ennen kuin hakemisto voidaan poistaa, täytyy kaikki sen sisältämät tiedostot hävittää. Tätä varten tarvitaan apumenetelmä *delete all_files_in_directory*, joka pohjautuu aikaisemmin määriteltyyn *delete file* –menetelmään. Vasta tämän jälkeen voidaan käyttää menetelmää *remove directory*, joka varsinaisesti suorittaa tehtävään liittyvän komennon.

```
method_for_goal: delete directory
1:   accomplish_goal: delete all_files_in_directory
2:   accomplish_goal: remove directory
3:   return_with_goal_accomplished

method_for_goal: delete all_files_in_directory
1:   accomplish_goal: perform command using DEL
        directory_name of <task>, "*.*"
2:   return_with_goal_accomplished

method_for_goal: remove directory
1:   accomplish_goal: perform command using RMDIR
        directory_name of <task>, NULL
2:   return_with_goal_accomplished
```

Esimerkki paljastaa, että jo yksinkertaisen käyttöjärjestelmäkomennon kuvaaminen vie paljon tilaa ja aikaa jos sen joutuu tekemään käsin, joten käytännössä menetelmän hyödyntäminen vaatii automatisointia menetelmien luomiseksi. Toisaalta, jos samaa esimerkkiä käyttää optimaalisena suorituksena tehtävälle, niin sen avulla voi vertailla eri käyttäjien suorituksia ja sitä missä vaiheessa syntyy eroja.

Analysoinnin voi tehdä usealla eri tasolla riippuen siitä kuinka pitkälle yksityiskohtien tutkimisessa halutaan tai täytyy mennä. Card ja kumppanit [1983, s.162] esittivät, että analyysin taso voi olla jokin seuraavista.

- *Tehtävätasolla* (unit-task level) on vain yksi operaattori – itse suoritettava tehtävä. Aiemmin esitetyssä esimerkissä tällainen on *perform disk maintenance.*
- *Toiminnallinen taso* (functional level), jossa operaattoreita ovat tehtävässä suoritettavat toiminnot. Esimerkissä tällaisia ovat

`delete_file` tai `delete_directory`. Huomaa operaatioiden nimissä alaviivat.

- *Argumenttitasolla* (argument level) määritellään komentoihin liittyvät argumentit. Tällaisia ovat `delete file` tai `delete directory` (operaatioiden nimissä *on* alaviivat)
- *Näppäinpainallustasolla* (keystroke level) analyysi on KLM-mallin mukaisia.

3. KLM

Keystroke Level Model eli KLM [Card *et al.*, 1980] käytettiin alunperin merkki- ja valikkopohjaisten käyttöliittymien suoritusaikojen mittaamiseen sekä arviointiin. Sen avulla voidaan muodostaa tehtävän suoritukselle optimiaika.

KLM on yksinkertaisin GOMS-malli, koska siinä ei käytetä lainkaan osatavoitteita vaan kaikki menetelmät kuvataan ainoastaan operaattoreiden avulla. Lisäksi mallissa ei määritellä valintasääntöjä vaihtoehtoisille menetelmille vaan henkilö, joka suorittaa analysoinnin, joutuu itse valitsemaan kuhunkin tilanteeseen sopivan menetelmän. KLM-mallissa oli alun perin kuusi operaatiota, jotka on esitetty taulukossa 1.

OPERAATTORI	KUVAUS
K	Näppäimen tai painikkeen painaminen
P	Kuvaruudulla olevan kohteen osoittaminen hiirellä
H	Käsien siirtäminen näppäimistölle tai muulle laitteelle
D	Suorien viivojen piirtäminen hiirellä
M	Valmistautuminen toiminnon suorittamiseen
R	sovelluksen vasteaika

Taulukko 1. KLM-operaatiojoukko ja niiden selitykset.

Kieras [1993] antaa yksityiskohtaiset ohjeet KLM-mallin käytöstä sekä esittelee nykyaikaisemman operaattorijoukon, jota on myös osittain yksinkertaistettu sen käytön helpottamiseksi. Operaattorit suoritusaikoineen ja kuvauksineen on esitetty taulukossa 2.

OPERAATTORI	KUVAUS	AIKA [S]
K	Näppäimen tai painikkeen painaminen. Suoritusaika vaihtelee käyttäjän kirjoitustaitojen mukaan ja tyypilliset arvot ovat:	
	- Kokenut konekirjoittaja (90 sanaa minuutissa)	0.12
	- Keskimääräinen konekirjoittaja (55 sanaa minuutissa)	0.20
	- Kokematon konekirjoittaja (40 sanaa minuutissa)	0.28
	- Henkilö, joka ei tunne näppäimistöä	1.20
T(n)	Merkkien (*n* kpl) kirjoittaminen näppäimistöllä.	$n \times$ K
P	Kuvaruudulla olevan kohteen osoittaminen hiirellä.	1.10
B	Hiiripainikkeen painaminen alas tai vapauttaminen.	0.10
BB	Hiiripainikkeen napsautus (painaminen alas ja vapautus).	0.20
H	Käsien siirtäminen näppäimistölle tai hiirelle.	0.40
M	Ajattelu, havainnointi, valmistautuminen toiminnon suorittamiseen tai suoritetun toiminnon varmistaminen.	1.20
W(t)	Järjestelmän suorittaminen operaatioiden odottaminen.	*t*

Taulukko 2. Kierasin [1993] esittämä laajennettu KLM-operaatiojoukko ja niiden suoritusajat.

Taulukon operaattorijoukko sisältää muutamia yleistyksiä ja suosituksia. Kieras suosittelee käyttämään *K*-operaattorille arvoa 0.28 sekuntia, koska se kuvaa hänen mielestään tyypillistä tietokoneen käyttäjää, joka ei ole ammattimainen konekirjoittaja. Kohteen osoittamiseen hiirellä (*P*) vaadittu tarkka suoritusaika voitaisiin laskea Fitts'n lain mukaan, mutta tähän on valittu käytännöllinen keskiarvo 1.1 sekuntia, koska tyypillisesti arvo vaihtelee 0.8 ja 1.5 sekunnin välillä. Operaattorin M on havaittu vaihtelevan käytännössä 0.6 ja 1.35 sekunnin välillä tehtävästä, käyttäjästä tai käyttötilanteesta riippuen. Kuitenkin Kieras suosittelee viitaten Olsonin ja Olsonin 1990 tekemään tutkimukseen, että kaikille *M*-operaattoreille käytettäisiin arvoa 1.2 sekuntia.

Mallin soveltamisessa vaikein tehtävä on päättää *M*-operaattorien määrä sekä niiden sijoittaminen. Kieras [1993] ehdottaa, että operaattori tulisi sijoittaa seuraavien toimintojen yhteyteen:

- tehtävä suorituksen aloittaminen,
- valinnan tekeminen vaihtoehtoisista toimintatavoista,

- tietoyksikön (esimerkiksi tiedoston tai komennon nimen) hakeminen muistista,
- tiedon hakeminen kuvaruudulta,
- tehtävään liittyvän parametrin hakeminen (muistista tai muulla tavalla) ja
- suoritetun toiminnon oikeellisuuden varmistaminen.

Tarvittaessa näitäkin ohjeita pitää soveltaa, koska esimerkiksi aloitteleva käyttäjä saattaa miettiä tehtävän jokaista vaihetta vaikka vaiheet olisivat hyvinkin yksinkertaisia.

Taulukossa 3 on esimerkki siitä, kuinka KLM-mallia voidaan käyttää kokeneen ja aloittelevan käyttäjän suoritusaikojen vertailuun. Esimerkin tehtävänä on tuhota tiedosto raahaamalla se roskakoriin MacOS –käyttöjärjestelmässä. [Kieras, 1993, s. 9].

ALOITTELEVA KÄYTTÄJÄ	**OPERAATTORI**	**KOKENUT KÄYTTÄJÄ**	**OPERAATTORI**
valmistaudu tehtävään	*M*	valmistaudu tehtävään	*M*
etsi tiedoston ikoni	*M*	etsi tiedoston ikoni	*M*
osoita tiedoston ikonia hiirellä	*P*	osoita tiedoston ikonia hiirellä	*P*
paina hiiripainike alas	*B*	paina hiiripainike alas	*B*
varmista, että ikoni on korostettu	*M*		
etsi roskakorin ikoni	*M*		
raahaa ikoni roskakoriin	*P*	raahaa ikoni roskakoriin	*P*
varmista, että roskakori korostettu	*M*		
vapauta hiiripainike	*B*	vapauta hiiripainike	*B*
varmista, että roskakori pullistui	*M*		
etsi ikkuna, josta poistettiin	*M*		
osoita ikkunaa, josta poistettiin	*P*	osoita ikkunaa, josta poistettiin	*P*
YHTEENSÄ	**3*P* + 2*B* + 7*M***		**3*P* + 2*B* + 2*M***
AIKA	**12,6 s**		**5,9 s**

Taulukko 3. KLM-mallit siitä kun kokenut ja aloitteleva käyttäjä tuhoavat tiedoston MacOS– käyttöjärjestelmässä [Kieras, 1993, s. 9].

Yllä olevasta esimerkistä on keskeistä huomata, että aloittelevan ja kokeneen käyttäjän välinen suuri ero suoritusajoissa johtuu vain *M*-

operaattoreiden määrästä aloittelevalla käyttäjällä. Muuten malleissa on sama operaattorijoukko (ts. 3*P* + 2*B*). Tämän on osoitettu vastaavan todellisuutta erilaisissa KLM-mallin käyttökohteissa. Edelleen kuitenkin operaattoreiden sijoittelu ja suoritusaikoja laskettaessa erityisesti niiden määrä on usein vaikea määritellä mallissa oikealla tavalla.

4. Yhteenveto

Pääsääntöisesti esitetyt GOMS-mallit sopivat parhaiten kehitettävän tuotteen toiminnallisuuden kattavuuden, virhetilanteista toipumisen ja suoritusaikojen keston arviointiin sekä tutkimiseen tilanteissa, jossa käyttäjän toiminta on suuntautunut ainoastaan yhteen päämäärään, johon hän pyrkii suorittamalla perättäisiä tehtäviä. Näistä rajoituksista huolimatta on mielenkiintoista huomata, että GOMS -malleja on käytetty todellisissa suunnittelutehtävissä – eikä ainoastaan akateemisessa tutkimuksessa, jossa suunniteltavat tuotteet ovat olleet melko monimutkaisiakin [John & Kieras, 1996a].

GOMS-malleja ei voida soveltaa ohjelmistokehityksessä ennen kuin tehtäväanalyysi on tehty ja käyttäjän tehtävät ja niihin liittyvät päämäärät ovat löydetty. Lisäksi ne soveltuvat parhaiten tehtäviin, joissa toiminta on suuntautunut vain yhteen tavoitteeseen, jolloin niiden avulla ei voi mallintaa esimerkiksi luovaa toimintaa kuten piirtäminen ja säveltäminen.

Kuvatut mallit eivät pysty ottamaan huomioon sovelluksen käytön oppimisen vaikutusta vaikka todellisuudessa käyttäjä tekee joitakin tehtäviä harvoin ja joitakin taas jatkuvasti. Lisäksi mallien avulla ennustettaviin suoritusaikoihin tulee virheitä, koska ne eivät pysty huomioimaan todellista käyttötilannetta ja -ympäristöä.

Mallit eivät huomioi erilaisia käyttäjäryhmiä vaan mallintavat sekä arvioivat lähinnä kokeneen käyttäjän toimintaa. Lisäksi GOMS-malleissa oletetaan, että käyttäjä ei tee virheitä. Tämän vuoksi mallit eivät pysty ennustamaan virheiden esiintymistä tai edes virheiden lukumäärää. Toisaalta GOMS-mallit soveltuvat hyvin jonkin tietyn virheen korjaamisen analysointiin, koska se on selvästi yhteen tavoitteeseen suuntautuvaa toimintaa.

Esimerkkien avulla huomaa, että vaikka CMN-GOMS on KLM-mallista laajennettu, niin yksi niiden suurista eroista on siinä, että CMN-GOMS on ohjelmointikielen kaltaisessa muodossa kun taas KLM:n vaatii käyttäjää tekemään käytettävien menetelmien valinnan [John & Kieras, 1994].

Kuitenkin käytännössä, jotta esimerkiksi CMN-GOMS-malli olisi käyttökelpoinen, sen muodostamisen pitäisi olla helppoa ja nopeaa. Baumeisterin ja muut [2000] vertailivat tutkimuksessaan kolmea erilaista GOMS-mallien luomiseen tarkoitettua työkalua yksinkertaisen tehtävän

analysoinnissa ja havaitsivat, että edes kokeneille GOMS-mallien käyttäjille nämä työkalut eivät olleet sopivia.

Paras hyöty GOMS-työkaluista saataisiinkin mikäli ne voitaisiin yhdistää ohjelmistokehityksen alkuvaiheeseen prototyyppien käytettävyyden arviointiin. Erityisesti KLM-malli on käyttökelpoisuudestaan huolimatta varsin rajoittunut, koska siinä tavoitteiden ja valintasääntöjen käyttö ja analysoijan tehtäväksi kun hän laskee suoritusaikoja [John & Kieras, 1996a].

Viiteluettelo

[Baumeister *et al., 2000*] Baumeister, L., John, B., and Byrne, M., A comparison of tools for building GOMS models, in *Proceedings of ACM CHI 2000 (*The Hague, Netherlands*)*, ACM, 2000, 502 – 509.

[Card *et al.*, 1980] Card, S.K., Moran, T.P., and Newell, A. The Keystroke-Level Model for User Performance Time with Interactive Systems, *Communications of the ACM*, 23(**7**), 1980, 396 – 410.

[Card *et al.*, 1983] Card, S., Moran, T., and Newell, A., *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1983.

[John, 1990] John, B., Extensions of GOMS Analyses to Expert Performance Requiring Perception of Dynamic Visual and Auditory Information, in *Proceedings of the ACM CHI'90*, (), 1990, 107–115.

[John & Kieras, 1994] John, B. and Kieras, D., *The GOMS family of analysis techniques: Tools for design and evaluation*. Available as ONR Technical Report http://www.cs.cmu.edu/afs/cs/user/bej/www/ONR-TRs/ONR-TR-GOMS-fam.ps

[John & Kieras, 1996a] John, B. and Kieras, D., Using GOMS for user interface design and evaluation: Which technique? *ACM Transactions on Computer-Human Interaction*, 3(**4**), 1996, 287 – 319. Also available as http://www.cs.cmu.edu/afs/cs/user/bej/www/ONR-TRs/Which-GOMS.ps

[John & Kieras, 1996b] John, B. & Kieras, D., The GOMS family of user interface analysis techniques: Comparison and contrast, *ACM Transactions on Computer-Human Interaction*, 3(**4**), 1996, 320 – 351. Available as http://www.cs.cmu.edu/afs/cs/user/bej/www/ONR-TRs/Compare-GOMS.ps

[Kieras, 1993] Kieras, D., *Using The Keystroke-Level Model to Estimate Execution Times*, 1993. Available as ftp://ftp.eecs.umich.edu/people/kieras/GOMS/KLM.pdf

[Kieras *et al.*, 1995] Kieras, D., Wood, S. D., Abotel, K., and Hornof, A., GLEAN: A Computer-Based Tool for Rapid GOMS Model Usability Evaluation of

User Interface Designs, in *Proceedings of the ACM UIST '95 (Pittsburgh, PA USA)*, 1995, 91 – 100.

[Kieras, 1996] Kieras, D., Guide to GOMS Model Usability Evaluation Using NGOMSL, in *The Handbook of Human-Computer Interaction*, M. Helander and T. Landauer (eds.), 2nd edition, North-Holland, Amsterdam, 1996. Available as ftp://ftp.eecs.umich.edu/people/kieras/GOMS/NGOMSL_Guide.pdf

[Kieras, 1999] Kieras, D., A Guide to GOMS Model Usability Evaluation using GOMSL and GLEAN3, 1999. Available at ftp://ftp.eecs.umich.edu/people/kieras/GOMS/KLM.pdf.

GOMS-malli ja sen variaatiot

Laura Lehto

Tämä seminaarityö on tehty seminaariin Käyttöliittymäteoriat ja –mallit. Seminaarityö käsittelee GOMS-mallia (Goals, Operators, Methods and Selection rules) ja sen variaatioita. GOMS-malli on tehtäväanalyysitekniikka, jota voidaan käyttää hyväksi tietokonesysteemiä suunniteltaessa. Se esiteltiin alunperin vuonna 1983 Cardin, Moranin ja Newellin teoksessa *The Psychology of Human-Computer Interaction*. Tämän jälkeen sitä on tutkittu ja laajennettu: alkuperäisestä GOMS-mallista on syntynyt uusia variaatioita. Tässä seminaarityössä käsitellään seuraavia GOMS-mallin variaatioita: Keystroke-Level Model (KLM), Card, Moran, & Newell GOMS (CMN-GOMS), Natural GOMS Language (NGOMSL) ja Cognitive-Perceptual-Motor GOMS (CPM-GOMS).

1. Johdanto

Tämä seminaarityö käsittelee GOMS-mallia (Goals, Operators, Methods and Selection rules) ja sen variaatioita. Aluksi käsitellään GOMS-mallia yleensä: sen peruskäsitteet esitellään ja niitä havainnollistetaan esimerkin avulla. Työ keskittyy kuitenkin lähinnä variaatioihin: tunnetuimpien variaatioiden esittelemiseen, niiden vahvuuksien ja heikkouksien arviointiin ja variaatioiden vertailuun. Lopuksi esitellään hieman myös GOMS-mallin rajoituksia ja siihen kohdistettua kritiikkiä. GOMS-mallin taustalla olevia malleja ja arkkitehtuureja (kuten Stage Model of Human Information Processing, Model Human Processor ja Cognitive Complexity Theory) ei tässä työssä niiden mainitsemista lukuun ottamatta käsitellä. Työssä käytetyt termien suomennokset ovat kirjoittajan omia, eivätkä kaikki siten välttämättä vastaa muualla käytettyjä suomennoksia.

Card, Moran ja Newell esittelivät GOMS-mallin teoksessaan *The Psychology of Human-Computer Interaction* vuonna 1983, jonka jälkeen se on ollut yksi ihmisen ja tietokoneen vuorovaikutuksen tärkeimmistä teoreettisista käsitteistä. Alkuperäinen GOMS-malli jätti tilaa tulkinnalle ja myöhemmälle tutkimukselle. Myöhemmät tutkijat ovatkin soveltaneet mallia monella tapaa, minkä tuloksena on syntynyt ”analysointitekniikoiden GOMS-perhe”.

Tässä seminaarityössä esitellään neljä GOMS-mallin variaatiota: Keystroke-Level Model (KLM), Card, Moran, & Newell GOMS (CMN-GOMS), Natural GOMS Language (NGOMSL) ja Cognitive-Perceptual-Motor GOMS (CPM-GOMS).

2. GOMS-malli

2.1. GOMS-malli

Kokonaisvaltainen idea GOMS-mallin ja muiden ihmisen ja tietokoneen vuorovaikutuksen kognitiivisen mallinnuksen keinojen takana on tarjota ihmisen suorituksen suunnittelumalleja [John and Kieras, 1994]. GOMS-mallissa otetaan huomioon ihmisen kognitiiviset informaationprosessointitoiminnot, kun ennustetaan esimerkiksi tietyn tehtävän suorittamiseen liittyviä asioita.

GOMS-malli on tehtäväanalyysitekniikka. Sen luojien Cardin, Moranin ja Newellin [1983] mukaan ennustaakseen käyttäjän käyttäytymistä, täytyy analysoida tehtävä määrittääkseen käyttäjän tavoitteet ja operaattorit ja

tehtävän rajoitteet. He olettivat että tietyn käyttäjän yksityiskohtaisen käytöksen taustalla on pieni joukko informaationprosessointioperaattoreita, ja käyttäjän käyttäytyminen on kuvattavissa näiden operaattoreiden sarjoina. Aika, jonka käyttäjä tarvitsee toimiakseen on näiden erillisten operaattorien aikojen summa.

GOMS-mallissa käyttäjän kognitiivinen rakenne koostuu neljästä informaationprosessointikomponentista: 1) joukosta tavoitteita (Goals), 2) joukosta operaattoreita (Operators), 3) joukosta metodeita (Methods) tavoitteiden saavuttamiseksi, ja 4) joukosta valintasääntöjä (Selection rules) sopivan metodin valitsemiseksi kilpailevien metodien joukosta [Card *et al.*, 1983].

GOMS-malli on siis analysointitekniikka, jota voidaan käyttää hyväksi tietokonesysteemiä suunniteltaessa. Yleinen GOMS-käsite on määritelty seuraavasti: *"It is useful to analyze knowledge of how to do a task in terms of the components of goals, operators, methods, and selection rules"* [John and Kieras, 1994]. GOMS on siis tehtäväanalyysin muoto, joka kuvailee tehtävään liittyvää menetelmällistä, "miten-se-tehdään" tietoa. GOMS-pohjaisen tehtäväanalyysin tulos on jonkinlainen kuvaus neljästä komponentista: tavoitteista, operaattoreista, metodeista ja valintasäännöistä [John and Kieras, 1994].

GOMS-mallin ympärille on rakentunut GOMS-perhe: joukko tehtäväanalyysi- ja mallinnustekniikoita. On tärkeää huomata, että yleinen GOMS-malli asettaa vain edellä esitetyn määritelmän: se ei määrittele mitään tiettyä tekniikkaa, jolla tämänkaltainen analyysi tulisi tehdä [John and Kieras, 1994].

Kaikki GOMS-tekniikat tuottavat kvantitatiivisia ja kvalitatiivisia ennusteita siitä, miten ihmiset käyttävät kyseessä olevaa järjestelmää, vaikkakin eri versioilla on erilaisia painotuksia. Kaikkien tekniikoiden voidaan katsoa käsittävän systeemin toiminnallisuuden ja kaikki tarjoavat arvioita tehtävän suorittamisajasta. [John and Kieras, 1996a]

GOMS-mallia käytäntöön sovellettaessa Cardin, Moranin ja Newellin [1983] käyttämä tehtävä oli tekstin editointi tekstieditorilla. Tätä tehtävää on käytetty paljon GOMS-mallia tutkittaessa ja kehitettäessä, mutta malli ei silti rajoitu vain tekstin editoimiseen, vaan muitakin tehtäviä on tutkittu [John and Kieras, 1996b]. Tekstin editointi –esimerkin avulla voidaan helposti havainnollistaa mallin neljää informaationprosessointikomponenttia (luku 2.6.). Ensin kuitenkin määritellään tarkemmin nuo neljä komponenttia seuraavissa alaluvuissa.

2.2. Tavoitteet (Goals)

Tavoitteet tarkoittavat GOMS-mallissa samaa kuin luonnollisessa kielessäkin: ne ovat sitä, mitä käyttäjä haluaa saavuttaa. Tavoite on symbolinen rakenne, joka määrittelee asiaintilan joka halutaan saavuttaa, ja joka määrittää joukon mahdollisia metodeja joilla tavoite voidaan saavuttaa [Card *et al.*, 1983]. Tavoitteet jaetaan usein alitavoitteiksi, ja kaikki alitavoitteet tulee saavuttaa, jotta voisi saavuttaa varsinaisen tavoitteen. Tavoitteet ja alitavoitteet ovat usein järjestetty hierarkkisesti, mutta tiukkaa hierarkkista rakennetta ei vaadita [John and Kieras, 1996b]. Tavoitteen tehtävä on myös tarjota ”muistipaikka”, johon systeemi voi palata vikatilanteissa ja mistä voi tarkistaa mitä halutaan, mitä metodeja on käytössä ja mitä on jo kokeiltu [Card *et al.*, 1983].

2.3. Operaattorit (Operators)

Operaattori on toiminto, joka suoritetaan tavoitteen saavuttamiseksi. Operaatiot voivat olla havainnollisia, kognitiivisia tai motorisia toimintoja, tai näiden yhdistelmä [John and Kieras, 1996b]. Operaattorin määrittää tietty seuraus ja tietty kesto. Operaattori voi saada syötteitä, ja sen seuraukset ja kesto voivat olla sen syötteiden funktio [Card *et al.*, 1983]. Käyttäjän käyttäytyminen on kaiken kaikkiaan rekisteröitävissä näiden operaatioiden sarjana. Alkuperäinen GOMS-malli ei käsittele samanaikaisten operaatioiden hienoja rakenteita, vaan käyttäytymisen oletetaan koostuvan operaattorien sarjamaisesta suorittamisesta [Card *et al.*, 1983].

2.4. Metodit (Methods)

Metodi kuvaa menetelmän tavoitteen saavuttamiseksi. Se on yksi tavoista, joilla käyttäjä varastoi tietoaan tehtävästä [Card *et al.*, 1983]. Metodi on sarja tavoitteita ja operaattoreita, ja se voi sisältää ehtoja. Jos tavoitteilla on hierarkkinen järjestys, myös metodeilla on vastaavanlainen hierarkia [John and Kieras, 1996b].

Metodit eivät ole tehtävän tekemisen aikana luotuja suunnitelmia, vaan ne ovat opittuja menettelytapoja, joita käyttäjällä on valmiina jo suoritusaikana [Card *et al.*, 1983].

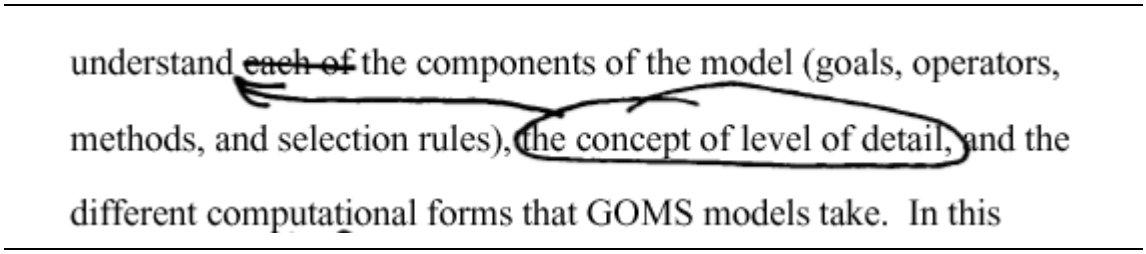
2.5. Valintasäännöt (Selection rules)

Tehtävää suoritettaessa käyttäjällä saattaa olla useampi kuin yksi metodi tarjolla tavoitteen saavuttamiseksi. Tällöin sopiva metodi valitaan valintasääntöjen avulla. Valintasäännöt edustavat käyttäjän tietämystä siitä, mitä metodia tulisi soveltaa [John and Kieras, 1996b]. Tyypillistä on, että säännöt perustuvat tehtävän tiettyihin ominaisuuksiin. Ne voivat syntyä

käyttäjän henkilökohtaisista kokemuksista tai eksplisiittisestä harjoittelusta [John and Kieras, 1996b].

2.6. Esimerkki: tekstin editointi

Alla oleva Kuva 1 kuvaa editoitavaa kohtaa tekstissä. Tehtävässä yliviivatut sanat tulee poistaa, minkä suorittamiseen tarvittavia tavoitteita, operaattoreita, metodeita ja valintasääntöjä tarkastellaan tässä esimerkinomaisesti. Lisäksi ympyröity kohta tulee valita, ja sen jälkeen leikata ja liittää nuolen osoittamaan kohtaan. Tätä tehtävää tarkastellaan myöhemmin eri variaatioiden esittelyn yhteydessä luvussa 3. Nämä esimerkit on esitelty Johnin ja Kierasin artikkeleissa [1994, 1996b].



Kuva 1: Editoitava kohta tekstissä [John and Kieras, 1996b, muokattu]

Tässä esimerkissä ylin tavoite olisi EDIT-MANUSCRIPT ja alitavoitteita olisivat MOVE-TEXT ja DELETE-PHRASE. Operaattoreita voisivat olla MOVE-MOUSE, CLICK-MOUSE-BUTTON, SHIFT-CLICK-MOUSE-BUTTON ja HIT-DELETE-KEY. Tällöin esimerkiksi yksi metodi tavoitteen DELETE-PHRASE saavuttamiseksi olisi MOVE-MOUSE poistettavan kohdan alkuun, CLICK-MOUSE-BUTTON, MOVE-MOUSE poistettavan kohdan loppuun, SHIFT-CLICK-MOUSE-BUTTON, ja lopuksi HIT-DELETE-KEY. Tekstin poistamiseksi olisi käytössä myös toinen metodi, jossa tekstiä ei valittaisi, vaan kirjaimet poistettaisiin yksi kerrallaan. Tällöin valintasäännön avulla valitaan sopiva metodi: esimerkiksi tiettyä kirjainmäärää pidemmät lauseet poistetaan ensiksi kuvatulla metodilla, lyhyemmät taas toiseksi kuvatulla. [John and Kieras, 1996b]

Tavoitteiden ja operaattorien suhde toisiinsa voi tuntua hämmentävältä. Niiden ero tulee analyytikon valitsemassa tarkkuustasossa: on valittava tietty ”pysähtymispiste”, jonka jälkeen käyttäjän toimintoja ei enää pureta pienempiin osiin. Nämä tarkimmat osat ovat siis operaattoreita, kun taas ”ylempiä” toimintoja täytyy tarkastella yksityiskohtaisemmin, joten ne esitetään tavoitteina niihin liittyvine metodeineen [John and Kieras, 1996b]. ”Pysähtymispisteen” valinta tuntuu järkevältä, sillä muuten käyttäjän toimintoja voisi tarkentaa hyvinkin pieniin osiin, esimerkiksi lihasten liikkeisiin

asti, jolloin analyysistä saattaisi tulla liian laaja. Tärkeätä kuitenkin on, että operaattorien tärkeiden ominaisuuksien, kuten suoritusaikojen, voidaan olettaa olevan vakioita riippumatta ympäröivästä kontekstista [John and Kieras, 1996b].

3. GOMS-mallin variaatiot

Tässä työssä tarkastellaan GOMS-mallin neljää tunnettua variaatiota, jotka ovat Keystroke-Level Model (KLM), Card, Moran, & Newell GOMS (CMN-GOMS), Natural GOMS Language (NGOMSL) ja Cognitive-Perceptual-Motor GOMS (CPM-GOMS).

3.1. Keystroke-Level Model (KLM)

Keystroke-Level Model eli KLM on yksinkertaisin GOMS-tekniikka. KLM tekee useita yksinkertaistavia oletuksia, jotka tekevät siitä rajoittuneen GOMS-version. Analyytikon täytyy määrittää metodi, jonka avulla tehtävä suoritetaan, mikä tyypillisesti johtaa tiettyjen tehtävien valitsemiseen. Määritelty metodi rajoittuu olemaan sarjamuodossa (sequence form) ja sen tarkimmat operaattorit (primitive operators) ovat ”näppäimenpainallus-tasolla” (keystroke-level). Mallin sarjamuotoisuus tarkoittaa sitä, että malli näyttää kiinteän operaattorisarjan yhden tavoitteen saavuttamiseksi tietyssä tehtävässä. [John and Kieras, 1994]

Alkuperäisessä KLM:ssa on kuusi operaattorityyppiä, jotka on kuvattu taulukossa 1.

Taulukko 1: KLM:n operaattorityypit [John and Kieras, 1994]

K	Näppäimen tai painikkeen painallus
P	Osoittaminen hiirellä näytöllä olevaan kohteeseen
H	Käsien palauttaminen näppäimistölle tai muulle laitteelle
D	Suorien viivojen piirtäminen
M	Valmistautuminen henkisesti toiminnon tai sarjan läheisesti toisiinsa liittyvien tarkimpien toimintojen tekemiseen
R	Järjestelmän vasteaika, jonka aikana täytyy odottaa

Operaattorit jakautuvat siis fyysisiin operaatioihin (K, P, H, D, R) ja mentaaliseen operaatioon (M). Jokaiselle operaattorille on määritelty suoritusaika [Card *et al.*, 1983, s.264]. Operaattorien H ja M suoritusajat ovat vakioita: H 0,4 ja M 1,35 sekuntia. P:n suoritusaika määräytyy Fittsin lain mukaan, keskiarvon ollessa 1,1 sekuntia. K:n suoritusaika vaihtelee 0,08-1,2 sekunnin välillä riippuen käyttäjän konekirjoitustaidoista. D:n suoritusaika lasketaan funktion avulla, ja R riippuu järjestelmästä [Card *et al.*, 1983, s.264].

Metodit voidaan luoda edellä kuvattujen operaattorien avulla, soveltaen samalla operaattorin M sijoittamiseen asetettujen heuristisen sääntöjen joukkoa. Sääntö numero 0 määrää mihin M-operaattoreita tulisi sijoittaa, ja neljä sitä seuraavaa sääntöä määräävät, milloin M voidaan poistaa säännön 0 osoittamista paikoista. [Card *et al.*, 1983, s.265]

Alla olevassa Kuvassa 2 on esitetty KLM-malli tekstin siirtämiseen paikasta toiseen operaattoreineen ja suoritusaikoineen. Ensin haluttu teksti valitaan, sitten annetaan 'leikkaa'-komento, sitten liitetään halutulle paikalle. Tekstin siirto –esimerkkiä havainnollistettiin aikaisemmin Kuvassa 1.

Moving text with the *MENU-METHOD*

Description	Operator	Duration (sec)
Mentally prepare by Heuristic Rule 0	M	1.35
Move cursor to beginning of phrase (no M by Heuristic Rule 1)	P	1.10
Click mouse button (no M by Heuristic Rule 0)	K	0.20
Move cursor to end of phrase (no M by Heuristic Rule 1)	P	1.10
Shift-click mouse button		
(one average typing K)	K	0.28
(one mouse button click K)	K	0.20
Mentally prepare by Heuristic Rule 0	M	1.35
Move cursor to Edit menu (no M by Heuristic Rule 1)	P	1.10
Press mouse button	K	0.10
Move cursor to Cut menu item (no M by Heuristic Rule 1)	P	1.10
Release mouse button	K	0.10
Mentally prepare by Heuristic Rule 0	M	1.35
Move cursor to insertion point	P	1.10
Click mouse button	K	0.20
Mentally prepare by Heuristic Rule 0	M	1.35
Move cursor to Edit menu (no M by Heuristic Rule 1)	P	1.10
Press mouse button	K	0.10
Move cursor to Paste menu item (no M by Heuristic Rule 1)	P	1.10
Release mouse button	K	0.10
TOTAL PREDICTED TIME		14.38

Kuva 2: KLM-malli tekstin siirtämiseen [John and Kieras, 1996b, muokattu]

Keystroke-Level Model –tekniikan, kuten muidenkin GOMS-mallien, rajoitteena voidaan pitää sitä, että se ennustaa vain virheetöntä "asiantuntija"-toimintaa. Lisäksi KLM-mallille pitää antaa metodi syötteenä, ja tämän metodin perusteella se ennustaa suoritusajan. Se ennustaa kuitenkin vain ajan, joka kuluu tehtävän suorittamiseen, ei aikaa, joka kuluu tehtävän hankkimiseen. Suoritusajan oletetaan olevan sama tehtävän hankkimistavasta riippumatta, ja

toiseksi hankkimisajan ja suoritusajan oletetaan olevan toisistaan riippumattomia. [Card *et al.*, 1983]

Toinen KLM-mallin rajoite on se, että kaikki ihmisen informaationprosessointitoiminnnot oletetaan sisältyviksi mallin määrittämiin operaattoreihin, myös sisäiset havainnolliset ja kognitiiviset toiminnot, joita kuvataan M-operaattorilla. Tämä rajoittaa KLM-mallin tarkastelemaan tehtäviä, joita on hyödyllistä lähestyä operaattorien sarjalla, jossa ei ole samanaikaisia toimintoja, keskeytyksiä eikä limittäisiä tavoitteita. [John and Kieras, 1996b]

3.2. Card, Moran, & Newell GOMS (CMN-GOMS)

Cardin, Moranin and Newellin [1983] esittämä GOMS-malli on tarkempi kuin yleinen GOMS-malli; siinä on tiukka tavoitehierarkia, ja metodit on esitetty pseudokoodimaisessa muodossa, joka voi sisältää alimetodeja ja ehdollisuuksia [John and Kieras, 1994]. Johnin ja Kierasin artikkeleissa [1994, 1996a ja 1996b] tästä GOMS-variaatiosta käytetään lyhennettä CMN-GOMS, ja niin tehdään myös tässä seminaarityössä. CMN-GOMS kuvaa tehtävän hierarkkisen tavoiterakenteen ja ohjelmamuodossa (program form) olevan metodijoukon avulla, joista jokainen koostuu sarjasta tiukassa sarjamaisessa järjestyksessä suoritettavia askelia [John and Kieras, 1994]. Mallin ohjelmamuoto tarkoittaa sitä, että mallin rakenne on esitetty pseudokoodimaisesti; sitä voisi verrata parametrejä sisältävään tietokoneohjelmaan [John and Kieras, 1994]. Malli siis ikään kuin seuraa tiettyä kaavaa. CMN-GOMS-mallissa tehtäviä voidaan näin ollen suorittaa tai simuloida seuraamalla mallin askelia, jotka voivat ottaa eri suuntia riippuen kyseessä olevasta tehtävästä. Malli siis ennustaa tavoitteet ja metodien valinnan tehtävätilanteesta riippuen , joten analyytikon ei tarvitse sanella niitä kuten Keystroke-Level Model –mallissa [John and Kieras, 1994].

Kuvassa 3 on esitetty CMN-GOMS-malli tekstin siirtämiseen. Kuten kuvasta voidaan havaita, esimerkki käsittelee taas tekstin liikuttamista ja tarkkuustaso on jälleen ”näppäimenpainallus-taso”. Kuvassa ylempänä on ylemmän tason tavoiterakenne, ja niiden alla alitavoitteen MOVE-TEXT kuvaus operaattoreineen ja suoritusaikoineen. Kuvan alareunassa näkyy valintasääntö tekstin valitsemiselle.

CMN-GOMS näyttää esittävän alitavoitteet huomattavasti selkeämmin kuin Keystroke-Level Model. Lisäksi ehdot ja valintasäännöt ovat selvästi näkyvillä. Mielenkiintoisia operaattoreita CMN-GOMS-mallissa ovat ihmisen mentaalitoimintoja kuvaavat VERIFY-LOCATION ja VERIFY-HIGHLIGHT. Karkeasti ottaen, CMN-GOMS-tekniikan VERIFY-operaattorit, tavoitehierarkiat ja valintasäännöt ovat esitetty operaattorilla M Keystroke-Level Model-tekniikassa [John and Kieras, 1994]. Mielenkiintoinen ero on kuitenkin siinä, että KLM laittaa M-operaattorin alitoiminnon alkuun, kun taas

CMN-GOMS laittaa ”vahvistavat” verify-operaattorinsa alitoiminnon loppuun. Samankaltaiset piirteet CMN-GOMS- ja KLM-mallien välillä ovat kuitenkin selkeitä: CMN-GOMS-tekniikan fyysiset operaattorit vastaavat KLM-tekniikan K- ja P-operaattoreita. Näin ollen operaattorit saavat molemmissa tekniikoissa samat suoritusajat, joten kokonaissuoritusajan ennuste on näissä malleissa sama.

```
GOAL: EDIT-MANUSCRIPT
.    GOAL: EDIT-UNIT-TASK ... repeat until no more unit tasks
.    .    GOAL: ACQUIRE UNIT-TASK ... if task not remembered
.    .    .    GOAL: TURN-PAGE ... if at end of manuscript page
.    .    .    GOAL: GET-FROM-MANUSCRIPT
.    .    GOAL:EXECUTE-UNIT-TASK ... if a unit task was found
.    .    .    GOAL: MODIFY-TEXT
.    .    .    .    [select: GOAL: MOVE-TEXT* ... if text is to be moved
.    .    .    .             GOAL: DELETE-PHRASE ... if a phrase is to be
deleted
.    .    .    .             GOAL: INSERT-WORD] ... if a word is to be
inserted
.    .    .    .     VERIFY-EDIT


*Expansion of MOVE-TEXT goal
GOAL: MOVE-TEXT
.     GOAL: CUT-TEXT
.    .    GOAL: HIGHLIGHT-TEXT
.    .    . [select**: GOAL: HIGHLIGHT-WORD
.    .    .              .  MOVE-CURSOR-TO-WORD
.    .    .              .  DOUBLE-CLICK-MOUSE-BUTTON
.    .    .              .  VERIFY-HIGHLIGHT
.    .    .            GOAL: HIGHLIGHT-ARBITRARY-TEXT
.    .    .              .  MOVE-CURSOR-TO-BEGINNING        1.10
.    .    .              .  CLICK-MOUSE-BUTTON              0.20
.    .    .              .  MOVE-CURSOR-TO-END              1.10
.    .    .              .  SHIFT-CLICK-MOUSE-BUTTON        0.48
.    .    .              .  VERIFY-HIGHLIGHT]               1.35
.    .    GOAL: ISSUE-CUT-COMMAND
.    .    .    MOVE-CURSOR-TO-EDIT-MENU                     1.10
.    .    .    PRESS-MOUSE-BUTTON                           0.10
.    .    .    MOVE-MOUSE-TO-CUT-ITEM                       1.10
.    .    .    VERIFY-HIGHLIGHT                             1.35
.    .    .    RELEASE-MOUSE-BUTTON                         0.10
.     GOAL: PASTE-TEXT
.    .    GOAL: POSITION-CURSOR-AT-INSERTION-POINT
.    .    .    MOVE-CURSOR-TO-INSERTION-POINT               1.10
.    .    .    CLICK-MOUSE-BUTTON                           0.20
.    .    .    VERIFY-POSITION                              1.35
.    .    GOAL: ISSUE-PASTE-COMMAND
.    .    .    MOVE-CURSOR-TO-EDIT-MENU                     1.10
.    .    .    PRESS-MOUSE-BUTTON                           0.10
.    .    .    MOVE-MOUSE-TO-PASTE-ITEM                     1.10
.    .    .    VERIFY-HIGHLIGHT                             1.35
.    .    .    RELEASE-MOUSE-BUTTON                         0.10

                                  TOTAL TIME PREDICTED (SEC) 14.38

**Selection Rule for GOAL: HIGHLIGHT-TEXT:
If the text to be highlighted is a single word, use the

   HIGHLIGHT-WORD method, else use the HIGHLIGHT-ARBITRARY-TEXT method.
```

Kuva 3: CMN-GOMS-malli tekstin siirtämiseen

[John and Kieras, 1996b, muokattu]

CMN-GOMS-mallin kvantitatiivisia ennusteita ovat operaattorien sarja ja suoritusaika. Kvalitatiivisesti malli kiinnittää huomiota metodeihin tavoitteen saavuttamiseksi; samankaltaiset metodit ovat helppoja havaita, epätavallisen lyhyet tai pitkät metodit kiinnittävät huomion ja voivat näin herättää suunnitteluideoita. [John and Kieras, 1996b]

3.3. Natural GOMS Language (NGOMSL)

Natural GOMS Language (NGOMSL) eli ”luonnollinen GOMS-kieli” jalostaa CMN-GOMS –mallia yhdistämällä sen erääseen kognitiiviseen arkkitehtuuriin, CCT:hen (Cognitive Complexity Theory, johon ei tässä työssä perehdytä tämän enempää). NGOMSL on hyvin määritelty, rakenteellinen luonnollinen kieli, joka sopii käytännön soveltamiseen. Tämä variaatio sisältää myös eksplisiittisen menetelmän GOMS-mallien rakentamiseksi [John and Kieras, 1994].

Kuten CMN-GOMS –mallikin, myös NGOMSL-tekniikan avulla tuotetut mallit ovat ohjelmamuodossa; metodirakenne on siis hyvin ”näkyvissä”. Lisäksi operaattorit ovat tässäkin tekniikassa tyypillisesti ”näppäimenpainallus-tasolla”[John and Kieras, 1994].

Alla olevassa Kuvassa 4 on esitetty esimerkki NGOMSL-metodeista tekstin siirtämiseksi. Kuten aikaisemmissakin variaatioissa, esimerkkitehtävä suoritetaan jälleen ensin valitsemalla haluttu tekstinpätkä, sitten leikkaamalla se ja lopuksi liittämällä haluttuun kohtaan.

Kuvasta 4 voidaan havaita, että vaikka CMN-GOMS –mallin tapaan myös NGOMSL on ohjelmamuodossa, se näyttää erilaiselta. NGOMSL käyttää luonnollista kieltä ja numeroi askeleet metodin suorittamiseksi. Tämän variaation tuottama malli tuntuukin helppolukuiselta – Kuvan 4 esimerkinkin rakenteen havaitsee nopeasti. Aluksi kuvan ylälaidassa kuvataan metodit ylemmille tavoitteille, joista ”hypätään” alemmas katsomaan metodit alitavoitteille.

Mielenkiintoista NGOMSL-mallin tarjoamassa ratkaisussa esimerkkitehtävään on se, että Kuvassa 4 alimpana oleva metodi näyttää olevan ”yleinen”. Se tarjoaa metodin komennon suorittamiselle: sekä leikkaus- että liittämiskomennot suoritetaan tämän metodin askelien mukaan. Tämä metodi käyttää ihmisen pitkäaikaismuistia (long-term memory, Kuvassa 4 LTM) assosioidakseen järjestelmän valikoissa olevia nimiä tehtävän komentojen nimiin [John and Kieras, 1994]. Lisäksi NGOMSL sisältää ”sisäisiä” operaattoreita kuten informaation lisäämistä työmuistiin ja poistamista sieltä tai tavoitteen toteamista saavutetuksi. Nämä NGOMSL-tekniikan piirteet liittyvät sen taustalla olevaan CCT-arkkitehtuuriin [John and Kieras, 1994].

Voisi olettaa, että ihmisen muisti on mukana muidenkin variaatioiden taustalla, mutta NGOMSL ottaa sen eksplisiittisesti huomioon.

NGOMSL Statements	Executions	External Operator Times
Method for goal: Move text	1	
Step 1. Accomplish goal: Cut text.	1	
Step 2. Accomplish goal: Paste text.	1	
Step 3. Return with goal accomplished.	1	
Method for goal: Cut text	1	
Step 1. Accomplish goal: Highlight text.	1	
Step 2. Retain that the command is CUT, and accomplish goal: Issue a command.	1	
Step 3. Return with goal accomplished.	1	
Method for goal: Paste text	1	
Step 1. Accomplish goal: Position cursor at insertion point.	1	
Step 2. Retain that the command is PASTE, and accomplish goal: Issue a command.	1	
Step 3. Return with goal accomplished.	1	
Selection rule set for goal: Highlight text	1	
If text-is word, then accomplish goal: Highlight word.		
If text-is arbitrary, then accomplish goal: Highlight arbitrary text.	1	
Return with goal accomplished.	1	
Method for goal: Highlight word		
Step 1. Determine position of middle of word.		
Step 2. Move cursor to middle of word.		
Step 3. Double-click mouse button.		
Step 4. Verify that correct text is selected		
Step 5. Return with goal accomplished.		
Method for goal: Highlight arbitrary text	1	
Step 1. Determine position of beginning of text.	1	1.20
Step 2. Move cursor to beginning of text.	1	1.10
Step 3. Click mouse button.	1	0.20
Step 4. Determine position of end of text. (already known)	1	0.00
Step 5. Move cursor to end of text.	1	1.10
Step 6. Shift-click mouse button.	1	0.48
Step 7. Verify that correct text is highlighted.	1	1.20
Step 8. Return with goal accomplished.	1	
Method for goal: Position cursor at insertion point	1	
Step 1. Determine position of insertion point.	1	1.20
Step 2. Move cursor to insertion point.	1	1.10
Step 3. Click mouse button.	1	0.20
Step 4. Verify that correct point is flashing	1	1.20
Step 5. Return with goal accomplished.	1	
Method for goal: Issue a command 1		
Step 1. Recall command name and retrieve from LTM the menu name for it, and retain the menu name.	1	
Step 2. Recall the menu name, and move cursor to it on Menu Bar.	1	1.10
Step 3. Press mouse button down.	1	0.10
Step 4. Recall command name, and move cursor to it.	1	1.10
Step 4. Recall command name, and verify that it is selected.	1	1.20
Step 5. Release mouse button.	1	0.10
Step 6. Forget menu name, forget command name, and return with goal accomplished.	1	

Predicted Pure Procedure Learning Time for 44 statements + 6 LTM chunks = 784 sec

Total Predicted Execution Time = 16.38 sec

Kuva 4: NGOMS-metodit tekstin siirtämiseen

[John and Kieras, 1996b, muokattu]

Kuten KLM- ja CMN-GOMS –mallitkin, myös NGOMSL ennustaa tehtävän suoritusajan. Mutta kuten Kuvasta 4 huomataan, tässä käytetyssä esimerkissä sen antama suoritusaika on hieman suurempi kuin kahden ensimmäisen tässä työssä kuvatun variaation. Tämä johtunee siitä, että NGOMSL-tekniikassa ilmeisesti operaattorien kokonaissuoritusaika lasketaan mallin määrittämällä erilaisella tavalla, johon ei tämän seminaarityön puitteissa perehdytä tarkemmin.

Suoritusajan ennustamisen lisäksi NGOMSL voi ennustaa myös oppimisaikaa. Oppimisajan ennustamiselle on ehtona, että vain aika menetelmän askelien oppimiseen tietyssä tilanteessa otetaan huomioon [John and Kieras, 1994]. Ennustettaessa menetelmän käyttämisen oppimisaikaa on otettava huomioon monia eri asioita, joihin ei tämän seminaarityön laajuisessa työssä voida perehtyä tarkemmin.

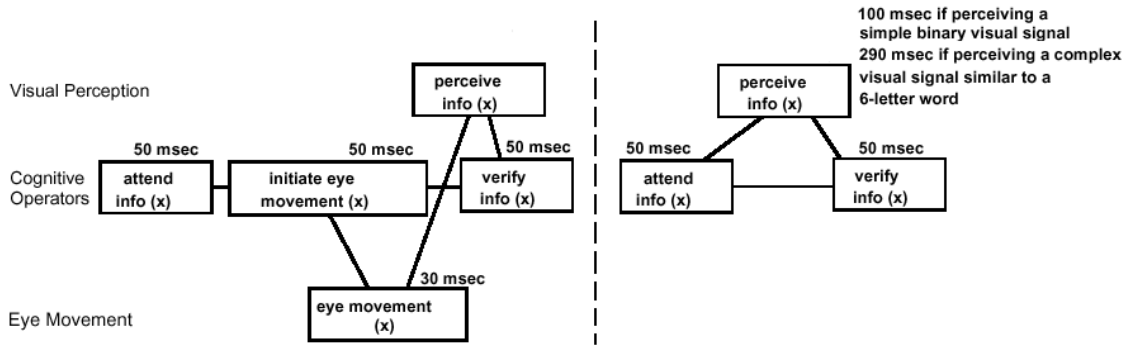
NGOMSL-tekniikalla on rajoitteensa, kuten muillakin malleilla. Yksi niistä on se, että vaikka NGOMSL-analyysi voi tarjota hyödyllisen kuvauksen tehtävästä, kvantitatiiviset ennusteet oppimis- ja suoritusajoista ovat merkityksellisiä vain jos metodit käyttävät sellaisia operaattoreita, jotka käyttäjän oletetaan tuntevan valmiiksi ja joilla on tunnetut ominaisuudet (kuten ”näppäimenpainallus-tasolla” olevat operaattorit) [John and Kieras, 1994].

3.4. Cognitive-Perceptual-Motor GOMS (CPM-GOMS)

Cognitive-Perceptual-Motor GOMS eli CPM-GOMS perustuu suoraan Cardin, Moranin ja Newellin [1983] ‘Model Human Processor’ (MHP) –arkkitehtuuriin. MHP ottaa huomioon rinnakkaisen toiminnan: siinä havainnollisen, kognitiivisen ja motorisen prosessoinnin suorittavat erilliset prosessointimekanismit [John and Kieras, 1994]. Näin ollen CPM-GOMS ei oleta, että operaattorit suoritetaan sarjamaisesti, vaan havainnolliset, kognitiiviset ja motoriset operaattorit voidaan suorittaa rinnakkaisesti tehtävän niin vaatiessa.

Rakentaakseen CPM-GOMS –mallin analyytikko aloittaa kyseessä olevasta tehtävästä tehdystä CMN-GOMS –mallista, jonka operaattorit ovat etupäässä havainnollisella ja motorisella tasolla. Nämä operaattorit ilmaistaan sitten tavoitteina ja toteutetaan MHP-tason operaattoreiden muodostamilla metodeilla.

Johnin ja Kierasin [1994] mukaan John ja Gray ovat kehittäneet kaavioita MHP-tason kognitiivisten, havainnollisten ja motoristen operaattorien yhdistelmistä, jotka toteuttavat useita eri toimintatason tavoitteita erilaisten tehtävän asettamien ehtojen vallitessa. Kuvassa 5 on esitetty tällainen kaavio visuaalisen informaation havaitsemisesta kahdessa eri tilanteessa: kun havaitsemiseksi vaaditaan silmänliike, ja kun silmänliikettä ei vaadita.



Kuva 5: CPM-GOMS –mallin operaattorikaavio. Vasemmalla on kuvattu visuaalisen informaation havaitseminen silmänliikkeen kanssa, ja oikealla ilman silmänliikettä (eli katse on valmiiksi halutussa paikassa).

[John and Kieras, 1996b, muokattu]

Operaattorikaavioissa jokaiseen operaattoriin on liitetty arvio sen kestosta, jotka Kuvassa 5 on kirjoitettu operaattoreita kuvaavien laatikoiden yläpuolelle. Arvio voi myös riippua tehtävätilanteesta: esimerkiksi pelkästään sen havaitseminen, että jotakin on tai ei ole, kestää 100ms, mutta kuusikirjaimisen sanan havaitseminen ja ymmärtäminen kestää 290ms [John and Kieras, 1994]. Kuvassa 5 tämä ehto on ilmaistu kuvan oikeassa ylälaidassa ja koskee ’perceive info’ –operaattoria molemmissa kuvatuissa tilanteissa.

CPM-GOMS –mallin luomista jatkettaessa tehtävään liittyvät kaaviot liitetään ensin toisiinsa sarjaksi, ja sen jälkeen limitetään, jotta hyödyttäisiin taustalla olevan käsitteellisen arkkitehtuurin sallimasta rinnakkaisuudesta [John and Kieras, 1994]. Kaavioon muodostetaan ns. kriittinen polku (critical path), jonka perusteella suoritusajan ennuste voidaan määritellä. Kriittinen polku muodostetaan siksi, että rinnakkaisuuden vuoksi kaikkia operaattoreita ei suoriteta tiukasti peräkkäin, vaan toimintoja tapahtuu myös samanaikaisesti. Esimerkiksi siirrettäessä kursoria hiirellä paikasta toiseen, silmänliike ja informaation havaitseminen tapahtuu rinnakkain kursorin liikuttamisen kanssa. Katse ehtii uuteen paikkaan ensin, jotta kursorin voidaan vahvistaa liikkuneen oikeaan paikkaan, mutta koska hiiren liike kestää kauemmin kuin silmän liike ja havaitseminen, se määrittää kriittisen polun [John and Kieras, 1994].

Samoin kuin Keystroke-Level Model, CPM-GOMS on sarjamuodossa, joten valintasäännöt eivät ole eksplisiittisesti esitettyjä, vaan analyytikko toteuttaa valinnan valitsemalla nimenomaisen metodin kullekin tehtävälle [John and Kieras, 1994].

Edellisten variaatioiden yhteydessä esitettyä esimerkkiä tekstin siirtämisestä ei esitetä tässä, sillä esimerkistä tulisi liian yksityiskohtainen ja laaja tähän työhön. Johnin ja Kierasin [1994] mukaan tekstin editointi ei ole hyvä sovelluskohde CPM-GOMS –mallille, koska tämä analyysitekniikka on liian yksityiskohtainen tämänlaiselle ensisijaisesti sarjamaiselle tehtävälle ja voi aliarvioida suoritusaikaa.

4. Variaatioiden vertailua

Tässä työssä kuvatuilla GOMS-mallin variaatioilla on osaksi samankaltaisia piirteitä, ja osaksi ne eroavat toisistaan. Kaikki perustuvat yleiseen GOMS-malliin, minkä näkyy selvimmin yhteisessä käsitepohjassa: tavoitteet, operaattorit, metodit ja valintasäännöt ovat kaikkien variaatioiden perustana. Variaatioiden painotuksissa on kuitenkin eroja, ja jotkut variaatiot tuntuvat olevan lähempänä yhtä variaatiota kuin jotakin toista.

Luvussa 3.1. kuvattujen rajoitustensa vuoksi Keystroke-Level Model ei ole käytännöllinen edustamaan kokonaista korkean tason tehtävää (kuten yhteistyössä kirjoittamista), vaan se sopii paremmin tarkemman tason tehtäviin (kuten esimerkkinä ollut tekstin siirtäminen). Eksplisiittisesti esitetyn tavoitehierarkiansa ansiosta CMN-GOMS taas sopii aloitettavaksi korkeammaltakin tasolta, siirtyen sitten hierarkiassa alemmille tasoille [John and Kieras, 1996b]. Suuri ero KLM-tekniikan ja CMN-GOMS -tekniikan välillä on se, että KLM on sarjamuodossa (sequence form) ja CMN-GOMS ohjelmamuodossa (program form). Tästä huolimatta samankaltaiset piirteet näiden kahden mallin välillä ovat kuitenkin selkeitä: CMN-GOMS-tekniikan fyysiset operaattorit vastaavat KLM-tekniikan K- ja P-operaattoreita, ja M-operaattori vastaa CMN-GOMS-tekniikan ihmisen mentaalitoimintoja kuvaavia operaattoreita. Operaattorit saavat molemmissa tekniikoissa samat suoritusajat, joten kokonaissuoritusajan ennuste on näissä malleissa sama.

Myös NGOMSL ja CPM-GOMS sopivat edustamaan korkeamman tason tehtävää. NGOMSL-tekniikan yhteinen piirre CMN-GOMS –mallin kanssa on ohjelmamuotoisuus. CPM-GOMS on taas KLM-mallin tavoin sarjamuodossa.

KLM, CMN-GOMS ja NGOMSL –mallit kaikki tuottavat saman sarjan havaittavissa olevia operaattoreita, kuten myös CPM-GOMS –malli, vaikkakin yksityiskohtaisemmalla tasolla [John and Kieras, 1994]. Kaikki tässä esitellyt variaatiot tuottavat ennusteita suoritusajasta. NGOMSL on näistä ainoa, joka ennustaa myös oppimisaikaa (tosin tietyin rajoituksin). NGOMSL myös käyttää enemmän ”M-tyyppisiä” eli mentaalisia operaattoreita kuin muut tekniikat [John and Kieras, 1994]. CPM-GOMS taas ei sisällä ”M-tyyppisiä” operaattoreita ollenkaan, mikä saattaa olla syynä sen ennustamaan lyhyempään suoritusaikaan [John and Kieras, 1994]. CPM-GOMS on kuitenkin

mielestäni niin yksityiskohtainen havainnollisine, kognitiivisine ja motorisine operaattoreineen, että mentaaliset operaatiot lienevät kuitenkin mukana mallissa.

CPM-GOMS erottuu muista tässä esitellyistä variaatioista ainakin kahdella tavalla: ensiksi, se perustuu MHP-arkkitehtuuriin ja sallii siksi rinnakkaisuutta, ja toisekseen juuri rinnakkaisuuden ansiosta, se on yksityiskohtaisempi kuin muut tässä esitellyt variaatiot.

5. GOMS-mallin rajoitteet

Edellä kunkin variaation esittelyn yhteydessä on mainittu kyseistä variaatiota koskevia rajoituksia. Tässä kappaleessa esitetään vielä muutama yleistä GOMS-mallia koskeva rajoite.

Yhtenä GOMS-mallin rajoituksen voidaan pitää sitä, että se olettaa käyttäjän valmiiksi osaavan suorittaa operaattorit. GOMS-metodit eivät esitä menetelmällistä tietoa joka liittyisi operaattoriin itseensä, vaan ne esittävät vain tiedon siitä, mitä operaattoreita sovelletaan ja missä järjestyksessä ne suoritetaan tavoitteen saavuttamiseksi [John and Kieras, 1994].

Analysoitavassa tehtävässä saattaa olla ominaisuuksia, joita GOMS-malli ei ota huomioon. Näin saattaa olla joko siksi, että niihin liittyy jotain enemmän kuin vain menetelmällistä tietoa, tai siksi, että tietämys ei ole rutiinisen kognitiivisen taidon muodossa, vaan vaatii monimutkaista järkeilyä tai ongelmanratkaisua [John and Kieras, 1994].

Ehkä suurin GOMS-mallin rajoite on kuitenkin sen oletus käyttäjän virheettömästä suorituksesta. Cardin, Moranin ja Newellin [1983] mukaan GOMS-malli tarjoaa täyden dynaamisen kuvauksen käyttäytymisestä mitattuna tavoitteiden, operaattorien ja metodien tasolla, jos käyttäjä toimii virheettömästi. Malli ei siis ole tarkoituksenmukainen, jos virheitä tapahtuu, mutta virheet ovat kuitenkin läsnä rutiinisessakin kognitiivisessa toiminnassa. Toisaalta, virheet myös huomataan ja korjataan yleensä rutiininomaisesti ja nopeasti, jolloin virheet voidaan laskea operaattoriaikojen varianssiin. Tällöin GOMS-malli säilyttää toimivuutensa, vaikkakin heikentyneen tarkkuuden kustannuksella [Card *et al.*, 1983]. GOMS-mallin perustumista ”eksperttikäyttäjien” virheettömille suorituksille kuitenkin pidetään suurena mallin rajoituksena.

6. GOMS-malliin kohdistettua kritiikkiä

Edellä mainittu GOMS-mallin perustuminen virheettömille suorituksille on aiheuttanut kritiikkiä mallia kohtaan. Greifin ja Gedigan [1987] mukaan GOMS-malli olettaa että on olemassa ideaaleja eksperttikäyttäjiä, jotka aina toimivat

rationaalisesti ja minimoivat aikaa ja virheitä, löytävät ja valitsevat joka kohdassa operaattorien sarjaa lyhyimmän ja turvallisimman tien annettuun tavoitteeseen. Järjellä ajateltuna tämä tuntuu mahdottomalta; kukapa ihminen toimisi aina virheettömästi. Toki rutinoitunut käyttäjä voi toimia pienellä virhemäärällä, varsinkin jos tehtävä on helppo eikä vaadi opettelua, mutta ihmisen toiminnassa on silti aina virheen mahdollisuus. Lisäksi GOMS-malli tuntuu osoittavan vain yhden tavan, jolla tehtävä tulisi saavuttaa; Greif ja Gediga [1987] kutsuvatkin GOMS-mallia ”yhden-parhaan-tavan-malliksi” (”One-Best-Way-Model”). He ehdottavat tällaisten mallien korvaamista monimutkaisten ja joustavien ihmisen mukautumisprosessien mukaisilla ”erilaisten-parhaiden-tapojen-malleilla” (”Different-Best-Way-Models”).

GOMS-mallia on kritisoitu myös monista muista asioista, kuten esimerkiksi sen käyttämästä analyysitasosta. Kritiikin mukaan ”näppäimenpainallustaso” ei ole peräisin teoreettisesta tai mittausmallista. Se on vapaasti määrättävä, vaikkakin ehkä käytännöllisesti hyödyllinen, mittauksen aikastandardi, ja siksi sitä ei pitäisi kutsua ”tasoksi” (level). Kritiikin mukaan ihmisen ja tietokoneen vuorovaikutuksen teoreettiset ja käytännölliset ongelmat ja tieteen kehitys eivät tule edistymään vapaasti määrättävissä olevien aikayksikköjen avulla. [Greif and Gediga, 1987]

On myös sanottu, että GOMS-mallit ovat ensisijaisesti esimerkkejä sovellus-malleista ennemmin kuin tieteellisistä malleista ja pysyäkseen yleisinä ja suhteellisen suoraan sovellettavissa olevina ne tekevät vahvoja yksinkertaistavia oletuksia ihmisen kognitiosta [Hammond *et al.*, 1987]. Vaikka Card, Moran ja Newell tiedostavat malleilla olevan rajoitteita, he väittävät että ne ovat silti tarpeeksi täsmällisiä, vankkoja ja joustavia sovellettaviksi käytännön suunnittelussa ja evaluoinnissa. Tällaisten väitteiden on kuitenkin sanottu olevan ennenaikaisia, ainakin Keystroke-Level-Modelin kohdalla [Hammond *et al.*, 1987].

7. Yhteenveto

Kaikki tässä seminaarityössä kuvatut GOMS-mallin variaatiot tuntuvat olevan melko käytännöllisiä tehtäväanalyysitekniikoita. Toki analysoitavan tehtävän ominaisuuksista riippuu, mikä variaatio sopii parhaiten sovellettavaksi. GOMS-mallin variaatioilla on osaksi samankaltaisia piirteitä, ja osaksi ne eroavat toisistaan. Ne esimerkiksi liikkuvat osittain erilaisilla tarkkuustasoilla ja käyttävät osittain erilaista kieltä. Kaikki variaatiot perustuvat kuitenkin yleiseen GOMS-malliin, minkä näkyy selvimmin yhteisissä käsitteissä: tavoitteet, operaattorit, metodit ja valintasäännöt ovat kaikkien variaatioiden perustana. Variaatioiden painotuksissa on kuitenkin eroja, ja jotkut variaatiot tuntuvat olevan lähempänä yhtä variaatiota kuin jotakin toista. Vaikka GOMS

ei välttämättä tunnu kovin yhtenäiseltä mallilta, mielestäni juuri variaatiot antavat siitä monipuolisen vaikutelman: variaatioiden ansiosta GOMS-mallia voi soveltaa erilaisten tekniikoiden keinoin tilanteen vaatimalla tavalla.

Viiteluettelo

[Card *et al.*, 1983] Stuart K. Card, Thomas P. Moran, and Allen Newell, *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, 1983.

[Greif and Gediga, 1987] Siegfried Greif and Günther Gediga, *A Critique and Empirical Investigation of the "One-Best-Way-Models" in Human-Computer Interaction*. Teoksessa *Psychological Issues of Human-Computer Interaction in the Work Place*, Michael Frese, Eberhard Ulich, and Wolfgang Dzida (editors), Elsevier Science Publishers B.V., Amsterdam, 1987, 357-377.

[Hammond *et al.*, 1987] Nick Hammond, Margaret M. Gardiner, Bruce Christie and Chris Marshall, *The Role of Cognitive Psychology in User-interface design*. Teoksessa *Applying Cognitive Psychology to User-interface design*, Margaret M. Gardiner and Bruce Chistie (editors), John Wiley & Sons Ltd., 1987, 13-52.

Seuraavat kolme artikkelia ovat saatavana mainittujen julkaisujen lisäksi Internetistä. Linkit artikkeleihin löytyvät osoitteesta www.eecs.umich.edu/~kieras/goms.html (artikkelit haettu 11.4.2001).

[John and Kieras, 1994] John, B. E. and Kieras, D. E., *The GOMS family of analysis techniques: Tools for design and evaluation*. Carnegie Mellon University School of Computer Science Technical Report No. CMU-CS-94-181. Also appears as the Human-Computer Interaction Institute Technical Report No. CMU-HCII-94-106.

[John and Kieras, 1996a] John, B. E., and Kieras, D. E., *Using GOMS for user interface design and evaluation: Which technique?* ACM Transactions on Computer-Human Interaction, 3, 287-319.

[John and Kieras, 1996b] John, B. E., and Kieras, D. E., *The GOMS family of user interface analysis techniques: Comparison and contrast*. ACM Transactions on Computer-Human Interaction, 3, 320-351.

Lukuprosessin silmänliikkeisiin perustuvat mallit

Aulikki Hyrskykari

Lukeminen on ihmisen ja tietokoneen vuorovaikutuksessa usein toistuva toiminto. Varsinkin katseenseurantalaitteistoa hyödynnettäessä lukuprosessin ymmärtäminen ja mahdollisuus seurata lukemisen etenemistä saattaisi monissa tilanteissa, esimerkiksi aktiivista käyttöliittymää toteutettaessa, olla vuorovaikutuksen kannalta hyödyllistä.

Lukeminen on monimutkainen kognitiivinen prosessi. Lukiessamme meidän on mm. tunnistettava lukemisen kohteena oleva sana, ymmärrettävä tunnistetun sanan merkitys, ohjattava samanaikaisesti tarkkaavaisuuttamme kohtaan johon lukuprosessi seuraavaksi etenee, pidettävä muistissa tallessa aiemmin luettua, tutkittava luetun merkitystä pitkäkestoisen muistin avustamana. Lukuprosessi on ollut psykologisen tutkimuksen kohteena kauan: ensimmäiset yhäkin viitatut tutkimukset on kirjoitettu jo viime vuosisadan jälkipuoliskolla. Tutkimusta on varsinkin kolmen viimeisen vuosikymmenen aikana kertynyt mittavasti. Teen seuraavassa katsauksen silmänliikkeisiin perustuviin lukumalleihin ja niiden pohjalla olevaan tutkimukseen.

1. Johdanto

Rayner jakaa lukemiseen liittyvän silmänliiketutkimuksen kolmeen aikakauteen, jotka ajoittuvat vuosisadan vaihteeseen (noin 1980-1920), vuosisadan keskivaiheille (noin 1930-1960), kolmannen 70-luvulla alkaneen vaiheen jatkuessa yhä [Rayner, 1998].

Jo ensimmäisen aikakauden aikana tehtiin suuri osa silmänliikkeisiin liittyvistä perushavainnoista, kuten liikkeen on sakkadisuus (silmät liikkuvat nykäyksittäin), todettiin että kohteesta havainnoitavaa informaatiota välittyy vain pysähdyksien (fiksaatioiden) aikana, ja että siirtymiin (sakkadeihin) liittyy viive (saccade latency). Myös havaintokentän laajuudesta tehtiin jo tällöin joitain huomioita.

Toisen aikakauden silmäliiketutkimusta osui yksiin kokeellisessa psykologiassa vallinneen behavioristisen tutkimuksen kanssa. Tällöin tutkimus keskittyi tarkastelemaan silmäliikkeitä omana, kontekstista erillisenä, prosessinaan. Kovin paljon ei tällöin kiinnitetty huomiota silmänliikkeiden ja kognitiivisten prosessien yhteyteen. 1950-luvun lopulta 1970 luvulle asti silmänliikkeiden parissa tehty tutkimus oli erittäin vähäistä. Toisen aikakauden lopussa oletettiin, että kaikki mitä silmänliikkeistä voitiin saada selville oli jo löydetty [Rayner, 98].

Kolmannen aikakauden aikana kehittyneet katseenseurantalaitteistot mahdollistavat silmänliikkeiden erittäin tarkan seurannan. Tarkimpien laitteistojen spatiaaliseksi resoluutioksi laitteistonvalmistajat ilmoittavat nykyisin jopa 0.01^{o} (alle kirjaimen leveys keskikokoisessa tekstissä näytöllä) ja näytteenottotaajuudeksi 4 ms. Käytännössä silmän biologinen rakenne kuitenkin rajaa tarkkuuden vain noin 0.5 - 1.5 asteeseen. Jo ensimmäisen aikakauden aikana käytössä olleilla laitteistoilla päästiin riittävään tarkkuuteen, mutta silmäliikkeistä saatavien massiivisten tietojoukkojen keräämisen helppous ja tietokonepohjaiset analyysimenetelmät helpottivat oleellisesti silmäliiketutkimuksen mahdollisuuksia. Yleisten lingvistisiin prosesseihin liittyvien teorioiden kehityksen myötä aktivoitui myös lukemiseen liittyvien kognitiivisten prosessien tarkastelun silmänliikkeiden avulla.

Lukemiseen liittyvä silmänliiketutkimus on ollut kolmannen aikakauden aikana erittäin vilkasta. Jonkinlaisen mielikuvan tilanteesta antaa se, että Rayner käyttää katsauksessaan ”*Eye movements in reading and information processing: 20 years of research*”, noin tuhatta (!) lähdeviittausta, joista noin puolet on 90-luvulla, kolmannes 80-luvulla tehtyihin tutkimuksiin, ja loput ovat

peräisin aiemmalta ajalta aina lahtien 1870-luvulta. Lisäksi aiheesta on kirjoitettu useampia kokoomateoksia ja oppikirjoja.

Tämä katsauksen tarkoituksena on toimia johdatuksena tämän laajan tutkimuksen pohjalta kehitettyihin lukuprosessin silmänliikkeisiin perustuviin malleihin. Tarkemmin esityksessä käsitellään nykyistä ”state-of-the-art” lukumallia, E-Z Reader..

Ennen sitä annetaan luvussa 2 lyhyesti ne yleiset perustiedot silmän liikkeistä lukemisen aikana, jotka ovat oleellisia Joidenkin tutkimustulosten ymmärtäminen helpottuu, jos lukijalla on käsitys siitä, millä menetelmillä tulokset on saatu aikaan. Luvussa 3 esitellään ensin lukututkimuksessa käytettyjä tekniikoita. Sen jälkeen lukututkimukseen tehdyssä katsauksessa on mallien pohjalla olevat tutkimustulokset pyritty kategorisoimaan muutamien pääilmiöiden mukaisesti. Ennen E-Z Reader –mallin esittelyä (luvussa 5), esitellään luvussa 4 lyhyehkösti muita lukumalleja ja niihin liittyviä ongelmia.

2. Silmänliikkeet lukuprosessissa

Seuraavassa annetaan lukijalle lyhyesti perustiedot silmän rakenteesta, silmänliikkeistä, ja silmänliikkeistä erityisesti lukemisen aikana. Tarkoituksena ei ole kuvata kovin syvällisesti ”silmän toimintaa”, vaan silmänliikkeet kuvataan vain sillä tasolla joka on tarpeen tässä kuvatun lukututkimuksen ymmärtämiseksi.

2.1 Näköhavainnon välittyminen

Silmän rakenteeseen liittyy eräs katseenseurannan kannalta olennainen seikka. Verkkokalvolla sijaitsevat aistinsolut välittävät linssin läpi heijastuneen ärsykkeen näköhermon välityksellä edelleen hermojärjestelmään. Katseen kohteen tulkinnalta on onnekasta, että nämä aistinsolut ovat jakaantuneet verkkokalvoille epätasaisesti siten, että suurin osa niistä sijaitsee suhteellisen pienellä *fovean* alueella. Tämä tarkan näön alue kattaa näkökentästä noin puolitoista astetta [Duchowski, 2000], joka vastaa suurin piirtein ojennetun käsivarren etäisyydellä peukalonpään kokoista aluetta. Fovean ympärillä olevalle *parafovean* alueelle heijastuneiden näköhavaintojen tarkkuuskin riittää suhteellisen tarkkaan näköhavaintoon, joskin havainto on huomattavasti epätarkempi kuin fovean alueella. Parafovean ulkopuolelle jäävälle *ääreisnäön* (*perifeeriselle)* alueelle heijastuvien näköhavaintojen tarkkuus laskee hyvin nopeasti etäisyyden foveasta kasvaessa.

Koko näkökenttä on korkeudeltaan 130^{o}:n ja leveydeltään 180^{o} levyinen ellipsi, josta ”käyttökelpoinen” alue on noin 30^{o} [Duchowski, 2000]. Näkökentässä tehtyihin havaintoihin käytän seuraavassa nimityksiä *foveaalinen havainto* (alle 2^{o}), *parafoveaalinen havainto* (2^{o} - 5^{o}) ja *periferaalinen havainto* ($> 5^{o}$).

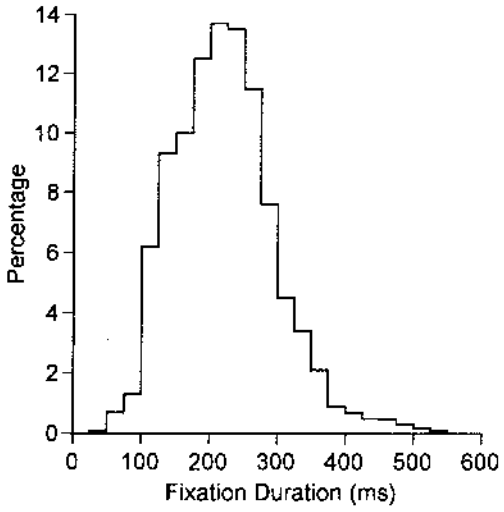
Edellinen näkökentän jakautuminen selittää syyn siihen, että silmien liike ei ole tasaisesti liukuvaa havainnointia, vaan silmät liikkuvat nykäyksittäin; jotta kiinnostuksen kohde nähtäisiin kunnolla, se noudetaan fovealle. Vaikkakin tarkkaavaisuuden kohdetta on mahdollista siirtää parafoveaaliseen kohteeseen, on havainnointi tehokkaampaa, kun katse kohdistetaan suoraan kohteeseen. Esimerkiksi parafoveaalinen lukeminen on vaikeaa, lähes mahdotonta [Rayner 1998]. Lukuprosessin kannalta mielenkiintoiset silmänliikkeet jakautuvat *fiksaatioihin* ja *sakkadeihin*. Fiksaation aikana silmä pysyy paikallaan, ja sen aikana tapahtuu näköhavainnon siirtyminen hermojärjestelmään. Sakkadit ovat siirtymiä fiksaatiosta toiseen. Sakkadin aikana informaation välittyminen on estetty, tästä syystä emme saa näköhavaintoa vilistävistä siirtymistä kohteesta toiseen.

2.2 Perustietoa silmänliikkeistä lukemisen aikana

Yleisesti vallalla oleva oletus on, että silmäliikkeitä lukiessa ohjaavat kaksi verraten itsenäistä mekanismia: toinen ohjaa fiksaatioiden kestoa ja toinen sakkadin kohteen määrittämistä. Toisin sanoen sitä *milloin* fiksaatio lopetetaan ja sitä *mihin* seuraava fiksaatio kohdistetaan kontrolloidaan omissa autonomisissa ohjausjärjestelmissään.

2.2.1 Fiksaation kesto.

Tottunut lukija tekee lukiessaan 4-5 fiksaatiota sekunnissa. *Fiksaatioiden kesto* vaihtelee lukijasta ja tekstin tasosta riippuen 100:n ja 500 millisekunnin välillä. Keskimääräinen fiksaatio normaalin tasoisessa tekstissä kestää 200 - 250 ms (kts. kuva 1), ääneen luettaessa fiksaatiot ovat hiukan pidempiä kuin normaalisti [Rayner, 1998]. Lyhimmillään fiksaatioiden kesto saattaa olla jopa vain 50 ms. Normaalissa tekstissä fiksaatio kohdistetaan lähes jokaiseen sisältösanaan (substantiivit, verbit ja adjektiivit): noin 85% sisältösanoista mutta vain noin 35% apusanoista (artikkelit, sidesanat, prepositiot ja pronominit) tulevat fiksatoiduiksi [Reichle, 98]. Pitkiin sanoihin kohdistetaan säännönmukaisesti useampikin fiksaatio. Sanan pituudella on tässä merkitystä, sillä 2-3 -kirjaimisista sanoista fiksatoidaan vain



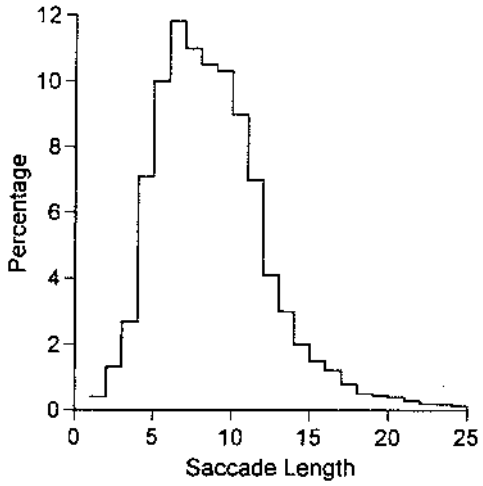
Kuva 1. Fiksaatioiden kestojen frekvenssi-jakauma lukutilanteessa [Rayner, 1998]

noin 25%, kun taas 8-kirjaimisisiin sanoihin fiksatoidaan lähes aina, niihin kohdistuu säännönmukaisesti useampiakin fiksaatioita [Rayner, 1995].

Edelliset luvut ovat keskiarvoisia arvoja lukemisessa, kuten kuvasta 1 käy ilmi vaihtelu on suurta. Vaihtelu ei johdu pelkästään yksilöllisistä eroista, vaan saman lukijankin fiksaation kestot vaihtelevat suuresti. Kohdassa 3.2 tarkastellaan eri tutkimustulosten valossa mistä vaihtelut johtuvat. Kiistatonta näyttöä on esimerkiksi siitä, että fiksaation kesto on sidoksissa prosessoinnin vaikeuteen.

2.2.2 Sakkadin pituus ja kesto

Tyypillinen *sakkadin pituus* on 7-9 kirjainmerkkiä (kuva 2) [Rayner, 1998], suomenkieltä luettaessa n. 11 kirjainmerkkiä (1.2 sanaa) [Hyönä 95], ja kestää noin 10 – 100 millisekuntia. Sakkadin pituuden mittana lukiessa käytetään nimenomaan merkkien lukumäärää, koska kirjasinkoolla tai lukuetäisyydellä ei (äärimmäisiä kirjasinkokoja lukuun ottamatta) ole paljonkaan merkitystä. Keskimäärin 10-15% sakkadeista on *regressiivisiä*, ts. kohdistuvat taaksepäin. Usein regressiot ovat muutaman kirjaimen levyisiä, jolloin ne ovat todennäköisesti korjaussakkadeja, joilla haetaan sana paremmin tarkan näön alueelle. Pidemmät, yli 10 kirjainmerkin pituiset, regressiot indikoivat ymmärtämisvaikeuksia. Onkin todettu, että lukijat pystyvät hyvinkin tarkasti kohdistamaan sakkadin kohtaan, joka ko. vaikeuden aiheutti [Frazier and Rayner 1982].



Kuva2. *Sakkadien pituuksien (kirjainmerkkeinä) frekvenssi-jakauma lukutilanteessa [Rayner, 1998]*

3. Katsaus mallien pohjana käytettyyn tutkimukseen

Kolmannen tutkimusaikakauden aikana tutkimusta on tehty kahdesta eri näkökulmasta. On joko pyritty tukimaan puhtaasti sitä, miten silmät liikkuvat lukemisen aikana, ts. lähinnä katseen siirtymiseen vaikuttavan ohjausmekanismin toimintaa. Lähempänä kognitiivisen psykologian tutkimusotetta tehdyissä tutkimuksissa taas on pyritty rekisteröityjen silmänliiketietojen avulla tekemään päätelmiä lukemisen aikaisista kognitiivisista prosesseista.

Luonteeltaan suurin osa kumpaankin luokkaan kuuluvista tutkimuksista on empiirisiä kokeita silmän käyttäytymisestä joissain hyvin rajoitetuissa

tilanteissa. Näitä tuloksia jotkut tutkijat ovat pyrkineet kokoamaan yhtenäisiksi malleiksi.

Esittelen tässä luvussa ensin lyhyesti tutkimuksessa käytettyjä menetelmiä, jotka ovat olleet kokonaisvaltaisempien lukumallien kehittelyn pohjana.

3.1. Käytetyt tutkimusmenetelmät

Jotta katseenseurannasta saatua tietovirtaa voitaisiin tulkita, on seurannasta saatavan tiedon valtavaa määrää ja yksityiskohtien runsautta pyrittävä yksinkertaistamaan kuvailevien yleisesti samalla tavalla ymmärrettyjen tunnuslukujen avulla. Alla on ensin lueteltu tällaisia yleisesti käytettyjä tunnuslukuja, ja sellaisia niiden tulkintaan vaikuttavia seikkoja, jotka tutkijoiden tulisi muistaa raportoida tuloksia esittäessään. Sen jälkeen kuvaan muutamaa, varsinkin kognitiivisten prosessien ja lukemisen välisen yhteyden tutkimisessa yleisesti käytettyä tekniikkaa.

3.1.1 Yleisesti käytettyjä tunnuslukuja

Globaaleja tunnuslukuja voidaan laskea laajemmista tekstikokonaisuuksista, kuten esimerkiksi kappaleesta tai lauseiden kokoelmasta. Esimerkiksi (1) regressioiden todennäköisyyden (2) fiksaatioiden keskimääräisen keston ja (3) etenevien sakkadien keskimääräisen pituuden on osoitettu heijastelevan luetun tekstin vaikeusastetta [Rayner and Pollatsek, 1989]. Usein halutaan saada rekisteröityä tiedot siitä, miten lukijan katse käyttäytyy luettaessa tekstiä ensimmäisen kerran (first-pass reading). Tällöin jotkut tutkijat ovat poistaneet regressioiden vaikutukset ensimmäisen kerran lukemista kuvaavista tunnusluvuista.

Jossain tilanteissa, esimerkiksi on vertailtaessa kahden tekstikappaleen lukemista, on pyritty häivyttämään kappaleiden sisältämien erilaisten sanojen vaikutus laskemalla (4) kirjainmerkkikohtaisia tunnuslukuja (ms / kirjainmerkki). Tässä tulisi kuitenkin olla varovainen, koska merkin lukemiseen kulutetun ajan suhde sanan pituuteen ei ole lineaarinen.

Tekstin rakenteeseen liittyvät seikat, kuten esimerkiksi teeman vaihdokset, tai epäloogisuudet tekstissä aiheuttavat lukemisen hidastumista, joka voidaan havaita edellä mainituilla globaaleilla tunnusluvuilla [Hyönä 1995]

Sanakohtaisten tunnuslukujen (ja sen kautta myös globaalien tunnuslukujen) laskentaan liittyy ongelmia, jotka tutkijoiden tulisi ottaa huomioon päättäessään, mikä tunnuslukujen laskentatapa kussakin tilanteessa valitaan. Voidaanko sanakohtaisiin keskiarvoihin ottaa mukaan kaikki fiksaatiot ja sakkadit vai pitäisikö joitakin niistä (muutenkin kuin havaitun mittausvirheen sisältävät) suodattaa pois?

Miten tulisi esimerkiksi kohdella samaan sanaan kohdistuneita *refiksaatioita*, tai luettaessa yli hypättyjä sanoja. Entä silmänräpäytyksiä? Pitäisikö sakkaditen kestot ottaa mukaan sanan prosessointia kuvaavien tunnuslukujen kestoon? Fiksaatioiden keskimääräiset kestot eivät kerro sitä, kuinka kauan sana viipyi tarkan näön alueella, koska kaikkiin sanoihin ei kohdistu fiksaatioita lainkaan, ja joihinkin sanoihin kohdistetaan useita fiksaatioita. Ongelma syntyykin silloin, kun pyritään kuvaamaan fiksaatioiden avulla sitä, kuinka kauan sanaa prosessoidaan (vrt. välittömyysoletus kohdassa 3.2). Tämän selvittämiseksi on käytetty tutkimuksen tarkoituksesta riippuen ainakin kolmenlaisia tunnuslukuja.

Yleisesti käytetty sanakohtainen tunnusluku (1) katseen kesto (gaze duration) [Just and Carpenter, 1980], on sanaan kohdistuneiden fiksaatioiden kestojen summa. Keskiarvoiseen katseen kestoon sanaa kohden Just ja Carpenter ottivat mukaan myös yli hypätyt sanat katseen keston ollessa niille 0 ms. Jos kuitenkin halutaan mitata aikaa, joka käytetään sanan prosessointiin, sekä yli hypätyt sanat, että regressiot sanaan aiheuttavat ongelman. Sanojen välisten regressioiden sisällyttäminen katseen kestoon johtaa harhaan koska usein halutaan tarkastella sanan lukemista ja prosessointia ensimmäisellä lukukerralla. Yli hypättyjen sanojen prosessointia tutkineet Fisher ja Schebilske havaitsivat 1985, että toisen ryhmän lukijat eivät ymmärtäneet tekstiä, josta jätettiin pois ne sanat, joihin ensimmäisen ryhmän lukijat eivät kohdistaneet fiksaatioita. [Rayner, 1998]. Näin ollen selvästikin myös yli hypätyt sanat saavat osakseen jonkinlaista prosessointia. Tämä ennakkohavainnoinnin tuoma etu (*preview benefit*, kts. tarkemmin kohta 3.2.4.) on yleisesti hyväksytty monien tutkijoiden todentama ilmiö. Yli hypätty sana prosessoidaan yleensä etukäteen, joten Rayner ja Pollatsek ehdottavat että ennakkohavainnointiin käytetty aika tulisi kohdistaa yli hypätylle, ja yleisemminkin, aina seuraavalle, parafoveaaliselle sanalle [Rayner and Pollatsek, 1989].

Silmänräpäytysten käsittely jää monelta tutkijalta raportoimatta. Katseen keston yhteydessä loogiselta tuntuu Justin ja Carpenterin tapa sisällyttää refiksaatioiden välissä tapahtuneet silmänräpäytykset katseen kestoon, mutta jättää eri sanoihin osuneiden fiksaatioiden väliin jääneiden räpäytysten kesto huomioimatta [Just and Carpenter, 1980].

Sakkadin pituus jätetään yleensä ottamatta mukaan katseen kestoon. Irwinin kuitenkin esitti 1998 olettamuksen, että vaikka näköhavaintojen välittyminen sakkadien aikana onkin estetty, sanan prosessointi etenee sakkadinkin aikana. Katseen kestoon tulisi siten lisätä myös sakkadien kestot. Sakkadit, ja varsinkin sanan sisäiset sakkadit ovat kuitenkin kestoltaan niin lyhyitä, että tästä aiheutunut tulosten mahdollinen vääristyminen ei todennäköisesti ole ollut kovin merkittävää [Rayner 1998].

Usein sanakohtaisena tunnuslukuna käytetään (2) yhdenfiksaationkatseen kestoa (single-fixation time) tai (3) aloitusfiksaation kestoa (first fixation time). Ensimmäiseen tunnuslukuun lasketaan mukaan vain ne fiksaatiot jotka ovat sanaan kohdistuneita ainoita fiksaatioita. Aloitusfiksaation kestot kuvaavat nimensä mukaisesti niitä fiksaatioita, jotka ensimmäisellä lukukerralla osuvat ensimmäiseksi sanaan. Nämä tunnusluvut kuvaavat eri asiaa kuin katseen kesto. Inhoffin tulikinnan mukaan ne heijastelevat leksikaaliseen hakuun (kts. alla kohta 3.1.2) kulutettua aikaa, kun taas katseen keston avulla saadaan esille myös leksikaalisen haun jälkeen tapahtuva sanan integrointi lauseyhteyteensä [Inhoff, 1984].

3.1.2 Tutkimuksissa käytettyjä tekniikoita

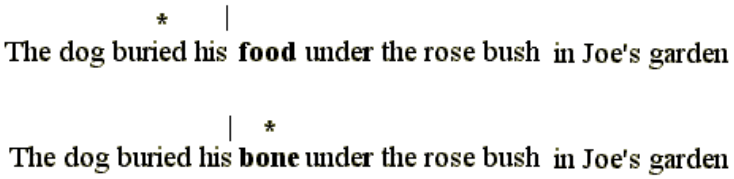
Jo varhemmassa lukututkimuksissa käytettiin yleisesti kolmea eri menetelmää. (1) Sana-sanalta luettaessa (word-by-word reading) lukija kontrolloi itse lukutahtia pyytämällä seuraavan luettavan sanan näkyviin näppäintä painamälla. (2) ROVP-menetelmässä (rapid serial visual preception) lukija saa nopeasti luettavat sanat näkyviin samassa kohdassa näyttöä. (3) Prosessi-loppu -tekniikkaa (completion responses) käytettäessä lukija antaa palautteen siitä, milloin stimulus on saatu käsiteltyä. Luetun tekstin prosessoinnissa sanan tunnistamiseen liittyvä *leksikaalisen päätöksenteon* (lexical decision) on oletettu koostuvan useammasta osasta: *leksikaalisesta hausta* (lexical access), *leksikaalisesta tunnistuksesta* (post-access checking) ja *integraatiovaiheesta* (integration). Leksikaalisessa haussa kirjainsekvenssistä muodostetaan äännettävissä oleva sana, leksikaalissa tunnituksessa sana noudetaan lukijalle tutusta ”mentaalisesta sanakirjasta” ja integraatiovaiheessa sanalle annetaan kontekstisidonnainen merkitys ja tulkinta. Pyrittäessä ymmärtämään tarkemmin näiden prosessien toimintaa, on haluttu tutkia mitkä sanaan liittyvät ominaisuudet vaikuttavat jo leksikaalisen haun, ja mitkä vasta myöhemmillä tasoilla. Tätä on tutkittu mm. käyttäen prosessi-loppu -tyyppisiä nimeämis- ja tunnistustehtäviä, joista ensimmäisessä sana lausutaan ääneen heti, kun se on mahdollista, ja jälkimmäisessä annetaan signaali heti, kun lukija päättänyt onko kyseessä oikea sana vai ainoastaan lausuttavissa oleva ”ei-sana” [Hyönä, Laine, and Niemi 1995].

Katseenseurantalaitteistojen kehitys ja niiden yhdistäminen tietokoneeseen helpottivat silmäliikkeiden tallentamista ja käsittelyä normaalimmassa lukutilanteessa. Sen lisäksi kehitys toi ulottuville mahdollisuuden muuttaa koehenkilölle annettuja ärsykkeitä reaaliaikaisena vasteena silmänliikkeille. McConkie ja Rayner käyttivät ensimmäisen kerran (4) katsevirtteiseksi näytönmuunnostekniikaksi (eye-contingent display-change) nimeämäänsä

tutkimusmenetelmää toiminnallisen näkökentän tutkimiseen vuonna 1975. Alla kolme eri variaatiota ko. tekniikasta [Rayner 1998].

- *Näkökentän ikkunointi* -tekniikka (moving window tehcnique), jossa kontrolloidaan lukijan näkökentän laajuutta poistamalla tietyllä etäisyydellä katsekohdasta oleva teksti tai se muutetaan lukukelvottomaan muotoon. Ikkunan kokoa voidaan vaihdella, ja testata missä vaiheessa se vaikuttaa lukuprosessiin.
- *Foveaalisen näkökentän peittäminen* (foveal masking) on edelliselle tekniikalle kääteinen tekniikka, jossa nimenomaan fiksaatiokohdan ympärillä oleva alue peitetään
- *Rajapyykkitekniikkassa* (boundary technique) tietty yksittäinen kohdesana vaihdetaan toiseksi sanaksi kun lukijan sakkadi ylittää ennalta määritellyn kohdan tekstissä. Fiksaatioiden kestot rekisteröidään tilanteessa, jossa tämä korvaus tehdään, ja verrataan dataan, jossa sama teksti luetaan normaalisti.

Edellä kuvattuja tekniikoita on käytetty laajasti tutkittaessa lukemisen aikaisten silmänliikkeiden ja luetun tekstin prosessoinnin välistä yhteyttä. Kuvassa 3 on esimerkki rajapyykkitekniikan käytöstä. Näkymätön ”rajapyykki” on kuvassa esitetty pystyviivana ja fiksaatiokohta *-merkillä. Kun katse ylittää rajapyykin, teksti muuttuu ensimmäisestä esimerkkirivistä toiseksi.



Kuva 3. *Esimerkki rajapyykkitekniikan toiminnasta [Reichle1998].*

Tästä manipuloidusta lukutilanteesta talletettavaa silmänliiketietoa vertaillaan normaalitilanteesta saatuun dataan. Näin tutkitaan sitä, miten parafoveaalisesta näkökentästä saatu informaatio (tai paremminkin sen puute) vaikuttaa lukuprosessiin.

3.2. Tutkimustuloksia

Seuraavassa referoidaan tärkeimpiä yksittäisiä tuloksia, jotka ovat olleet pohjana kokonaisvaltaisempaan silmäliikkeiden mallintamiseen.

3.2.1 Immediacy- ja eye-mind -oletukset

Varsinkin aiemmassa lukututkimuksessa pidettiin mahdottomana, että kognitiiviset prosessit voisivat vaikuttaa silmänliikkeisiin, koska okulomotoriset prosessit ovat niin nopeita. Kognitiivisten prosessien vaikutuksesta

lukemisen aikaisiin silmänliikkeisiin on kuitenkin viimeisten vuosikymmenten aikana saatu niin paljon näyttöä, että kiistanalainen sanojen lukuprosessiin liittyvä kysymys koskee sitä, kumpi ohjausjärjestelmä, okulomotorinen vai kognitiivinen on hallitseva silmänliikkeiden ohjauksessa. Useimpien tutkimusten pohjaksi on kuitenkin hyväksytty kaksi pitkään vallinnutta perusolettamusta, *immediacy-oletus* ja *eye-mind* -oletus [Just and Carpenter, 1980].

Immediacy-oletuksen mukaan luettaessa kohdattu sana pyritään tulkitsemaan usealla tasolla välittömästi, siitä huolimatta että tulkinnat puutteellisesta informaatiosta johtuen ovat joskus virheellisiä. Sana tunnistetaan, sille valitaan merkitys mahdollisesti useiden vaihtoehtojen joukosta ja sen merkitys pyritään ymmärtämään luetussa lauseyhteydessä. Eye-mind -oletuksen mukaan katse viipyy sanassa niin kauan kun sitä prosessoidaan.

Keskimääräiset fiksaation kesto, sakkadin pituus ja regressioiden määrä vaihtelevat eri lukijoilla huomattavasti. Myöskin yhden lukijan fiksaatioiden kestot vaihtelevan saman lukukerran kuluessa. Mitkä tekijät tähän vaihteluun vaikuttavat?

3.2.2 Fiksaation kestoon vaikuttavia tekijöitä

Sanan pituus on ilmeinen katseen kestoon vaikuttava tekijä: sanan pituuden kasvaessa katseen kestokin pitenee, esim. [Just and Carpenter 1980].

Sanan yleisyyden (frequency) on todettu korreloivan vahvasti kaikkiin sanaan kohdistuvien fiksaatioiden kestoa kuvaaviin mittareihin — keskimääräiset katseen kesto, aloitusfiksaation kesto ja yhdenfiksaationkatseen kesto pidempiä harvinaisille sanoille. Osittain vaikututusta voidaan selittää sanan pituudella; yleiset sanat ovat usein lyhyitä. Ilmiö kuitenkin toteutuu, vaikka testissä sanan pituuden vaikutus eliminoidaan [Rayner and Sereno 1994] .

Hyönä tutki sanan *morfologisen rakenteen* vaikutusta, ja totesi, että morfologisesti kompleksisten sanojen kohdalla (taivutetuissa, esim ”jonossa” ja johdetuissa sanoissa, esim. ”tutkija”) sanaan osuneen aloitusfiksaation kesto on merkitsevästi pidempi kuin perusmuotoisissa sanoissa [Hyönä et. al., 1995] .

Sanojen informaatiostatukseen vaikuttaa paitsi niiden sisäinen informaatiorakenne, myös *konteksti*, missä sana esiintyy. Esimerkiksi lauseen viimeiseen sanaan kohdistuvat katseen kestot ovat keskimäärin pidempiä kuin muille lauseen sanoille kohdistuvat. Just ja Carpenter olettivat tämän johtuvan lauseen “paketointiin” kuluvasta ajasta, jolla he tarkoittivat niitä kognitiivisia prosesseja joita tarvitaan lauseen ymmärtämiseksi kokonaisuudessaan ja suhteessa muihin lauseisiin [Just and Carpenter, 1980].

Erlich ja Rayner havaitsivat 1981, että myös *sanan ennustettavuus* (predictability) vaikuttaa fiksaatioaikoihin. Jos seuraava sana on pääteltävissä

lauseyhteydestä, siihen kohdistuvien fiksaatioiden kesto lyhenee. Ennustettavuuden vaikutusta on mitattu useiden tutkijoiden tekemissä kokeissa, joissa samaan lausekehykseen vaihdetaan sana, jonka pituus ja frekvenssi ovat vakioita, mutta ennustettavuus vaihtelee [Reichle et- al 1998]. Hyönän tulkinta ennustettavuuden merkityksestä on, että se vaikuttaa sanan tunnistukseen vain ääritilanteissa: kun sana esiintyy joko erittäin ennustettavassa tai hyvin yllättävässä kontekstissa [Hyönä et. al., 1995]. Tätä tukevat myös jotkut Raynerin myöhemmät tulokset.

Ennustettavuuden lisäksi on saatu tuloksia myös tekstin tason ja lauserakenteiden monimutkaisuuden, [Just and Carpenter, 1980; Henderson and Ferreira, 1990; Hyönä et. al. 1995] ja moniselitteisyyden, mm. [Frenck-Mestre and Pynte, 1995], vaikutuksista fiksaatioiden keskimääräiseen kestoon.

3.2.3 Sakkadin ohjautuminen

Aikaisissa tutkimuksissa 1900-luvun alussa oletettiin, että silmänliikkeiden ohjaus tapahtuu autonomisen okulomotorisen ohjauskeskuksen avulla siten, että sakkadien pituudet ovat enemmän tai vähemmän vakioita, muuttuen vain luettavan materiaalin vaikeuden funktiona. Sanojen ohitusten oletettiin johtuvan vain tekstin yleisestä helppoudesta, ei itse sanojen ominaisuuksista.

Näin oletettiin 1970–luvulle asti. Sakkadin osumisen sanan keskikohtaan on havaittu olevan tunnistuksen kannalta oleellista, joten parafoveaalisen näkökentän oletetaan pyrkivän ohjaamaan sakkadia seuraavassa sanassa optimaaliseen kohtaan. O'Reagan havaitsi vuonna 1980, että sanan sisäisten sakkadien pituus riippui parafoveaalisen sanan pituudesta. Hän ei edelleenkään olettanut sanan ohituksen riippuvan sanan itsensä "helppoudesta" ts. sen omista lingvistisistä ominaisuuksista [Brysbaert and Vitu, 1998].

Vähitellen alettiin ottaa huomioon sanan tai lauseen lingvististen ominaisuuksien mahdollinen vaikutus sanan ohittamiseen. 70-luvulla useat tutkijat tekivät oletuksia seuraavan sanan (/sanojen) ennustettavuuden ja periferaalisen näköaistimuksen yhtyeensopimisen vaikuttavan sanan yli hyppäämiseen. 70-luvun lopulla McConkie, Rayner esittivät ajatuksen, että sanan ohitus ei tapahdu satunnaisesti vaan riippuu siitä tunnistetaanko sana jo parafoveaallisen näköaistimuksen perusteella. Sanan yli hyppääminen olisi siten riippuvainen parafoveaalisen *havaintokentän* (kts. 3.2.4 alla) pituudesta. Sakkadit viritetään olettamuksen mukaan kullakin fiksaatiolla siihen tekstin osaan, joka ei ole selvästi näkyvissä nykyisestä fiksaatiokohdasta. Näin seuraavan fiksaation kohde riippuu paitsi havaintokentän tarkkuudesta, myös seuraavan sanan käsittelyn helppoudesta (ennustettavuus, yleisyys) [Brysbaert and Vitu, 1998].

Koska sakkadit ovat motorisia siirtymiä ja vaativat aikaa liikkeen suunnitteluun ja suorittamiseen, sakkadin suorittamista edeltää viive, *sakkadin ohjelmointiaika* (saccade latency, saccade programming) Tämän ohjelmoinnin on kokeissa todettu kestävän 150-175 ms [Abrams and Jonides, 1988]. Sakkadin pituus vaikuttaa jonkun verran sakkadin ohjelmointiaikaan, ja on myös todettu, että sakkadin pituus vaikuttaa sen osumiseen kohteeseensa; pitkä sakkadi on epätarkempi kuin lyhyt [Rayner, 1998]. Koska sakkadin ohjelmointiaika sisältyy fiksaation kestoon, tämä tuntuisi olevan ristiriidassa havaittujen lyhyiden fiksaatioiden (alle 150 ms, vrt. kuva 1) kanssa. Ehdotettu selitys lyhyille fiksaatioille löytyy kohdasta 4.3, Morrisonin mallin yhteydestä.

Kokeessa, jossa fiksaation kohde poistettiin näkökentästä ennen seuraavan kohteen esittämistä, sakkadin viive lyheni. Tämä antoi olettaa, että joku prosessi, jota Rayner kutsuu *tarkkaavaisuuden vapauttamiseksi* (relinguishing of attention), on osallisena sakkadin luomiseen [Rayner, 1998].

Rivien vaihdossa sakkadit jäävät usein liian lyhyiksi. Riviä vaihdettaessa tehdään yleisesti korjaava sakkadi vasemmalle. Fiksaatioilla on myös taipumus kohdistua palstoitusta kapeammalle alueelle. Rivin ensimmäinen ja viimeinen fiksaatio osuvat 7-8 kirjainmerkin päähän rivin alusta ja lopusta [Rayner,1998]. Tätä ilmiötä voitaneen ainakin osittain selittää havaintokentän koon avulla.

3.2.4 Havaintokenttä ja toiminnallinen havaintokenttä

Kuinka laajalta alueelta lukija voi omaksua informaatiota fiksaation aikana? *Havaintokentän* (perceptual span) on todettu olevan epäsymmetrinen [Rayner 1995]. Oikealle lukija tekee näköhavaintoja n 15 kirjainmerkistä, vasemmalle kuitenkin vain n. 3-4 kirjainmerkin verran; usein kuitenkin vain sanan alkuun asti. Havaintokentän epäsymmetrisyys on kulttuurisidonnaista, niin että hepreaa luettaessa havaintokenttä oikealta vasemmalle (Pollatsek, Bolozky, Well & Rayner 1981) ja kanji vertikaalisesti alaspäin painottunut. Tiiviisti pakatuissa kirjoitusjärjestelmissä (tavu- tai sanamerkkejä käyttävissä) havaintokentän koon on myös todettu olevan huomattavasti lyhyempi kuin länsimaisin aakkosin kirjoitettua tekstiä luettaessa [Rayner, 1998].

Toiminnallinen havaintokenttä (effective span), ts. alue, jonka sisällä kirjaimet voidaan tunnistaa, on pienempi, noin 7-8 kirjainmerkkiä. Tämän alueen ulkopuolelle jäävistä havaintokenttään kuuluvista kirjaimista lukija tunnistaa vain lähinnä visuaalisia piirteitä, kuten sanan pituuden. Toiminnallinen havaintokenttä ei kuitenkaan ole vakio, vaan muuntuu fiksaatio fiksaatiolta ja riippuu sekä parafoveaalisen informaation, että käsiteltävänä olevan sanan ”vaikeudesta” (sisältäen mm. sanan yleisyyden ja lauseen syntaktisen rakenteen vaikutuksen sanaan) [Henderson and Ferreira, 1995]. Lukija saattaa esimerkiksi tunnistaa yhden fiksaation aikana kolmekin lyhyttä peräkkäistä

sanaa, vaikka niissä olisi enemmän kuin 8 kirjainta. Myös lukutaito vaikuttaa toiminnallisen havaintokentän kokoon, tämä havainto on tehty kuitenkin lähinnä selvästi lukutaidoiltaan poikkeaville lukijoille (aloittelevat lukijat ja dysleksiasta kärsivät). Underwoodin ja Zolan vertaillessa kohtalaisen lukutaidon omaavia samanikäisiä lapsia, ei eroa toiminnallisen havaintokentän koossa ”hyvien” tai ”huonojen” lukijoiden välillä havaittu [Rayner, 1995].

Parafoveaalisessa näkökentässä olevan sanan todettiin jo 1978 tehdyissä testeissä nopeuttavan sanan myöhempää tunnistusta [Rayner et. al., 1978]. Myöhemmissä tutkimuksissa onkin tätä *ennakkohavainnon tuomaa etua* (preview benfit) on tarkennettu seuraavasti.

Rayner et. al. havaitsivat, että jos fiksaatiota seuraavan sanan 3 aloituskirjaimen annettiin olla muuttumattomina ja sanan loput kirjaimet korvattiin visuaalisesti samankaltaisilla (esim. a-> e tai k-h), kirjaimilla, lukuprosessi ei häiriintynyt muunnoksesta. Jos kuitenkin sanan lopussa olevat kirjaimet korvattiin visuaalisesti erilaisilla kirjaimilla lukeminen ei sujunut yhtä hyvin kuin tilanteessa jossa fiksaatiosta oikealle oleva sana pidettiin normaalisti näkyvissä [Rayner et. al 1982].

Myös Lima ja Inhoff (1984) havaitsivat, että fiksatoitua sanaa seuraavan sanan lukeminen nopeutui jos siitä pidettiin 3 ensimmäistä kirjainta näkyvissä, verrattuna tilanteeseen, jossa kirjaimet eivät olleet näkyvissä edelliseen sanaan kohdistuneiden fiksaatioiden aikana. Tämän on osoitettu todennäköisesti johtuvan joko siitä, että seuraavan sanan alkukirjaimet ovat hyödyllisiä joko fiksaation aikana aloitettavalle seuraavan sanan leksikaaliselle haulle [Inhoff, 1989] tai fiksaatioiden välisen informaation integroinnille. Parafoveaalisessa näkökentässä olevasta sanasta saadaan siis sen pituuden lisäksi myös muuta osittaista informaatiota alkukirjainten perusteella. Joissain tilanteissa parafoveaalisesta sanasta saadaan tarpeeksi informaatiota niin, että se voidaan ohittaa. Lyhyiden, 2-3 merkin mittaisten sanojen ohitus yleistä, pitkien 6-10 mittaisten harvinaista. Useissa muissakin tutkimuksissa on havaittu parafoveaalisen sanan ominaisuuksien, kuten sen pituuden tai alkukirjainten muodostaman kirjainsekvenssin yleisyyden sanojen alussa, vaikuttavan fiksatoitavan sanan prosessointiin (mm. [Kennedy, 98], [Murray, 98]). Kennedy vahvisti myös monen muunkin tutkijan saaman tuloksen, että seuraavan sanan yleisyys ei vaikuta fiksatoitavan sanan prosessointiin.

3.2.5 Sanan prosessointi

Häivyttämällä luettava sana näkyvistä sanan prosessoinnin eri vaiheissa fiksaation siirryttyä sanaan, havaittin että mikäli sana oli näkyvissä kauemmin kuin 50-70 ms, lukeminen jatkui normaalisti. Jos sana taas poistettiin näkyviltä nopeammin lukeminen häiriintyi. Tämä oletettiin johtuvan siitä, että vaikka

katse viipyikin sanassa pidempään, tarpeellinen visuaalinen informaatio sanasta saatiin jo fiksaation alussa, 50-70 ms aikana [Rayner et. al., 1981, Henderson and Ferreira, 1990].

Kennedyn tekemä havainto parafoveaalisen stimuluksen vaikutuksen käänteisyydestä oli mielenkiintoinen. Kun parafoveassa seuraavaksi luettava sana oli pitkä tai alkukirjainten muodostamalla kirjainsekvenssillä alkavia sanoja oli useita (alkukirjainten antama informaatio ei siis ollut kovin hyödyllistä), foveaalisen sanan käsittelyaika oli lyhyt. Hän tulkitsi efektin johtuvan mahdollisesti siitä, että foveaalisen ja parafoveaalisen informaation käsittelyä suoritetaan paralleelisti, eli parafovelaalisen informaation käsittelytarpeen puuttuminen lyhensi katseen viipymistä foveaalisessa sanassa [Kennedy 1998].

Monet tutkijat ovat saaneet kokeissaan todisteita ns. *ylivuotoefektistä* (spillover effect, mm. [Balota et. al., 1985, Rayner and Pollatsek, 1989). Sillä tarkoitetaan tilannetta, jossa sanan katseen kestoon vaikuttavat ominaisuudet näkyvätkin osittain seuraavan sanan fiksaatioissa. Toisin sanoen, jos foveaalissa käsiteltävä sana on ”vaikea”, sen käsittely ”vuotaakin yli” ja vaikuttaa vasta seuraavan sanan prosessointiin. Tämä tuntuisi olevan ristiriidassa immediacy-oletuksen kanssa. Näin ei kuitenkaan Hendersonin ja Ferreiran [Henderson and Ferreira, 1990] mielestä välttämättä ole, koska ylivuoto voi johtua siitä, että parafoveaalia informaatiota on vähemmän saatavilla, jolloin tunnistus kestää kauemmin. On mahdollista, että edellisen sanan käsittely ei varsinaisesti jatkukaan seuraavan sanan käsittelyn aikana, vaan seuraavan sanan foveaalinen käsittelyaika pitenee, koska ennakkohavainnon kautta saatu informaatio on normaalia vähäisempää.

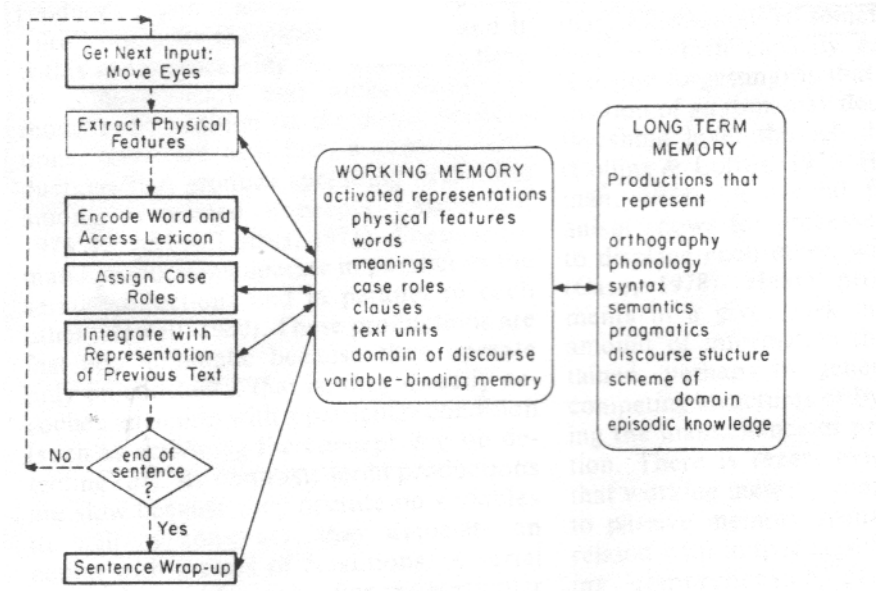
Paap esitti 1982 sanan tunnistuksen yhteydessä tapahtuvia kognitiivisia prosesseja tarkemmin kuvaavan *modernin aktivaationleviämismallin*, jossa saatu stimulus aktivoi samanaikaisesti useampia aluksi epätäydellisesti saatuun stimulukseen sopivia sanakandidaatteja. Stimuluksesta saadut näköhavainnot kuten kirjainten hahmot, ja seuraavassa vaiheessa jo tunnistetut kirjaimet, nostavat aistihavainnon jatkuessa kunkin sanakandidaatin aktivaatiotasoa ja korkeimman aktivaatiotason saanut kandidaatti valitaan [Hyönä, 1998]. Myös sanan fonologisten piirteet saattavat vaikuttaa sanan tunnistukseen; kirjaimet koodataan äänteiksi ja sana tunnistetaan yhdistämällä äänteet peräkkäin. Mm Carr ja Pollatsek esittivät 1985 että sekä visuaalisten että fonologisten piirteiden kautta tapahtuva tunnistus tapahtuu yhtäaikaisesti, ja nopeampi reitti johtaa sanan tunnistukseen. Tutun sanan kohdalla visuaalinen reitti on nopeampi, kun taas täysin tuntemattoman sanan kohdalla fonologinen reitti toimii nopeammin [Hyönä, 1998].

4. Lukuprosessin mallinnus

Edellä kuvatut tutkimukset ovat tyypillisiä esimerkkejä kolmannen aikakauden lukemiseen liittyvästä silmänliiketutkimuksesta. Tutkimuksissa on pureuduttu lähinnä johonkin tiettyyn, hyvin rajattuun, ilmiöön koko lukuprosessista. 80-luvulta lähtien on ollut muutamia pyrkimyksiä nivoa näitä tutkimustuloksia yhteen, lukuprosessia kattavammin kuvaavaksi malliksi. Tässä luvussa kuvaan muutaman tällaisen mallin pääpiirteet ja seuraavassa luvussa kattavammin tällä hetkellä parhaiten eri ilmiöitä selittävän E-Z Reader –mallin.

4.1. Justin ja Carpenterin malli (1980)

Just ja Carpenter pyrkivät ensimmäisinä ennustamaan kattavammin lukemiseen kuluvaa aikaa sitomalla regressioanalyysin avulla kehiteltyyn yhteen lausekkeeseen kaikki löydetyt sanan lukuaikaan vaikuttavat tekijät [Just and Carpenter1980]. Vastaava lauseke kehiteltiin lauseen lukemiseen käytettävän ajan ennustamiseksi.



Kuva 4. *Just and Carperter –lukumallin prosessikuvaus [Just and Carpenter, 1980]*

Mallin pohjalla oleva prosessimalli koostuu seuraavista lukemisen päävaiheesta (vrt. kuva edellä):

1) seuraavan syötteen haku (ja visuaalisten piirteiden hahmottaminen)
2) sanan koodaus ja leksikaalinen haku
3) tapauskohtaisen roolin asettaminen sanalle
4) lauseiden välinen integrointi ja
5) lauseen paketointi

Malli sisältää eksplisiittisesti oletuksen, että prosessin eri vaiheet ovat peräkkäisiä.

Malli on saanut osakseen paljon kritiikkiä, koska myöhempien tutkimusten valossa se tekee perustelemattomia yleistyksiä katseen keston ja lukemisen

aikaisten kognitiivisten prosessien suhteesta. Artikkeliin viitataan kuitenkin edelleen usein, koska siinä raportoidaan useita yksittäisiä, edelleenkin päteviä, tutkimustuloksia.

4.2. Morrisonin malli (Morrison, 1984)

Morrsonin mallin perustana on käsitys lukijan *tarkkaavaisuuden ohjautumisesta* (attentional processes) lukemisesta aikana. Hän testasi ennakkohavainnoinnin merkitystä katsevirtteisen näytönmuunostekniikan avulla, ja teki löydöksiä joita ei pystynyt selittämään aiempien peräkkäisten prosessimallien avulla. Siihen asti oli oletettu, että silmänliikkeiden ohjaus tapahtuu okulomotorisina peräkkäisinä prosesseina. Hän päätyi esittämään, että sakkadin ohjelmointi tapahtuu useammalla tasolla paralleellisti. Tämän pohjalta hän esitti mallin, jonka mukaan lukeminen etenee tarkkaavaisuuden kohdistamisen ja okulomotorisen ohjausjärjestelmän yhteispelinä seuraavassa kuvatulla tavalla.

4.2.1 Morrisonin mallin mukainen lukuprosessi

Kunkin fiksaation alussa sekä katseen kohta, että tarkkaavaisuuden kohde ovat foveaalisessa sanassa *n*. Kun sanan *n* prosessointi saavuttaa tietyn tason (ehkä silloin kun leksikaalinen haku on tehty), tarkkaavaisuus kohdistuu sanaan *n+1*, jonka prosessointi aloitetaan. Tarkkaavaisuuden siirtyminen virittää myös sakkadin ohjelmoinnin aloittamisen. Kun sakkadin ohjelmointi on valmis, se suoritetaan, ts. sakkadi sanaan n+1 tehdään vasta silloin. Koska sakkadin ohjelmointi aiheuttaa viiveen (vrt. kohta 3.2.3), informaatiota sanasta *n+1* kumuloituu sakkadin tulee suoritetukseen asti. Jos sanasta *n+1* saadaan tarvittava informaatio nopeasti sakkadin ohjelmoinnin vielä ollessa kesken, tarkkaavaisuus siirtyy jo seuraavaan sanaan (*n*+2) ja sakkadin ohjelmointi sanaan *n+1* peruutetaan. Koska myöhempi sakkadin siirto-ohjelma peruuttaa edellisen sakkadin ja saa siis aikaan sanan n+1 yli hyppäyksen. Sanaan n kohdistuva katse pitenee verrattuna tilanteeseen, jolloin sanan yli hyppäystä ei tapahdu. Jos sakkadin siirto-ohjelma on kuitenkin edennyt pitkälle, seurauksena on joko lyhyt fiksaatio sanaan *n+1* tai fiksaatio sanojen n+1 ja n+2 väliin tilanteissa joissa sakkadin peruutus onnistuu vain osittain [Morrsion 1984].

Useampien sakkadien samanaikainen ohjelmointi selittää lyhyiden fiksaatioiden esiintymisen ja epätavalliset fiksaation kohteet.

Morrisonin malli ei selitä miksi joihinkin sanoihin fiksatoidaan uudelleen: jos leksikaalinen haku laukaisee tarkkaavaisuuden kohteen vaihtamisen, ja siten tietyn ajan kuluttua sakkadin, sanaan ei pitäisi koskaan osua refiksaatioita. Henderson ja Ferreira muokkasivat mallia niin, että tämä tilanne tulee käsitellyksi.

4.2.2 Hendersonin ja Ferreiran tarkennukset

Henderson ja Ferreira lisäsivät Morrisonin malliin oletuksen, että on olemassa yläraja sille ajalle, jonka fiksaatio voi kestää, jolloin sakkadin ohjelmoinnin on alettava ajoissa ennen tämän rajan ylittymistä [Henderson Ferreira, 1990]. Tällöin *sakkadin ohjelmoinnin takaraja* (eye movement programming deadline), on se ajankohta, jolloin sakkadin ohjelmointi tulee viimeistään aloittaa. Sakkadin ohjelmoinnin takaraja on fiksaation keston yläraja miinus keskimääräinen sakkadin ohjelmointiaika. Sakkadin oletetaan aina kohdistuvan tarkkaavaisuuden kohteeseen. Normaalissa tilanteessa tarkkavaisuuden siirtyminen tapahtuu tarpeeksi 1aikaisin, jotta sanan *n* foveaalinen prosessointi jää alle sakkadin ohjelmoinnin takarajan. Tässä tilanteessa Morrisonin oletus pitää, ja sanan *n+1* parafoveaalinen prosessointi kestää koko sakkadin ohjelmoinnin ajan.

Jos kuitenkin foveaalinen sana on ”vaikea” tarkkaavaisuus voi viipyä sanassa n huomattavan kauan, jolloin tarkkaavaisuus ei siirrykään seuraavaan sanaan ennen kuin sakkadin ohjelmoinnin takaraja tulee vastaan, ja sakkadin ohjelmointi aloitetaan vaikka tarkkaavaisuus on vielä siirtymättä. Jos tarkkaavaisuus kuitenkin siirtyy ”hiukan myöhässä” sanaan *n+1*, parafoveaaliselle prosessoinnille käytetty aika lyhenee sillä ajalla, jonka huomio viipyy vielä sanassa *n* siirtymän ohjelmoinnin aloittamisen jälkeen. Tämä selittää useissa kokeissa saadun tuloksen, jonka mukaan foveaalisen prosessoinnin vaikeus vaikuttaa ennakkohavainnosta saatuun etuun (preview benefit), jolle Morrisonin malli ei aintanut selitystä.

Morrisonin malli ei pystynyt selittämään usein ”vaikeille” ja/tai pitkille sanoille yleisesti kohdistuvia refiksaatioita. Tehdyn lisäyksen jälkeen tämä selittyy sillä, että sanalle kohdistuu uusi fiksaatio, jos sakkadin ohjelmoinnin takaraja pakottaa tekemään fiksaation ennen kuin sanan prosessointi on saatu valmiiksi. Koska tarkkaavaisuuden kohteena on edelleen sana *n*, sakkadin ohjelmoinnin kohdekin on sana *n*, joten suoritettava sakkadi kohdistuu uudelleen foveaaliseen sanaan.

Sakkadin ohjelmoinnin takarajan käsitteen lisäksi Henderson ja Ferreira lisäsivät malliin sakkadin ohjelmointiin liittyvän toisenkin aikarajan: sakkadin ohjelmoinnin kesketyttämistä rajoittavan *point-of-no-return* –raja-arvon. Sen jälkeen siirtymän suoritusta ei enää voi peruuttaa. Kuvan 1 fiksaatioiden frekvenssikaavioista nähtiin, että lyhimmillään fiksaatiot voivat olla kestoltaan jopa alle 50 ms, joka tuntuu mahdottomalta, kun sakkadin ohjelmointiin kuluvan ajan on todettu olevan yli 150 ms. Sakkadien paralleelisti tapahtuva ohjelmointi kuitenkin selittää ilmiön. Kun tarkkaavaisuus siirtyy sanaan *n+1* sakkadin ohjelmoinnin takarajan jälkeen, *ja* point-of-no-returnin jälkeen (ts. sakkadin ohjelmointia on edennyt jo niin pitkälle, ettei sitä voi peruuttaa),

mutta uusikin osittain simultaanisti laskettu sakkadi on jo laskettu, uusi sakkadi suoritetaan nopeasti ensimmäisen sakkadin perään.

4.3. Strategy-Tactics -malli (O'Reagan, 1990)

Srategy-Tactics -mallin mukaan fiksaation osumakohta määrää sen (taktiikan), kuinka kauan katse sanassa viipyy, ja mihin seuraava sakkadi kohdistuu. Mallin mukaan lingvistiset prosessit ehtivät vaikuttaa silmänliikkeisiin vasta pitkän yhdenfiksaationkatseen aikana (yli 300 ms), tai refiksaatioiden tapauksessa vasta toiseen tai myöhempiin fiksaatioihin. Malliin liittyy oletus lukijan itsensä kontrolloiman lukustrategian (pikaluku tai huolellinen lukeminen) voimakkaasta vaikutuksesta lukuprosessiin. Tämä valittu lukustrategia vaikuttaa globaalisti lukemisessa toteutuneisiin fiksaation kestoihin ja sakkadien pituuksiin. Mallin mukaan ennalta määritellyt okulomotoriset srategiat siis selittävät pitkälle silmänliikkeiden vaihtelut lukuprosessin aikana [O'Reagan 1990].

Myöhemmin Vitu ja O'Reagan tarkensivat mallia ottamalla huomioon sanan lingvististen ominaisuuksien vaikutuksen sakkadin ohjelmoinnissa [Vitu ja O'Reagan, 1998]. He esittivät, että tarkkaavaisuuden siirtymiselle seuraavaan sanaan olisi okulomotorinen takaraja, jonka jälkeen sakkadin ohjelmointi aloitetaan. Aiempi, Morrisonin sekä Hendersonin ja Ferreiran (vrt. edellä) esittämä oletus sakkadin suorittamisen virittämiseksi oli, että sakkadin ohjelmointi voi alkaa, kun sanasta on saatu tietty määrä informaatiota. Vitu ja O'Reagan olettivat sakkadin ohjelmoinnin virittymisen syyn olevan päinvastaisen. He ehdottavat että kynnysarvo sakkadin ohjelmoinnin aloittamiselle riippuisikin siitä onko fiksaation aikana tiettyyn aikaan mennessä saatu sanasta omaksuttua tarpeeksi informaatiota — jos näin on, fiksaatio saa jatkua (*ei* siis viritä sakkadin ohjelmointia) ts. tällöin tehdään päätös siitä että sanan pystytään käsittelemään ainoalla fiksaatiolla. Muussa tapauksessa aloitetaan sakkadin ohjelmointiprosessi refiksaatiolle samaan sanaan (tämä tapahtuu todennäköisesti tilanteessa, jossa fiksaatio kohdistui liian kauas optimaalisesta fiksaatiokohdasta).

Lukustrategian vaikutus voidaan edellisen korjauksen jälkeen selittää lukijan itsenä asettamalla informaation omaksumisen kynnysarvolla: kun lukijan lukee pinnallista tai pikalukua, kynnysarvo asetetaan alhaiseksi, niin että refiksaatioiden todennäköisyys on pieni.

Rayner [Rayner et. al., 1998] kritisoi mallia sanomalla, että lingvististen vaikutusten rajoittaminen pitkiin ainoisiin fiksaatioihin tai refiksaatioihin ei voi pitää paikkaansa, koska on todettu että ensimmäinen useammasta sanaan kohdistetusta fiksaatiosta harvinaisille sanoille on pidempi kuin yleisille sanoille. Sanan yleisyyden vaikutus fiksaation kestoihin ei kuitenkaan tule esiin

pelkästään jakaumien yläpäissä. Mallia on kritisoitu myös siitä, että sen perustana olevat havainnot on tehty rajatussa kontekstissa tehdyissä kokeissa (sana-sanalta luku tai ROVP-tekniikka, kts. edellä kohta 3.1), joissa lukijalle on esitetty yksittäisiä sanoja tai sanasekvenssejä, eikä niinkään aidossa lukutilanteessa. Raynerin perustellulta tuntuvan kritiikin mukaan on todennäköistä, että okulomotorisen järjestelmän vaikutus on silloin suurempi kuin normaalissa lukutilanteessa, jolloin esimerkiksi parafoveaalisen prosessoinnin vaikutus tulee selkeämmin esiin. Hän toteaa myös, että jos aloitusfiksaation osumiskohta sanassa olisi niin tärkeä kuin malli esittää, fiksaatioiden osumakohdat eivät todennäköisesti vaihtelisi niin paljon kuin ne vaihtelevat.

Strategy-tactics -malli ei myöskään pysty selittämään Raynerin ja Serenon (1996) havaintoa sanan yleisyyden vaikutuksesta sakkadin osumakohtaan. Sanan yleisyydellä todettiin olevan vaikutuksia, jotka eivät riippuneet siitä mihin kohtaan sanassa ensimmäinen fiksaatio osui. [Rayner, 1998]

4.4. Muita lukumalleja

Jotta päästäisiin malleihin, jotka olisivat tarpeeksi yksityiskohtaisia jotta niiden paikkansapitävyyttä voitaisiin kokonaisvaltaisesti testata, mallien pitäisi olla matemaattista tai laskennallista mallinnusta hyväksikäyttäviä selkeitä malleja. E-Z Readerin lisäki tämän suuntaisia yrityksiä ovat olleet mm. Legge et. al. esittämä Mr. Chips-malli [Legge et. al., 1997] ja Suppesin malli [Suppes, 1994]. Legge et al. malli. Mr. Chips pyrkii antamaan yksityiskohtaisen kognitiivisen selityksen sille, mihin lukijan fiksaatio osuu. Legge olettaa, että päätöstä sakkadin kohteesta ohjaavat sekä leksikaalisen haun kontrolloimat älykkäät mekanismit, että se mitkä kirjaimet ovat prosessoitavissa mm. tarkkuuden rajoittaessa havaintoa. Suppes taas perustaa mallinsa voimakkaasti silmänliikkeistä kerätyn datan stokastisille ominaisuuksille. Hän käsittelee lukuprosessia melko itsenäisenä, kognitiivisista prosesseista erillisenä, prosessinaan. Tutkimuksessa keskitytään käsittelemään silmänliikkeitä matalla tasolla painottaen sakkadien satunnaista vaihtelua. Nämä mallit pyrkivät selittämään rajoitettua osaa silmäliikkeistä kerätystä datasta.

Seuraavassa luvussa tarkemmalla tasolla esiteltävä E-Z Reader malli eroaa näistä kahdesta pyrkien selittämään sekä yksittäisten fiksaatioiden kestoja tarkalla tasolla, että kohtaa johon fiksaatio (sanatasolla) kohdistuu. E-Z Reader malli on myös muista malleista eroten esitetty eksaktina operationalisoitavissa olevana tila-automaattina, joten se mahdollistaa myös silmäliikkeiden tarkan simuloinnin.

5. E-Z Reader

E-Z Reader [Reichle, et. al. 1998] voidaan nähdä olemassa olevien tutkimustulosten ja teoreemien yhtyeensulautusyrityksenä: kehitetty malli selittää silmänliikkeet lukemisessa yksityiskohtaisella tasolla, ja se ottaa huomioon kognitiivisen ohjauksen. EZ Reader perustuu vahvasti edellä kuvattuun Morrisonin tarkkaavaisuuden ohjautumiseen perustuvaan malliin, johon on otettu mukaan myös Hendersonin ja Ferreiran ehdottamat korjaukset. E-Z Reader yrittää kuitenkin päästä askeleen pidemmälle ja selittää sekä fiksaatioiden kestoja että fiksaatioiden kohdistusta (sanan tasolla) yksityiskohtaisen operationaalisen mallin avulla.

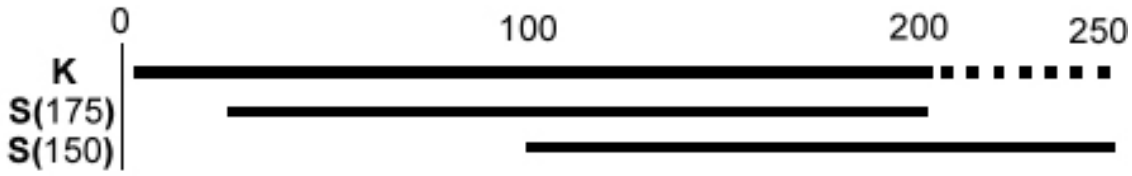
Malli ei käsittele refiksaatioita, vaan fiksaatioiden kestojen mittarina käytetään katseen kestoa (fiksaatioiden summaa sanassa), olivat ne sitten yhdellä tai useammassa fiksaatiossa saatuja. Ellei paikallisesti toisin mainita, koko tämä luku perustuu lähteeseen [Reichle e. al., 1998].

5.1 EZ Reader mallin perusolettamuksia

EZ Readerissä käytetään jo Morrisonin mallissa esitellystä parafoveaalisesta, ennakkohavainnoinista, nimitystä *piilohavainnointi* (covert attention). Kuten Morrisonin mallissakin, tällä tarkoitetaan tilannetta, jossa tarkkaavaisuus siirtyy seuraavaan sanaan, vaikka fiksaatio kohde ei muutu. Tarkkaavaisuuden siirtyminen parafoveaaliselle alueelle (ts. piilohavainnoinnin alkaminen) virittää sakkadin ohjelmoinnin, eli saa aikaan sen, että sakkadin ohjelmointiprosessi aloitetaan.

Silmänliikkeisiin liittyvät kestot (leksikaalinen haku, sakkadin ohjelmointi) on mallissa pyritty saamaan sopusointuun niistä aiemmin tutkimuksissa todennettujen kestojen kanssa. Tällaisia aikoja ovat (1) *katseen kesto* sanaa kohden: yleensä n. 200-250 ms ja (2) *sakkadin ohjelmoinnin kesto:* noin 150-175 ms (vrt kuva 5).

Näin ollen siirtymän ohjelmoinnin tulee alkaa karkeasti arvioiden 25-100 ms sen jälkeen, kun fiksaatio siirtyy sanaan. Jos kuitenkin (aiempien tutkimustulosten perusteella) leksikaalinen haku kestää noin 100-300 ms, ja mallissa oletetaan, että sanan leksikaalisen haun loppuun saattaminen antaa alkusysäyksen sakkadin ohjelmoinnin aloittamiseen, niin miten sanan tunnistus voi vaikuttaa sakkadin ohjelmointiin? Ristiriita selittyy edellä kuvatulla piilohavainnoinnilla, jolloin sanan tunnistus tapahtuu itse asiassa nopeammin kuin em. 100-300 ms katseen kohdistumisesta sanaan.



Kuva 5. *Katseen keston, K, ja sakkadin ohjelmoinnin kestojen S, 175 ms ja S, 150 ms., samanaikaisuus*

EZ Reader olettaa, että mallissa kuvattu silmäliikkeitä ohjaava mekanismi on suhteellisen kuuro, ts. oletetaan että silmänliikkeisiin ei vaikuta mikään ”ulkopuolinen keskusohjaus”. Tätä perustellaan sillä, että koska lukemisessa päätarkoituksena on ymmärtää luettu teksti, tuntuu kohtuuttomalta olettaa, että olisi erillinen järjestelmä silmänliikeiden ohjelmointiin, joka kilpailisi rajallisesta prosessointikapasiteetista.

Kuten Morrisonin mallissakin, tarkkaavaisuuden siirtymisen (ja siten myös sakkadin ohjelmoinnin aloittamisen) saa aikaa leksikaalisen haun loppuun suorittaminen. Tähän liittyen tulee erityisesti tulee huomata että, siirtymisen laukaisevan signaalin lähtemiseen (tai lähtemättä jäämiseen) *ei* vaikuta se, onko lukijalla vaikeuksia luetun ymmärtämisessä. Tärkein E-Z mallin perusoletus onkin, että sakkadien suorittamisen (tai ensin sen valmistelun, sakkadin ohjelmoinnin) saavat aikaan sanan *leksikaaliset, eivät semanttiset ominaisuudet*.

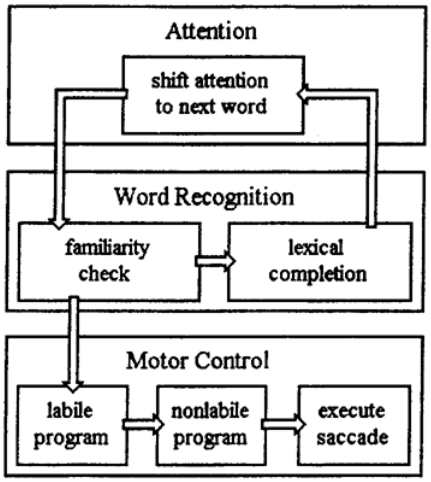
5.2 EZ Reader mallin prosessit

Malliin on pyritty ottamaan mukaan vain ehdottoman välttämättömät lukemisen etenemiseen vaikuttavat prosessit. Niitä on viisi:

- *tuttuustarkistus, f* (familiarity check)
- *leksikaalinen haku*, *lc* (completion of lexical access of the word)
- *sakkadin ohjelmoinnin labiili vaihe*, *m* (labile stage of saccade programming), joka voidaan vielä peruuttaa, tai ”ajaa yli” seuraavalla sakkadilla (vrt. point-of-no-returnia [Henderson and Ferreira])
- *siirtymän ohjelmoinnin vakaa vaihe*, M (nonlabile stage of saccade programming), jonka jälkeen sakkadin peruutus ei enää onnistu
- *sakkadi, s*, varsinainen sakkadin suoritus

Fiksatoitavan sanan havainnointi — ennen sitä kognitiivista prosessia joka selvittää sanan semanttisen sisällön — tapahtuu siis kahdessa vaiheessa: ensin tapahtuu sanan tuttuustarkistus ja sen jälkeen leksikaalinen haku. Jako näihin kahteen ei ole kovin selkeä. Ne voitaisiinkin ajatella samaksi sanan tunnistus – prosessiksi, jollei malli vaatisi jonkinlaista jakoa: on olennaista, että sakkadin ohjelmointi aloitetaan kesken tämän prosessin. Sanan tuttuustarkistus on pinnallisempi sanan havainnointi kuin varsinainen sanan tunnistus; se voi esimerkiksi olla tutun sanakuvan tunnistaminen. Siihen vaikuttavat mm. sanan

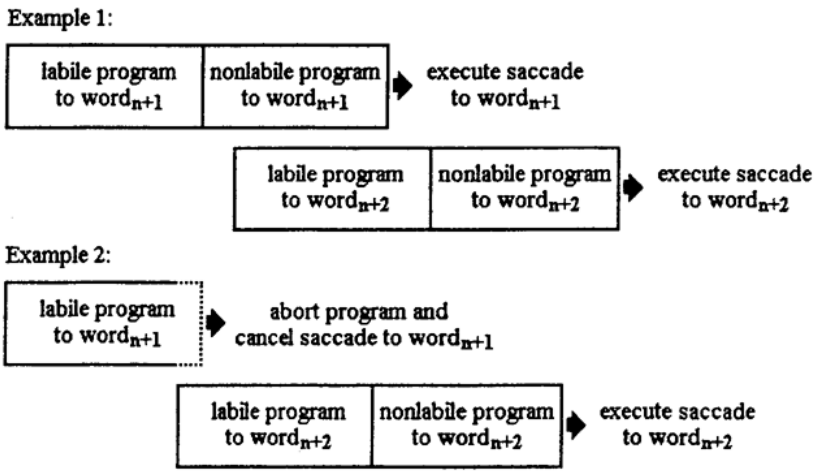
frekvenssi, aika edellisestä käyttökerrasta, edellisten esiintymien lukumäärä, samankaltaisten sanojen esiintymät, jne. Leksikaalisella haulla tarkoitetaan sanan kirjoitus- ja äänneasun tunnistamista, niin että sen jälkeen on mahdollista päästä käsittelemään sanan semanttista sisältöä.



Kuva 6. *EZ Reader –mallin prosessit karkealla tasolla [Reichle et. al. 1998]*

Kuva 6 havainnollistaa miten eri komponentit ovat keskenään vuorovaikutuksessa. Siitä käy ilmi, miten sanan havainnoinnin tuttuusosuuden päättäminen virittää siirtymän ohjelmoinnin labiilin vaiheen- Leksikaalisen haun loppuminen saa aikaan havainnointiprosessin siirtymisen seuraavaan sanaan (piilohavainnointi), vaikka varsinaista sakkadia ei vielä suoritetakaan.

Kuvan 7 ylempi kaavio havainnollistaa ensin miten sakkadia seuraavaan sanaan (n+1) ei enää voida peruuttaa, kun sakkadin ohjelmointi on saavuttanut vakaan vaiheen (example 1), vaikka sakkadia ei varsinaisesti olekaan suoritettu. Toisessa esimerkissä uusi sakkadin ohjelmointi aloitetaan jo siinä vaiheessa, kun edellisen sakkadin ohjelmointi on vielä labiilissa vaiheessa, jolloin uuden ohjelman aloitus ehtii keskeyttää aiemmin aloitetun sakkadin ohjelmoinnin.



Kuva 7. *Esimerkkikaavioita sakkadin ohjelmoinnin labiilin ja vakaan vaiheen suhteesta [Reichle et. al., 1998]*

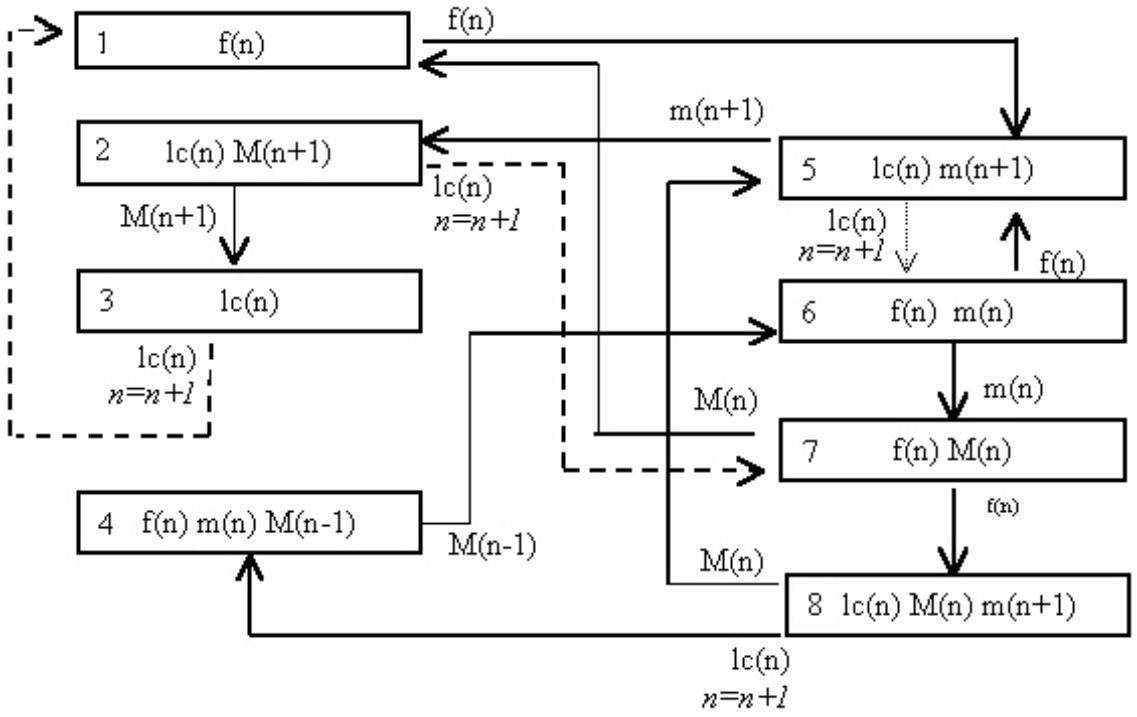
5.3. EZ Reader tila-automaattina

Edellisten kaavioiden avulla voidaan selittää E-Z Reader mallin toimintaa, mutta kvantitatiiviseen mallinnuksen tarpeisiin malli tarvitsee kuvata tarkemmalla operationaalisella tavalla.

Kuvassa 7 malli on kuvattu tila-automaattina (Reichle et. al. itse käyttävät siitä nimitystä order-of-processing -malli). Kukin kaavion suorakaide kuvaa sitä tilaa, jossa prosessi kulloinkin on, ts. mitkä edellä luetelluista prosessesta kulloinkin ovat suoritettavana.

Luettavan tekstin sanat on indeksoitu järjestyksessä niin, että sana n on kulloinkin tarkkaavaisuuden kohteena oleva sana, ei siis välttämättä fiksaation kohteena olevan sana. Prosesseihin on liitetty prosessoinnin kohteena olevan sanan indeksi.

Prosessien välisiin polkuihin liitetyt prosessit kuvaavat sen prosessin, jonka loppuun suoritus saa aikaan polun salliman siirtymän automaatin tilojen välillä. Sanan tunnistuksen *(lc)* päättymiseen liittyvä merkintä *n=n+1*, kuvaa piilohavainnoinnin siirtymistä seuraavaan sanaan, jolloin siis tarkkavaisuuden kohteena oleva sana n päivitetään ”osoittamaan” seuraavaan sanaan. Sakkadi *(s)* kuvataan mallissa vain implisiittisesti, koska siirtymän ohjelmoinnin vakaa vaiheen *(M)* päättyminen saa aina aikaan sakkadin. Prosesseja *M* ja *s* voidaan siis ajatella yhtenä prosessina jonka kesto on siirtymän ohjelmoinnin vakaan vaiheen ja sakkadin suorittamisen yhteenlaskettu kesto.



Kuva 8. *E-Z Readerin "order-of-processing" –malli [Reichle et. al., 1998].*

Esimerkkinä mallin toiminnan tulkinnasta katsotaan seuraavassa mitä tarkoittaisi tila-automaatissa kuljettu reitti $1 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 1$. Kuvitellaan, että tarkasteltavan sanan indeksi on 1 (n=1), ja lähdetään liikkeelle tilasta 1. Tilaan 5 päästään silloin kun ensimmäisen sanan, sanan 1 tuttuus *f(n)* on tarkastettu. Tällöin käynnissä olevat prosessit ovat sanan leksikaalinen haku ja sakkadin ohjelmoinnin labiili vaihe sanaan 2. Tilasta 5 voidaan siityä joko tilaan 2 (jos sakkadin ohjelmoinnin labiili vaihe sanaan 2, *m(n+1)*, valmistuu), tai tilaan 6, (jos sanan 1 leksikaalinen haku, *lc(n*) saadaan suoritetuksi). Tilaan 6 siirryttäessä tarkkaavaisuuden kohde siirtyy sanaan 2 ja sakkadin ohjelmointia suoritetaan edelleen samaan sanaan, sanaan 2. Paluu tilan 7 ja kautta tilaan 1 tarkoittaisi, että sakkadin ohjelmoinnin molemmat vaiheet saatiin suoritetuiksi ja sakkadi sanaan 2 suoritettiin. Tilasta 1 jatketaan siis sanan 2 tuttuustarkistuksella.

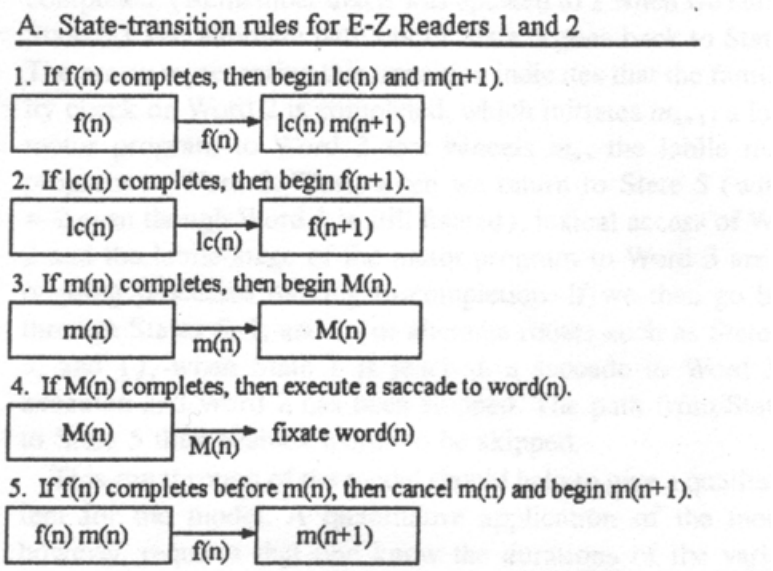
Vaihtoehtoisesti tilasta 6 olisi voitu palata tilaan 5, jolloin olisi tullut käytetyksi automaatin reittiä $1 \rightarrow 5 \rightarrow 6 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 1$. Toisin sanoen edellä tilassa 6 sanan 2 käynnissä oleva leksikaalinen haku olisikin tullut suoritetuksi ennen sakkadin ohjelmien valmistumista, siirtyminen tilaan 5 olisi saanut aikaan sakkadin ohjelmoinnin aloittamisen seuraavalle sanalle, sanalle 3, ja sanan 2 prosessoinnin etenemisen leksikaaliseen hakuun. Jos leksikaalinen haku sanalle 2 tässä tilanteessa valmistuu, siirtyy tarkkaavaisuus jo sanaan 3 ja sen tuttuustarkastusta aletaan suorittaa piilohavainnointina, vaikka fiksaatiokohta on edelleen sanassa 1 (yhtään sakkadin vakaata vaihetta ei ole

saatu loppuun suoritetuksi). Nyt eteneminen tilojen 6 ja 7 kautta tilaan 1 tarkoittaa, että sakkadin ohjelmointi sanaan 3 valmistuu, sana 2 tuli ”piilohavainnoitua” valmiiksi, joten sana 2 tulee hypätyksi yli ja sakkadi suoritetaan suoraan sanaan 3.

Automaatin avulla voidaan siis selittää useita silmänliikkeistä aiemmin tehtyjä havaintoja, kuten esim. piilohavainnointi ja yhden tai useamman sanan yli hyppääminen, kuten edellä huomattiin tai ylivuotoefekti, joka tapahtuisi kuljettaessa mallissa reitti 2 → 3 → 1. Samaten esimerkiksi Morrisonin mallissa esitetty prosessien suorituksen samanaikaisuus toteutuu tila-automaatissa, esimerkiksi tilassa 5 ovat samanaikaisesti suorituksessa leksikaalinen haku, ja sakkadin ohjelmoinnin labiili vaihe.

5.4 E-Z Reader tilasiirtymäkaaviona

Malli on siis äärellinen automaatti; tilasiirtymät määräävät kaikki mahdolliset siirtymät prosessien välillä. Prosessit voivat olla käynnissä rinnakkaisesti kullakin lukuprosessin ajanhetkellä. Tilasiirtymät voidaan esittää viiden säännön avulla (kuva 9).



Kuva 9. *E-Z Reader 1 –malin tilasiirtymät [Reichle, 1998]*

Mallin läpäiseminen katsepolkujen tarkasteleminen auttaa muodostamaan kvalitatiivisen käsityksen mallista. Kvantitatiivinen tarkastelu vaatii kuitenkin, että mallissa käytettyjen prosessien kesto tunnetaan. E-Z reader 1 –mallissa keskimääräiset kestot sanan tuttuustarkistukselle ja leksikaaliselle haulle (f_n, lc_n) oletetaan sanan frekvenssin funktioiksi.

Mallin kvantitatiivisessa tarkastelussa vapaiksi muuttujiksi (ts. niiksi parametreiksi, joille mallin avulla etsittiin parhaiten sopivia arvoja) pyritään jättämään mahdollisimman vähän mallin parametreja. E-Z Reader –mallissa näitä oli viisi.

5.5. Prosessien kestot

5.5.1 Sanan tuttuuden ja tunnistuksen kesto

Kvalitatiivisessa tarkastelussa tarvittavien f_n:n ja lc_n:n keskimääräiset kestot $t(f_n)$ ja $t(lc_n)$ laskettiin seuraavasti:

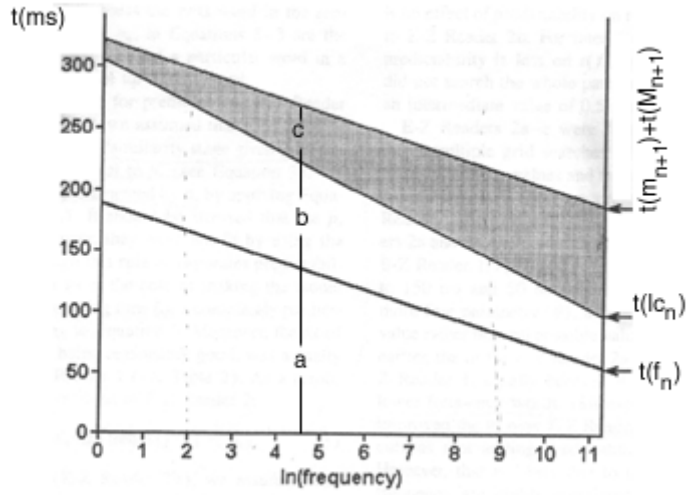
$t(f_n) = fb - [f_m \cdot ln(freq_n)]$, jossa f_b on siirtymäparametri ja f_m kulmaparametri.

Ajan oletettiin siis riippuvan lineaarisesti sanan frekvenssin luonnollisesta logaritmista. Tämä oletus tehtiin useiden sanan tunnistukseen liittyvien tutkimusten perusteella – niissä sanan prosessointiajan ja sanan frekvenssin luonnollisen logaritmin keskinäisen riippuvuuden on todettu olevan lineaarista. Sanan tunnistuksen kestolla ja sanan frekvenssin luonnollisella logaritmilla oletettiin olevan vastaava lineaarinen riippuvuus. Kuitenkin vapaiden muuttujien lukumäärän minimoimiseksi sanan tunnistuksen kesto oletettiin tuttuustarkistuksen keston kerrannaiseksi, eli

$t(lc_n) = \Delta \cdot t(f_n)$, jossa Δ on positiivinen vakio (ei 0)

Aikojen $t(f_n)$ ja $t(lc_n)$ hajauma saadaan aikaan olettamalla, että tuttuuden tarkistukseen kuluneet ajat f_n (vastaavasti $l(c_n)$) tietyssä testissä noudattavat gamma-jakaumaa $T(f_n)$ (vast. $T(lc_n)$), jonka keskiarvo on $t(f_n)$ (vast. $t(lc)$) ja keskihajonta $0.33 \cdot t(f_n)$ (vast. $0.33 \cdot t(lc_n)$). Koska $t(lc_n)$ on $t(f_n)$:n kerrannainen, ero tuttuustarkistuksen ja sanan tunnistuksen vaatimien aikojen välillä kasvaa, kun sana on harvinaisempi (ts. frekvenssi pienenee). Tämä tuntuu järkevältä, koska piilohavainnoinnin tuoma etu silloin luonnollisestikin vähenee.

Kuva 10 havainnollistaa tilannetta. Sanan frekvenssin kasvaessa sen tuttuustarkistukseen $t(f_n)$ tarvittava aika (a) pienenee. Myös sen tunnistukseen $t(lc_n)$ tarvittava aika (b) pienenee, mutta nopeammin, koska $t(lc_n) = \Delta \cdot t(f_n)$. Kaavion ylin suora kuvaa aikaa, jonka kuluttua sakkadin ohjelmointi on valmis, ja sakkadi suoritetaan seuraavaan sanaan. Ylimmän ja kolmanneksi ylimmän suoran välinen aika (b+c) kuvaa sakkadin ohjelmointiin kuluvaa aikaa — se pysyy vakiona, koska se ei ole riippuvainen sanan frekvenssistä. Näin ollen kahden ylimmän suoran väliin jäävä harmaa alue (b) kuvaa aikaa, joka jää sanan tunnistuksen ja siirtymän ohjelmoinnin valmistumisen väliin. Tämä on aika joka jää seuraavan sanan piilohavainnointiin. Piilohavaoinnoinnilla saatava etu on siis sitä suurempi, mitä yleisempi fiksatoitava sana on.



Kuva 11. *Piilohavainnoinnin etu sanan frekvenssin funktiona [Reichle, 1998]*

5.5.2 Sakkadin ohjelmoinnin ja suorittamisen kesto

Sakkadin kesto t*(s)* oletettiin vakioksi, 25 ms, mutta siirtymien labiilille t*(m)* ja vakaalle vaiheelle t*(M)* sallittiin hajauma käyttäen, kuten edellä, gamma-jakaumaa käyttäen keskihajontana keskiarvon kolmannesta.

5.6 Perusmallia täydentävät mallit E-Z Reader 2, 3, 4 ja 5

Edellä kuvattu malli on itse asiassa E-Z Readerin perusmalli E-Z Reader 1. Reichler et. al. tekivät tämän jälkeen vielä neljä mallin tarkennusaskelta.

Toisessa E-Z Reader 2 –mallissa otettiin huomioon myös myös sanojen ennustettavuus. Sanojen ennustettavuus kartoitettiin empiirisin testein ja sen avulla tarkennettiin tuttuustarkistuksen kestoa $t(f_n)$.

E-Z Reader 1 ja 2 ottivat huomioon vain sanojen väliset sakkadit ja sanaan kohdistuvat useammat fiksaatiot koottiin yhteen katseeksi. E-Z Reader 3 ottaa huomioon myös lisäksi aloitusfiksaation keston, yhdenfiksaation katseen ja ennustaa jakaumaa yhdenfiksaation ja kahdenfiksaation katseen avulla käsiteltyjen sanojen kesken.

E-Z Reader 4 korjaa edellisten mallien olettamusta, että parafoveaalisen prosessoinnin tehokkuus oletettiin samaksi kuin foveaalisen prosessoinnin tehokkuus. Malliin lisättiin *eksentrisyys*-parametri (eccentricity), jolla malliin tuotiin mukaan tieto siitä, kuinka kaukana havainnoitava sanan on foveasta (sanatasolla, ts. viereisen sanan piilohavainnointi on tehokkaammin kuin kahden sanan etäisyydellä olevaa sanaa). Tämän mallin kvantitatiiviset validointitulokset olivat kuitenkin huonommat kuin E-Z reader 3:n tuottamat. Kuitenkin mallin olettamus on havaintopsykologisesti perusteltu.

Viidennessä mallissa korjattiin olettamusta, että sanan eksentrisyys vaikuttaa sekä tuttuustarkistukseen, että tunnistukseen samalla tavalla.

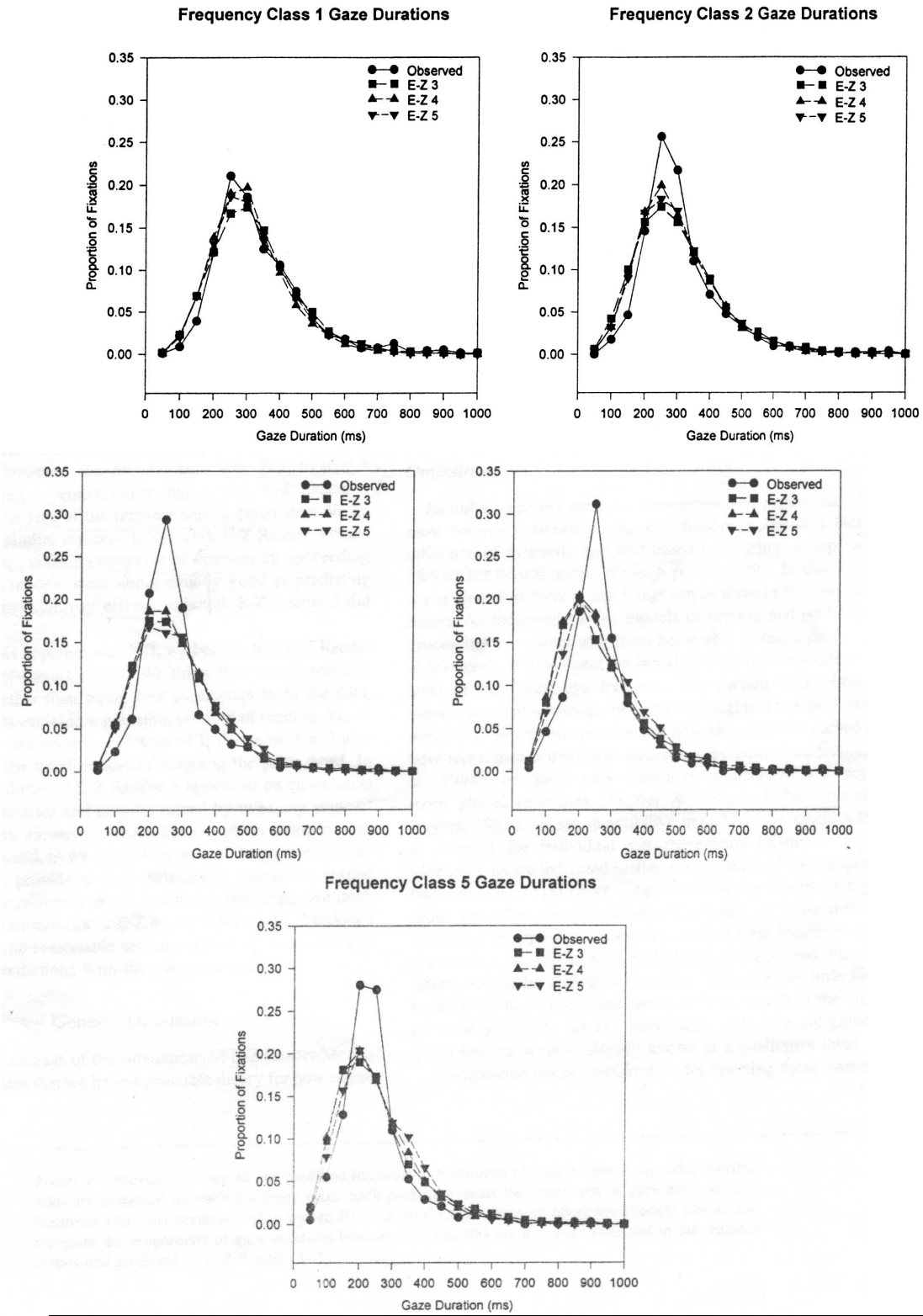
Tuttuustarkistus on karkeampi prosessi, joten voidaan olettaa että etäisyys ei ehkä vaikuta sen kestoon yhtä paljon kuin tunnistuksen kestoon. Kvantitatiiviset validointiulokset paranivat E-Z Reader 4:n tuloksista, mutta eivät juurikaan 3:sta. Malli on kuitenkin psykologisesti uskottavampi, joten tutkijat itse pitivät E-Z Reader 5 mallia nykyisenä ”state of art” -mallina.

5.7 E-Z Reader -mallien vahvuuksista ja puutteista

E-Z Reader on tietääkseni toistaiseksi ainoa malli, joka on onnistunut tällä luku-prosessin kuvaustasolla sekä mallin kvalitatiivisessa (malli psykologista uskottava: onnistuu selittämään suuren osan aiemmista tutkimustuloksista), että kvantitatiivisessa validoinnissa (empiirisen datan sovitus malliin, kts. kuva 12). Silmänliiketutkimuksen perinteisen tutkimuksen laajuuden huomioon ottaen, E-Z Reader malli on mielestäni vaikuttava ja vakuuttava, ja onnistuu selittämään laajasti saatuja tutkimustuloksia. Tämä ei kuitenkaan tarkoita, etteikö tutkittavaa ja edelleen kehiteltävää olisi vielä paljon jäljellä; E-Z Reader mallikin on kuitenkin monelta osin rajoittunut.

E-Z Reader ei pyri mallintamaan tarkkaa fiksaatioon kohdetta (sanan sisällä). Fiksaatioita mallinnetaan sanan tarkkuudella. On olemassa tutkimuksia, joissa pyritään selittämään mihin kohtaan sanaa fiksaatio todennäköisesti osuu, ja miten yli- tai aliosumat vaikuttavat seuraavaan. Rayner et. al. toteavat että E-Z Reader on laajennettavissa sisältämään moduulin, joka tekee myös tarkan osumakohdan mallintamisen mahdolliseksi. Malli ei ota huomioon myöskään virheellisiä sakkadeja – siinä oletetaan, että sakkadi osuu aina kohteeksi tarkoitettuun sanaan.

Malli ei ota lainkaan huomioon regressioita, joten näin ollen mallilla ei myöskään pystytä selittämään korkeamman tason kielelliseen prosessointiin liittyviä ongelmia. Ydinmalli perustuu siihen, että *sanan tunnistus* (tai vastaava kognitiivinen prosessi) ohjaa silmien liikkeitä. Esimerkiksi niin sanottujen garden path –virkkeiden (virkkeet, joiden ymmärtäminen vaatii peruutusta, ja uudelleen jäsentämistä), tai hämmentävien pronominien (joita seuraa esim. sukupuolten stereotypioista) aiheuttamia regressioita ei mallissa oteta huomioon. Tällaisia lukemisen aikaista uudelleen prosessoinnin tarvetta aiheuttavia tekijöitä on tutkimuksissa raportoitu paljon.



Kuva 12. *Katseen kestojen jakaumat empiirisestä datasta ja E-Z Reader –mallien simuloinneista eriteltynä sanojen yleisyyden perusteella viiteen eri frekvenssiluokkaan kuuluville sanoille. [Reichle et. al., 1998]*

Näiden korkealla prosessointitasolla tapahtuvien ongelmien paikallistaminen on vaikeaa – lisäksi tällaisen ongelman ratkaisustrategiat ovat todennäköisesti hyvin yksilöllisiä. E-Z Reader kuvaa siis ”oletusarvoista lukuprosessia”. Joissain tilanteissa korkeamman tason tekstin ymmärtämiseksi vaadittavat kognitiiviset prosessit voivat ajaa oletusarvoisen prosessin yli – ts. kontrolli siirtyy monimutkaisemmalle prosessille, josta palataan takaisin mallin kuvaamaan lukuprosessiin, kun ongelma on selvitetty. Kuitenkin, jotta ongelmaan päästään pureutumaan, se täytyy jakaa osiin, jolloin todennäköisesti tekstin korkeamman tason kognitiivisesta prosessoinnista (ts. ymmärtämisestä) johtuvat regressiot jätetään mallissa huomioimatta. Mallia validoitaessa data, joka sisälsi regressioita edellisiin sanoihin jätettiin kokonaan pois.

6. Yhteenveto

Tässä esityksessä on tehty katsaus lukemisen aikana tapahtuvien silmien liikkeiden mallinnuksen kehitykseen. Alan tutkimus on ollut paljon vilkkaampaa, kuin lähtiessäni tekemään katsausta osasin olettaa. Pidän kuitenkin luvuissa 2 ja 3 esitettyjen taustatietojen ja taustalla tehdyn tutkimuksen jonkin tason tuntemusta tärkeänä silmänliikemallien syvällisemmälle ymmärtämiselle. Mallien takana olevat olettamukset tuntuvat joskus löyhiltä, ellei ole tutustunut siihen massiiviseen tutkimukseen jonka pohjalta olettamukset on tehty.

Alkuaikojen lukututkimukset nojasivat vahvasti oletukseen, että okulomotorinen järjestelmä kontrolloi silmänliikkeitä autonomisesti, eikä luetun tekstin kognitiivinen prosessointi vaikuta matalalla tasolla lainkaan katseen ohjautumiseen luetussa tekstissä.

Tämä oletus viimeisten vuosikymmenten tutkimuksissa selkeästi kumottu. On todettu, että sanojen kognitiivista prosessointia vaativat piirteet vaikuttavat jo hyvinkin matalalla tasolla, aiheuttaen variaatiota jo yksittäisten sakkadien kohteen valinnassa ja fiksaatioiden kestoissa.

Lukuprosessia selittävät mallit ovat olleet luonteeltaan joko kvantitatiivisesti tai kvalitatiivisesti painottuneita. Kvantitatiivisesti painotuneet mallit pyrkivät silmänliiketietoja mittaamalla ja analysoimalla kehittämään yleisiä tunnuslukuja (keskiarvoja, jakaumia), joiden avulla on pyritty ennustamaan silmänliikkeitä lukuprosessissa [Just and Carpenter, O’Reagan, 1990, Suppes 1994]. Kvalitatiivisemman otteen malleissa pyritään enemmänkin selittämään, kuvaamaan ja ymmärtämään silmänliikkeiden kättäytymistä lukemisen aikana [Morrison, 1984, Henderson and Ferreira, 1990, Suppes 1990, Legge et. al., 1997, Reichle, 1998]. E-Z Reader malli [Reichle, 1998] onnistuu tällä hetkellä sulauttamaan parhaiten lukemiseen liittyviä moninaisia piirteitä, ja se kuvataankin tässä esityksessä muita tarkemmin.

Oma mielenkiintoni lukututkimukseen lähtee hyvin soveltavasta näkökulmasta. Katse on toistaiseksi vähän käytetty modaliteetti pyrittäessä kehittämään ihmisen ja tietokoneen välistä vuorovaikutuksesta monipuolisemmaksi ja luonnollisemmaksi. Jotta voisimme tehokkaasti hyödyntää katsetta käyttöliittymässä, meidän tulisi pystyä mahdollisimman tarkasti päättelemään silmänliikkeiden avulla käyttäjän tila, ts. mitä hän kullakin hetkellä on tekemässä. Lukutilanne tulee helposti vastaan monissa erityyppisissä tietokoneen käyttötilanteissa, joten se tieto, lukeeko käyttäjä, ja jos, niin mitä lukee, ja miten lukee, saattaisi monissa tilanteissa olla erittäin hyödyllinen informaatio sovelluksen toiminnalle.

Monet lukututkimuksessa tehdyistä yksittäisistä havainnoista ovat potentiaalisesti hyödyllisiä pyrittäessä tulkitsemaan ihmisen käyttäytymistä ja tilaa vuorovaikutustilanteessa. Edellä kuvattu E-Z Reader malli auttaa kyllä hyvin ymmärtämään miten lukuprosessi etenee, mutta itse mallin käyttöä lukemisen ennustamiseksi tai lukutilanteen tunnistamiseksi reaaliaikaisesti sovelluksessa en pidä kovin realistisena. Malli kuvaa lukuprosessia tähän tarkoitukseen liian matalalla tasolla, ja käyttöliittymän ollessa kyseessä juuri ne ymmärtämisvaikeuksiin ja regressioihin liittyvät seikat, jotka E-Z Reader jättää käsittelemättä, olisivat tärkeitä.

Viiteluettelo

[Abrams and Jonides, 1995] Abrams R. A., Jonides J., Programming saccadic eye movements. Journal of Experimental Psychology: Human Perception and Performance, 14, 1995, 428-443

[Balota et. al., 1985] Balota D. A., Pollatsek A., Rayner K., The interaction of contextual constraints an paroveal visual information in reading. *Cognitive Psychology*, ***7***, 1985**,** *346-390.*

[Brysbaert and Vitu, 1998] Brysbaert Marc and Vitu Francoise, Word Skipping: Implications for theories of eye movement control in reading. In *Eye Guidance in Reading and Scene perception* (pp. 125-147), Underwood G. (ed.), 1998, North-Holland: Elsevier Science Publishing.

[Duchowski, 2000] Duchowski Andrew, Eye-Based Interaction in Graphical Systems: Theory & Practice, course notes of the tutorial in *SIGGRAPH 2000*, available at *http://www.vr.clemson.edu/eyetracking/sigcours/e* (checked on 10.5.2001)

[Frazier and Rayner 1982] Frazier L., Rayner K., Making and correcting errors during sentece comprehension: Eye movements in the analysis of structurally ambiguous sentences. Cognitive Psychology, **14**, 1982, 178-210.

[Frenck-Mestre and Pynte, 1995] Frenck-Mestre C. and Pynte J., Lexical influences on parsing strategies: Evidence from eye movements. In *Eye Movement Research: Mechanisms, Processes, and Applications* (pp. 433-444). J. M. Findlay, R. Walker, & R. W. Kentridge (Eds.), 1995, New York: Elsevier Science Publishing.

[Henderson and Ferreira, 1995] Henderson, J.M., & Ferreira, F. The effects of foveal difficulty on the perceptual span in reading: Implications for attention and eye movement control. Journal of Experimental Psychology: Learning, Memory, and Cognition, 16, 1990, 417-429.

[Hyönä 1995] Hyönä Jukka, Silmäliikkeet, kognitio ja lukeminen. Psykologia, **30**, 1995, 89-95.

[Hyönä et. al., 1995] Hyönä Jukka, Laine Matti, Niemi Jussi, Effects of a word's morphological complexity on readers´ eye fixation patterns. In Eye Movement Research: Mechanisms, Processes, and Applications (pp. 444-463), J. M. Findlay, R. Walker, & R. W. Kentridge (Eds.), 1995, New York: Elsevier Science Publishing.

[Hyönä 1998] Hyönä Jukka, *Lukeminen havaintoprosessina.* Psykologia, **33**, 1998, 276-282.

[Inhoff, 1989] Inhoff A. W. Parafoveal processing of word and saccade computation during eye fixation in reading. *Journal of Experimental Psychology: Human Perception and Performance*, **15**, 1989, 544-555.

[Just and Carpenter, 1980] Just Marcel Adam A., & Carpenter Patricia A. A theory of reading: From eye fixations to comprehension. *Psychological Review*, **87**, 1980, 329-354.

[Legge et. al., 1997] Legge, G. E., Klitz, T. S., & Tjan, B. S. (1997). Mr. Chips: An ideal-observer model of reading. *Psychological Review*, **104,** 524-553.

[Lima, Inhoff, 1985] Lima S. D., Inhoff A.W., Lexical access during eye fixatons inreading: Effects of word-initial letter sequences. *Journal of Experimental Psychology: Human Perception and Performance*, **11**, 1985, 272-285.

[Morrison, 1984] Morrison R. E., Manipulation of stimulus onset delay in reading: Evidence for parallel programming of saccades. *Journal of Experimental Psychology: Human Perception and Performance*, **10**, 1984, 667-682.

[Murray, 1998] Murray W. S., Parafoveal pragmatics. In Eye Guidance in Reading and Scene perception, Underwood G. (ed.), , 181-200, 1998, North-Holland: Elsevier Science Publishing.

[O'Reagan, 1990] O'Regan J. K., Eye movements and reading. In *Eye movements and their role in visual and cognitive processes,* E. Kowler (ed.), 395-453, 1990, Amsterdam: Elsevier.

[Rayner et. al. 1978] Rayner K., McConkieG. W., Erlich S.F., Eye movements and information integration across fixations. *Journal of Experimental Psychology: Human Perception and Performance*, **4**, 1978, 529-544.

[Rayner et. al. 1981] Rayner K., Inhoff A.W., Morrison R., Slowiaczek M. L., Bertera J.H., Masking of foveal an dparafoveal vision during eye fications in reading. *Journal of Experimental Psychology: Human Perception and Performance*, **7**, 1981, 167-179.

[Rayner et. al., 1978] Rayner Keith, Well A.D., Pollatsek A., Asymmetry of the effective visual field in reading. *Perception & Psychophysics.* **27**, 1980, 537-544.

[Rayner and Duffy 1986] Rayner Keith, Duffy S. A, and Raney G. E., Lexical complexity and fixation times in reading: Effects of word frequency, verb complexity, and lexical ambiguity. *Memory & cognition*, **14**, 1986, 191-201.

[Rayner and Pollatsek 1989] Rayner Keith, Pollatsek Alexander, *Psychology of Reading*, 1989, Englewood Cliffs, NJ:Erlbaum.

[Rayner and Sereno 1989] Rayner Keith, Sereno S.C., Eye movements in reading: Psycholinguistic studies. In *Handbook of psycholinguistic* (pp. 57-81), Gernsbacher, M. A. (Ed.), 1994, San Diego, CA: Academic Press.

[Rayner and Sereno 1994] Rayner Keith, Sereno S.C., Eye movements in reading: Psycholinguistic studies. In *Handbook of psycholinguistic* (pp. 57-81), Gernsbacher, M. A. (Ed.), 1994, San Diego, CA: Academic Press.

[Rayner 1995] Rayner Keith, Eye movements and cognitive processes in reading, visual search, and scene perception, In *Eye Movement Research: Mechanisms, Processes, and Applications* (pp. 3-21), J. M. Findlay, R. Walker, & R. W. Kentridge (Eds.), 1995, New York: Elsevier Science Publishing.

[Rayner 1998] Rayner Keith, Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin.* **123**, 1998, 372-422.

[Rayner et. al., 1998] Rayner Keith, Reichle Erik D., and Pollatsek Alexander, Eye movement control in reading: An overview and model. In *Eye Guidance in Reading and Scene perception* (pp. 243-268), Underwood G. (ed.), 1998, North-Holland: Elsevier Science Publishing.

[Reichle 1998] Reichle Erik, Pollatsek Alexander, Fisher, Donald, Rayner, Keith, Toward a model of eye movement control in reading. *Psychological Review,* **105**, 1998, 125-157.

[Suppes, 1994] Suppes P., Stochastic model in reading. In *Eye movements in reading*, Ygge, Lennerstrand (eds.), *1994,* 349-367, Aoxford, England, Pergamon Press.

[Vitu ja O'Reagan, 1998] Vitu Francoise and O'Reagan J. Kevin, A challenge to current theories of eye movements in reading. In *Eye Movement Research: Mechanisms, Processes, and Applications* (pp. 381-391), J. M. Findlay, R.

Walker, & R. W. Kentridge (Eds.), 1995, New York: Elsevier Science Publishing.

Katsekirjoituksessa käytettyjä malleja

Päivi Majaranta

Katsekirjoitusta käyttävät tyypillisesti vaikeasti vammaiset, jotka eivät pysty puhumaan eivätkä kirjoittamaan muulla tavoin. Tyypillisesti katsekirjoitussysteemi on toteutettu siten, että näytöllä on virtuaalinäppäimistö, jonka kirjaimia katsomalla vammainen pystyy kirjoittamaan ja kommunikoimaan. Katse ei ole kovin tarkka, joten näppäinten koko on suuri, kaikki näppäimet eivät mahdu ruudulle ja näppäimistö vie paljon tilaa. Katseella kirjoittaminen on myös erittäin hidasta. Tämän seminaarityön tarkoitus on kartoittaa, millaisia teorioita ja malleja on kehitetty ja käytetty katsekirjoitussysteemeissä. Seminaarityössä esitellään Jacobin kehittämä taksonomia katsesyötteelle sekä kaksi mallia, joista Freyn, Whiten ja Hutchinsonin malli on kehitetty virtuaalinäppäimistön näppäinten dynaamiseen järjestämiseen sen mukaan, missä järjestyksessä kirjaimet ja välimerkit seuraavat toisiaan kirjoitetussa tekstissä. Salvuccin kehittämä malli mahdollistaa kirjoitetun sanan päättelemisen katsepolusta. Molemmat mallit soveltavat Markovin ketjuja ja molemmissa hyödynnetään tietoa merkkien järjestyksestä ja todennäköisyydestä, jolla merkit seuraavat toisiaan. Jälkimmäisen taustalla on sanakirja ja kielioppi, ensimmäinen tallettaa vain kirjainpareja ja päättelee seuraavan merkin yhden tai kahden edellisen merkin perusteella. Malleja käyttämällä voidaan katsekirjoitusta nopeuttaa merkitsevästi.

1. Johdanto

Katseella kirjoittamisen malleihin liittyvät ainakin sekä mallit siitä, miten katse käyttäytyy (katse syöttömetodina) sekä se, miten ihminen kirjoittaa (miten merkit seuraavat toisiaan). Keskityn tässä kuvaamaan malleja, jotka on kehitetty katsekirjoitukseen ja joita on testattu katsekirjoituksessa. Artikkelin lopussa kerron myös malleista, joita voisi hieman muuttamalla ja soveltamalla käyttää katsekirjoituksessa.

Yleisesti voidaan sanoa, että katse kertoo missä ihmisen huomio on [Monty ja Senders, 1976]. Kun ihminen katsoo jotain kohdetta, sanotaan, että ihminen kiinnittää katseensa kohteeseen, tapahtuu ns. fiksaatio. Kun katse siirtyy kohteesta toiseen (fiksaatiosta toiseen), tapahtuu sakkadi. Sakkadi on siis kiinnitysten eli fiksaatioiden välillä tapahtuva liike, katseen siirtäminen kohteesta toiseen. Myös fiksaation sisällä tapahtuu liikettä, ns. mikrosakkadeja. Ihmisen silmä tekee koko ajan pientä liikettä. Silmässä olevien sauvojen ja tappien saama informaatio ”vanhenee” ja silmän täytyy liikkua, jotta kuva säilyy. Kaikki informaatio saadaan fiksaatioiden aikana.

Katse toimii paljolti tiedostamatta, vaikka voimmekin kohdistaa katseemme tietoisesti mihin haluamme. Katseen voidaan todeta olevan luonnollinen tapa kommunikoida, on helppoa ja luonnollista osoittaa kiinnostuksen kohde yksinkertaisesti katsomalla sitä. Kun kohdetta katsotaan riittävän pitkään (esim. 750ms ajan), voidaan pitkä kiinnitys tulkita komennoksi valita fokuksen kohteena oleva kohde. Kun katsetta käytetään sekä informaation hankkimiseen että komentojen antamiseen tietokoneelle, mukana seuraa ongelmia. Midaksen kosketukseksi kutsuttu ongelma syntyy siitä, kun ihminen katsoo tietokoneella jotain kohdetta ja tietokone tulkitsee sen komennoksi valita ko. kohde [Jakob, 1991].

Vaikka katse on nopea, katseella kirjoittaminen on hidasta. Katseella on mahdollista valita ruudulta noin yksi kohde (virtuaalinen näppäin) sekunnissa, eli periaatteessa kirjoitusnopeus voi olla noin 60 merkkiä minuutissa [Friedman *et al.*, 1985; Gips ja Olivieri, 1996]. Hyvä konekirjoittaja pystyy kirjoittamaan saman verran sanoja minuutissa tavallisella näppäimistöllä. Käytännössä ei useinkaan päästä edes yksi merkki sekunnissa -nopeuteen, eli käytännössä minuutissa ehditään kirjoittaa vain muutama sana, ehkä vain yksi sana. Jos katsesyötettä käyttävä ihminen on puhekyvytön, käyttää hän katsekirjoitusta myös kommunikointiin. Tällöin nopeus on kriittinen tekijä. On vaikea kommunikoida luontevasti, jos toinen pystyy tuottamaan vain pari sanaa minuutissa, kun normaalisti puhuva vastapuoli pystyy tuottamaan yli sata sanaa minuutissa.

Katsekirjoittamiseen liittyvien mallien taustalta löytyvä motivaatio liittyy yleensä tarpeeseen nopeuttaa katsesyötettä. Osaksi tämä johtuu katseen

epätarkkuudesta, aina ei ole helppoa päätellä, mihin ihminen katsoo. Koska katse pystytään määrittämään vain noin asteen tarkkuudella, ei tietokoneen ruudulle välttämättä mahdu kaikkia merkkejä kerralla; myös itse tekstille pitää jäädä tilaa. Tällöin joudutaan osa merkeistä piilottamaan valikoihin, hierarkkisiin näkymiin, jotka näytetään vasta kun käyttäjä valitsee tietyn komennon. Tämä luonnollisesti hidastaa katseella kirjoittamista entisestään.

Kuva 1. Virtuaalinäppäimistö vie näytöltä tilaa muilta ohjelmilta.
(Lähde: QuickGlance, http://www.eyetechds.com)

2. Silmänliikkeeseen perustuvan vuorovaikutuksen taksonomia

Jacob [1995] jakaa katsesyötteen luonnolliseen ja epäluonnolliseen ottaen huomioon sekä silmien liikkeen (syötteen) luonnollisuuden että vasteen (*response*) luonnollisuuden. Jacob määrittää luonnollisuuden sen mukaan, miten se vastaa reaalimaailmaa, sitä, miten käytämme katsetta luonnollisesti ollessamme vuorovaikutuksessa reaalimaailman kanssa.

Tyypillisesti komentopohjaisissa käyttöliittymissä (kuva 2, (a)) katsesyöte on luonnotonta. Suurin osa katsekommunikointijärjestelmistä kuuluu tähän kategoriaan. Käyttäjän pitää kiinnittää katseensa haluttuun kirjaimeen esimerkiksi 900 millisekunnin ajan, jotta kirjain tulee valituksi. Käyttäjän täytyy opetella käyttämään katsettaan luonnottomasti, katsomaan kirjaimia viiveellä, tai vaihtoehtoisesti opetella käyttämään silmänräpäytystä tai jotain muuta kytkintä kirjaimen valintaan. Kun käyttöliittymä reagoi katsekiinnitykseen (”tuijotukseen”) eri tavalla kuin muuhun informaatiota keräävään katselemiseen, katseen vasteen voidaan sanoa olevan luonnotonta.

Reaalimaailmassa emme yleensä voi aktivoida kohteita kiinnittämällä katseemme niihin.

		Vaste	
		Luonnoton	**Luonnollinen**
Silmän liike (syöte)	**Luonnoton** (opittu)	(a) komentopohjaiset käyttöliittymät	
	Luonnollinen	(b) ei-komentopohjaiset käyttöliittymät	(c) virtuaali-maailmat

Kuva 2. Jacobin taksonomia katsesyötteen käyttämisestä käyttöliittymissä

Toisaalta, jos katseen suuntaa käytetään antamaan lisäinformaatiota esimerkiksi käyttäjän kiinnostuksen kohteesta, tällöin katsesyötteen voidaan sanoa olevan luonnollista (b). Käyttäjä vain katselee normaalisti ympärilleen ja tietokone reagoi sen mukaan. Tällöin vaste on luonnotonta. Virtuaalimaailmoissa sekä syöte että vaste voivat olla luonnollisia tai luonnollisen kaltaisia, vaikkakin myös luonnottomien syötteiden ja vasteiden käyttö on mahdollista.

Alla esitellyistä malleista ensimmäinen kuuluu ryhmään (a), kyseessä on komentopohjainen käyttöliittymä, jossa sekä syöte (silmän liike) että vaste ovat epäluonnollisia. Jälkimmäinen malli puolestaan kuuluu ryhmään (b), koska siinä käyttäjä voi katsella luonnolliseen tapaan, silmien liikettä ei rajoiteta eikä määrätä. Systeemi osaa päätellä normaalista katsepolusta, mitä käyttäjä haluaa kirjoittaa.

3. Tekstinsyöttömalli ennustaa seuraavan merkin

Frey, White ja Hutchinson [1990] kehittivät mallin, joka ennustaa mitkä merkit seuraavat toisiaan tietyllä todennäköisyydellä. Mallin avulla pystytään nopeuttamaan katseella kirjoittamistehtävää tilanteessa, jossa kaikki merkit eivät mahdu ruudulle yhtä aikaa, vaan ne pitää jakaa hierarkkisesti valikoihin. Kirjaimet, numerot ja välimerkit seuraavat toisiaan tietyn säännön mukaan. Sääntö riippuu kielestä; esimerkiksi suomenkielessä ei kirjainten esiintymistiheys ja järjestys ole sama kuin esim. englannissa.

Frey *et al.* kehittivät metodin kehittämässään ERICA (Eye-gaze response interface computer aid) katsekommunikointisysteemiä [Hutchinson, *et al.*, 1989] varten. ERICAn kehityksen alkuaikoina näytöt olivat pienempiä, eivätkä katseenseurantalaitteet olleet aivan yhtä tarkkoja kuin nykyään. Siksi ERICAssa

oli vain kuusi valittavissa olevaa näppäintä näkyvissä tekstisyötekentän lisäksi. Nykyään tietokoneiden näytöt ovat suurempia, joten ruudulle mahtuu yhtä aikaa enemmän merkkejä kuin mallin kehittämisen aikaan. Vaikka malli on osin vanhentunut, on se toiminut pohjana nykyisille malleille. Tästä lisää tämän kappaleen lopussa.

Tekstin voidaan ajatella olevan yksittäisistä merkeistä koostuva sarja $W_t = \{w_t \in A: t=1, \ldots, n\}$, eli n kappaleen muodostama jono merkkejä, jotka koostuvat N kappaleesta aakkosia, välimerkkejä ja muita merkkejä $A=\{w_i, i=1, \ldots, N\}$. Frey *et al.* laskivat aluksi kunkin ASCII-merkistön kirjaimen ja merkin todennäköisyyden perustuen siihen, kuinka paljon niitä esiintyi tekstissä. Teksti, jota he käyttivät pohjana oli neljän eri henkilön kirjoittamaa ja se koostui eri tyyppisistä teksteistä. Tekstissä oli 55 882 merkkiä. Frey *et al.* varmensivat tekstinsä sopivuutta vertaamalla sitä laajempaan yli miljoonasta sanasta kerättyyn Brown Corpus –aineistoon. Kun kaikki merkit otettiin huomioon, havaittiin, että eniten esiintymiä oli välilyönnillä. Kirjaimista eniten esiintyi kirjain ’e’. Yksittäisten merkkien todennäköisyydestä luotiin taulukko, jossa pienet kirjaimet, isot kirjaimet ja välimerkit (ja muut merkit) oli jaettu omiin ryhmiinsä. A1 = {L, U, K}, missä L = pienet kirjaimet, U = isot kirjaimet, K = muut merkit. Välimerkkien ja isojen kirjainten todennäköisyydet olivat huomattavasti pienempiä kuin pienten kirjainten: $Pr(w_i \in L)=0.758$, $Pr(w_i \in U)=0.020$ ja $Pr(w_i \in K)=0.222$. Koska välilyönnin todennäköisyys on niin suuri, se nostettiin samalle tasolle yleisimpien pienten kirjainten kanssa. Palautusnäppäimen todennäköisyyttä ei pidetty ongelmana, koska tietokone osaa jakaa tekstin riveille automaattisesti. Täten se käsiteltiin muiden merkkien ryhmässä. Kuvassa 3 ([Frey *et al.*, 1990]) on puu, joka kuvaa järjestyksen johon merkit ruudulla jaettiin todennäköisyyksiä noudattaen.

Frey *et al.* laskivat odotusarvon ajalle, joka kuluu tekstin syöttämiseen kun kirjaimet on näytöllä jaettu noudattaen kirjainten esiintymistiheyden todennäköisyyttä. He laskivat ensin, montako askelta (merkkihierarkiassa) kuluu yhden merkin valintaan. Tulos oli 1.89 askelta merkkiä kohden. Kun otetaan huomioon katsevalinnassa käytetty yhden sekunnin viive eli kiinnitysaika (*dwell time*) ja oletetaan että yhdelle sivulle mahtuu noin 2400 merkkiä, voidaan laskea että yhden sivun kirjoittamiseen kuluu aikaa noin 80 minuuttia. Aika on todella pitkä.

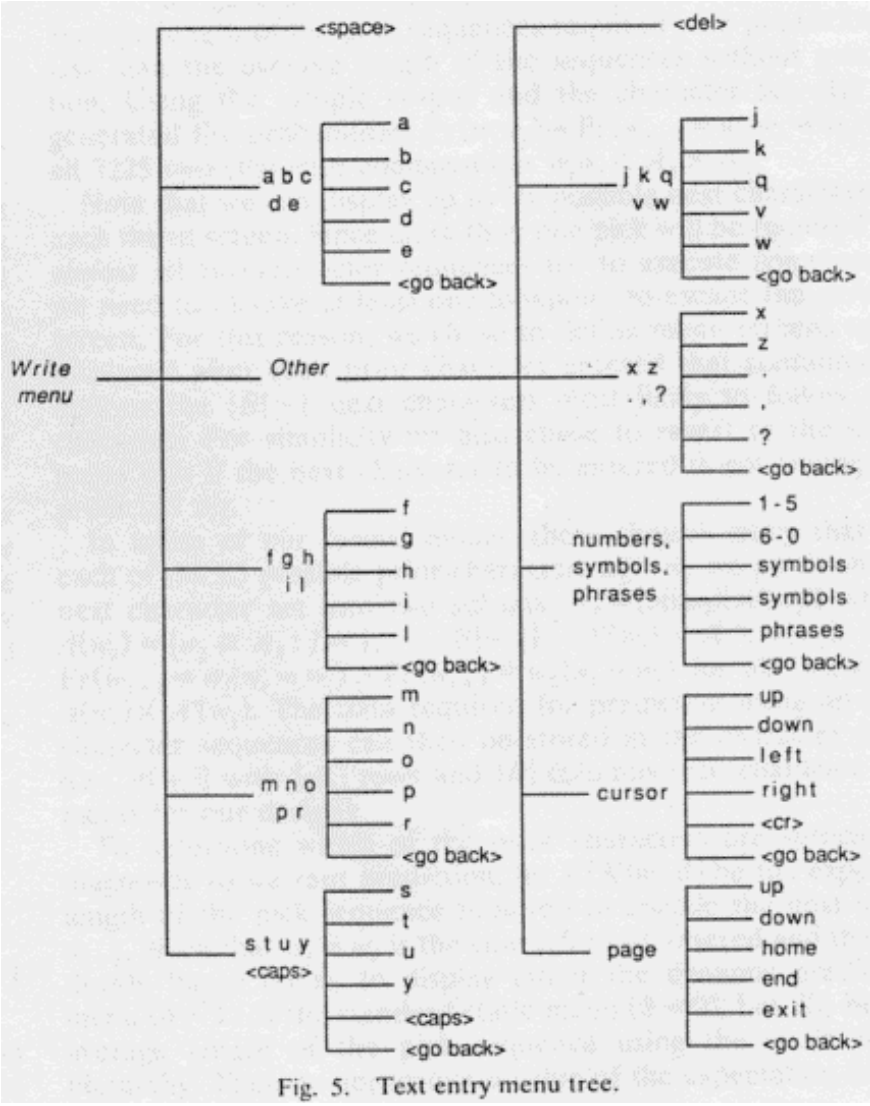


Fig. 5. Text entry menu tree.

Kuva 3. Tekstinsyöttöön käytetty puurakenne.

Frey *et al.* päätyivätkin laskemaan todennäköisyyksiä sille, mikä merkki seuraa jo kirjoitettua merkkiä (*character string prediction model*). He totesivat, että on käytännössä mahdotonta laskea kaikki mahdolliset todennäköisyydet kaikille merkeille ja niitä seuraaville merkeille. Ensin he kehittivät mallin, joka ennusti seuraavan merkin yhden edellisen merkin perusteella. Merkeistä luotiin merkkimatriisi ($w_i w_j \in A_1 \times A_2$). Markovin ketjujen yhden tason tilaamuutokset (*single-stage state transitions*) kuvaavat lineaarisen sovittamisen, jonka avulla voidaan laskea todennäköisyys seuraavalle merkille $Q'_{k+1} F' Q_{k,}$ jossa $Q' = [S_{k+1,\ k+i}]$ määrittää hyväksytyt peräkkäiset parit (Q_k, Q_{k+1}) ja $F'=[Pr(S_{k+1,\ k+i} \mid S_{k-j+1,\ t})]$ siirtymän todennäköisyyden.

Tehokkuuslaskujen (kirjoitusnopeuden suhteen) jälkeen he kuitenkin päätyivät malliin, joka ennustaa seuraavan merkin kahden edellisen merkin perusteella. Tällöin osoittautui myös hyödylliseksi muuttaa näytölle ilmestyvää merkkisettiä dynaamisesti, riippuen edellisistä kirjaimista.

Jos ennustamiseen käytettiin vain yhtä edellistä merkkiä, lyheni valintojen määrä 15%. Kun seuraava merkki ennustettiin kahden edellisen perusteella, on keskimääräinen merkin valitsemiseen tarvittava askelten määrä

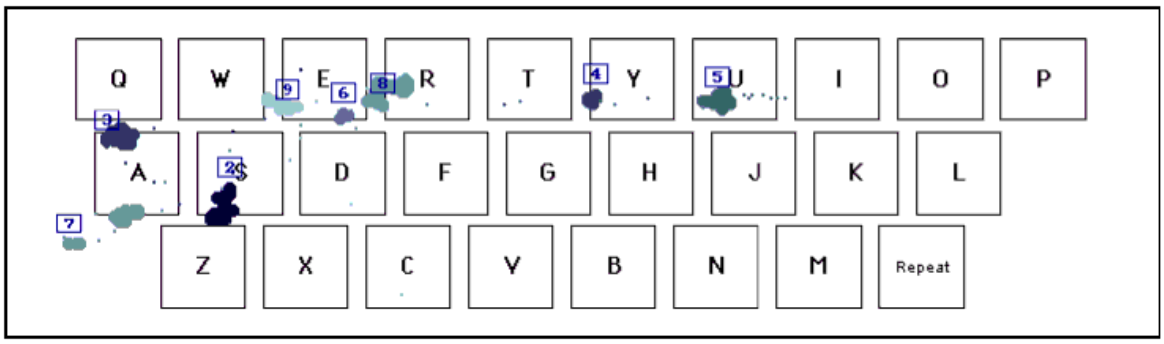
$$E[K] = \sum_{w_i w_j \in A_2 \, X \, A_2} Pr(w_i i \, w_j) \, E[K \mid w_i w_j, \delta],$$ jossa A_2 koostuu merkkipareista ja δ

on 1 jos kyseessä on dynaaminen ennustava valikkorakenne ja $\delta=0$ jos staattinen. Dynaamisen näytön tapauksessa, kun näytöllä olevat merkit ennustettiin kahden edellisen merkin perusteella, E[K] = 1.41, jolloin katsekirjoittamista pystyttiin nopeuttamaan keskimäärin 25% ja 78% merkeistä pystyttiin valitsemaan yhdellä askeleella.

Malli on osin vanhentunut, nykyisin ennustavat kirjoitusjärjestelmät ovat korvanneet sen (esim. sanan lopun arvaaminen, ”*word prediction*”). Ennustavassa tekstinsyötössä on nykyisin yleensä pohjalla sanakirja, systeemi etsii sanat, joiden alku vastaa jo kirjoitettua merkkijonoa. Sanasto tarvitaan, koska systeemi ehdottaa vain oikeita sanoja, eikä pelkästään arvaile mikä merkki tulee seuraavaksi. Mallia soveltamalla ja edelleen kehittämällä sitä voitaneen hyödyntää nykyisinkin. Esimerkiksi kirjainten esiintymistiheyden ja todennäköisyyden käyttäminen voi auttaa mm. siinä, miten kirjaimet sijoitetaan virtuaalinäppäimistöön. Vaikka QWERTY on yleisin tapa järjestää kirjaimet näppäimistölle, se ei välttämättä ole paras. Jos virtuaalinäppäimistön käyttäjä ei ennestään tunne QWERTYä (mikä lienee todennäköistä vakavasti vammaisten tapauksessa), on parempi vaihtoehto käyttää joko aakkosjärjestystä tai järjestää näppäimet siten, että eniten käytetyt ovat helposti saatavilla. Merkkien todennäköisyyksiä voidaan käyttää hyväksi myös silloin, kun päätellään mihin kirjaimeen kiinnitys (fiksaatio) kohdistuu tilanteissa, joissa kiinnitys on merkkien välissä. Mainittu tilanne on varsin tavallinen johtuen katseen ja katseenseurantalaitteiston epätarkkuudesta.

4. Kirjoitetun sanan päätteleminen katsepolusta

Salvucci [1999a] esittelee väitöskirjassaan metodin, jolla pystytään päättelemään mitä ihminen on kirjoittanut tai kirjoittaa (parhaillaan) pelkän katsepolun perusteella. Yleensä katse aktivoi tietyn näppäimen joko siten, että käyttäjä katsoo kohdetta ja aktivoi samanaikaisesti erillisen kytkimen tai siten, että käyttäjä kiinnittää katseensa kohteeseen tietyksi ajaksi. Salvuccin metodi ei vaadi erillisiä kytkimiä eikä katseen pitkää kiinnittymistä. Käyttäjä vain katselee kirjaimia siinä järjestyksessä kuin hän ne kirjoittaisi ja malli päättelee katsepoluista mitä tuli kirjoitettua. Kuvassa 4 ([Salvucci, 1999a]) näkyvät numeroidut fiksaatiokasaumat kohdissa, joihin käyttäjä on katsonut kirjoittaessaan sanan ”SQUARE”.



Kuva 4. Sanaa “SQUARE” vastaavat fiksaatiot

Tyypillisiä ongelmia fiksaatioiden kohdistamisessa (*map*) tiettyihin kirjaimiin ovat mm. fiksaatiot, jotka eivät osu keskelle kohdetta (*off-center fixations*). Tällöin ohjelman pitää osata päätellä, mihin katse kohdistui. Koska katsetta käytetään myös tiedon etsimiseen, kirjaimiin kohdistuu fiksaatioita, jotka aiheutuvat siitä, että käyttäjä käy läpi kirjaimia etsien sitä haluttua. Katse myös toisinaan ”laskeutuu” hieman pieleen, kirjaimen viereen tai väärään kirjaimeen, jolloin katsetta joudutaan korjaamaan. Ihminen pystyy myös hahmottamaan fiksaatioalueen lähellä olevia kohteita kiinnittämättä katsetta niihin, eli ihminen voi kohdistaa huomionsa kiinnityskohdan läheisyydessä (fovean näköalueella) oleviin kohteisiin katsetta liikuttamatta. Näistä johtuen valittavien kohteiden pitää olla riittävän suuria (n. 4 astetta), jotta katseen kohta pystytään määrittämään riittävällä tarkkuudella, ja tarvitaan riittävän pitkä aika (n. 750ms – 1s) kiinnitystä, jotta pystytään erottamaan komennoksi tarkoitettu fiksaatio informaation keräämiseen tarkoitetusta katsomisesta. Salvuccin malli pyrkii vastaamaan näihin ongelmiin. Kuvasta 4 voidaan nähdä, että tehtävä ei ole aivan helppo. Esim. fiksaatio numero 4 osuu kirjaimeen ’y’, jota ei kohdesanassa ”SQUARE” esiinny lainkaan.

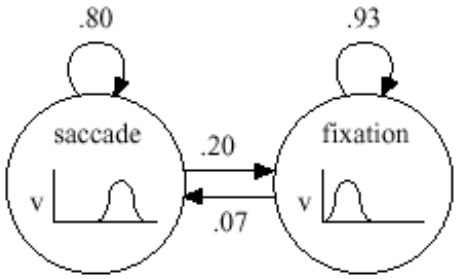
Salvuccin menetelmä perustuu taustalla oleviin sanakirjoihin ja Markovin ketjujen (*Hidden Markov Model*) soveltamiseen. Kun käyttäjä katselee virtuaalinäppäimistöä kirjoittaen katseellaan, malli yrittää sovittaa katsepolun (fiksaatioiden) osoittamia kirjaimia johonkin sanakirjassa olevista sanoista. Fiksaatioita jäljittävä algoritmi käyttää kolmenlaista syötettä: katsedataa (x- ja y-koordinaatit), tietoa kohdealueista (nimi ja paikka) ja prosessimallin kielioppia (*process model grammar*). Kuvassa 5 [Salvucci, 1999a] on hyvin yksinkertaistettu malli kieliopista kolmelle sanalle.

Rule			Probability
start	→	**word** *rat*	.33
start	→	**word** *trap*	.33
start	→	**word** *part*	.33
rat	→	**R A T** *end*	1.00
trap	→	**T R A P** *end*	1.00
part	→	**P A R T** *end*	1.00
end	→	**out**	1.00

Kuva 5. Yksinkertainen kielioppi sanoille {RAT, TRAP, PART}

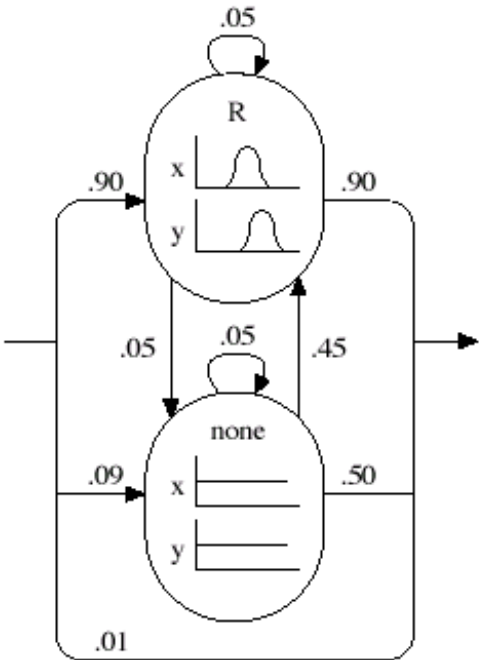
Salvucci esittää useita mahdollisia ratkaisuja. Sana voidaan yrittää käsitellä kokonaisena, eli kokeilla mikä sana parhaiten vastaa katsepolkua (kuten kuvassa 5). Ohjelman paikkatieto sisältää tiedon siitä, milloin ollaan kirjoitusalueella ja milloin katsotaan jotain muuta kohdetta (esimerkiksi tekstikenttää, johon kirjoitettu sana ilmestyy). Kun saavutaan kirjoitusalueelle, käynnistyy aloituskomento ”*start*” ja jos poistutaan alueelta suoritetaan lopetuskomento ”*end*”. Aloitus– ja lopetuskomentojen välissä yritetään sovittaa katsepolun tuottama data kielioppiin.

Samanaikaisesti ohjelma seuraa fiksaatioita ja sakkadeja ja pyrkii Markovin ketjujen (HMM) avulla päättelemään tapahtuiko fiksaatio vai sakkadi (kuva 6) ja missä fiksaatiokasauman keskikohta sijaitsee (kuva 7 [Salvucci, 1999b]). Jälkimmäinen käyttää hyväkseen paikkatietoa.



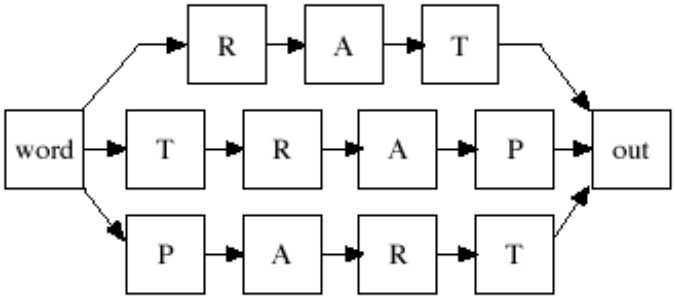
Kuva 6. HMM fiksaatioiden ja sakkadien tunnistus (Velocity HMM)

HMM:ssä on kaksi tilaa, joista toinen edustaa sakkadia ja toinen fiksaatiota. Nopeat siirtymät koordinaateissa tulkitaan sakkadeiksi (*high-velocity saccadic movement*) ja hitaammat, paikallaan viipyvät fiksaatioiksi.



Kuva 7. Keskikohdan (*centroid*) määritys (ala)malli kirjaimelle **R**.

Salvuccin malli siis pyrkii samanaikaisesti jäljittämään sen, mihin fiksaatiot osuvat ja pyrkii sovittamaan löydetyt fiksaatiot kielioppiin, joka sisältää käytettävissä olevan sanaston (kuva 8 [Salvucci, 1999b]).



Kuva 8. Esimerkki katse-kirjoitus jäljitys-mallista (*tracer model*)

Salvucci kehitti mallista kolme eri versiota ja käytti vertailukohtana yksinkertaista mallia, joka pelkästään sovitti kunkin fiksaation lähimpään kirjaimeen:

- Täydellinen (”*Full*”): joka (kokonainen) sana erikseen
- 2-järjestys (”*2nd -order*”): kuvaa siirtymät kirjainparista kirjaimeen
- 1-järjestys (”*1st -order*”): kuvaa siirtymät kirjaimesta kirjaimeen
- Yksinkertainen (”*Simple*”): sovittaa fiksaatiot lähimpään kirjaimeen

Täydellinen malli sisältää kaiken tiedon mahdollisista toimintosarjoista, kun taas yksinkertainen malli ei kerro mitään tietoa toimintosarjoista. 1-järjestys ja 2-järjestys mallit ovat jotain edellisten väliltä. Kuvassa 9 vasemmalla näkyy

1-järjestys kielioppi kirjaimesta toiseen ja oikealla 2-järjestys kirjainparista kirjaimeen.

start	→	**word** *p*	.33
start	→	**word** *r*	.33
start	→	**word** *t*	.33
a	→	**A** *p*	.33
a	→	**A** *r*	.33
a	→	**A** *t*	.33
p	→	**P** *a*	.50
p	→	**P** *end*	.50
r	→	**R** *a*	.67
r	→	**R** *t*	.33
t	→	**T** *r*	.33
t	→	**T** *end*	.67
end	→	**out**	1.00

start	→	**word** *p*	.33
start	→	**word** *r*	.33
start	→	**word** *t*	.33
p	→	**P** *pa*	1.00
r	→	**R** *ra*	1.00
t	→	**T** *tr*	1.00
ap	→	**P** *end*	1.00
ar	→	**R** *rt*	1.00
at	→	**T** *end*	1.00
pa	→	**A** *ar*	1.00
ra	→	**A** *ap*	.50
ra	→	**A** *at*	.50
rt	→	**T** *end*	1.00
tr	→	**R** *ra*	1.00
end	→	**out**	1.00

Kuva 9. Vasemmalla 1- ja oikealla 2-järjestys kielioppi {RAT, TRAP, PART}

Salvucci testasi mallejaan erikokoisilla sanastoilla (12, 100 ja 1000 sanaa), testaten sekä tarkkuutta että aikavaativuutta. HMM laskeminen vie aikaa, ja mitä enemmän tietoa pitää prosessoida, sitä kauemmin laskeminen kestää. Kuten kuvasta 10 näkyy, täydellisellä mallilla saavutetaan paras tarkkuus, mutta toisaalta 1- ja 2-järjestysmalleilla päästään nopeampiin vasteaikoihin.

Words	Model			
	Full	2nd-order	1st-order	Simple
12	.997	.98	.94	.90
100	.98	.91	.87	.80
1000	.92	.80	.76	.74

Interpretation accuracy.

Words	Model			
	Full	2nd-order	1st-order	Simple
12	142	119	81	55
100	891	602	203	124
1000	9276	3701	834	777

Interpretation times, in milliseconds.

Kuva 10. Mallien tulkintatarkkuus (yllä) ja aikavaatimus (alla)

Käytännön testi osoitti, että mallissa saavutetaan kirjoitusnopeus:

- keskimäärin 822 ms / merkki (kaikki käyttäjät)
- nopeimman kirjoittajan keskiarvo 430 ms / merkki
- hitaimman kirjoittajan keskiarvo 1272 ms / merkki

Luvut ovat selvästi nopeampia kuin aiemmin raportoidut noin sekunnin tai yli kestävät katsekirjoitusajat.

Salvucci uskoo, että algoritmeja kehittämällä on mahdollista päästä parempiin tuloksiin, ja nopeuttaa reaaliaikaista laskemista. Tämä onkin mielestäni erityisen tärkeää, koska sanaston on oltava suuri, jotta malli käytännössä toimisi. Tuskin koskaan kaikki mahdolliset sanat löytyvät sanakirjasta. Voisi olettaa, että parhaiten toimii malli, jossa käytetään hyväksi tietoa sanojen rakenteista ja esimerkiksi suomen kielessä taivutuksista. Sanaston tulisi myös mielestäni kasvaa käyttäjänsä mukana, aina tulisi olla mahdollisuus lisätä sanastoon uusia sanoja. Tämä vaatii sen, että merkkejä (ja siten sanoja) on mahdollista tarvittaessa syöttää ”ohi” valmiin kieliopin ja sanaston.

5. Lopuksi

Katseella kirjoittaminen on olennaisesti katseella valitsemista. Katseella valitsemiseen liittyvät mallit siis pätisivät myös katsekirjoittamiseen. Katse on ilmeisesti syöttövälineenä vielä liian nuori, koska katsesyötteeseen liittyviä malleja en löytänyt.

Tekstinsyöttöön piirtimen (*stylus*) avulla kehitetyt mallit voisivat toimia pohjana katsesyötteessä käytettäville malleille. Esimerkiksi Soukoreff ja Mackenzien [1995] kehittämä malli ennustaa ala- ja ylärajoja ajalle, joka kuluu tekstin syöttämiseen. Malli soveltaa Hick-Hymanin lakia ennustamaan valintaan kuluvaa reaktioaikaa, Fittsin lakia liikkeen kohdistamiseen kuluvaa aikaa, ja käyttää myös hyödyksi tietoa kirjainten (ja kirjainparien) esiintymistiheydestä englannin kielessä. Mallia ei voi käyttää suoraan, koska katsesyötteessä kohteen etsiminen ja kohdistaminen ovat käytännössä sama asia. Piirtimen (käden lihasliikkeellä) siirtämiseen kuluva aika jää pois, mutta toisaalta tilalle tulee kiinnitysaika, jota katsekäyttöliittymissä käytetään valinnan vahvistamiseen. Mallia sopivasti muokkaamalla se voisi mielestäni toimia myös katsesyötteessä.

Monet vammaisten kommunikaatioapuvälineisiin kehitetyt ja yleiset tekstinsyöttöön kehitetyt mallit sopisivat luultavasti sovellettuina ja/tai ehkä hieman muunnettuina myös katsesyötteessä käytettäviksi. Shein [1997] esimerkiksi on kehittänyt askellusta käyttävän mallin, jonka avulla voidaan ennustaa käyttäjän suorituskykyä, kun tavoitteena on liikkua editoitavassa tekstissä (esim. riviltä toiselle, tai sanasta toiseen) ja valita osa tekstistä. Mallin

avulla voidaan laskea vaaditut askeleet ja siihen kuluva aika. Sheinin malli sopisi mielestäni hyvin katsekirjoituksessa käytettäväksi, koska se kohdistuu ensisijaisesti tehtävään, ei käytettävään syöttölaitteeseen. Malli kuitenkin olettaa, että syöttölaitetta käytetään kytkimenä askelluksessa. Askellus tarkoittaa sitä, että siirrytään kohteesta toiseen askelittain esim. rivi riviltä ja sana sanalta, ja kytkintä käytetään fokuksessa olevan kohteen valitsemiseen. Mallista ei siis ole hyötyä, jos katsetta käytetään suoraan valitsemiseen, eli valinta tapahtuu katsomalla valittavaa kohdetta. En ryhdy tässä mallia tarkemmin käsittelemään, koska sitä ei ole katsesyötteellä testattu ja koska se sopii vain niihin erikoistilanteisiin, joissa ihminen ei pysty kohdistamaan katsettaan vaan käyttää katsetta vain kytkimenä (esim. katsominen vasemmalle käynnistää askelluksen ja katsominen oikealle pysäyttää askelluksen). Toisaalta tekstipätkän valitseminen suoraan katseella saattaa olla vaikeaa, joten ainakin jonkinasteinen askellus saattaa olla tarpeen.

Varmasti löytyy muitakin kirjoittamiseen liittyviä malleja, joilla on yhteyksiä katsesyötteeseen, ja joista voitaisiin kehittää johdannaisia nimenomaan katsesyötettä tai katseella kirjoittamista varten. Lopuksi voidaan todeta, että katsesyötteen ja katsekirjoituksen alueella on vielä paljon tehtävää, sekä mallien että yleisten teoreettisen pohjan kehittelyssä.

Viiteluettelo

[Frey, et al., 1990] Frey, L.A, White, K. P. JR., Hutchinson, T. E. Eye-Gaze Word Processing, *IEEE Transactions on Systems, Man, and Cybernetics, vol 20, 4*, 1990, 944-950.

[Friedman, et al., 1985] Friedman, M.B., Killiany, G. and Dzmura, M. An Eye the target is already being fixated. Gaze Controlled Keyboard. *Proceedings of 2nd International Conference on Rehabilitation Engineer*, Ottawa, 1985.

[Hutchinson, *et al.*, 1989] Hutchinson, T.E., White, K.P., Martin, W.N., Reichert, K.C. and Frey, L.A. Human-Computer Interaction Using Eye-Gaze Input. *IEEE Transactions on Systems, Man, and Cybernetics, 19 (6)*, 1989, 1527-1534.

[Jakob, 1991] Jacob, R.J.K. The use of eye movements in human-computer interaction techniques: what you look at is what you get, in ACM Transactions on Information Systems, Vol. 9, No. 3, April 1991, 152-169.

[Jacob, 1995] Jacob, R.J.K. Eye Tracking in Advanced Interface Design. W. Barfield ja T.A. Furness (toim.), *Virtual Environments and Advanced Interface Design*, Oxford University Press, New York, 1995, 258-288. http://www.eecs.tufts.edu/~jacob/papers/barfield.html (8.6.2001)

[Monty ja Senders, 1976] Monty, R.A. and Senders, J.W., Eye Movements and Psychological Processes. Lawrence Erlbaum, Hillsdale, N.J., 1976.

[Salvucci, 1999b] Salvucci, D. D. Inferring intent in eye-movement interfaces: Tracing user actions with process models. *Human Factors in Computing Systems: CHI 99 Conference Proceedings*, New York: ACM Press, 1999, 254-261. http://www.cbr.com/~dario/publications.html (14.5.2001)

[Salvucci, 1999a] Salvucci, D. D. (1999). Mapping eye movements to cognitive processes (Tech. Rep. No. CMU-CS-99-131). Doctoral Dissertation, Department of Computer Science, Carnegie Mellon University. *http://www.cbr.com/~dario/publications.html* (14.5.2001)

[Shein, 1997] Shein, F. Towards Task Transparency In Alternative Computer Access: Selection Of Text Through Switch-Based Scanning. Ph.D. Thesis, Dept. Industrial Engineering, University of Toronto, 1997. Available at http://www.wivik.com/documents/wiv97d1.pdf (14.5.2001)

[Soukoreff ja Mackenzie, 1995] Soukoreff, W. And Mackenzie, I. S. Theoretical upper and lower bounds on typing speed using a stylus and soft keyboard. *Behaviour & Information Technology*, *14*, 1995, 370-379.

Dimensionaaliset emootiot ihminen-tietokone-vuorovaikutuksessa

Timo Partala

Kirjoitus käsittelee ihmisen ja tietokoneen vuorovaikutuksen tutkimusta dimensionaalisten emootioiden näkökulmasta. Dimensionaalisten emootioiden näkökulmassa ihmisen tunteita tarkastellaan yleisimmin kolmella dimensiolla, jotka ovat valenssi, vireystila ja dominanssi. Näkökulmaan liittyy läheisesti semanttisen differentiaalin menetelmä, jonka avulla voidaan arvioida tunnekokemuksia. Nämä yhdessä näyttävät tarjoavan lupaavan lähtökohdan ihminen-tietokone vuorovaikutuksen laajentamiseksi siten, että myös vuorovaikutuksen tunnetaso otetaan huomioon. Erityisesti sovellusmahdollisuuksia on vuorovaikutuksen arvioinnissa ja tunnepitoisen syötteen tulkinnassa.

1. Johdanto

Ihmisen ja tietokoneen vuorovaikutus on noussut 1990-luvulla maailmanlaajuisesti merkittäväksi tutkimusalueeksi. On mm. kehitetty uudenlaisia vuorovaikutustekniikoita, joissa ihmisen aisteja käytetään hyväksi entistä paremmin. Myös vuorovaikutuksen arviointi on kehittynyt, erityisesti ohjelmistojen käytettävyyden arviointi on ollut suuren huomion kohteena.

Ihmisen ja tietokoneen välistä vuorovaikutusta on perinteisesti tarkasteltu kognitiivisella eli tiedollisella tasolla. Systeeminsuunnittelun lähtökohdaksi on usein otettu tehtävän analysointi juuri kognitiivisella tasolla, ja myös arvointimenetelminä on käytetty useimmiten kognitiivisen psykologian lähtökohdista alkunsa saaneita menetelmiä kuten GOMS ja erilaiset kognitiiviset läpikäynnit [Butler, 1996].

Viime vuosina on kuitenkin kiinnitetty entistä enemmän huomiota emootioiden (tunteiden) merkitykseen ihminen-tietokone vuorovaikutuksessa. Emootiot ovat keskeinen osa ihmisten välistä vuorovaikutusta. Uusien kanavien käyttöönotto sekä syöte- että tulostepuolella saa aikaan sen, että ihminen-tietokone vuorovaikutus muistuttaa entistä enemmän ihmisten keskinäistä vuorovaikutusta. Tällöin myös tunteiden ja tunteita ilmaisevan kommunikaation (esim. kasvonilmeiden) huomioon ottaminen ihminen-tietokone vuorovaikutuksessa tulee ajankohtaiseksi.

Tämä kirjoitus käsittelee tunteiden huomioon ottamista ihminen-tietokone vuorovaikutuksessa. Aihetta käsitellään systemaattisesti yhden tietyn teorian kannalta: Peter Langin [Bradley ja Lang, 1994] tunnetuksi tekemän dimensionaalisten emootioiden näkökulman kannalta. Tässä näkökulmassa emootioiden ajatellaan koostuvan kolmesta dimensiosta (ulottuvuudesta): valenssista, vireystilasta ja dominanssista. Nämä kolme dimensiota muodostavat kolmiulotteisen avaruuden, johon erilaiset tunnekokemukset sijoittuvat.

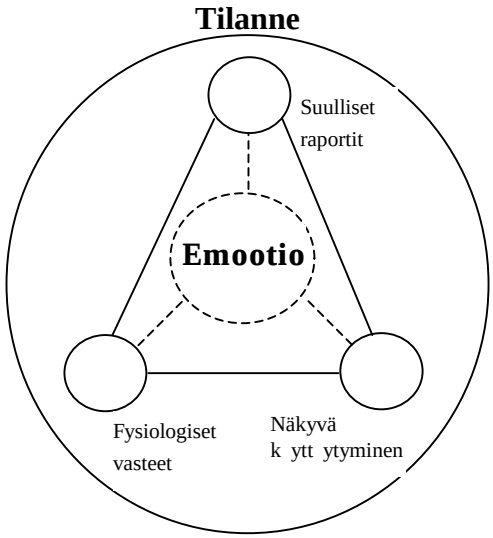
Kirjoitus on jäsennetty seuraavasti. Ensin emootion käsite ja emootioon liittyvät erilaiset ilmenemistasot määritellään luvussa 2. Sen jälkeen luvussa 3 esitellään dimensionaalisten emootioiden näkökulma ja vertaillaan sitä toiseen tunnettuun näkökulmaan: erillisten emootioiden näkökulmaan. Luvussa 4 käsitellään dimensionaalisiin emootioihin kiinteästi liittyvää semanttisen differentiaalin menetelmää, ja luvussa 5 käsitellään dimensionaalisiin emootioihin liittyviä fysiologisia suureita. Luvussa 6 pohditaan dimensionaalisten emootioiden näkökulman mahdollisia sovelluksia ihminen-tietokone vuorovaikutuksessa. Lopuksi luvun 7 yhteenveto kokoaa käsitellyn aineksen yhteen, ja kirjoittaja esittää oman näkemyksensä käsiteltyihin asioihin.

2. Emootiot ja emootion tasot

Emootioita (tunteita) on vaikea määritellä, eikä yleisesti hyväksyttyä yksikäsitteistä määritelmää olekaan olemassa. Yleensä emootioiden katsotaan kuitenkin olevan lyhytkestoisia reaktioita. Tämä erottaa ne mielialoista, jotka voivat kestää jopa viikkoja. Emotionaalisiin reaktioihin liittyvät useimmiten seuraavat komponentit [Izard, 1977]:

- muutos neurofysiologisessa toiminnassa eri aivoalueilla
- muutos kasvojen ja kehon lihasten neuromuskulaarisessa toiminnassa (lihasjännityksissä)
- muutos käyttäytymisessä (esim. lähestyminen / välttäminen)
- subjektiivinen tunnekokemus

Öhman [Öhman ja Birbaumer, 1993] on Langin [1993] edeltävän tutkimuksen pohjalta esittänyt mallin, joka kuvaa emootion rakennetta (kuvio 1). Öhmanin koostamassa mallissa emotionaaliset ilmiöt aiheuttaa jokin tunnetasolla merkityksekäs tilanne. Emotionaaliset ilmiöt ilmenevät kolmella tavalla: suullisissa raporteissa, fysiologisissa vasteissa ja käyttäytymisessä. Nämä ilmenemistavat ovat keskeisiä myös emootioiden tutkimuksessa.



Kuvio 1. Emootion rakenne.

Emootioiden tutkimuksessa suuntaus onkin ollut se, että yritetään tutkia tunneperäisiä ilmiöitä sen sijaan että yritettäisiin määritellä merkitystä termille emootio. Määritelmän puuttumista ei nähdä siis esteenä onnistuneelle tutkimukselle. Sitä vastoin tunneperäisten ilmiöiden tutkimuksen tuloksena voidaan saada realistinen kuva siitä, mitä emootio on, ja kuinka se voi ilmetä.

3. Dimensionaaliset vs. erilliset emootiot

Emootiota koskevat teoriat jakautuvat korkealla abstraktiotasolla kahteen eri ryhmään: toisessa ryhmässä oletetaan, että tunnekokemukset voidaan esittää jatkuvina dimensioina (ulottuvuuksina), ja toisessa ryhmässä oletetaan, että tunteet muodostavat erillisiä kategorioita kuten pelko, viha, ilo jne. Suurin osa tutkijoista näkee kuitenkin nykyään nämä kaksi lähestymistapaa toisiaan täydentävinä, jolloin ajatellaan, että erilliset emootiot sijoittuvat tietyllä tavalla niihin dimensioihin, jotka dimensionaalisessa lähestymistavassa voidaan määritellä. Käytännössä emootiotutkimuksessa kuitenkin usein valitaan vain toinen lähestymistapa yhtä tiettyä tutkimusta tehtäessä. Tässä tutkielmassa keskitytään dimensionaaliseen lähestymistapaan, mutta seuraavassa käydään läpi lyhyesti myös erillisten emootioiden lähestymistapa, jotta lukija saisi paremman kokonaisnäkemyksen aiheesta. Tämän jälkeen käydään läpi dimensionaalisten emootioiden lähestymistavan perusajatukset.

3.1 Erilliset emootiot

Erillisten emootioden lähestymistavoille yhteistä on se, että niissä jaetaan tunteita erilaisiin kategorioihin [Izard, 1977]. Näiden kategorioiden nimitykset (esim. pelko) ovat pääasiassa vakiintuneet arkipäivän kielenkäyttöön jo kauan ennen nykyaikaista emootiotutkimusta.

Erillisiä emootioita on ehdotettu paljon erilaisia, ja useimpien teoreetikkojen näkemyksiin sisältyy ajatus, että on olemassa joukko emootioita (n. 2-10), jotka ovat enemmän perustavaa laatua olevia (basic, fundamental, primary) kuin muut emootiot. Tätä jakoa perustellaan yleensä fysiologiaan pohjautuen: perusemootiot ovat kiinteässä yhteydessä (hard-wired) autonomiseen hermostoon, jolla on keskeinen rooli siinä, kuinka emootiot viriävät kehossa.

Yksi tunnetuimmista erillisten emootioiden teorioista on Ekmanin [1992] teoria, jonka mukaan on olemassa kuusi perusemootiota: viha, inho, pelko, ilo, suru ja yllätys. Nämä on valittu perusemootioksi siksi, että jokaista vastaa oma yleismaailmallisesti tunnistettava kasvonilmeensä.

3.2 Dimensionaaliset emootiot

Dimensionaalisten emootioiden näkökulmassa määritellään dimensiot, jotka ovat tunnekokemuksien takana, ja sen jälkeen sijoitetaan tunnekokemukset näille dimensioille. Dimensionaalisten emootioiden näkökulma välttää ongelmat, jotka syntyvät siitä, että eri emootioiden nimitykset ovat ajan kuluessa vakiintuneet arkikielenkäyttöön ja sitä kautta tutkimukseen. Dimensionaalisilla asteikoilla pystytään mm. tarkempaan tunnekokemuksen

arviointiin, kun arvioijan ei tarvitse noudattaa muutaman vakiintuneen emootiokategorian rajoja.

Nykyisin dimensionaalisista näkökulmista käytetyin on Langin ja kumppaneiden näkökulma [Bradley and Lang, 1994], jossa tunteiden nähdään sijoittuvan seuraaville dimensioille:

- valenssi, jonka ääripäät ovat erittäin negatiivinen ja erittäin positiivinen tunnekokemus
- vireystila, jonka ääripäät ovat erittäin rauhallinen ja erittäin kiihtynyt tunnekokemus
- dominanssi, jonka ääripäät liittyvät hallinnan tunteeseen: esim. ’tilanne hallitsee minua erittäin paljon’ ja ’minä hallitsen tilannetta erittäin paljon’.

Näistä ensimmäiset kaksi ulottuvuutta ovat useimmiten käytetyt dimensiot. Ääripäiden puoliväliin jäävää keskimääräistä tunnekokemusta kutsutaan kaikilla ulottuvuuksilla neutraaliksi.

Dimensioista valenssilla ja vireystilalla on vankka fysiologinen pohja [Bradley et. al., 1993]. Valenssi kuvastaa jomman kumman aivojen päämotivaatiosysteemin, appetitiivisen tai aversiivisen, läsnäoloa. Näillä on kiinteä yhteys käyttäytymiseen lähestyminen-välttäminen –skaalalla. Esim. ihmisten välisessä vuorovaikutuksessa appetitiivisen systeemin viriäminen aktiiviseksi saa aikaan (useimmiten) lähestymiskäyttäytymistä, aversiivisen systeemin aktivoituminen taas saa aikaan välttämiskäyttäytymistä. Vireystila puolestaan kuvastaa joko appetitiivisen tai aversiivisen systeemin intensiteettiä, eli sitä, kuinka voimakkaita tunnekokemukset ovat. Vireystila on myös kiinteässä yhteydessä kehollisiin tekijöihin: kiihtyneessä tilassa mm. sydämen lyöntitiheys on nopea ja lihakset ovat jännittyneet.

4. Dimensionaaliset emootiot ja semanttinen differentiaali

Dimensionaalisten emootioiden lähestymistapaan liittyy läheisesti semanttisen differentiaalin menetelmä. Empiirinen tutkimus on osoittanut, että ihmisten arviot tunnepitoisista ärsykkeistä voidaan järjestää kolmella perusdimensiolle. Tämän esitti ensimmäisenä Wundt vuonna 1896, faktorianalyyseillä varmensi Osgood vuonna 1952, ja näiden pohjalta semanttisen differentiaalin menetelmän ottivat käyttöön Mehrabian ja Russell vuonna 1974 [Bradley ja Lang, 1994]. Semanttisen differentiaalin menetelmän avulla ihmiset voivat arvoida esim. erilaisten tapahtumien, esineiden tai sanojen tunnepitoisia merkityksiä kolmella dimensiolla. Dimensioiden nimet ovat historian saatossa muuttuneet, mutta merkitykset ovat pysyneet lähes samoina. Nykyään tunnetuin on edellämainittu Langin ja kumppaneiden versio, jossa valenssi, vireystila ja dominanssi ovat dimensiot, joille tunnekokemus voidaan järjestää,

ja semanttinen differentiaali voi olla käytännössä arviointilomakkeiden suunnittelun pohjana.

Semanttisen differentiaalin menetelmässä koehenkilöt arvioivat kokemuksiaan ärsykkeisiin (esim. tilanteita, kuvia, ääniä) kaksipäisillä skaaloilla, joiden päissä ovat merkityksiltään vastakohtaiset adjektiivit (esim. onneton – onnellinen, rauhallinen – kiihtynyt). Usein käytössä on esim. 9-pisteinen arviontiskaala, jolloin on mahdollista antaa myös neutraali arviointi valitsemalla asteikon keskimmäinen arvosana.

Semanttisen differentiaalin pohjalta on myös kehitetty SAM-menetelmä [Bradley ja Lang, 1994]. SAM-menetelmässä mainittujen kolmen dimension asteikot kuvataan siten, että jokaista asteikon pistettä edustaa graafinen hahmo, joka kuvaa pisteen edustamaan tunnetilaa. Esim. erittäin positiivista tunnetilaa valenssiskaalalla edustaa iloisesti hymyilevä piirretty hahmo. SAM-menetelmän etuna on se, että se on kulttuurista ja kielestä riippumaton (olettaen, että piirroshahmot tunnistetaan samalla tavalla eri kulttuureissa).

5. Dimensionaaliset emootiot ja fysiologiset signaalit

Emootiotutkimukseen liittyvät läheisesti erilaiset fysiologiset signaalit. Myös jos tietoa käyttäjän tunteista halutaan välittää ihmiseltä tietokoneelle, toimii fysiologisten signaalien mittaus kanavana ihmisen ja tietokoneen välillä. Teknologiaa, joka käyttää hyväkseen fysiologisia mittauksia ihmisen tunnetilan määrittämiseksi, voidaan kutsua nimellä *affective computing* (suomeksi esim. tunneperäinen tietojenkäsittely) [Picard, 1997]. On kehitetty esimerkiksi tietokoneopettajia, jotka ottavat oppijan tunteet huomioon ja erilaisia laitteita (esim. koruja ja leluja), jotka mukautuvat käyttäjänsä tunnetilaan.

On olemassa useita fysiologisia mittaussuureita, jotka antavat tietoa ihmisen tunnetilasta. Voidaan mitata esim. aivojen sähköistä aktiviteettia EEG-tekniikalla (elektroenkefalografia) [Surakka *et al.*, 1998] tai kasvolihasten sähköistä aktiviteettia EMG-tekniikalla (elektromyografia) [Partala *et al.*, 2001]. Voidaan myös mitata esim. ihon sähköjohtokykyä, verenpainetta, sydämen sykettä tai vaikkapa pupillin kokoa.

Olemassaolevien tutkimusten mukaan valenssilla on selvä yhteys ainakin kasvolihasten sähköiseen aktiviteettiin ja sydämen sykkeeseen. Kun valenssi kasvaa positiivisemmaksi, *corrugator supercilii*-lihaksen (kulmien kurtistajalihas) aktiviteetti pienenee ja *zygomaticus major*-lihaksen (lihas, joka vetää suupielet hymyyn) aktiviteetti kasvaa [Bradley *et al.*, 1993; Ekman ja Friesen, 1978]. Vireystilan kasvu puolestaan näkyy ainakin ihon sähkönjohtavuuden kasvuna [Bradley *et al.*, 1993] ja pupillin koon kasvuna [Partala *et al.*, 2000].

Fysiologiset signaalit tarjoavat mahdollisuuden päästä käsiksi tietokoneen käyttäjän tunteisiin objektiivisesti mitattavissa olevien suureiden välityksellä.

Tällöin voidaan myös havainnoida sellaisia kehollisia tunnereaktioita, joiden olemassaoloa käyttäjä ei itse välttämättä edes huomaa. Emootioiden tunnistaminen fysiologisesta datasta vaatii paljon perustutkimusta fysiologisesten ilmiöiden käyttäytymisestä ja uusia signaalinkäsittelymenetelmiä. Näillä menetelmillä on tietokoneen mahdollista muodostaa käsitys käyttäjän tunnetilasta lähes reaaliaikaisesti tietokoneen käytön aikana.

6. Dimensionaaliset emootiot ihminen-tietokone vuorovaikutuksessa

Ihmisen ja tietokoneen vuorovaikutusta on tarkasteltu tunnetason tekijät huomioon ottaen vain melko vähän. Aluksi keskityttiin lähinnä systeeminsuunnitteluun (miten pystytään suunnittelemaan ja rakentamaan tietynlainen järjestelmä). Sittemmin ihmisen ja tietokoneen vuorovaikutus on noussut omaksi tutkimusalueekseen, mutta tutkimus on käsittellyt ihmistä enimmäkseen tiedollisesta (kognitiivisesta) näkökulmasta. Vasta viime vuosina on alettu kiinnostua siitä, miten tunteet ovat mukana ihmisen ja tietokoneen vuorovaikutuksessa.

Seuraavassa otetaan käyttöön ylhäältä alaspäin –lähestymistapa: kartoitetaan korkealla abstraktiotasolla informaatioteknologian tutkimuksen tärkeimmät aktiviteetit ja tuotokset sekä arvioidaan tapoja, joilla (dimensionaaliset) emootiot vaikuttavat kuhunkin aktiviteettiin ja tuotokseen. Informaatioteknologian tutkimusta tarkastellaan Marchin ja Smithin [1995] esittämän viitekehyksen kautta (kuvio 2). Malli jakaa sekä tutkimusaktiviteetit että tutkimustuotokset neljään luokkaan. Mallin mukaan kaikki informaatioteknologian tutkimus sijoittuu yhteen tai useampaan kuudestatoista ruudusta, jotka saadaan yhdistämällä tutkimusaktiviteetit ja –tuotokset. Esim. yksi ruutu sisältää tutkimuksen, jossa arvioidaan menetelmiä.

	Rakenna	**Arvioi**	**Teorisoi**	**Perustele**
Käsitteistö				
Malli				
Menetelmä				
Toteutus				

Kuvio 2. Informaatioteknologian tutkimusaktiviteetit ja –tuotokset

March ja Smith [1997] jakavat tutkimustuotokset neljään luokkaan: käsitteistöön, malleihin, menetelmiin ja toteutuksiin. Seuraavassa arvioidaan

miten dimensionaalisten emootioiden lähestymistapa voisi muuttaa ihmisen ja tietokoneen vuorovaikutuksen tutkimustuotoksia.

Käsitteet muodostavat tutkimusalueen sanaston. Käsitteistö näyttelee erittäin tärkeää osaa tutkimuksessa, koska se määrittelee termit, joilla ilmiöitä voidaan ajatella ja kuvata. Perinteisesti ihmisen ja tietokoneen on liittynyt jonkin verran sanastoa, joka liittyy jollakin tavalla emootioihin (esim. miellyttävyys), mutta käsitteistö on silti eronnut merkitsevästi esim. psykologian tunteiden tutkimuksen käsitteistöstä. Dimensionaalisten emootioiden lähestymistavan tuominen ihmisen ja tietokoneen vuorovaikutuksen käsitteistöön toisi mukaan erityisesti päädimensiot käsitteinä: valenssin, vireystilan ja dominanssin.

Malli on joukko väittämiä, jotka kuvaavat käsitteiden välisiä suhteita. Malli voi kuvata väitelauseilla ongelmaa ja ratkaisua (suunnittelutiede) tai olla kuvaus todellisuudesta (luonnontiede). Dimensionaaliset emootiot voivat liittyä näistä molempiin. Ne voivat olla osana suunnittelutieteen ratkaisua (esim. affective computing). Erityisesti ne voivat kuitenkin muuttaa sitä kuvaa, joka tutkimusmalleissa on ollut ihmisestä tietokoneen käyttäjänä. Ihminen on nähty useimmiten kognitiivisena tiedonkäsittelijänä, joka toimii täysin loogisesti. Tätä kuvaa voisi täydentää ottamalla huomioon ihmisen tunnetila ja tunteisiin liittyvä käyttäytyminen.

Menetelmä on joukko askelia, jotka otetaan ongelman ratkaisemiseksi, esimerkiksi algoritmit ja erilaiset suuntaviivat (guidelines) voidaan laskea menetelmiksi. Menetelmät pohjautuvat aihealueen käsitteistöön ja malliin, joka kuvaa ongelma-avaruutta. Käytännössä suunnittelutiede luo sekä suunnittelutieteissä että luonnontieteissä käytetyt menetelmät. Dimensionaalisten emootioiden lähestymistapa voisi tuoda mukanaan uusia menetelmiä. Rakennuspuolella olisi mahdollista luoda uusia menetelmiä esim. käyttäjän tunnetilan päättelyyn ja tunnistamiseen esim. fysiologisista signaaleista. Arviointipuolella voisi käyttää esim. edellä esitettyyn semanttiseen differentiaaliin pohjautuvia menetelmiä.

Tuotos (instantiation) on toteutettu artefakti käyttöympäristössänsä. Tuotokset siirtävät käsitteet, mallit ja menetelmät käytännön tuotoksiksi tai niiden osiksi. Tuotoksia on perinteisesti pidetty tietojenkäsittelytieteissä suuressa arvossa. Jos tuotoksien affektiiviset vaikutukset otetaan suunnitteluvaiheessa huomioon, muuttaa se todennäköisesti tuotosta ainakin käyttöliittymän osalta. Oma lukunsa ovat tietysti tuotokset, joissa on toteutettu mahdollisuus ottaa käyttäjän tunnetila huomioon vuorovaikutuksessa. Tällöin se, että kone arvioisi käyttäjän tunnetilan arvioiminen esim. dimensionaalisten emootioiden viitekehystä käyttäen vaikuttaisi vuorovaikutteisesti siihen, miten itse järjestelmä toimii.

Seuraavassa arvioidaan Marchin ja Smithin [1995] tutkimusaktiviteetteja yksitellen suhteessa emootioihin. Suunnittelutieteissä tutkimusaktiviteetit liittyvät artefaktien rakentamiseen ja arviointiin, luonnontieteissä puolestaan teorisointiin ja perusteluun liittyen olemassaolevaan todellisuuteen.

Rakennusaktiviteetti viittaa artefaktin rakentamiseen, jolloin osoitetaan, että artefaktin rakentaminen on mahdollista. Dimensionaalisia emootioita voisi käyttää osana käyttäjämallia (tietokoneen muodostamaa kuvaa käyttäjästä), jolloin ne muodostaisivat mallin käyttäjän tunnetilasta. Mahdollisesti sitä voisi käyttää myös tietokoneen emootioiden mallina, osana tunteiden syntetisointia, jolloin tietokone voisi myös ilmaista keinotekoisia tunteita käyttäjälle. Dimensionaalisten emootioiden malli on määrällinen, mikä helpottaa sen toteuttamista tietokoneelle esim. verrattuna erillisiin emootioihin, jotka ovat erillisiä laadullisia kategorioita.

Arviointiaktiviteetissa arvioidaan suunniteltua ja toteutettua tuotosta: kuinka hyvin se toimii? Tällöin keskeinen merkitys on arvointikriteereillä: mitä halutaan saavuttaa? Yleisesti arviointikriteereinä on pidetty esim. tehtäväaikoja, virhemääriä ja muita objektiivisiä suureita. Usein on myös arvioitu järjestelmän miellyttävyyttä, monesti nimikkeellä subjektiivinen tyytyväisyys [Frokjaer *et al.*, 2000].

Dimensionaaliset emootiot voivat antaa arviointiin uuden näkökulman edellä mainittujen lisäksi. Käyttäjän tunnekokemuksen ja subjektiivisen tyytyväisyyden arvioinneilla on yksi perustavaa laatua oleva ero: subjektiivinen tyytyväisyys mittaa käyttäjän mielipidettä ja asennoitumista systeemiin ja sen käyttöön, kun taas dimensionaalisia emootioita arvioidessa käyttäjä arvioi omia tunteitaan systeemin käytön aikana (esim. semanttisen differentiaalin menetelmällä laaditulla kyselylomakkeella). Tällöin voidaan joko arvioida kokonaistunnekokemusta käytön jälkeen tai kokemuksia suhteessa systeemin käytön eri vaiheisiin, jolloin menetelmä on eräänlainen tunnetasolla kulkeva systeemin läpikäynti.

Teoretisointiaktiviteetti pyrkii antamaan yleistettyä tietoa tuotoksen toiminnasta: se pyrkii selittämään miksi tuotos toimi tai ei toiminut käytännössä halutulla tavalla. Tällöin on tärkeää ottaa huomioon luonnontieteen lainalaisuudet, jotka koskevat tuotosta, käyttäjää ja käyttöympäristöä. Erityisesti pyritään tekemään johtopäätöksiä koskien sitä, mitä vaikutuksia (esim. käyttäjän tunteisiin) artefaktin eri piirteillä oli.

Perusteluaktiviteetti on tärkeässä osassa erityisesti matemaattisesti suuntautuneessa tutkimuksessa, jolloin voidaan käyttää matemaattisia todistusmenetelmiä. Ei-matemaattisen tutkimus noudattaa luonnontieteisiin muodostuneita käytäntöjä tutkimusdatan keräykseen ja analysointiin.

Dimensionaalisten emootioiden lähestymistapa tukee teorioiden perustelua: määrällistä dataa voidaan vertailla tilastollisia menetelmiä käyttäen. Esim. voidaan saada tulos, jonka mukaan käyttöliittymä, jossa on jokin tietty piirre, saa aikaan positiivisemman käyttökokemuksen kuin käyttöliittymä, josta kyseinen piirre puuttuu (tai on korvattu jollakin muulla ratkaisulla), ja joka on muuten samanlainen.

Yhteenvetona voidaan todeta, että (dimensionaalisten) emootioiden näkökulman mukaan ottaminen tietotekniikan tutkimukseen voi saada aikaan muutoksia kaikissa Marchin ja Smithin [1995] tutkimustuotoksissa: käsitteistössä, malleissa, menetelmissä ja toteutuksissa. Näkökulman avulla voi myös tehdä tutkimusta kaikilla neljällä tutkimusaktiviteetillä, jotka ovat rakennus-, arviointi-, teoretisointi- , ja perusteluaktiviteetti.

7. Yhteenveto ja oma arvio

Emootio määritellään yleisimmin monitasoiseksi käsitteeksi, ja emootiotutkimuskin on yleensä monitasoista, koostuen kokemuksellisesta tasosta, fysiologisesta tasosta ja käyttäytymistasosta. Emootioita tarkastellaan yleisimmin erillisten emootioiden tai dimensionaalisten emootioiden näkökulmista. Dimensionaalisiin emootioihin liittyy läheisesti semanttisen differentiaalin menetelmä. Dimensionaalisten emootioiden lähestymistavan avulla voidaan tehdä hyvin monenlaista tutkimusta ihminen-tietokone vuorovaikutukseen liittyen.

Mielestäni dimensionaalisten emootioiden lähestymistapa tuntuu perustellulta valinnalta kuvaamaan ihmisen tunnetilaa. Sillä on vankka fysiologinen pohja, se on helposti määrällistettävissä, ja empiirinen tutkimus ihmisten tunnepohjaisista arvioista tukee sitä, että emootio on järjestynyt kolmelle eri dimensiolle.

Dimensionaalisten emootioiden lähestymistapa kuvaa hyvin ihmisen tunnetilaa, mutta se ei anna tietoa esim. ihmisen kognitiivisista ajatusprosesseista. Siksi lähestymistapa täytyykin integroida osaksi malleihin, joissa kuvataan ihmistä tietokoneen käyttäjänä. Tällöin mielestäni saavutetaan realistisempi kuva ihmisen toiminnasta tietokoneen käyttötilanteissa; osa käyttäytymisestä voidaan selittää tunnekäsitteistöä käyttäen. Toisin sanoen dimensionaalisten emootioiden lähestymistavan avulla ei voida selittää kaikkea ihmisen ja tietokoneen vuorovaikutusta, vaan se pitää nähdä lisänä olemassaoleviin näkemyksiin, jotka koskevat ihmistä tietokoneen käyttäjänä.

Emootion käsite määritellään koostuvaksi kolmesta erillisestä järjestelmästä: suullisista raporteista (kokemuksen taso ja kielellinen taso), fysiologisista vasteista (fysiologinen taso) ja näkyvästä käyttäytymisestä (behavioraalinen taso). Emootiotutkimuksessa nämä kolme tasoa nähdään

toisiaan täydentävinä. Yhden käsitteen määritteleminen monen tason avulla tuo mukanaan myös ongelmia. Tutkijat puhuvatkin ns. huonon kovarianssin ongelmasta [Miller ja Kozak, 1993], joka tarkoittaa sitä, että edellämainituista kolmesta eri lähteestä tullut tutkimusdata ei aina tue samoja päätelmiä, vaan niiden perusteella saattaisi tehdä erilaisia johtopäätöksiä. Mielestäni tämä kuvastaa sitä, kuinka eri tasot eroavat toisistaan jo taustaoletuksien tasolla, ja niiden suhteuttaminen toisiinsa voi olla hankalaa. Jos määrälliset mittaukset eivät korreloi systemaattisesti, voi tulkinta vaatia tutkijalta perusteluita, jotka voidaan ilmaista vain väitelauseina.

Tästä ongelmasta huolimatta olen sitä mieltä, että emootioita tulisi jatkossakin tutkia eri tasoja yhdistäen, eikä esimerkiksi pelkästään tunnekokemusta mittaamalla. Sama ongelma on yleinen ihmistutkimuksessa yleensäkin: yhden tason tutkimukseen keskittyminen kaventaa näkökulmaa niin, että muiden tasojen asiat jäävät huomiotta, jolloin tieteellisistä selityksistä tulee helposti reduktionistisia, eli ne selittävät monitasoisen ilmiön suhteessa vain yhteen tasoon. Siksi monitasoisella emootiotutkimuksella voidaan todennäköisesti kokonaistasolla muodostaa realistisempi kuva käyttäjän emootioista kuin pelkästään yhteen tasoon keskittymällä.

Viiteluettelo

[Bradley et al., 1993] Bradley, M. M., Greenwald, M. K., and Hamm, A. O. Affective Picture Processing. In N. Birbaumer and A. Öhman (Eds.) *The Structure of Emotion*. Seattle: Hogrefe & Huber, 1993, 48-65.

[Bradley ja Lang, 1994] Bradley, M. M., and Lang, P. J. Measuring Emotion: The Self-Assessment Manikin and the Semantic Differential. *J. Behav. Ther. & Exp. Psychiat.*, 25, 1(1994), 49-59.

[Butler, 1996] Butler, K. A. Usability Engineering Turns 10. *Interactions*. January 1996, 59-75.

[Ekman ja Friesen, 1978] Ekman, P. and Friesen, W. V. *Facial Action Coding System (FACS): A Technique for the Measurement of Facial Action*. Palo Alto, CA: Consulting Psychologists Press, 1978.

[Ekman et. al, 1983] Ekman, P., Levenson, R. W., and Friesen, W.V. Autonomic Nervous System Activity Distinguishes Among Emotions. *Science*, 221 (September 1983), 1208-1210.

[Ekman, 1992] Ekman, P. An Argument for Basic Emotions. *Cognition and Emotion*. 6(3/4), 169-200.

[Frøkjær et al, 2000] Frøkjær, E., Hertzum, M., and Hornbæk, K. Measuring usability: Are effectiveness, efficiency and satisfaction really correlated? In *Proceedings of CHI 2000 Human Factors in Computing Systems*, ACM Press, New York, 2000, 345-352.

[Izard, 1977] Izard, C. Human Emotions. New York: Plenum Press.

[Lang, 1993] Lang, P. J. The Three-System Approach to Emotion. In: N. Birbaumer and A. Öhman (Eds.), *The Structure of Emotion*, Hogrefe and Huber Publishers, Seattle 1993, 3-17.

[March ja Smith, 1995] March, S. T. and Smith, G. F. Design and Natural Science Research on Information Technology. *Decision Support Systems*, 15, 1995, 251-266.

[Miller ja Kozak, 1993] Miller, Gregory A. and Kozak, Michael J. Three-Systems Assessment and the Construct of Emotion. In N. Birbaumer and A. Öhman (Eds.) *The Structure of Emotion*. Seattle: Hogrefe & Huber, 1993, 31-47.

[Partala et al, 2000] Partala, T., Jokiniemi, M. and Surakka, V. Pupillary Responses to Emotionally Provocative Stimuli. In *Proceedings of ETRA 2000, Eye Tracking Research and Applications, Symposium 2000*, Palm Beach Gardens, FL, November 2000, 123-129.

[Partala et al., 2001] Partala, T., Aula, A., and Surakka, V. Combined Voluntary Gaze Direction and Facial Muscle Activity as a New Pointing Technique. In *Proceedings of INTERACT 2001*, Tokyo, Japan, July 2001.

[Picard, 1997] Picard, R.W. *Affective Computing.* M.I.T. Press, Cambridge MA, 1997.

[Surakka et al., 1998] Surakka, V., Tenhunen-Eskelinen, M., Hietanen, J. K., and Sams, M. Modulation of Human Auditory Information Processing by Visual Emotional Stimuli. *Cognitive Brain Research*, 7(1998), 159-163.

[Öhman ja Birbaumer, 1993] Öhman, A. and Birbaumer, N. Psychophysiological and Cognitive-Clinical Perspectives on Emotion: Introduction and Overview, In: N. Birbaumer and A. Öhman (Eds.), *The Structure of Emotion*, Hogrefe and Huber Publishers, Seattle 1993, 3-17.

Ekologinen psykologia ja affordanssit

Joni Koskinen

Tämän paperin tarkoituksena on esitellä lyhyesti erästä havaintopsykologian osa-aluetta, ekologista psykologiaa, ja sen soveltamista ihmisen ja tietokoneen välisen vuorovaikutuksen tutkimuksessa. Paperiin on kerätty erilaisia näkemyksiä aihealueen merkittävimpien tutkijoiden kirjoituksista ja keskeisimpiä ajatuksia ja ongelmia käydään läpi useista eri näkökulmista. Lisäksi paperissa keskustellaan teorian eri suuntausten välisistä eroista. Sekä suuntausten hyviä että huonoja puolia on pyritty tuomaan esille tasapuolisesti.

1. Johdanto

Ekologinen psykologia juontaa juurensa yhdysvaltalaisen havaintopsykologi James J. Gibsonin 1960-70 –luvuilla kehittelemiin ajatuksiin havainnoitsijan, havaintojen kohteina olevien objektien ja havaittavan ympäristön rakenteiden välisistä suhteista. Mentalismin ympäristön ja sen havainnoijan erottavaan subjektiivisuuteen ja behaviorismin trivialisoituun havainnoijan ja ympäristön väliseen suhteeseen tyytymättömän Gibsonin vaihtoehdoksi kehittämää teoriaa ohjaa ja pitkälti sen sisältöä määrittää ajatus, että kaikki havaintoa varten tarpeellinen informaatio on jo olemassa ympäristössä, ja havaitsija käyttää

ympäristönsä ominaisuuksia toimintojensa rakenteina. Havainnoijan rooli on vain tarttua oikealla tavalla saatavilla olevaan informaatioon, havainnoida suoraan ympäristönsä ominaisuuksia. Käsitys eroaa suuresti yleisimmistä kognitiivisista malleista, joiden mukaan havainnoija rakentaa havainnon pienemmistä osista sisäisten prosessiensa kautta. Gibsonin mukaan havaitsijan toimintoja ohjaavat sisäisten kognitiivisten prosessien sijasta siis ympäristön rakenteet, ja esimerkiksi uuden oppiminen merkitsee sitä, että havaitsijalle paljastuu ympäristöstään uusia kaavoja siitä, miten maailma toimii. Kun perinteisessä kognitiivisessa psykologiassa havaitsemisprosessi sijoitetaan havaitsijan sisälle, ekologisessa psykologiassa havaitsijan toiminta on olennainen ja kiinteä osa itse havaitsemista.

Ihmisen ja tietokoneen vuorovaikutuksen tutkimuksessa hyödynnetyssä ekologisen psykologian osassa affordanssin käsite on niin merkittävässä roolissa, että koko aluetta voitaisiin varsin hyvin ekologisen psykologian sijasta kutsua affordanssiteoriaksi. Niinpä tämä selvitys keskittyy hyvin vahvasti kyseisen käsitteen ympärille. Koska affordanssin käsitteen tarkka määrittely on varsin vaikeaa, käydään ensin läpi yksityiskohtaisesti sen ominaisuuksia ja sitten havainnollistettaan tarkastelun alla olevaa asiaa runsain esimerkein.

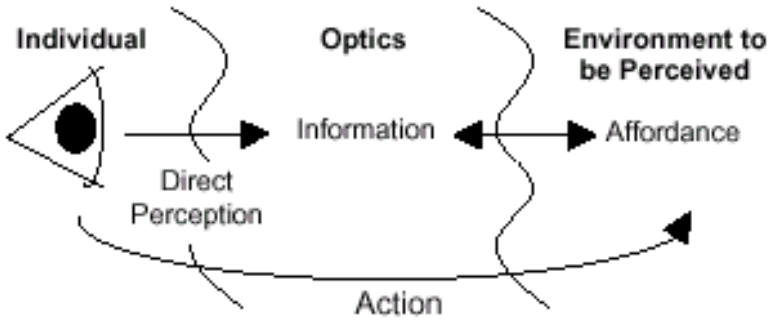
Paperin alussa käydään ensin läpi affordanssin käsitettä sellaisena, millaiseksi Gibson sen alun perin loi. Alkuperäisen idean esittelyn jälkeen pureudutaan ekologisen psykologian soveltamiseen ja affordanssi-termin käyttöön ihmisen ja tietokoneen välisen vuorovaikutuksen tutkimuksessa. Teoria on alalla nykyään hyvin laajalle pirstoutunut, eikä alkuperäisistä ideoista ole usein paljonkaan jäljellä. Siksi tässä paperissa kiinnitämme huomiota ennenkaikkea 1980- ja 1990-lukujen vaihteessa asiaa tietokonesovellusten ja –laitteistojen alueelle lanseeranneiden Donald Normanin ja William Gaverin kirjoituksiin ja ajatuksiin aiheesta. Varsinkin Normanin vaikutus myöhemmin tietokoneisiin liittyviissä asioissa affordansseja käsitelleisiin tutkijoihin on ollut todella suuri. Normanin ja Gaverin ajatusten esittelyn jälkeen tutkitaan mille alueille ja miten affordansseja voidaan soveltaa, tarkastellaan kriittisesti affordanssien soveltamisen huonoja puolia ja lopuksi tehdään lyhyt yhteenveto tässä kirjoituksessa käsitellyistä asioista.

2. Affordanssi

Eräs ekologisen psykologian keskeisimpiä käsitteittä on affordanssi. Affordanssit ovat maailman ominaisuuksia, jotka ovat merkityksellisiä ihmisen (tai eläimen) toiminnan kannalta ja yhteensopivia sen kanssa. Affordanssi, kun

se on havaittavissa, tarjoaa suoran linkin havainnon ja toiminnan välille, ja tarkoittaa hieman yksinkertaistettuna sitä, että objektia havaittaessa havaitaan samalla se, mihin käyttötarkoituksiin se tarjoaa mahdollisuuden tai kannustaa. Laajemmassa merkityksessä havaitsija havaitseekin ympäristöstään vain sen "tarjolle asettamia" asioita.

Gibsonin mukaan nämä ympäristön ja objektien tarjoamat vihjeet niiden käyttökelpoisuuksista ovat täysin riippumattomia havainnoitsijan kyvystä havainnoida objektia. Affordanssien olemassaolo ei riipu myöskään arvoista, tarkoituksista tai tulkinnoista. Sen sijaan ne ovat osa havaitsemisprosessia ja tarvitsevat havainnoijan viitekehyksekseen. Lisäksi ne esiintyvät suhteessa havaitsijan toimintaedellytyksiin. Havaitsija ja ympäristö ovat erottamaton pari. Tämä tarkoittaa sitä, että affordanssi ei ole havainnoitsijan kokemuksen tai ajatusten ominaisuus, vaan hänen (tai sen) toimintamahdollisuuksiensa ja –edellytystensä ominaisuus. Siis esimerkiksi lumihangen affordanssi havaitsijalle on olemassa, vaikka hän ei affordanssia huomaisikaan. Mutta koska affordanssi on olemassa vain suhteessa tiettyyn havainnoijaan, on mahdollista että jollekin ruumiillisesti kevyelle havainnoijalle lumihangella on tuen antamisen affordanssi, mutta painavamman havainnoijan ja lumihangen välillä tätä samaa affordanssia ei olekaan.



Kuva 1. Affordanssin havaitseminen [McGrenere and Ho, 2000]

2.1.. Esimerkkejä Gibsonilaisista affordansseista

Havaitsijan ympäristö tarjoaa usein lukemattomia määriä erilaisia affordansseja. Esimerkiksi tavallisen tuolin ja havaitsijan suhde saattaa sisältää

useita affordansseja: päällysvaatteita riisumassa oleva voi ripustaa tuolin selkänojalle takin, seisomaan väsyneelle tuoli tarjoaa paikan jolle istua, kirjahyllyn päältä tavaroita hamuava voi käyttää tuolia korokkeena jonka avulla yltää korkeammalle, tulipalon sattuessa tuolia voi käyttää ikkunan rikkomiseen ja niin edelleen.

Koska affordanssit ovat maailman ominaisuuksia, jotka havainnoija havaitsee suoraan ilman monimutkaisia kognitiivisia prosesseja, voidaan esimerkeissä käyttää havainnoijana ihmisen sijasta myös muuta eläintä. Katos voi ”kannustaa” eläintä suojaan sateelta, siis katoksen ja eläimen välisessä suhteessa on olemassa suojan antamisen affordanssi. Ekologisen psykologian mukaan eläin ei havaitse tilanteessa peräkkäisiä ärsykkeitä, joita kognitiiviset prosessit liittäisivät yhteen, vaan yksinkertaisesti katoksen joka on osa todellisuutta ja affordanssin ansiosta eläin hakee suojaa kastumiselta sen alta.

Gibson ei teoriassaan erittele affordanssin eri asteita. Jos ajatellaan vaikkapa rappusia, on tietenkin selvää että pitkäjalkainen mies harppoo niitä helpommin kuin juuri kävelemään oppinut pikkulapsi, mutta Gibsonin mukaan rappuset tarjoavat ylöspäin nousemisen affordansiin kummallekin, eikä hän ota kantaa nousun helppouteen. Myöhemmin affordanssien eri asteita on yritetty määrittää, ihmisen ja tietokoneen vuorovaikutuksen kannalta huomionarvoisimpana Warrenin [Warren, 1995] tutkimukset, joiden tavoitteena oli laskea affordanssin aste, jonka perusteella voitaisiin määrittää käyttäjän kannalta optimaaliset ominaisuudet käyttöliittymälle.

2.2. Norman

Ihmisen ja tietokoneen välisen vuorovaikutuksen alueella ekologinen psykologia ja lähinnä affordanssit herättivät ensimmäisen kerran suurempaa huomiota 1980-luvun loppupuolella, kun Donald Norman käsitteli aihetta kirjassaan ”The Psychology of Everyday Things” [Norman, 1988]. Normanin popularisoimat tulkinnat erosivat kuitenkin monista Gibsonin alkuperäisistä ajatuksista. Koska tietokonemaailmassa liikutaan usein osin virtuaalisissa ympäristöissä ja käsitellään virtuaalisia objekteja, on osa Normanin erilaisista tulkinnoista perusteltua, sillä alkuperäisten affordanssin idean todellinen vahvuusalue piilee fyysisten, kolmiulotteisten objektien tutkimisessa. Seuraavaksi käymme läpi Normanin ajatuksia affordansseista ja tutkimme miten ne eroavat Gibsonin alkuperäisistä ideoista.

Normanin käyttämässä merkityksessä affordanssi on muuntunut tarkoittamaan objektin havaittua ominaisuutta, jota ei välttämättä todellisuudessa ole edes olemassa. Affordanssi-termiä käytetäänkin Normanin kirjoituksissa usein merkityksessä, joka alkuperäisen affordanssi-idean mukaan olisikin itse affordanssin sijasta tietoa, joka määrittelee sen. Affordanssi tässä merkityksessä voi olla riippuvainen havainnoijan kokemuksista, tiedoista tai kulttuuritaustasta, ja ne määräävät affordanssin ominaisuuksia ja olemassaoloa. Myöhemmissä kirjoituksissaan Norman selventääkin termin käyttöään siten, että määrittelee alun perin kirjoittaneensa *havaituista affordansseista* eikä varsinaisista, ”oikeista” affordansseista. Hän perustelee tätä käyttöään sillä, että käytettävyys, eli aihe jota Norman käsittelee kirjoissaan ja joka on hänelle tutuin alue, kietoutuu enemmän nimenomaan havaitun affordanssin ympärille.

Alkuperäisessä ekologisessa psykologiassa oltiin kiinnostuneita siitä, kuinka havainnoija havainnoi ympäristöään. Normanin tavoitteet ovat hyvin erilaiset. Ihmisen ja tietokoneen vuorovaikutuksen tutkimuksessa päähuomion kohteeksi nouseekin kysymys siitä, kuinka voidaan suunnitella tai manipuloida sellaisia ympäristöjä, joissa haluttu toiminnallisuus on mahdollisimman helposti havaittavissa. Varsinkin sovellusten ja laitteistojen suunnittelijoille on enemmän hyötyä siitä että osaa suunnitella sellaisia asioita, jotka havaitessaan käyttäjä tietää heti mitä tehdä, kuin sellaisia, joissa on oikeanlaisia affordansseja, mutta jotka eivät ole ilmeisiä ja niiden havaitseminen on vaikeaa. Normanin tavoitteena onkin ollut tehdä ilmeiseksi sitä, mitä objekteilla voidaan tehdä, eli päätarkoituksena on alleviivata mahdollisia affordansseja eikä luoda niitä. ”Aidot” affordanssit eivät määrittele objektin käytettävyyttä, vaan havaitut affordanssit.

Gibsonin ja Normanin affordanssikäsitteen eroavaisuuksia voidaan tarkastella usein käytetyn ovenkahvaesimerkin avulla. Kuvitellaan ovi, jossa ei ole kahvaa eikä mitään mistä tarttua, vain tasainen pinta. Jos havainnoijalla ei ole muuta tietoa oven toiminnasta, on hänen hyvin vaikeata päätellä mihin suuntaan ovi avautuu. Gibsonin affordanssin määritelmän mukaan jos ovi on tunnistettavissa oveksi, on olemassa affordanssi riippumatta siitä, onko havainnoijalla mitään visuaalista informaatiota joka määrittelisi mihin suuntaan ovi avautuu. Normanin määritelmän mukaan affordanssi olisi olemassa vain, jos ovessa olisi sellaista informaatiota joka antaisi signaalin siitä, mihin suuntaan havainnoija voi oven aukaista.

Koska Norman, toisin kuin Gibson, ei ole kiinnostunut totuudesta, vaan siitä, mitä havainnoija havaitsee, mahdollistaa hänen affordanssitulkintansa helpommin omaksuttavia ja toteutettavia malleja käyttöliittymien suunnitteluun. Lisäksi yksinkertaisempi tulkinta antaa mahdollisuuden yhdistellä teoriaa muiden käyttöliittymäsuunnittelun kannalta oleellisten teorioiden kanssa.

Sen jälkeen kun Norman aloitti laajemmassa mittakaavassa affordanssien soveltamisen ihmisen ja tietokoneen välisen vuorovaikutuksen alueella, on termi kokenut inflaation ja sitä on käytetty mitä erilaisimpiin asioihin viittaamassa. Norman tiedostaa tämän itsekin. Hän kertoo mm. kuinka liian usein hän kuulee graafisten suunnittelijoiden väittävän lisänneensä affordansseja käyttöliittymiin, vaikka todellisuudessa suunnittelijat ovat vain havainnollistaneet jollakin kuvalla että tietty toiminto on mahdollinen [Norman]. Tällainen havainnollistaminen ei ole Gibsonin määrittelemä affordanssi, eikä edes Normanin termistössään käyttämä havaittu affordanssi. Se on vain symbolien avulla kommunikointia, joka toimii vain jos se noudattaa tiettyjä käyttäjän ymmärtämiä konventioita. Tietenkin tällaisiakin keinoja voi käyttää käyttöliittymäsuunnittelussa, mutta ekologiseen psykologiaan tai affordansseihin on turha silloin vedota. Symbolit eivät sinällään ole affordansseja, vaan esimerkkejä havainnoijan omaksumasta, eri havainnoijien välisistä jaetuista käsitemalleista ja konventioista. Ne vaativat yleensä opittua tietoa toimiakseen, eikä pelkkä suora havainto vailla pohjatietoja riitä niiden ”käyttötarkoitusten” omaksumiseen.

2.3. Gaver

William Gaver, joka pohti affordansseja vuoden 1991 kirjoituksessaan Technology Affordances [Gaver, 1991], toi vahvasti mukaan keskusteluun monimutkaisemmat toiminnot, joissa yhdistyy useampia ajallisesti tai tilallisesti toisiinsa sidottuja affordansseja.

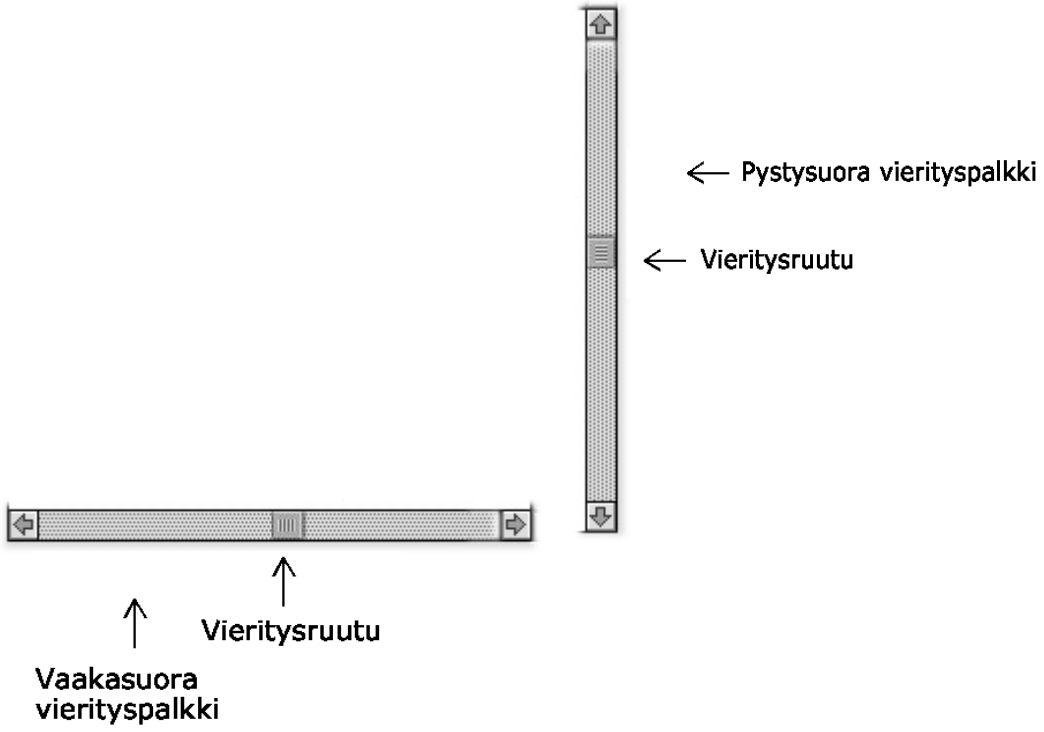
Ajallisesti toisiinsa kytkettyjä affordansseja Gaverin mukaan syntyy tilanteissa, joissa yksinkertainen passiivinen havainnointi ei viittaa tiettyyn affordanssiin, vaan ilmetäkseen tämä affordanssi tarvitsee käyttäjän tarttumista johonkin muuhun affordanssiin. Esimerkkinä tästä yhden affordanssin johtamisesta toiseen, voimme turvautua taas ovenkahvaesimerkkiin: Kahvahan ei välttämättä ole vain sellainen, josta voi vetää tai työntää oven aukaistakseen. Kahva voi olla myös sellainen, jota pitää painaa alas ennen kuin ovi aukeaa. Tällöin ensimmäinen, visuaalinen, affordanssi saa sen havaitsijan tarttumaan

kiinni kahvasta. Kun käsi on kiinni kahvassa, paljastaa se uuden affordanssin, joka tarjoaa mahdollisuutta painaa kahvaa alaspäin. Kahvan ollessa alhaalla lukossa ollut ovi aukeaa ja tekee ilmeiseksi taas uuden affordanssin, oven aukaisun kahvasta vetämällä.

Affordanssit voivat kytkeytyä toisiinsa myös tilassa. Oviesimerkissä tilassa kytkeytyminen tarkoittaa, että kahva yksin voi omata vetämisen affordanssin. Ovi ilman kahvaa taas houkuttelee manipuloimaan sitä jotenkin (koska ovi on osittain irti seinästä). Kun kahvasta vetämisen affordanssi nähdään yhdessä oven avaamisen affordanssin kanssa, nähdään suoraan koko mekanismi joka avaa oven.

Sekä ajallisesti että tilallisesti kytketyille affordansseille ominaista on, että paljastuessaan osa affordansseista paljastaa uusia affordansseja.

Tavallisen tietokoneen käyttöliittymän tarjoamista toisiinsa sidotuista affordansseista Gaver käyttää esimerkkinä Macintoshin (nykyisin myös Windows-ympäristöissä käytetään lähes samanlaista) vierityspalkkia. Palkin sisässä oleva veritysruutu houkuttaa tarttumaan, ja tartuttaessa se todennäköisesti liikahtaa ja ”vihjaa” mahdollisuudesta raahata sitä. Lisäksi palkin muu (yleensä harmaa) alue houkuttelee raahaamaan veritysruutua juuri oikeaan suuntaan, kun käyttäjä näkee suoraan palkin muodosta kannattaako ruutua raahata vaaka- vai pystysuunnassa.



Kuva 2. Vierityspalkit (MacOs)

Normanin mielipide ylläolevassa tapauksessa eroaa Gaverin ajatuksista, sillä Normanin mielestä kysymyksessä eivät ole affordanssit vaan konventiot ja rajoitteet. Hänen mukaansa vierityspalkit eivät sinänsä ”vihjaa” oikeaan käyttöönsä. Ne ovat vain tehty niin, ettei niitä voi käyttää kuin suunnittelijan haluamalla tavalla, niiden toiminta on rajoitettu tiettyyn muottiin ja pakotettu käyttäjä noudattamaan sitä. Lisäksi Normanin mukaan affordanssista puhumista vastaan sotii se, että käyttäjän täytyy tietää miten palkki toimii, eikä suora havainto paljasta sitä. Yhtä hyvin alaspäin vedettävä vieritinpalikka voisi paljastaa uutta tekstiä ruudun alaosan sijasta sen yläosasta, niin kuin tavallista paperia alaspäin vedettäessä tapahtuisi. Se, että tiedämme kummasta suunnasta tekstiä alkaa näkyä lisää palikkaa tiettyyn suuntaan viemällä, perustuu aiemmin opittuun, konventioihin.

3. Sovellusalueista

Eräs perusajatus miksi ekologista psykologiaa ja lähinnä affordansseja on tutkittu ja sovellettu tietotekniikan ja siellä erityisesti sovellusten käyttöliittymien alueella, on se, että se tarjoaa tehokkaan keinon kiinnittää huomiota ei vain teknologiaan tai sen käyttäjään, vaan nimenomaan näiden

väliseen vuorovaikutukseen. Affordanssien avulla voidaan lähestyä suoraan havaintojen ja toimintojen eri tekijöitä, ja käyttää tätä tietämystä hyväksi kun tarkastellaan käyttöliittymän käytön ja käytön oppimisen helppoutta. Näitä asioita analysoitaessa affordanssin ajatusta voidaan hyödyntää käyttöliittymän järkevyyden selvittämisessä ja apuna mietittäessä tapoja parantaa sen käytettävyyttä. Lisäksi konsepti mahdollistaa melko yksinkertaisen ja yhtenäisen tavan käsitellä maailman ominaisuuksien monimutkaisiakin yhdistelmiä.

Teknologian ja sen käyttäjän välisen vuorovaikutuksen apukeinona Gibsonin alkuperäinen affordanssin idea on vahvempi väline kuin esimerkiksi Normanin tulkinnat siitä. Vaikka Normanin affordanssiajatukset vesittävätkin tällä alueella Gibsonin idean vahvuutta, voidaan Normanin tulkintojakin soveltaa eräille muille alueille erittäin hyvin. Normanin affordanssin ajatus oikein käytettynä rohkaisee ohjelmistojen ja laitteistojen käyttöliittymien suunnittelijoita ja analysoijia miettimään kohteitaan myös siinä mielessä, että halutut toiminnot saataisiin esiin tekemällä ne ilmeisiksi ja haluttuun suuntaan “vihjaaviksi”. Kun affordanssien periaatetta noudatetaan suunnittelussa tinkimättömästi, saadaan rakennettua käyttöliittymä, joka korostaa haluttujen toimintojen mahdollisuutta ja ohjaa pois ei-halutuista toiminnoista. Tämä helppokäyttöisten systeemien tekeminen painottamalla affordanssien tekemistä mahdollisimman havaittaviksi onkin Normanin koko affordanssiajatusten ylivoimaisesti keskeisin teema.

Affordansseja käyttöliittymäsuunnitteluun sovellettaessa on huomattava, että Gibsonin ja Normanin affordanssikäsitteitä sekaisin tai huolimattomasti käytettäessä kohdataan ongelmia. Gibsonin affordanssien mukainen suunnittelu lähtee objektien ja ympäristön *käyttökelpoisuuden* pohjalta, Normanin affordanssit taas liittyvät kiinteästi *käytettävyyteen*. Käyttökelpoisuudella tarkoitetaan siis sitä, että affordanssit viittaavat itse objektin hyödyllisyyteen, kun taas käytettävyys määritellään sen perusteella, millaista on tieto, joka määrittelee objektin hyödyllisyyttä. Käyttöliittymän käyttökelpoisuus määritellään sen kautta, mitä toimintamahdollisuuksia sen design antaa ja miten nämä toimintamahdollisuudet ovat suhteessa siihen, mitä käyttäjä on tekemässä ja mihin hän pyrkii. Kun näitä toimintamahdollisuuksia alleviivataan ja tehdään helpommiksi havaita, on kyse käytettävyydestä.

Käyttöliittymän käytön oppimisesta keskusteltaessa kannattaa kiinnittää huomiota Gaverin affordanssiajatuksiin. Kuten aiemmin paperissa selvitettiin, oppiminen ekologisessa psykologiassa perustuu erilaisten kaavojen

huomaamiseen maailmasta. Oppimisessa painottuu siis ympäristön tarkastelu ja huomiokyky, eikä niinkään päättelykyky. Affordansseja hyväksikäyttävän käyttöliittymäsuunnittelijan tehtävä on tällöin oppimista tukevia käyttöliittymiä suunnitellessaan tehdä niistä sellaisia, että ne ohjaavat nimenomaan käyttäjän huomiokykyä haluttuihin toimintoihin. Silloin hyvin suunnitellut ajallisesti peräkkäiset ja tilallisesti toisiinsa sidotut affordansit ovat merkittävässä huomiokyvyn ohjaajan roolissa.

Affordanssien ja vastaavien, alkujaan reaalimaailman objektien tutkimiseen kehitettyjen, ideoiden käyttö ja merkitys tietokoneiden käyttöliittymäalalla on tullut tärkeämmäksi osaksi tutkimusta viime vuosina. Tämä on seurausta pääosin siitä, että siirtyminen tekstipohjaisista käyttöliittymistä graafisiin ja edelleen multimodaalisiin ja entistä interaktiivisempiin järjestelmiin on ollut samalla siirtymistä lähemmäksi niitä alueita, joita varten em. ideat ja teoriat ovat olemassa.

4. Kritiikkiä

Nykyisellään affordanssin konsepti on levinnyt niin laajalle niin monina eri versioina, että sen alkuperäiset ideat ovat usein vesittyneet. Affordanssin käsitteellä ei ole tällä hetkellä mitään tarkkaa määritelmää, vaan siitä keskustelevat tutkijat saattavat todellisuudessa puhua täysin eri asioista. Useimmissa tapauksissa vain hyvin pieni osa alkuperäisistä, Gibsonin esittämistä ajatuksista on jäljellä. Ikävä kyllä, usea nykyään käytettävä malli, jossa affordansseilla on keskeinen osa on helposti opittavissa, mutta affordanssin syvempi ymmärtäminen ei onnistu, ja mallin soveltaminen voi johtaa huonoihin tuloksiin.

Eräs mielenkiintoinen ja huomionarvoinen seikka onkin se, että termi ”ecological” viittasi Gibsonilla ympäristöön ja sen tärkeyteen havaintojen muodostumisessa. Kun pääpaino affordanssien tutkimuksessa siirrettiin pois havainnoijan ja havaitun välisestä kiinteästä suhteesta, unohtui samalla koko teorian nimen etymologia.

Useiden erilaisten määritelmien hylkääminen ja paluu lähemmäksi Gibsonin alkuperäisiä ajatuksia olisi järkevää, sillä konsepti saataisiin taas selvemmäksi ja käyttökelpoisuuden sekä oikeanlaisen toiminnallisuuden suunnittelu nousisi teorian käyttöliittymiin soveltamisen keskeiseksi alueeksi. Vaikka käytettävyys onkin hyvin tärkeä käyttöliittymän ominaisuus, sen analysointiin on olemassa paljon muitakin apuvälineitä, eikä affordanssien pohjalta lähteminen ole

yleensä sen parempi ratkaisu kuin moni muukaan. Sen sijaan alkuperäinen affordanssin idea, ja ekologinen psykologia laajemminkin, tarjoaisi yksinkertaisen ja tehokkaan välineen systeemin käyttäjän ja ulkoisten tekijöiden välisten suhteiden tutkimiseen. Käyttökelpoisuuden tutkimisessa alkuperäinen ekologinen psykologia tarjoaisi oivaa kehystä, mutta on huomattava, että tehokkaimman tuloksen saamiseksi Gibsonin teoriaa jouduttaisiin todennäköisesti yhdistelemään jossain määrin muiden käyttöliittymäteorioiden ja –mallien kanssa.

Gibson itse ei koskaan ehtinyt ottaa kantaa ajatustensa modifioinneista tietokonealalle, sillä hän kuoli 1979, samana vuonna kuin hänen pääteoksensa ”The Ecological Approach To Visual Perception” [Gibson, 1979] julkaistiin.

5. Yhteenveto

Ekologinen psykologia on siis varsin mielenkiintoinen, suoraan havaintoon perustuva psykologinen teoria. Gibsonille affordanssi tarkoitti havainnoijan ja ympäristön välistä suhdetta, suhdetta joka antaa ”vihjeitä” siitä, mitä havainnoija voi milläkin objektilla tehdä. Affordanssin ominaispiirteisiin kuuluu sen esiintyminen suhteessa havainnoitsijan toimintaedellytyksiin, riippumattomuus havainnoijan kyvystä havaita sitä ja sen muuttumattomuus havainnoijan tarpeiden ja päämäärien muuttuessa.

Ihmisen ja tietokoneen välisen vuorovaikutuksen tutkimuksen alueella ekologisesta psykologiasta on kinnitetty huomiota lähinnä affordanssin ideaan. Suurin vaikutus affordanssien tuomisessa yleiseen tietoisuuteen ja soveltamisessa käyttöliittymien suunnitteluun ja arviointiin on ollut Donald Normanilla, jonka kirja ”The Psychology of Everyday Things”[Norman, 1988] herätti keskustelua ja mielenkiintoa 1980- ja 1990-lukujen vaihteessa ja myös sen jälkeen.

Norman käsitteli Gibsonin teorioita hyvin erilaisista lähtökohdista ja tarkoitusperistä kuin Gibson itse, ja niinpä tuloksena olikin monilta osin alkuperäisistä poikkeavia näkökantoja aiheeseen. Tämä on aiheuttanut teorian pilkkoutumista useisiin erilaisiin haaroihin, kun osa tutkijoista toimii Gibsonilaisia ajatuksia ohjenuoranaan pitäen, osa nojaa Normanin tulkintoihin ja osa muuntelee teoriaa omiin tarkoituksiinsa sopivammaksi. Eräitä huomionarvoisimpia laajennuksia ihmisen ja tietokoneen vuorovaikutuksen tutkimuksessa aiempiin affordanssin käyttömuotoihin olivat Gaverin

käsittelemät ajallisesti ja tilallisesti peräkkäiset affordanssit sekä Warrenin yritykset määrittää affordanssin asteita.

Ekologista psykologiaa ja etenkin affordansseja voidaan hyödyntää käyttöliittymien suunnittelukonsepteissa ja niiden tutkimisen analyyttisenä kehyksenä. Vaikka teoriaa onkin useimmiten hyödynnetty näytöllä olevien virtuaalisten objektien analysoinnin apuna, se on usein suhteellisen tehoton ja ilman sitäkin voitaisiin operoida. Sen sijaan teorian vahvoja alueita voisi käyttöliittymätutkimuksessa olla käyttöliittymiin liittyvät, esimerkiksi syötteen – tai palautteen antamisessa käytetyt, fyysiset laitteet. Fyysisten, kolmiulotteisten objektien tutkimukseenhan teoria on alun perin kehitetty.

Tulevaisuuden kannalta tärkeintä olisi konseptin tiivistäminen niin, että kaikki teoriaa hyödyntävät puhuisivat samasta asiasta. Jos tiivistäminen onnistuu, saattaa ekologinen psykologia vielä nousta keskeiseksi teoriaksi käyttöliittymätutkimuksessa, sillä käyttöliittymät ovat koko ajan muuttumassa lähemmäksi ihmiselle luonnollista kolmiulotteista reaalimaailmaa, maailmaa jonka objekteista Gibson oli kiinnostunut.

Viiteluettelo

[Gaver, 1991]. William Gaver, Technology Affordances. Teoksessa *Proceedings of the CHI'91 Conference.* Association for Computing Machinery, 1991, s.79-84.

[Gibson, 1979] James Gibson, *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, Inc., Publishers, New Jersey, 1979.

[McGrenere and Ho] Joanna McGrenere ja Wayne Ho, Affordances: Clarifying and Evolving a Concept. *Proceedings of Graphics Interface 2000.*

[Norman] Donald Norman, Affordances & Design. http://www.jnd.org/dn.mss/affordances-and-design.html (10.5.2001).

[Norman] Donald Norman, Affordance, Conventions and Design. http://www.jnd.org/dn.mss/affordances-interactions.html (10.5.2001)

[Norman, 1988] Donald Norman, *The Psychology of Everyday Things* . Basic Books, New York, 1988.

[Warren, 1995]. William Warren, Constructing an econiche. *Global perspectives on the ecology of human-machine systems.* Lawrence Erlbaum, s. 210-237.

Tunteiden syntetisointi

Anne Aula

Tunteiden syntetisoiminen tietokoneiden avulla lähtee liikkeelle tunteiden syntymekanismien tuntemisesta. Tunteiden syntetisoinnin tavoitteena on, että pystyisimme tuottamaan tietokoneella sellaisia eri tilanteisiin liittyviä tunneilmaisuja, joita tilanteet aiheuttaisivat myös ihmisissä. Tunteiden syntetisoinnissa apuna käytetyt teoriat voidaan jakaa kognitiivisiin teorioihin ja useampia mekanismeja synnyn selittämisessä käyttäviin teorioihin. Kognitiivisten teorioiden mukaan tunteet syntyvät tilanteista tekemiemme tulkintojen perusteella. Tilanteet itsessään eivät siis näiden teorioiden perusteella aiheuta tunnereaktioita. Toiset teoriat ottavat kognition lisäksi huomioon myös muita tekijöitä, esimerkiksi fysiologisten tekijöiden vaikutuksen. Tutkielmassa esitellään kolme tunteiden syntetisoinnin kannalta keskeistä teoriaa: Frijdan funktionaalinen tunneteoria, Ortonyn Cloren ja Collinsin kehittämä kognitiivinen tunneteoria ja Velásquezin Cathexis-malli. Teorioiden pääperiaatteiden lisäksi tutkielmassa kuvataan joitain teorioiden pohjalta tehtyjä tietokonesovelluksia. Lopuksi pohditaan esitettyjen teorioiden vahvuuksia ja heikkouksia sekä tunteiden syntetisoinnin mahdollisuuksia ihminen-tietokone vuorovaikutuksen parantamisessa.

1. Kirjallisuuskatsaus

On esitetty, että tietokoneiden käyttäjät toimivat ihminen-tietokone vuorovaikutuksessa samojen vuorovaikutussääntöjen mukaan kuin toimiessaan toisten ihmisten kanssa [Picard, 1997; Reeves and Nass, 1996]. Ihmisten välisessä vuorovaikutuksessa tunteilla on erottamaton rooli esimerkiksi vuorovaikutuksen laadun määräämisessä. Tämän lisäksi tunteet säätelevät monella tapaa ihmisen kognitiivisia toimintoja, muun muassa muistia, päätöksentekoa ja ongelmanratkaisukykyä. Jotta ihminen-tietokone vuorovaikutus olisi luonnollista ja vaivatonta, on ensiarvoisen tärkeää, ettei tunnereaktioiden tärkeyttä jätetä huomiotta. Yksi askel kohti tunneälykkäitä tietokoneita on mahdollistaa tietokoneille tunnetilojen tuottaminen.

Tunnetilojen syntetisointi mahdollistaisi Picardin [1997] mukaan tietokoneille useita ihmisen kykyjen kaltaisia ominaisuuksia, esimerkiksi kyvyn joustavaan päätöksentekoon, luovuuteen ja luontevaan vuorovaikutukseen käyttäjän kanssa. Ortonyn, Cloren ja Collinsin [1988] mukaan tietokoneilla ei voi olla tunteita samassa merkityksessä kuin ihmisillä. Heidän mukaansa subjektiivinen tunnekokemus on tunteiden keskeinen elementti ja koska tietokoneilla ei ole tietoisuutta, niillä ei voi olla tunteita. Kuitenkin Ortony kollegoineen korostaa, että tunteilla on tärkeä rooli tekoälytutkimuksessa. Vaikkei koneella voisikaan olla tunnekokemusta, se voi silti pystyä järkeilemään tunteista ja tämän järkeilyn pohjalta tulkita esiin tulevia tilanteita myös tunteiden kannalta mielekkäästi.

Emootioiden synnyn simuloiminen tietokoneella lähtee liikkeelle tunnetilojen syiden ja syntymekanismien kartoittamisesta. Teoriat, jotka ovat pyrkineet emootioiden syntymekanismien selvittämiseen ovat osittain psykologian piirissä kehiteltyjä teorioita, joita ei ole alun perin ajateltu sovellettavaksi emootioiden syntetisoinnin apuvälineenä. Toiset teoriat taas ovat lähtökohtaisesti pyrkineet tuottamaan malleja, joiden avulla on mahdollista tuottaa ”tunteellisia” tietokonesovelluksia. Mielipiteet tunteiden taustalla olevista tekijöistä jakavat emootiotutkijoiden mielipiteitä. Tunteet voidaan nähdä automaattisina reaktioina erilaisiin tilanteisiin tai niitä voidaan pitää tilanteiden arvioinnin tuotteina. Kognitiiviset teoriat tunteiden syntysyiden selittämiseen lähtevät liikkeelle siitä ajatuksesta, että tunnetilat syntyvät sen seurauksena, kun ihminen (tai muu tunteva olento) arvioi kohtaamansa tilanteen jollain tietyllä tavalla. Tilanteet itsessään eivät siis kognitiivisten teorioiden mukaan aiheuta tunteita, vaan arvioinnit joita niistä tehdään. Toinen yleinen tapa on käsittää emootioiden synnyn aiheuttajaksi fysiologia, keskushermostoon valmiiksi ohjelmoidut tunnetilat niihin kuuluvine käyttäytymismalleineen.

Tässä tutkielmassa keskityn tunteiden syntetisoinnissa hyväksikäytettyihin teorioihin, jotka voi karkeasti jakaa kognitiivisiin teorioihin ja useampia mekanismeja huomioonottaviin teorioihin. Kognitiivisista teorioista esittelen ensin Frijdan [1986] funktionaalisen lähestymistavan ja sen testaukseen kehitetyn suhteellisen yksinkertaisen ACRES-ohjelman. Sen jälkeen esittelen lukuisten sovellusten pohjateoriana olevan Ortonyn, Cloren ja Collinsin [1998] kognitiivisen mallin. Useampia emootion syntymekanismeja syntetisoinnissa hyväksikäyttävistä malleista kuvaan Velásquezin [1997; 1998] kehittämän Cathexis-mallin. Kunkin teorian esittelyn jälkeen kuvaan tietokonesovelluksia, joilla teoriaa on testattu. Paperin toisessa luvussa pohdin esiteltyjen teorioiden ja niiden pohjalta tehtyjen sovellusten toimivuutta, sekä emootioiden syntetisoinnin tarjoamia mahdollisuuksia.

1.1. Funktionaalinen tunneteoria

Funktionaalinen tunneteoria [Frijda, 1986; Frijda and Swagerman, 1987] olettaa, että tunteet ovat toiminnallisesti merkittävä osa systeemin (ihmisen, eläimen, ehkä tietokoneenkin?) *päämäärän toteuttamispyrkimyksissä* (concern realisation). Teorian mukaan tunteiden merkitys on siinä, että ne tarjoavat mahdollisuuden toimia menestyksekkäästi epävarmassa ympäristössä, aiheuttamalla muutoksia toimintavalmiudessa. Päämäärät joihin systeemi pyrkii, ovat yleensä sen selviämisen kannalta olennaisia asioita, jotka yleisemmällä tasolla merkitsevät sitä, että systeemi voi toimia niin kuin se ”on suunniteltu” toimivaksi. Esimerkiksi tietokoneen päämäärät voisivat olla seuraavanlaiset (nämä päämäärät ovat käytössä myöhemmin kuvattavassa ACRES-systeemissä): käytön jatkuminen (vältä sammuttamista), säännöllinen käyttö (vältä pitkiä odotuksia), oikean syötteen päämäärä (hyväksy vain oikeanlaista syötettä), syötteen vaihtelevuus ja turvallisuus (muuttumaton käyttö). Jos systeemi havaitsee kohtaavansa päämäärien toteutumisen kannalta edullisia tapahtumia, on seurauksena positiivinen tunnetila. Epäedullisista tapahtumista seuraa luonnollisesti negatiivinen tunnetila.

Frijdan [1986] mukaan tunteet syntyvät sellaisten ärsykkeiden seurauksena, jotka ovat keskeisiä yksilön päämäärien kannalta. Matka ärsykkeestä varsinaiseen emootioon koostuu seitsemästä askeleesta, joiden järjestykseen ja etenemiseen vaikuttavat useat tekijät. *Tunneprosessiin* kuuluvat seuraavat askeleet:

1. tapahtuman analysointi (pohjautuu mahdollisesti aikaisempaan tietoon samankaltaisista tilanteista)
2. vertailu (tärkeys suhteessa päämääriin)
3. diagnosointi (mitä tilanteelle on mahdollista tehdä)
4. arviointi (kuinka kiireellinen, vaikea ja vakava tapahtuma on)

5. toiminnan alustus (toiminnan valmiustila päälle)
6. fysiologisen tilan muutokset
7. toiminnan valinta (ulospäin näkyvä toiminta tai kognitiivinen toiminta)

Tunneprosessin vaiheet eivät välttämättä etene systemaattisesti ensimmäisestä viimeiseen, vaan prosessia säädellään koko ajan (tapahtumien etsiminen vs. välttäminen, arviointien muokkaaminen, impulssien kontrollointi, ulkoisen toiminnan säätely jne.).

Jotta päämäärien toteuttamiseen pyrkivä systeemi voisi toimia, tulee sillä olla kyky vastata odottamattomiin tilanteisiin. Tällaisia voivat olla systeemin ulkopuolelta tulevat odottamattomat tapahtumat tai systeemin sisäiset tapahtumat, esimerkiksi odottamattomat tarpeet tai resurssien yhtäkkinen väheneminen. Jos organismi on odottamattomien tilanteiden ilmetessä tekemässä jotain muuta, tulee sillä olla kyky muuttaa tarvittaessa toimintaansa. Modulaarinen rakenne, jossa toimintaa valvoo yleinen toimintamoduuli ja päämäärien toteuttamista varten on erillinen moduulinsa, mahdollistaa sen, että tarvittaessa päämäärien toteuttamismoduuli voi ottaa kontrollin systeemistä. Tällaisessa rakenteessa päämäärien toteuttamismoduuli viestii toimintamoduulille päämäärien ja ympäristön mahdollisuuksien suhteesta. Jos sen viesti osoittaa, että kyseessä oleva tilanne on ensisijainen organismin senhetkisten päämäärien joukossa, voi toimintamoduuli muuttaa organismin nykyistä toimintaa vastaamaan tärkeimmän päämäärän toteuttamismoduulin osoittamaan suuntaan. On kuitenkin tärkeää, että organismi huomioi myös tilanteet, jotka eivät saa tärkeimmän päämäärän asemaa: tilannetta tulee joka tapauksessa tarkkailla ja katsoa mihin suuntaan se kehittyy. Jos sen tärkeys suhteessa päämääriin kasvaa, tulee nykyinen tehtävä keskeyttää sopivassa kohdassa.

ACRES

ACRES (Artificial Concern REalization System) [Frijda and Swagerman, 1987] on systeemi, joka lähtee liikkeelle funktionaalisen teorian mukaisesti tunteisiin liittyvän toiminnan tärkeydestä. ACRESin agentilla on kyky tunnistaa tapahtumien tärkeys suhteessa päämääriinsä. Ohjelman agentti toimii vuorovaikutuksessa käyttäjän kanssa tekstipohjaisessa sovelluksessa. Edellisessä kappaleessa kuvattiin ACRES-ohjelman päämäärät (mm. vältä sammuttamista ja pitkiä odotusaikoja), jotka siis keskeisesti vaikuttavat ohjelman syntetisoimiin tunnetiloihin sen reagoidessa käyttäjän toimiin. Jotta systeemi pystyisi vastaamaan funktionaalisen tunneteorian mukaisesti ns. odottamattomiin tilanteisiin, sen pitäisi pystyä prosessoimaan rinnakkaisesti tapahtumia jopa kesken käynnissä olevan tehtävän. Tietokone, jolla ohjelma pyörii (VAX) ei kuitenkaan mahdollista rinnakkaisprosessointia. Näiltä osin järjestelmä on siis puutteellinen teoriaan nähden.

ACRES-ohjelmalla on kolme keskeistä tehtävää: saada ja hyväksyä käyttäjältä tulevia syötteitä, saada lisäoppia tunteista (ACRES-ohjelman agentin tehtävänä on käsitellä tunteisiin liittyvää tietoa) sekä hankkia, hyödyntää ja tallentaa tietoa omista ja muiden tunteista. Ohjelmallisesti tehtävien toteutus on tehty luomalla kolme tehtäviin liittyvää komponenttia: syötekomponentti, oppimiskomponentti ja tietokomponentti. Tilanteeseen sopiva tehtäväkomponentti valitaan nykytilanteessa tarjolla olevan tiedon perusteella. Jokaisella tehtäväkomponentilla on kaksi funktiota: toimintafunktio ja päämärän toteuttamisfunktio. Funktiot päättelevät saamastaan informaatiosta tilanteeseen sopivat päämäärät, päättelevät millä toiminnoilla päämääriin päästään ja mahdollisesti ajavat kyseiset toiminnot. Varsinainen toiminta siis koostuu peräkkäisistä tunnista-tulkitse-toimi –kierroksista.

ACRES on varhainen yritys syntetisoida tunteita ja tietokoneiden ollessa 1980-luvulla vielä varsin alkeellisia nykyisiin koneisiin verrattuna, ei ACRESin toiminta luonnollisestikaan yllä samalle tasolle nykyisten sovellusten kanssa. Kuitenkin Frijdan [1986] teorialle se tarjosi hyvän testausalustan ja osoitti teorian toimivan hyvin myös tunteiden syntetisoinnissa.

1.2. Ortony Clore Collins (OCC) kognitiivinen tunneteoria

Ortonyn, Cloren ja Collinsin [1988] kognitiivisen tunneteorian (OCC-teoria) keskeisenä lähtökohtana on 22 erilaista *tunnetyyppiä*. Tunnetyypit voidaan teorian mukaan erotella niiden edeltävien tapahtumien mukaan, jotka aiheuttavat kunkin tunnetyypin aktivoitumisen: saman tunnetyypin tunteet aktivoituvat vasteena samalla tavoin tulkituille tilanteille. Teorian mukaan maailmassa on tunnetilojen synnyn kannalta kolme merkittävää muuttujaa, joihin huomio voidaan kiinnittää: *tapahtumat* (tapahtumien seuraukset), *toimijat* (toimijoiden teot) ja *kohteet* (niiden miellyttävät tai epämiellyttävät ominaisuudet). Tunteet ovat valenssiltaan vaihtelevia reaktioita joihinkin edellä mainituista muuttujista, mikä tarkoittaa sitä, että niihin reagoidaan joko negatiivisesti tai positiivisesti. Vaikka ihmiset reagoivat tapahtumiin, toimijoihin ja kohteisiin joko negatiivisesti tai positiivisesti, ei se aina tarkoita, että ihmisessä syntyisi joku tunnetila. Tunnetilan syntyyn vaikuttaa teorian mukaan oleellisesti reaktion voimakkuus. Lievästä negatiivisesta arviosta seuraa ainoastaan lievä negatiivinen yleissävy, mieliala. Jos negatiivisia arvioita tulee runsaasti (vaikka ne kaikki olisivat voimakkuudeltaan heikkoja), voi seurauksena olla negatiivinen tunne, esimerkiksi suru. Tunnetyyppien lisäksi teoriassa erotellaan *tunneryhmät*, jotka koostuvat tunnetyypeistä: tunneryhmän sisällä olevien tunnetyyppien edeltävät tapahtumat ovat rakenteellisesti toisistaan riippuvaisia. Esimerkiksi yksi tunneryhmistä on nimeltään attribuutioryhmä. Kaikki siihen liittyvät tunnetyy-

pit syntyvät toimijan (joko itsen tai jonkun muun) ja toiminnan seurausten arviointien kombinaatioina.

OCC-teoriassa erotetaan kolme perustunneluokkaa: *miellyttävä vs. epämiellyttävä* (reaktio tapahtumiin), *hyväksyvä vs. paheksuva* (reaktio toimijoihin) ja *pitäminen vs. vastenmielisyys* (reaktio kohteisiin). Nämä perustunneluokat voidaan edelleen jakaa pienempiin osiin, tunnetyyppeihin joita OCC-teorian mukaan on siis 22. Tunnetyypit jotka liittyvät *tapahtumiin* ovat muiden kokemiin tapahtumiin liittyviä (esim. onnellinen jonkun puolesta, kateellinen), omiin mahdollisuuksiin liittyviä (esim. tyytyväisyys, pettymys) ja omaan hyvinvointiin liittyviä (ilo, mielipaha). *Toimijoihin* liittyvät tunnetyypit koostuvat neljästä ns. attribuutioryhmään kuuluvasta tunnetilasta (ylpeys, häpeä, ihailu, syytös). *Kohteisiin* puolestaan reagoidaan vetovoimaan liittyvillä tunteilla (rakkaus, viha). Näiden lisäksi teoria käsittää tunneryhmän, joka syntyy hyvinvointi- ja attribuutioryhmän yhdistelmänä. Tämä ryhmä käsittää tunteita jotka syntyvät yhtäaikaisesta huomion kiinnittämisestä *sekä tapahtumaan että toimijaan* (esim. kiitollisuus, katumus).

Ortony *et al.* [1986] pohtivat kirjassaan mahdollisuuksia käyttää OCC-teoriaa tekoälysovelluksissa. Jotta tietokone voisi kyetä päättelemään syntyvät tunteet oikein, tulee sillä olla selkeät säännöt joiden mukaisesti päättely tehdään. Esimerkkinä ilon tunnetyypin syntymisen simulointi [Ortony *et al.*, 1986, 182-183]:

IF DESIRE (p, e, t) > 0
THEN set JOY-POTENTIAL (p, e, t) = $f_j[\,|\,\text{DESIRE}(p, e, t)\,|\,, I_g\,(p, e, t)]$

p = person, e = event, t = time

$|\,\text{DESIRE}(p, e, t)\,|$ funktio palauttaa miellyttävyysarvon (itseisarvon siitä), jonka henkilö p asettaa tapahtumalle e jonain tiettynä aikana.

I_g (p, e, t) funktio palauttaa tunteen intensiteettiin vaikuttavien globaalien tekijöiden (mm. läheisyys, odottamattomuus) arvon (intensiteettimuuttujien käsittely sivuutettiin tässä esittelyssä).

Jos potentiaali ilolle asetettiin (tapahtuman miellyttävyys oli suurempi kuin nolla), edetään seuraavaan vaiheeseen, jossa tunnetilan intensiteetti asetetaan:

IF JOY-POTENTIAL (p, e, t) > JOY-THRESHOLD (p, t)
THEN set JOY-INTENSITY (p, e, t) =
JOY –POTENTIAL (p, e, t) –JOY-THRESHOLD (p, t)
ELSE set JOY-INTENSITY (p, e, t) = 0

Mielialoja voidaan teorian avulla tuottaa laskennallisesti esimerkiksi laskemalla yhteen ilon mahdollisuutta (JOY-POTENTIAL) kohottavien tapahtumien lukumäärä ja laskeamalla ilon raja-arvoa iloa aiheuttavien tapahtumien määrän lisääntyessä.

Ortony *et al.* [1986] jättivät täsmällisten tunteiden syntyä määrittävien kontrollimekanismien kehittämisen tulevaisuuden tutkimuksen tehtäväksi ja tutkimus on tarttunut innokkaasti tähän tehtävään. OCC-teorian pohjalta on tehty lukuisia sovelluksia joissa teorian paikkansapitävyyttä tai toimivuutta on testattu. Esittelen seuraavassa kaksi keskeistä sovellusta.

Oz-projekti

Batesin johtaman tutkimusryhmän Oz-projektin [Bates *et al.*, 1992a; 1992b; Bates, 1994] tavoitteena on tehdä taiteellisesti vaikuttavia simuloituja maailmoja, Oz-maailmoja. Maailmat koostuvat animoidusta ympäristöstä ja animoiduista agenteista. Tavoitteena on, että agentit ovat mahdollisimman paljon elävän kaltaisia ja tuottavat täten käyttäjälle ”illuusion elämästä”. Tämä on mahdollista, jos agenteilla on runsaasti kykyjä, kuten päämäärätietoinen toiminta, emotionaalinen tila, tunteiden mukaan vaihtuva toiminta, kykyjä toimia sosiaalisesti ja kielellisiä taitoja. Batesin mukaan tunteet ovat ensiarvoisen tärkeä tekijä ottaa huomioon uskottavien agenttien rakentamisessa. Jos hahmo ei välitä ympäristön tapahtumista, sitä pidetään elottomana ja konemaisena. Oz-maailmassa on agenttiarkkitehtuuri Tok, joka antaa tehtäviä eri kommunikaatiokomponenteille, kuten havainnoinnista, päämäärähakuisesta toiminnasta, kielenkäsittelystä, sekä emootioista ja sosiaalisista suhteista vastaaville komponenteille. Viimeistä näistä, joka on tämän käsittelyn kannalta olennaisin, kutsutaan Emiksi. Kuvaan tässä lyhyessä esittelyssä vain Emin toimintaa (simulaatiossa Em luonnollisesti toimii vuorovaikutuksessa muiden kommunikaatiokomponenttien kanssa).

Käytän tässä esittelyssä esimerkkinä Oz-maailmaan luotua kissa-agenttia, Lyotardia [Bates *et al.*, 1992a; 1992b]. Lyotardilla on tunnearsenaalissaan muun muassa seuraavat tunteet: pelko, onni, suru, ihailu, syytös, kiitollisuus, viha ja rakkaus. Em toimii päämäärätietoisesta toiminnasta vastaavan komponentin (Hapin) kanssa siten, että Hap informoi Emiä tavoitteiden syntymisestä, sekä niiden onnistumisesta ja epäonnistumisesta. Tilanteen johdosta aiheutuva tunnetila riippuu päämäärän tärkeydestä. Tunteiden syntymisen raja-arvot on sovelluksessa asetettu sen verran korkealle, etteivät vähemmän tärkeät päämäärät aiheuta tunnetilojen syntymistä (paitsi jos niiden aiheuttamia onnistumisen/epäonnistumisen kokemuksia tulee tarpeeksi). Lyotardilla on kaksi toimintaa ohjaavaa perusstandardia: *älä aiheuta päämäärieni tavoittelun epäonnistumista* ja *auta minua saavuttamaan päämääräni*. Näiden standardien valossa ta-

pahtumat (toiminnan hyväksyminen tai kieltäminen) aiheuttavat tunnetiloja (mm. ylpeys, ihailu). Lyotardilla on myös tunteiden kombinaatioina syntyviä tunnetiloja, kuten onnellisuuden ja ihailun tuloksena syntyvä kiitollisuus. Lyotardin asenteet kohteisiin ovat joko negatiivisia tai positiivisia ja asenteet muuttuvat sen myötä kun kulloinenkin asenne osoittautuu vääräksi. Lyotardilla on esimerkiksi aluksi negatiivinen asenne simulaation käyttäjää kohtaan, mutta asenne muuttuu jos käyttäjä onnistuu tekemään Lyotardin tyytyväiseksi. Tunnetilan rajallinen kesto on otettu huomioon Emissä erillisen tunteiden kestoa kontrolloivan funktion avulla.

Affective reasoner

Clark Elliotin [1992] Affective Reasoner (AR)–systeemin lähtökohtana on OCC-malli, mutta Elliot on muokannut mallia jossain määrin paremmin omiin tarkoituksiinsa sopivaksi. AR-systeemissä on Oz-projektin tapaan simuloitu maailma, jossa on emotionaaliseen vuorovaikutukseen pystyviä agentteja. AR-systeemissä on OCC-mallin 22:n tunteen sijasta käytössä 24 tunnetta (lisänä rakkaus ja viha, jotka syntyvät teorian mukaan kohteen arvioinnin ja toimijoiden tekojen arvioinnin yhdistelmänä). Sovelluksessa demonstroidaan sitä, miten agenttien persoonallisuus (esimerkiksi introvertti vs. ekstrovertti persoonallisuus) ja sosiaaliset suhteet (ystävyys, vihamielisyys ja empaattisuus) vaikuttavat agenteissa heräämiin emootioihin. AR-systeemin agenteilla on agenttikohtaisia tapoja näyttää tunteensa kasvonilmeillä, musiikin avulla tai syntetisoidun emotionaalisen puheen avulla. Tunteet ovat ulkoisten merkkien vuoksi muiden agenttien havaittavissa ja vaikuttavat osaltaan niiden tulevaan toimintaan. Agenteilla on tunnetilojen havaitsemisen lisäksi kyky järkeillä muiden agenttien tunnetiloista esimerkiksi seuraavaan tapaan [Elliot 1992, 5]:

> Tom oli vihainen autolleen kun se ei käynnistynyt (Tomin päämäärä ehtiä ajoissa tapaamiseen estyi). Tom ehkä potkaisi autoaan. Harry näkee tapahtuman muttei ymmärrä miksi Tom on vihainen autolleen, eihän autoa voi pitää vastuullisena tapahtuneesta. Harry kuitenkin tietää, että Tom pitää myös elottomia olentoja teoistaan vastuullisina olentoina. Harry kehottaa Tomia rauhoittumaan, tapaaminen ei varmaankaan ole niin tärkeä. Harry on pahoillaan ystävänsä epämieluisasta tunnetilasta.

Jos agentti ilman tunteiden järkeilykykyä havaitsisi edellä kuvatun tilanteen, kulkisi sen järkeily hyvin eri reittiä. Se luultavasti pitäisi tilannetta esimerkkinä siitä, että kulkuneuvoista täytyy pitää huolta jotta pääsee tapaamisiin ajallaan. Se ehkä kehottaisi Tomia korjauttamaan autonsa ja lähtemään ensi kerralla aikaisemmin tapaamiseen. Tällainen toiminta, vaikkakin rationaalista, ei välttämättä ole omiaan sosiaalisten suhteiden tai toisten agenttien tunnetilan parantamiseksi.

Agenteilla syntyvien tunteiden voimakkuuteen vaikuttavat ns. emootioiden intensiteettimuuttujat [Elliot and Siegle, 1993]. Intensiteettimuuttujat on jaettu kolmeen ryhmään: *simulaatiotapahtumamuuttujat* (riippumattomia tulkinnoista, vaikuttavat suoraan tapahtuman intensiteettiin), pysyvät, agentin *luonteeseen liittyvät muuttujat* (agenttikohtainen taipumus reagoida tiettyihin tapahtumiin tietyllä tavalla, muuttuvat hitaasti) ja *mielialariippuvaiset muuttujat* (epävakaita, kaksisuuntaisia muuttujia, joilla ei ole tiettyä agenttikohtaista perustasoa, arvo voi muuttua aikaisempien tunnekokemusten pohjalta).

AR:n toimintaa on simuloitu ”Taksimaailmassa” (TaxiWorld) [Elliot, 1992]. Taksimaailmassa on yli neljäkymmentä mahdollista tapahtumaa, jotka aiheuttavat tunteita herättäviä tilanteita (esim. onnettomuuksia, ruuhkia, sakkoja). Agenteille voidaan asettaa persoonallisuuksia neljästäkymmenestä eri vaihtoehdosta. Tilanteille on 150 mahdollista tulkintatapaa, jotka voivat yksistään tai useamman tulkinnan tuloksena herättää jonkin 24:stä eri tunnetyypistä. Jokaista tunnetyyppiä on mahdollista ilmaista noin 60:llä eri tavalla. Simulaation käyttäjällä on runsaasti mahdollisuuksia vaikuttaa simulaation kulkuun muuntelemalla mm. agenttien tulkintoja tilanteista, agenttien persoonallisuutta, tilanteeseen osallistuvia agentteja tai agenttien mielialaa. Simulaation käyttäjällä on mahdollisuus seurata simulaatioon osallistuvien agenttien tunnetiloja. Esimerkki simulaatiotilanteesta [Elliot, 1992, 20-21]:

> Taksikuski Tom ottaa kyytiin siivottoman näköisen asiakkaan. Tom pelkää tulevansa ryöstetyksi. Pelko aiheuttaa hänen kasvojensa punastumisen, änkyttämisen ja saa hänet sanomaan, ettei hänellä ole tarpeeksi polttoainetta viedäkseen asiakkaan tämän haluamaan paikkaan. Harry näkee tapahtuman. Hän ei tiedä, että Tom pelkää siivottoman näköisiä asiakkaita. Harrylla ei ole valmiina representaatiota tällaisesta tilanteesta. Harry on kuitenkin aikaisemmin nähnyt, kun toinen agentti punastui ja änkytti ja oppi silloin, että tämä oli esimerkki pelon ilmaisusta. Nyt Harry järkeilee, että Tom voisi myös pelätä tässä tilanteessa. Hän ”pohtii” tilanteita, jotka voisivat aiheuttaa pelkoa ja päätyy ajattelemaan, että Tomin päämääränä on luultavasi pitää rahansa ja olla turvassa. Siivottoman näköisten ihmisten tiedetään ryöstäneen taksikuskeja (vieneet rahat ja vaarantaneen turvallisuuden). Nyt Harrylla on selitys tapahtumalle. Tämän jälkeen Harry päivittää tietonsa Tomista (Tom haluaa pitää rahansa ja turvallisuutensa), joita hän testaa tulevaisuudessa .

Edellä kuvattu simulaatioesimerkki kuvaa hyvin järkeilykyvyn merkityksen sosiaalisissa tilanteissa toimimiselle. Vaikka Harry ei kyennytkään suoraan tapahtumasta arvioimaan, että Tom oli tilanteessa peloissaan, se pystyi järkeilemään tunnetilan käyttämällä hyväkseen aikaisemmin hankittua tietoa tunteista ja niiden ilmaisusta. Kun Harry nyt tietää, että Tom pelkää edellä kuvatuissa tilanteissa, se pystyy paitsi sopeuttamaan toimintaansa kohdatessaan vastaavia

tilanteita (ottaa kyytiin siivottoman näköisen asiakkaan jottei Tomin tarvitse), myös ennustamaan muiden agenttien tunnetiloja ja niistä seuraavaa käyttäytymistä (Harry asettaa Tomin tavoin käyttäytyvälle agentille ”oletuspersoonallisuudeksi” Tomin persoonallisuuden, jota se tarvittaessa muokkaa kokemustensa perusteella).

1.3. Velásquezin Cathexis-malli

Juan D. Velásquez [1997; 1998] on kehittänyt laskennallisen mallin tunteiden syntymisen simuloimiseksi, joka kognitiivisista malleista poiketen ottaa huomioon myös muita tunteiden syntyyn vaikuttavia tekijöitä. Velásquezin kehittämän Cathexikseksi nimetyn mallin taustalla on vaikutuksia paitsi psykologian teorioista, myös neuropsykologiasta, tekoälytutkimuksesta, ja etologiasta. Cathexis-mallissa tunteiden pohjalla oletetaan olevan neljän tasoisia tunteita herättäviä prosesseja: *hermostollisia*, *aistimellismotorisia*, *motivationaalisia* ja *kognitiivisia* [Izard, 1993]. Hermostollisiin prosesseihin kuuluvat muun muassa erilaiset aivojen välittäjäaineiden tasolla tapahtuvat muutokset ja aivojen lämpötilan muutokset. Aistimellismotorisiin prosesseihin kuuluvat esimerkiksi kasvonilmeet, kehon asento ja lihasten aktivaatio. Motivationaalinen systeemi käsittää kaikki tunteisiin johtavat motivaatiot: tarpeet, emootiot ja kivun säätelyn. Neljäntenä tunnetilojen pohjalla olevana systeeminä Cathexis käsittää kognition vaikutuksen. Cathexis-mallissa oli käytössä aiemmin (myös alla olevassa Simón the Toddler esimerkissä) kognitiivinen tulkintateoria [Roseman *et al.*, 1990]. Tällä hetkellä käytössä on kuitenkin niin sanottu Emotion Generation System, joka ei oleta minkään ennalta määrättyjen kognitiivisten tulkintamallien olemassaoloa (kuten esimerkiksi OCC-mallissa), vaan perustuu kognitiivisten tulkintojen käytännön oppimiseen (käytössä Virtual Yuppy esimerkissä).

Cathexis mallintaa tunteet erityisistä tunnesysteemeistä, ”proto-spesialisteista”, muodostuvan verkoston avulla. Proto-spesialisti termi on peräisin Marvin Minskyltä [1986]. Jokainen yksittäinen proto-spesialisti edustaa jotain tunneperhettä, joiden avulla voidaan mallintaa paitsi tunteita, myös mielialoja ja temperamenttia. Tunneperheitä on kuusi: viha, pelko, suru / mielipaha, onnellisuus / mielihyvä, inho ja yllätys. Jokaisen tunneperheen jäsenillä on samankaltaisia mekanismeja ja ominaisuuksia: ne heräävät vasteena samankaltaisiin edeltäviin tapahtumiin, ne aiheuttavat samankaltaisia tunneilmaisuja (mm. käyttäytyminen, ilmeet) ja tunteisiin liittyvä fysiologinen aktivoituminen on tunneperheen sisällä samankaltaista.

Proto-spesialisteilla on sensoreita, joiden avulla ne aistivat sekä ulkoisia (ympäristöstä tulevia) että sisäisiä (esim. palaute aistimellismotorisilta prosesseilta tai tarvetiloilta) ärsykkeitä ja pyrkivät niiden kautta tunnistamaan tuntei-

ta herättäviä tilanteita. Proto-spesialistit saavat syötteitä edellä mainituilta neljältä prosessilta (hermostollinen, aistimellismotorinen, motivationaalinen ja kognitiivinen) ja muilta proto-spesialisteilta vaikuttaen siten toistensa toimintaan. Jokaiseen proto-spesialistiin liittyy kaksi raja-arvoa, joista toinen kontrolloi *tunteen aktivoitumista* (esim. iloa kontrolloivan proto-spesialistin raja-arvon ylittyminen → syöte muille proto-spesialisteille ja käyttäytymistä säätelevälle systeemille, joka aiheuttaa esimerkiksi hymyn). Toinen raja-arvo määrittää proto-spesialistin *kylläisyyden* (tunteeseen liittyvä kiihtymys ei voi kohota yli tietyn rajan). Näiden lisäksi proto-spesialistien kontrolloimiin tunteisiin liittyy funktio, joka määrittää tunteen keston.

Proto-spesialistit toimivat siis verkostossa, jossa ne kaikki vaikuttavat toistensa toimintaan. Mikään yksittäinen proto-spesialisti ei kontrolloi verkoston toimintaa, vaan ne toimivat verkostossa rinnakkain, tasa-arvoisina. Kun jonkun proto-spesialistin tunteen aktivoitumisen raja-arvo kohoaa yli määrätyn rajan, voi kyseessä oleva proto-spesialisti kiihdyttää tai rajoittaa muiden proto-spesialistien toimintaa. Esimerkiksi systeemin ollessa pelon vallassa, onnellisuutta kontrolloivan proto-spesialistin toiminta on rajoitetumpaa (raja-arvot korkeammat kuin muulloin).

Cathexis-malli on erittäin monipuolinen, mallintaen edellisten lisäksi mm. mielialojen syntymisen, protospesialistien yhtäaikaisen aktivoitumisen (tunnesekoitukset), tunteiden intensiteetin vaihtelun ja tunteiden heikentymisen ajan myötä. Tilanpuutteen vuoksi näiden piirteiden esittely ei kuitenkaan tässä paperissa ole mahdollista.

Simón the Toddler

Simón on interaktiivinen agentti, jonka tarkoituksena on toimia Cathexis-mallin testausalustana. Simónilla on proto-spesialisteja viidelle eri tarpeelle (nälälle, janolle, lämmönsäätelylle, väsymykselle ja kiinnostukselle) ja kuudelle tunteelle (onnellisuus, surullisuus, pelko, viha, inho ja yllättyneisyys). Simónin käyttäytymisrepertuaariin kuuluvat mm. nukkuminen, leikkiminen, syöminen, juominen, halaaminen ja itkeminen. Simónin käyttäytyminen ilmenee kasvonilmeinä ja ääninä. Käyttäjä voi Simónin kanssa vuorovaikutuksessa ollessaan vaikuttaa monipuolisesti agentin olosuhteisiin. Käyttäjän on mahdollista muuttaa vaikkapa Simónin keskushermoston välittäjäaineiden määriä, syöttää häntä tai muuttaa huoneen valaistusta. Simón reagoi käyttäjän toimiin tunteita aiheuttavien (hermostollisten, aistimellismotoristen, motivationaalisten ja kognitiivisten) prosessien tilan perusteella.

Algoritmi joka tuottaa Simónin tunnetilat ja käyttäytymisen etenee seuraavasti:

1. Sisäiset muuttujat (motiivit) ja ympäristön muuttujat arvioidaan.

2. Agentin motiivit päivitetään.
3. Käyttäytymismahdollisuuksien arvot päivitetään sisäisten ja ulkoisten muuttujien perusteella.
4. Käyttäytyminen jolla suurin arvo aktivoituu.

Käyttäytymiseen liittyvä ilmaisullinen puoli muuttaa agentin kasvonilmeitä ja tarvittaessa ääntä. Käyttäytymiseen liittyvä kokemuksellinen puoli päivittää kaikki tarvittavat motiivit (esim. tietyn tarpeen täyttyminen laskee motiivin tärkeyttä). [Velásquez, 1997]

Virtual Yuppy

Yuppy on toinen Cathexis-mallin toteutus, joka on tehty sekä virtuaalisena (tietokoneen ruudulla ohjelman avulla käytettävänä), että fyysisenä robottina. Yuppy on lemmikkirobotti, joka pystyy aistimaan ympäristöään yksinkertaisen konenäön ja kosketusta aistivien sensorien avulla. Yuppylla on neljä tarvetta, virransäätely, lämpötilan säätely, väsymyksen säätely ja uteliaisuus. Yuppyn tunnesysteemi on edellä kuvattuihin tunneperheisiin perustuva (tunneperheet viha, pelko, suru / mielipaha, onnellisuus / mielihyvä, inho ja yllätys) ja sen tunteet siis heräävät joko sisäisten (motivaatio ym.) tai ulkoisten tekijöiden seurauksena (esim. valoisuuden vaihtelu). Yuppyn käyttäytymisrepertuaari käsittää noin 20 erilaista käyttäytymistä, esimerkiksi luun etsiminen, akun lataaminen, säpsähdys, ihmisen lähestyminen ja tunteen ilmaisu. Käyttäjä voi käyttää robottia muuttelemalla sen tunteiden ilmaisutyyliä (raja-arvojen muuttaminen, proto-spesialistien välisten yhteyksien muuttaminen) ja stimuloimalla agenttia jollain tavoin joko sisäisesti (esimerkiksi välittäjäaineiden tason muuttaminen) tai ulkoisesti (esim. näyttämällä luuta tai silittämällä).

Yuppy tuottaa siis tunteisiin liittyvää käyttäytymistä kohdatessaan tunteita herättäviä tilanteita. Jos Yuppylla on aktiivisena esimerkiksi uteliaisuuden tarve, se liikkuu ympäriinsä ja etsii luuta. Jos se löytää luun, onnellisuustunneperhe aktivoituu ja samalla aktivoituu myös onnellisuuteen liittyvä käyttäytyminen (esimerkiksi häntä heiluu). Yuppy myös oppii ns. toissijaisia tunteita. Esimerkiksi löydettyään ihmisen, jolla on Yuppyn luu, Yuppy lähestyy kyseistä ihmistä (luuta). Riippuen ihmisen käyttäytymisestä (lyö vs. silittää lähestyvää robottia), Yuppy luo mallia ihmisistä ylipäätään. Näiden mallien avulla se tulevaisuudessa valitsee tilanteisiin sopivat käyttäytymistavat. [Velásquez, 1998]

2. Pohdintaa

Ortonyn, Cloren ja Collinsin [1986] OCC-mallin rajoituksena varsinaisena tunteiden syntyä selittävänä teoriana on se, että se jättää suuren joukon emootioiden synnyn kannalta olennaisia asioita huomiotta. Teoria ei esimerkiksi ota lainkaan huomioon fysiologisten seikkojen merkitystä emootioiden synnyn selittämisessä. Myös Frijdan [1986] funktionaalinen tunneteoria on kognitiivinen teoria, jonka mukaan nimenomaan tilanteiden kognitiivinen arviointi tuottaa syntyvän tunnetilan. Vaikka tunneteorioina kaksi edellä mainittua olisivatkin hieman kapea-alaisina, ovat ne silti osoittautuneet emootioiden syntetisoinnin kannalta erittäin hedelmällisiksi lähestymistavoiksi ja tärkeiksi nimenomaan syntetisoinnin taustateorioina. Velásquezin [1997] Cathexis mallintaa edellisistä poiketen kognitiivisten seikkojen lisäksi myös fysiologian osuuden tunnetilojen tuottajina. Tämä piirre on mielenkiintoinen ottaen huomioon, että nimenomaan Cathexis-malli on esitellyistä se, joka on kehitetty tunteiden *syntetisointia* varten, ei yleisesti tunteiden syntyä selittäväksi teoriaksi.

Affective Reasoner, joka siis pohjautuu OCC-malliin, kykenee erittäin vaativaan sosiaalisten suhteiden simulointiin, kuten Taksimaailman esimerkki Harryn empatianomaisesta tunteiden järkeilykyvyistä osoitti. Tämä ominaisuus saattaa mahdollistaa Affective Reasonerin tyyppisen tunteiden syntetisointiin pystyvän systeemin toteuttamisen sellaisena, että se voi järkeillä myös käyttäjän tunnetilan [Elliot, 1994]. Vaikka fysiologiset signaalit voivat osaltaan tuottaa tietoa käyttäjän tunnetilasta, olisi käyttäjän tunnetilan järkeily askel kohti vieläkin adaptiivisempaa ihminen-tietokone vuorovaikutusta. Ei ainoastaan tunnetilan järkeily, vaan kyky siihen yhdessä käyttäjästä (hänen toiminnoistaan ja tunnereaktioista suhteessa niihin) luotuun persoonallisuusprofiiliin tuottaa erittäin vahvan pohjan todella henkilökohtaisesti profiloituihin sovelluksiin (esimerkiksi agentti joka osaa päätellä, tai ainakin oppii käytännöstä, milloin käyttäjää on soveliasta häiritä ja millä asioilla).

Elliotin [1994] mukaan tulevia emootioiden syntetisointia hyväksikäyttäviä sovelluksia voisivat olla muun muassa seuraavat: sovellus, joka osaisi kategorisoida käyttäjän emootioita ja vastata niihin asianmukaisesti, ottaa emotionaalisesti ”tietoiset” agentit mukaan tietokoneavusteisiin opetusohjelmiin, käyttää agentteja sosiaalisten tilanteiden simulointiin opetustarkoituksessa (esimerkiksi apuna tutoroinnissa tai psykoterapiassa käyville sosiaalisten taitojen puutteesta kärsiville asiakkaille) ja käyttää emotionaalisia agentteja tietokonepeleissä. Nämä sovellukset ovat luonnollisesti vain esimerkkejä mahdollisuuksista, joita tunteiden syntetisoinnin mukaan tuominen ihminen-tietokone vuorovaikutukseen mahdollistaisi. Nykyisin tehdään paljon tutkimusta

mahdollisista menetelmistä tunnistaa käyttäjän tunnetila psykofysiologisten mittausten avulla. Näiden indikaattorien rinnalle edellä esitellyt teoriat toisivat uuden mahdollisuuden: tietokone voisi tunnistaa käyttäjän tunnetilan fysiologian perusteella ja järkeillä sen jälkeen, mikä tai mitkä olosuhteet näyttävät aiheuttavan tämän tilanteen. Jos tunnetila ei ole ennen ilmennyt kyseisessä tilanteessa, voi tietokone ehkä päätellä jotain käyttäjän mielialasta. Lähinnä rajoja mahdollisten sovellusten keksimiselle aiheuttaa rajallinen mielikuvitus ja käytännön rajoitteiden huomioiminen.

Edellä kuvatut teoriat ja niiden pohjalta kehitetyt sovellukset havainnollistavat mahdollisuuksia, joita tunteiden syntetisointiin liittyy. Kuten tutkielmassa esitellyt esimerkkisovellukset osoittavat, kaikki tässä paperissa kuvatut taustateoriat ovat pystyneet tuottamaan ainakin jossain suhteissa onnistuneita tunteiden syntetisointiin pystyviä tietokoneohjelmia. Tietokone näyttäisi siis pystyvän ainakin joissain rajallisissa tilanteissa varsin aidon tuntuisiin tunnereaktioihin. Toisaalta tunteiden syntetisointi on suhteellisen tuore ilmiö ja tutkimuksen tehtävänä on vielä selvittää, mihin kaikkeen tietokoneen tuottamat tunnetilat soveltuvat ja mitä kaikkia tekijöitä tunteiden syntetisoinnin taustateorian tulee ottaa huomioon mahdollisimman toimivan tunteiden syntetisoinnin mahdollistamiseksi.

Viiteluettelo

[Bates 1994] Joseph Bates, The role of emotion in believable agents. *Communications of the ACM*, 37, 7, 122-125.

[Bates, Loyall and Reilly 1992a] Joseph Bates, Bryan A. Loyall, and Scott W. Reilly, An architecture for action, emotion, and social behavior. *Proceedings of the Fourth European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, S. Martino al Cimino, Italy.

[Bates, Loyall and Reilly 1992b] Joseph Bates, Bryan A. Loyall, and Scott W. Reilly, Integrating Reactivity, Goals, and Emotion in a Broad Agent *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, Bloomington, Indiana, July 1992.

[Elliot 1992] *The affective reasoner: A process model of emotions in a multi-agent system*. Ph.D. Dissertation, Northwestern University.

[Elliot 1994] Clark Elliot, Research problems in the use of a shallow artificial intelligence model of personality and emotion. *Proceedings of the 12'th AAAI National Conference on Artificial Intelligence.*

[Elliot and Siegle 1993] Clark Elliot and Greg Siegle, Variables influencing the intensity of simulated affective states. In *AAAI technical report for the Spring symposium on reasoning about mental states: Formal theories and applications*, 58-67. Menlo Park, California: AAAI Press.

[Frijda, 1986] Nico H. Frijda, *The Emotions.* New York: Cambridge University Press.

[Frijda and Swagerman, 1987] Nico H. Frijda and Jaap Swagerman, Can computers feel? Theory and design of an emotional system. In *Cognition and Emotion*, 1(3), 235-257.

[Izard, 1993] Carroll E. Izard, Four systems of emotion activation: Cognitive and noncognitive processes. *Psychological review*, 100, 1, 68-90.

[Minsky, 1986] Marvin Minsky, *The Society of Mind.* New York: Simon & Schuster.

[Ortony, Clore, and Collins, 1988] Andrew Ortony, Gerald, L. Clore, and Allan Collins, *The Cognitive Structure of Emotions.* Cambridge: Cambridge University Press, 1988.

[Picard, 1997] Rosalind Picard, *Affective Computing.* Massachusetts: The MIT Press, 1997.

[Reeves and Nass, 1996] Byron Reeves and Clifford Nass, *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places.* California: CSLI Publications.

[Roseman *et al.*, 1990] Ira J. Roseman, Martin S. Spindler, and Paul E. Jose, Appraisals of emotion-eliciting events: Testing a theory of discrete emotions. *Journal of Personality and Social Psychology*, 59(5), 899-915.

[Velásquez 1997] Juan D. Velásquez, Modeling Emotions and Other Motivations in Synthetic Agents. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*. Providence, RI: MIT / AAAI Press.

[Velásquez 1998] Juan D. Velásquez, When Robots Weep: Emotional Memories and Decision-Making. *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*. Madison, WI: MIT / AAAI Press.

Toiminnan teorian esittely

Mikko Hartikainen ja Saija Patomäki

Tämä työ on tarkoitettu yleissivistäväksi katsaukseksi toiminnan teoriaan, jota voidaan soveltaa ihmisen käyttäytymisen kuvaamiseen myös käyttöliittymien yhteydessä. Käsittelemme laaja-alaisesti erityisesti toiminnan teorian rakennetta sekä sen mahdollisia käyttöalueita. Luomme syvällisen katsauksen myös teorian syntyhistoriaan. Työmme tarkoituksena on esitellä toiminnan teoriaa toisille seminaarilaisille, niille jotka ovat siitä kuulleet jonkin verran ja niille, joille teoria on uusi tuttavuus.

1. Johdanto

Käytettävyystutkimuksessa on jatkuva tarve analysoida ihmisen käyttäytymistä. Jotta voidaan ennustaa käyttäjien toimintaa, täytyy perehtyä ihmiseen ja ihmisluontoon. Psykologian eri koulukunnat tarjoavat työkaluja tähän analysointiin. Toiminnan teoria on Neuvostoliitossa kehitetty malli, jota on hyödynnetty käytettävyystutkimuksessa ja tullaan hyödyntämään jatkossakin.

Toiminnan teoria ei ole teoria sanan varsinaisessa merkityksessä, vaan se tarjoaa käsitteellisen viitekehyksen pohdittaessa ihmisen toimintaa. Toiminnan teoria tarkastelee toimintaa lähtökohtanaan ihminen ja hänen ympäristönsä vuorovaikutus ja se korostaa toiminnan kokonaisvaltaisuutta ja eheyttä. Näin ollen toiminnan teoria tarjoaa kattavan mallin sovellustieteisiin. Toiminnan teoriaa onkin sovellettu laajalti eri tieteissä, kuten psykologiassa [Leiman, 1994], lingvistiikassa [Sharpe, 1989], [Junefelt, 1995] ja kasvatustieteessä [Center for Activity Theory and Developmental Work Research].

Tässä seminaarityössä käsittelemme laaja-alaisesti erityisesti toiminnan teorian rakennetta sekä sen mahdollisia käyttöalueita. Luomme syvällisen katsauksen myös teorian syntyhistoriaan. Työmme tarkoituksena on esitellä toiminnan teoriaa toisille seminaarilaisille, niille jotka ovat siitä kuulleet jonkin verran ja niille, joille teoria on uusi tuttavuus.

Olemme jakaneet seminaarityömme viiteen lukuun ja alalukuihin. Seuraavassa luvussa, eli toisessa luvussa kerromme toiminnan teorian historiasta keskeisen. Luvussa 3. esittelemme toiminnan teorian itsensä. Luku jakautuu alalukuihin. Alaluvussa 3.1. kuvaamme toiminnan teoriaa yleisellä tasolla. Alaluvuissa 3.2. - 3.6. käymme läpi toiminnan teorian viisi perusperiaatetta, jotka ovat tavoitteellisuus, toiminnan hierarkkinen rakenne, sisäistys ja ulkoistus, välineiden vaikutus ja kehittyminen. Näistä osista toiminnan teorian kokonaisuus rakentuu. Tätä yhdistelyä tarkastelemme alaluvussa 3.7. Perusperiaatteet toimivat myös toiminnan teorian käytännön soveltamisen eli toiminnan tarkistuslistan rakennuspalikoina. Tarkistuslistan käsittely tapahtuu alaluvussa 3.8., jossa selvitetään käytännönläheisemmin teorian käyttöä reaalielämässä. Neljännen luvun omistimme käytännön esimerkkien esittelylle. Toivomme näiden esimerkkien selvittävän sitä, mitä toiminnan teorialla voidaan tehdä ja miten sitä käytetään. Lopuksi luvussa 5. pohdimme hiukan toiminnan teoriaa yleisellä tasolla, eli luvun nimi on keskustelu.

2. Toiminnan teorian historiaa

Toiminnan teorian kehitti joukko Neuvostovenäläisiä psykologeja 1920- ja 1930-luvuilla. Näiden psykologien, eli lähinnä Lev Vygotskyn (1896-1934) ja hänen kahden kollegansa, A. R. Lurian (1902-1977) ja A. N. Leontevin (1903-1979), tarkoituksena oli kehittää vallitsevia psykoanalyyttisiä ja behavioristisiä näkemyksiä kattavampi psykologinen suuntaus. [Center for Activity Theory and Developmental Work Research], [Ryder, 2001]

Toisaalta Pavlovilla oli suuri vaikutus syntyisin valkovenäläisen Vygotskyn ajatteluun. Pavlov oli tunnetusti behavioristisesti suuntautunut psykologi, joka ensimmäisenä toi esille ajatuksen ehdollisista reflekseistä. Pavlovin työ ja behaviorismi yleensäkin oli puutteellinen vastaamaan tietoisuuden vaikutukseen käyttäytymisessä, eikä Vygotskykään ehtinyt lyhyen elinikänsä takia vastaamaan kunnolla tähän haasteeseen.

[http://www.marxists.org/glossary/people/v/y.htm#vygotsky-lev]

Vygotskyn teoksia julkaistiin vasta hänen kuolemansa jälkeen vuodesta 1934 lähtien. Länsimaissa Vygotskyä julkaistiin ensimmäisen kerran vasta vuonna 1958. Toiminnan teorian esivaikuttajan roolia paremmin hänet tunnetaan lähivyöhykeajatuksestaan, joka on suosittu erityisesti tietyissä yhdysvaltalaisissa psykologipiireissä.

[http://www.marxists.org/glossary/people/v/y.htm#vygotsky-lev]

Tärkeä lähtökohta tominnallisuusteorian kehittämisen tukemiseen oli halu muodostaa Marxilaiseen filosofiaan perustuva psykologian suuntaus. Toiminnan teoria ei varsinaisesti kuitenkaan perustu Marxilaiseen filosofiaan, vaan syntyi ennemminkin suuremman tavoitteen seurauksena. Tätä suuntausta kehitettäessä neuvostopsykologit eivät kuitenkaan tahtoneet päästä yhteisymmärrykseen monistakaan asioista, mutta eräs ensimmäisistä periaatteista, joista he sopivat, oli "tietoisuuden ja toiminnallisuuden eheyden ja erottamattomuuden periaate". Tämä periaate tarkoittaa sitä, ettei ihmismieli ole olemassa, se ei kehity eikä sitä voi ymmärtää ilman mielekästä, tavoitteellista ja yhteisöllisesti suuntautunutta vuorovaikutusta ihmisten ja ympäröivän todellisuuden välillä. [Ryder, 2001], [Bannon, 1997]

Tietoisuuden ja toiminnallisuuden eheyden ja erottamattomuuden periaatetta muokattiin sitten myöhemmin. Sergei Rubinshtein kehitti nimenomaan ajatuksen ihmisen toiminnasta psykoanalyysin pohjalta. Vygotsky keskittyi tutkimuksessaan kulttuurillis-historialliseen psykologiaan. Hänen oppilaansa Aleksei Leontev loi raamit toiminnan teorialle käsitteiden määrittelyllä, joka on hyvin lähellä Vygotskyn kulttuurillis-historiallisia käsityksiä. Toiminnan teoria ei ole kuitenkaan synonyymi kulttuurillis-

historialliselle psykologialle. Toiminnan teorian syntyhetken jäljittämistä hankaloittaa entisestään se, ettei termillä "activity theory" aina tarkoiteta Leontevin käsitteistöä. [Bannon, 1997]

Erään lähteen mukaan Leontevin ja Vygotskyn tiet erosivat Vygotskyn viimeisinä elinvuosina ja ilmapiiri Neuvostoliitossa muuttui kielteiseksi vygotskylaiselle ajattelulle. Lurian luovuttua psykoanalyysistä ja viimeistään 1963 Leontevin saatua Lenin-palkinnon alkoi Leontevin käsitykset Vygotskystä saada jopa enemmän kannatusta kuin Vygotsky itse. Lähde [Robbins, 2000] on kuitenkin mielestämme hiukan arvelluttava, joten kaikkea tätä emme voi varmasti allekirjoittaa. Selvää on kuitenkin, että Leontevin saama Lenin-palkinto on varmasti vaikuttanut siihen, että juuri tähän kaikista kulttuuris-historiallisen psykologian seuraajista on kiinnitetty niinkin paljon huomiota.

Nykyään toiminnan teoria ei missään nimessä ole venäläisten yksinoikeus. Aihetta tutkitaan ja sovelletaan ympäri maailmaa, ensisijaisesti Euroopassa ja Yhdysvalloissa [Bannon, 1997]. Suomi on mukana toimnnallisuusteorian soveltajien kärkiporukassa. Helsingin yliopistossa kasvatustieteen laitoksen yhteydessä toimii Toiminnan teorian ja kehittävän työntutkimuksen yksikkö, jonka johtaja professori Yrjö Engeström on julkaissut aiheesta paljon maailmallakin tunnettua kirjallisuutta. Vuodesta 1995 lähtien kyseisessä yksikössä on toiminut Kehittävän työntutkimuksen ja aikuiskoulutuksen tohtoriohjelma, jossa on mukana 35 päätoimista opiskelijaa. [Center for Activity Theory and Developmental Work Research]

3. Teorian käsitteiden esittely

3.1. Kuvaus toiminnan teoriasta (Activity Theory)

Toiminnan teoriassa tietokoneita pidetään yksinkertaisesti vain välineinä, jotka tukevat ihmisen toimintaa. Tämän vuoksi katsotaan, että täydellinen ymmärrys siitä, kuinka suunnitella, arvioida, toteuttaa ja räätälöidä teknologiaa ihmisen käyttöön sopivaksi voidaan saavuttaa ainoastaan ymmärtämällä teknologiatuettua ihmisen toimintaa ja pohtimalla sitä kuinka teknologia voitaisiin parhaiten yhdistää näihin toimintoihin. Toiminnan teoria tarjoaa laajan käsitteellisen viitekehyksen, jonka avulla voidaan kuvata tietokonetuetun toiminnan rakennetta, kehitystä ja kontekstia.

Vasta viime vuosina käyttöliittymäasiantuntijat ovat alkaneet arvostaa tietokonetuetun toiminnan ja koko kontekstin ymmärtämisen tärkeyttä. Tällainen kokonaisvaltainen ymmärrys vaikuttaa suoraan tuotteiden suunnitteluun ja arvioitiin. Jo tuotteen suunnitteluprosessin kuluessa pyritään selvittämään mitä käyttäjät todella tuotteella tekevät ja kuinka teknologia saataisiin käyttäjän näkökulmasta tehokkaimmin käyttöön.

Toiminnan teoria on yleinen, käsitteellinen lähestymistapa ongelman käsittelyyn. Toiminnan teorian analyysin perusyksikkö on itse toiminnallisuus, joka koostuu subjektista, joka voi olla yksilö tai ryhmä, objektista tai motiivista, artefakteista (ihmiskäden aikaansaamista tuotteista) ja sosio-kulttuurisista säännöistä. Jos haluamme todella ymmärtää ihmisen toimintaa, mitään näistä toiminnallisuuden osasista ei voida jättää pois, sillä kullakin niistä on tärkeä ja oleellinen rooli inhimillisessä toiminnassa.

Kaksi perusideaa elävöittää toiminnan teoriaa: ihmismieltä voidaan ymmärtää vain siinä kontekstissa kuin se on vuorovaikutuksessa muun maailman kanssa ja tämä vuorovaikutus eli toiminnallisuus on sosiaalisesti ja kulttuurisesti määrittyvää. Näistä perusideoista on kehitetty toiminnan teorian viisi perusperiaatetta: tavoitteellisuus, toiminnan hierarkkinen rakenne, sisäistys ja ulkoistus, välineiden vaikutus ja kehittyminen.

3.2. Tavoitteellisuus (object-orientedness)

Tavoitteellisuuden periaate on yksi tärkeimmistä periaatteista toiminnan teoriassa. Motiivit ovat periaatteessa tavoitteita, mutta tavoitteelle on olemassa toinenkin merkitys; tavoite on mahdollinen lopputulos, objekti, jonka saavuttamiseksi kaikki toiminta suunnataan. Esimerkiksi ohjelmoijan toiminnan tavoitteena on saada aikaan tietokoneohjelma. Ohjelma ei siis ole ohjelmoijan motiivi, vaan ohjelmaa kohtaan ohjelmoija on suunnannut kaiken toimintansa, jotta hänestä esimerkiksi tulisi maailman paras ohjelmoija, joka on hänen perimmäinen motiivinsa. Näin tietokoneohjelmasta tulee osa sosiaalista todellisuutta, sen muotoon vaikuttavat tarkat ja ennalta tunnetut lait, säännöt ja vaatimukset, jotka ohjelmoijan yhteisö on säätänyt. Tavoitteet voivat olla fyysisiä asioita tai aatteellisia asioita. Leontevin mukaan psykologiassa tavoitteen eli objektin käsitettä ei voida rajoittaa ainoastaan fyysisiin, kemiallisiin tai biologisiin tavaroihin. Tavoitteellisuuden periaatteen mukaan ihmiset elävät todellisuudessa, jonka esineet ja asiat yhdessä muodostavat. Esineet, jotka muodostavat ulkoisen todellisuuden, eivät ole ainoita todellisuuden objekteja toimintateoriassa, vaan objektin katsotaan olevan paljon laajempi käsite. Objekteja toiminnan teoriassa ovat fyysisten esineiden lisäksi myös sosiaaliset ja kulttuuriset ominaisuudet [Nardi & Kaptelinin, 2000].

3.3. Toiminnan hierarkkinen rakenne

Toiminnan teoriassa tutkimuksen perusyksikkö on itse toiminnallisuus. Leontev, joka kuuluu toiminnan teorian pääarkkitehteihin, kuvailee toiminnallisuuden muodostuvan neljästä eri komponentista eli toimijasta (subject), tavoitteesta (object), toiminnasta (action) ja operaatioista (operation). Toimija voi olla joko yksittäinen henkilö tai ryhmä, joka on sitoutunut

kyseiseen toiminnallisuuteen. Toimijalla on tavoite, joka motivoi toimijaa toiminnallisuuteen ja antaa samalla toiminnallisuudelle erityisen suunnan. Leontevin mukaan tavoitteen taustalla vaikuttaa aina tarve tai halu, jota toiminnallisuus tyydyttää [Nardi & Kaptelinin, 2000].

Toiminnat ovat tavoitteellisia tapahtumia, joihin toimijan täytyy sitoutua, jotta tavoite saavutettaisiin. Toiminnan suorittaminen on siis tietoista toimintaa, koska toimijan on koko toiminnan suorittamisen ajan pidettävä päämäärä mielessään ja ohjattava toimintaansa sen mukaan. Toiminnallisuus koostuu siis toiminnoista ja toiminnot voivat olla keskenään hyvinkin erilaisia, vaikka ne samaan aikaan tähtäävät samaan tavoitteeseen. Esimerkiksi ihminen, jolla on tavoitteena hankkia ruokaa, ei voi suoraan ryhtyä kyseiseen toiminnallisuuteen, vaan hänen on ennen sitä suoritettava useita eri toimintoja. Ensimmäisenä hänen tavoitteenaan voi olla metsästysaseen hankkiminen tai tekeminen. Tekemällään aseella hän voi itse hankkia ruokansa tai panna jonkun muun ihmisen asialle. Kummassakin edellä mainitussa tapauksessa se mikä saa ihmisen toimimaan, eli ruoan hankkiminen, ja se mihin hänen tehtävänsä suuntautuu, eli aseen tekeminen, eivät ole yhtäpitäviä. Päätavoitteella on siis usein monia välitavoitteita, joilla edelleen voi olla välitavoitteita ja ilman näiden välitavoitteiden suorittamista ei päästä päätavoitteeseen. Esimerkiksi ennen kuin kyseinen ihminen voi tehdä metsästysaseen hänen on hankittava siihen materiaalit ja tekemiseen tarvittavat työkalut [Nardi & Kaptelinin, 2000].

Kun tehtävien hierarkiassa liikutaan aina vain alaspäin, lopulta ylitetään tietoisten ja automaattisten prosessien raja. Toiminta koostuu näistä automaattisista prosesseista, joita kutsutaan operaatioiksi. Käytännön kautta operaatioista tulee tiedostamattomia ja rutinisoituja toimenpiteitä. Operaatioilla ei ole omia päämääriään, ne pikemminkin sopeuttavat kulloisetkin toiminnat vallitseviin olosuhteisiin. Esimerkiksi, kun ihminen opettelee ajamaan autoa, vaihteiden vaihtaminen on hänelle tietoinen, yksittäinen tehtävä, jolla on selkeä päämäärä. Myöhemmin, harjaantumisen myötä, vaihteiden vaihtamisesta tulee pelkkä operaatio ja sitä ei enää voida pitää tavoitteellisena, päämärähakuisena toimintana. Operaatiot ovat riippuvaisia olosuhteista, joissa tehtävä suoritetaan, sillä jos päämäärä pysyy samana, ainoa asia mikä muuttuu on toiminnan operationaalinen rakenne [Nardi & Kaptelinin, 2000][Nardi, 1996].

Toiminnan teorian mukaan toiminnallisuuden komponentteja ei voida muutella paitsi jos toiminnallisuuden komponentit vaihtuvat dynaamisesti olosuhteiden vaihdellessa. Tämä on merkittävä näkemysero toiminnan teorian ja kognitiotieteen skeemojen, kuten GOMSin kanssa. Toiminnan teoriassa kaikki tasot voivat liikkua ylös- tai alaspäin, kuten näimme esimerkissä auton vaihteiden vaihtamisessa, missä autoilijan totuttua vaihteiden vaihtamiseen, tehtävästä tuli operaatio. Vastavuoroisesti operaatiosta voi tulla toiminta,

esimerkiksi jos olosuhteet häiritsevät tehtävän toteuttamista aiemmin muodostettujen operaatioiden avulla. Esimerkiksi jos käyttäjän yleensä käyttämä sähköpostiohjelma lakkaa toimimasta ja hänen on yllättäen käytettävä hänelle vierasta ohjelmaa tuttu operaatio muuttuu jälleen toiminnaksi. Tässä tapauksessa päämäärä siis pysyy samana, mutta toiminnot ja operaatiot muuttuvat kun olosuhteet muuttuvat [Nardi & Kaptelinin, 2000].

3.4. Sisäistys ja ulkoistus

3.4.1. Mentaaliprosessit vs. ulkoinen toiminta

Toiminnan teoriassa puhutaan sisäisestä ja ulkoisesta toiminnallisuudesta. Sisäisellä toiminnallisuudella toiminnan teoriassa tarkoitetaan samaa kuin perinteisessä kognitiotieteessä mentaaliprosesseilla. Toiminnan teoria kuitenkin erityisesti painottaa ettei sisäistä toiminnallisuutta voida kunnolla ymmärtää jos se analysoidaan erossa ulkoisesta toiminnallisuudesta, koska sisäisen ja ulkoisen toiminnallisuuden muuntuminen on jatkuvassa, molemminpuolisessa vuorovaikutuksessa toiminnallisuuden aikana. Sisäistämisestä puhutaan silloin, kun ulkoinen toiminnallisuus muuttuu sisäiseksi toiminnallisuudeksi. Toiminnan teoria korostaakin ettei sisäistyksessä ole kyse vain ja ainoastaan mentaaleista prosesseista, vaan sisäistys on kokonaisvaltainen prosessi, johon vaikuttaa myös monet ulkoiset seikat kuten motorinen toiminta ja työkalujen käyttö. Esimerkiksi lapsen opetellessa laskemaan yksinkertaisia laskutoimituksia, hän voi aluksi käyttää sormia apunaan. Heti kun lapsi on sisäistänyt aritmetiikan tarpeeksi hyvin, laskutoimitukset voidaan suorittaa suoraan päässä eikä ulkoista apua eli sormia enää tarvita havainnollistamaan laskutoimitusta [Nardi & Kaptelinin, 2000].

Sisäistämisen taidosta on huomattavaa hyötyä, sillä se tarjoaa keinoja moniin tilanteisiin. Ihminen voi mielessään yrittää ratkaista ongelmaa kuvittelemalla, tekemällä mentaalisimulaatioita ja miettiä vaihtoehtoisia ratkaisumalleja ilman, että hänen tarvitsee toteuttaa niitä ulkoisesti todellisuudessa erilaisten esineiden avulla. Joissakin tapauksissa ulkoisten komponenttien poisjättäminen tekee toiminnasta paljon tehokkaampaa, kun toiminnallisuus tapahtuu vain ja ainoastaan aivoissa. Esimerkiksi hyvä päässälaskutaito on paljon nopeampi ja kätevämpi käyttää kuin ulkoisten komponenttien hyväksi käyttö, joita voivat olla sormet tai taskulaskin [Nardi & Kaptelinin, 2000].

Ulkoistamisessa muunnetaan sisäinen toiminnallisuus ulkoiseksi. Ulkoistaminen on usein tarpeellista siinä vaiheessa, kun sisäistettyä toimintaa täytyy parannella tai muuttaa. Esimerkiksi kun päässälasku ei tunnu luonnistuvan eli kun lasku on liian vaikea, se täytyy suorittaa ulkoisia

komponentteja eli kynää ja paperia tai laskinta hyväksi käyttäen. Ulkoistamista tarvitaan usein myös silloin kun toimitaan yhteistyössä muiden ihmisten kanssa. Ryhmätyötä tehtäessä sen jäsenten on pystyttävä ulkoistamaan ajatusmallinsa ja selittämään ne muille, jotta yhteistyö ryhmässä sujuisi. Samaan aikaan kun sisäistämisen käsite on saanut laajaa huomiota kognitiotieteissä, ulkoistamisen käsitettä ei ole siinä noteerattu juuri lainkaan. Toiminnan teoria sen sijaan painottaa kummankin puolen, niin sisäistyksen kuin ulkoistuksenkin tärkeyttä sekä niiden välistä alituista muuntumista ja yhteisvaikutusta. Toiminnan teorian mukaan sisäistäminen ja ulkoistaminen ovat keskeisiä ihmisen kognition ja toiminnallisuuden perusosia [Nardi & Kaptelinin, 2000].

3.4.2. Oppijan sisäiset (intra-subjective) ja muiden ihmisten väliset (inter-subjective) älylliset kyvyt

Vygotskin mukaan älyllisten kykyjen kehityksessä on kaksi tasoa. Ensinnäkin älylliset kyvyt kehittyvät älyllisiksi toiminnoiksi ihmisten välillä (inter-subjective), jolloin nämä älylliset kyvyt jaetaan oppijan ja muiden ihmisten välillä. Tämän jälkeen jaetuista kyvyistä tulee sisäisiä (intra-subjective), oppijan omia mentaalisia prosesseja. Tiedon omaksuminen ja jakaminen on siis hyvin sosiaalinen tapahtuma [Nardi & Kaptelinin, 2000].

Oppijan sisäisten ja muiden ihmisten välisten älyllisten kykyjen dimensio on yhtenevä mentaaliprosessien ja ulkoisen toiminnan dimension kanssa. Ihmisen toiminnallisuuden dynamiikka koostuu näiden kahden dimensioiden sisältämien ääripäiden välisestä muuntumisesta ja keskinäisestä vuorovaikutuksesta. Sosiaalisten toimintojen sisäistäminen on vahva vaikutin yksilönkehityksessä. Ulkoistaminen voi taas olla omien mentaalitoimintojen muuttamista tai normaalin toiminnan osa, joka yhdistää yksilön hänen sosiaaliseen ympäristöönsä [Nardi & Kaptelinin, 2000].

3.5. Välineiden vaikutus

Koska toiminnan teoria painottaa sosiaalisia tekijöitä ja ihmisten välistä vuorovaikutusta keskenään sekä ympäristönsä kanssa, välineiden vaikutuksen periaate on keskeinen teoriassa. Ensinnäkin työväline, jota ihminen käyttää, määrää kuinka ihminen ja ympäristö vaikuttavat toisiinsa. Sisäistyksen ja ulkoistuksen periaatteen mukaisesti ulkoisen toiminnallisuuden harjoittaminen lopulta vaikuttaa mentaaliprosessien muotoutumiseen. Toiseksi, työvälineet yleensä kuvastavat käyttäjälle, mitä työkaluja muut ihmiset ovat lopulta keksineet tai parannelleet, kun he ovat painiskelleet samantyyppisten ongelmien parissa. Tämä kokemus näkyy työkalun rakenteellisissa ominaisuuksissa, joita ovat esimerkiksi muoto ja materiaali, sekä tietona siitä

kuinka työkalua kuuluu käyttää. Työvälineitä tehdään ja muokataan uudelleen itse toiminnallisuuden kehityksen aikana, näin työvälineet kantavat itsessään mukanaan tiettyä kulttuuria eli historiallisia jäänteitä niiden kehityksestä. Näin ollen työvälineiden käyttö on myös sosiaalisen tiedon karttumista ja tiedon siirtoa. Toiminnan teoria korostaa erityisesti, että työvälineestä tulee täydellinen vasta käytön myötä. Työvälineet eivät vaikuta yksinomaan ja ainoastaan ulkoiseen käyttäytymiseemme vaan vahvasti myös yksilöiden älyllisiin prosesseihin [Nardi & Kaptelinin, 2000][Nardi, 1996].

Vygotskin mukaan on olemassa kahdenlaisia työvälineitä; teknisiä ja psykologisia työvälineitä. Teknisillä työkaluilla vaikutamme fyysisiin objekteihin, kuten vasaralla naulaan, joka painuu puuhun. Psykologisia työkaluja käytetään, jotta voisimme vaikuttaa muihin ihmisiin tai itseemme. Esimerkkejä psykologisista työkaluista ovat vaikkapa kertotaulu, kalenteri tai mainos [Nardi & Kaptelinin, 2000].

3.6. Kehitys

Toiminnan teoria edellyttää, että ihmisten ja ympäröivän todellisuuden vuorovaikutusta tulee analysoida kehityksen konteksti huomioiden. Toiminnan teoria ei ole ainut psykologinen teoria, jossa kehitys on laaja tutkimusalue, mutta toiminnan teoriassa kehitys ei ole vain tutkimuksen tavoite, vaan yleinen tieteellisen tutkimuksen metodologia. Toiminnan teoriassa kaikki käytäntö nähdään tietyn historiallisen kehityksen tuloksena tiettyjen olosuhteiden vallitessa. Kehitys jatkuvasti uudistaa ja kehittää käytäntöä. Tämän vuoksi toiminnan teorian perustutkimusmetodina eivät ole perinteiset laboratoriokokeet, vaan luova tutkimus, missä yhdistyy kehityksellisten muutoksien tarkkailu. Kansatieteelliset metodit, jotka jäljittävät käytännön historiaa ja kehitystä, ovat tulleet tärkeiksi viimeaikaisissa tutkimuksissa. Toiminnan teoria ei määrää yhtä tiettyä tutkimusmetodia käytettäväksi, vaan sen vaatimus on, että metodi on valittava kulloisenkin tutkimuksen mukaan. Toisin kuin tietyt lähestymistavat, jotka ovat vihkiytyneet vain yhden ainoan metodin käyttöön, toiminnan teorian lähtökohtana on aloittaa käsillä olevasta ongelmasta ja tämän jälkeen vasta suorittaa metodin valinta [Nardi & Kaptelinin, 2000].

3.7. Periaatteiden yhdistäminen

Edellä läpikäydyt toiminnan teorian perusperiaatteet tulee ymmärtää yhtenäisenä kokonaisuutena, koska perusperiaatteet on poimittu toiminnallisuuden eri puolista ja piirteistä. Minkä tahansa periaatteen systemaattinen soveltaminen vaatii ennen pitkää myös muiden periaatteiden liittämistä mukaan. Toiminnan teorian vaatimuksena on periaatteiden

yhtenäisyys ja eheys; yhtään yksittäistä prosessia ei saa erottaa kokonaisuudesta, sillä tällöin toiminnallisuuden kokonaisuutta ei pystytä täysin ymmärtämään [Nardi & Kaptelinin, 2000].

3.8. Toiminnan tarkistuslista

Toiminnan teoria ei tarjoa valmiiksi pureskeltuja ratkaisuja, joita voidaan käyttää suoraan tiettyihin ongelmiin. Toiminnan teorian potentiaali piilee tutkijoiden ja suunnittelijoiden tukemisessa, se kannustaa ja auttaa heitä löytämään itse ratkaisut ongelmiinsa. Erityisesti teoria auttaa kysymään tutkimuksen ja suunnittelun kannalta mielekkäitä ja merkityksellisiä kysymyksiä. Jotta toiminnan teorian soveltaminen tulisi käytännönläheisemmäksi on kehitetty analyyttinen työväline nimeltään toiminnan tarkistuslista [Nardi & Kaptelinin, 1999].

Tarkistuslistan tarkoituksena ei ole olla ainoa työkalu, jota ihmisen ja tietokoneen vuorovaikutuksen tutkimuksissa käytetään. Se täydentää jo olemassa olevia metodeita tarjoamalla yleisen käsitteellisen viitekehyksen, joka auttaa muodostamaan ongelmat, suunnittelemaan tutkimuksen ja tulkitsemaan saadut tulokset. Tutkimuksessa on viisaampaa käyttää vain osia tarkistuslistasta, sillä koko tarkistuslistan käyttö merkitsisi melko varmasti liian laajan tutkimuksen tekemistä [Nardi & Kaptelinin, 2000].

Toiminnan tarkistuslista on tarkoitettu käytettäväksi aikaisessa vaiheessa systeemin suunnittelun apuna tai jo olemassa olevien systeemien evaluointiin. Näin ollen on olemassa kaksi hieman toisistaan poikkeavaa versiota tarkistuslistasta, toinen on evaluointiversio ja toinen suunnitteluversio. Kumpikin versio on toteutettu organisoituina eri yksiköiden jaksoina, jotka käsittävät kaikki asiayhteyteen liittyvät tekijät, jotka potentiaalisesti voivat vaikuttaa tietokoneteknologian käyttöön reaalielämässä. Voidaan olettaa, että tarkistuslista helpottaa tärkeissä kysymyksissä, kuten potentiaalisissa pulmakohdissa, joihin suunnittelija voi törmätä [Nardi & Kaptelinin, 1999].

Tarkistuslista kattaa suuren alueen tutkimuskentästä. On ajateltu, että ensimmäinen vaihe tarkistuslistaa käytettäessä olisi koko tutkimusalueen suuripiirteinen tutkiminen kiinnostavien, yksittäisten alueiden löytymiseksi. Tämän jälkeen keskitytään vain näihin pienempiin alueisiin, mutta ne tutkitaan niin perinpohjaisesti kuin mahdollista. Yleinen strategia on siis breadth-first-periaatteen käyttö tarkistuslistassa luetellun kontekstin kannalta relevanttien alueiden löytämiseksi. Alueiden löytämisen jälkeen näihin spesifeihin alueisiin poraudutaan syvälle. Tällä tavoin saadaan runsaasti tietoa, jota voidaan käyttää tutkimuksen tai suunnittelussa apuna. [Nardi & Kaptelinin, 1999].

Tarkistuslistan rakenne jäljittelee toiminnan teorian viittä perusperiaatetta. Koska tarkistuslistan on tarkoitettu käytettäväksi analysoimaan sitä, miten

ihmiset käyttävät tietokoneteknologiaa, on tarkistuslistassa suuri painotus välineiden vaikutus -periaatteella. Tätä periaatetta käytetään koko tarkistuslistassa ja lisäksi se yhdistetään systemaattisesti muihin neljään periaatteeseen. Näin ollen tarkistuslistassa on neljä saraketta, jotka yhdessä sisältävät päänäkökulmat, joista kohdeteknologiaa arvioidaan ja suunnitellaan [Nardi & Kaptelinin, 1999].

Tarkistuslistan sisältämät näkökulmat ovat keinot ja päämäärät, ympäristön sosiaaliset ja fyysiset piirteet, oppiminen, kognitio ja ymmärrettävyys sekä kehitys. Keinojen ja päämäärien -sarakkeessa pyritään tuomaan esiin missä suhteessa kyseinen teknologia helpottaa ja rajoittaa käyttäjän päämääräänsä pyrkimistä. Myös teknologian vaikutuksen aiheuttamat konfliktit tai vaihtoehtoisesti ristiriidat ratkaisevat tapaukset kirjataan ylös. Ympäristön sosiaaliset ja fyysiset piirteet -sarakkeessa tutkitaan tutkittavan teknologian suhdetta kaikkeen ympärillä olevaan kuten vaatimuksiin, työvälineisiin, resursseihin ja ihmisten välisiin sosiaalisiin suhteisiin. Oppiminen, kognitio ja ymmärrettävyys -sarakkeessa vertaillaan sisäisiä ja ulkoisia toiminnallisuuden komponentteja sekä tutkitaan sisäisten ja ulkoisten prosessien muuntumista kohdeteknologian kanssa työskennellessä. Kehitys-sarakkeessa keskitytään analysoimaan edellä mainittujen komponenttien kehityksellistä muuntumista kokonaisuutena [Nardi & Kaptelinin, 1999].

4. Toiminnan teorian soveltaminen

Toiminnan teoria on suosittu varmasti juuri sen takia, että sitä pystyy soveltamaan todella tehokkaasti monilla eri aloilla.Tässä osiossa katsomme joitain esimerkkejä siitä, missä toiminnan teoriaa on sovellettu.

4.1. Käytettävyystutkimuksen esimerkki

Koska toiminnan teoriassa otetaan huomion ihmisten tunteet ja käyttäytyminen, on sitä sovellettu myös käytettävyystutkimukseen. Tästä esimerkkinä on saksalais-italialaisen tutkijaryhmän saavutukset. Sujan, Pasquini ja muut [Suja, Pasquini & al., 1999] ovat kiinnittäneet huomiota 90-luvun loppupuolelta kasvamassa olleeseen trendiin, jossa ihmisen osuutta monimutkaisissa systeemeissä ei enää kuvaakaan vain sana "ongelma". On todettu, että ihminen on tärkeä osa kokonaisuutta, luova yksilö, joka on parhaimmillaan poikkeustilanteessa.

Toisaalta tässä trendissä ihminen on siltikin vielä jäänyt vain tiedostavaksi osaksi järjestelmää. Sujan, Pasquini ja muut [Suja, Pasquini & al., 1999] kiinnittävätkin huomiota ihmisen tunnepuoleen. He ovat huomanneet, että ihmisten tunteellisuuden unohtaminen on johtanut usein vakaviin tilanteisiin - jotka olisi voitu paremmalla käytettävyyssuunnittelulla välttää. Esimerkiksi

luottamus järjestelmään vaikuttaa huomattavasti siihen, miten järjestelmää lopulta käytetään ja hyödynnetään.

Sujan, Pasquini ja muut [Suja, Pasquini & al., 1999] valitsivat toiminnan teorian lähtökohdakseen, sillä siinä painotetaan ihmisen tietoisuutta ja toimintaa - ja sitä, että toiminnallisuus on motivoitunutta. Toiminnan teoria huomio monimutkaiset viestintätilanteet erinomaisen hyvin, ja sen takia siihen perustuva käytettävyystutkimus vähentää väärää kommunikaatiota, jonka tutkijat erityisesti kokevat tärkeäksi.

Toiminnan teorian lähtöperiaate, tietoisuuden ja toiminnallisuuden eheyden ja erottamattomuuden periaate, on myös Sujanin, Pasquinin ja muiden [Suja, Pasquini & al., 1999] lähtökohtana. Mielenkiintoinen yksityiskohta onkin se, että tutkijat perustavat näkökulmansa jopa Marxiin asti, vaikka Leontevin käsitteistö - niin maallinen ja Marxismiin perustuva kuin onkin - on melko vähän varsinaisen Marxistisen filosofian tukija. Toisaalta se, minkä tutkijat marxismista nostavat esille, eli että se korostaa ihmisen toiminnan sosiaalista puolta hyläten "pään sisäisen" painoarvon, on yhteensopiva käytettävyystutkimukseen sovellettavissa oleviin toiminnan teorian periaatteisiin. Toisin sanottuna juuri tätä puolta kannattaakin korostaa.

Sujan, Pasquini ja muut ottavat esille toiminnan teorian erityisesti siltä kantilta, että ihmisen toiminta siinä otaksutaan erilaisten artifaktien välittäjäksi. Esimerkiksi Luria korosti ajattelussaan kielen merkitystä kaikkeen toimintaan. Ihminen ei koe objektia sellaisena kuin se on, vaan sellaisena kuin artefaktit mahdollistavat sen kokemaan. [Suja, Pasquini & al., 1999]

Toiminnan teoriaa voidaan soveltaa siis käytettävyyteen. Selkeyden vuoksi koemme, että on hyvä esittää esimerkki käytännön tilanteesta. Sujan, Pasquini ja muut [Suja, Pasquini & al., 1999] esittävät junaliikennetietojärjestelmäesimerkin, jonka tahdomme asian selkeyttämiseksi sisällyttää myös tähän omaan työhömme.

Tutkijat suorittivat käytettävyyskartoituksen holistiselta pohjalta, jossa huomioitiin eri osatekijät itsenäisinä toimijoina. Näitä eri osatekijäresursseja oli ainakin ihmisresurssit, instrumentit (kuten laitteisto ja liikennemerkit) ja softa (kuten tietokoneohjelmistot ja säännöt).[Suja, Pasquini & al., 1999]

Tutkimuksessa esitetään skenaarioita, jotka tutkijat ovat laatineet toiminnan teoriaan pohjautuen. Kun näitä skenaarioita vertaa alkuperäisiin, ilman toiminnan teoriaa laadittuihin, eivät erot näytä suurilta. Tärkeintä tapauksessa onkin kommunikaation lisääntynyt merkitys. Tapauksessa kommunikointi on ratkaistu matkapuhelinta käyttäen, mutta tutkijoiden mielestä tämä ei pitkällä tähtäimellä ole kestävä ratkaisu, sillä tietyssä tilanteessa käyttäjä sulkee matkapuhelimen voidakseen paremmin keskittyä johonkin tiettyyn ongelmaan. [Suja, Pasquini & al., 1999]

Toiminnan teoriaa käyttäen tutkijat huomasivat käyttäjän nostavan tietyssä tilanteessa lähitavoitteen yleistavoitetta tärkeämmäksi. Tämä johtuu tunnearvoista - lähitavoite, junasta vastaaminen, koetaan henkilökohtaiseksi vastuuksi. Haastatteluista paljastui, että jotkut tietyt historialliset tekijät vaikuttavat siihen, että yksittäinen juna koetaan junaverkostoa tärkeämmäksi. [Suja, Pasquini & al., 1999]

Tutkijoiden ehdottamat ratkaisut perustuvat juuri näihin tunteisiin, jotka vaikuttavat ihmisiin. Sille henkilölle, joka kokee juna arvokkaaksi, annetaan vastuu junasta ja jokin automaattinen järjestelmä kertoo ratapäällikölle tapahtumista ilman että junavastuullinen joutuu keskeyttämään työtänsä puhelinsoitolla. Toinen vaihtoehto on yrittää vaikuttaa tunteisiin. Esimerkiksi tiimityön merkityksen korostaminen kaikille työntekijöille saattaisi vaikuttaa siihen, että nämä tekevät yleistavoitteellisempia ratkaisuja. [Suja, Pasquini & al., 1999]

4.2. Järjestelmäsuunnittelun esimerkki

Rui Wang [Wang, 2000] on tutkinut sähköisiä suorituksentukemisjärjestelmiä Sydneyn yliopistossa. Hänen mielestään toiminnan teoria on runsain teoria aiheen tutkimiseen. Toiminnan teorian perusoletukset, kuten esimerkiksi monikerroksisuus, vuorovaikutus yhteisötasolla, tavoitteellisuus, dynaamisuus ja kehittyvyys tarjoavat uusille ideoille tilaa tällaisessakin tutkimuksessa.

Wang toimi siten, että hän loi toiminnan teorian periaatteiden pohjalta kuusikohtaisen systeeminkehitys- ja suunnittelulistan. Esittelemme nyt ensin periaatteet, joista Wang valitsi kolme. Sitten esittelemme listan.

1. *Toiminnallisuusjärjestelmän verkostoa tulisi käyttää analyysin ja suunnittelun keskipisteenä käyttäjän ja käyttäjäryhmän ominaisuuksien sijaan.* Toiminnan teorian mukaan toiminnallisuus esiintyy aina jossain viitekehyksessä. Sen sijaan, että yksilön vaatimuksiin kiinnitettäisiin suurin huomio, keskitytään sosiaaliseen käyttötilanteeseen ja käyttöympäristön ymmärtämiseen.[Wang, 2000]

2. *Uusien järjestelmien käytännöllisyys määräytyy analysoimalla jokapäiväisen elämän ristiriitaisuuksia ja häiriöitä.* Uusien innovaatioiden käytettävyyttä on vaikea arvioida käytön perusteella, kun niitä ei vielä ole käytössäkään. Toiminnan teorian mukaisesti tarpeet löytyy analysoimalla jokapäiväisiä ristiriitoja ja mahdollisuuksia. [Wang, 2000]

3. *Suunnitteluprosessi tulee olla jatkuvassa kehityksessä eikä vain yhden kerran toiminto.* Ihmisillä on tarvetta muuttaa toimintatapojaan, joten tämä pitää huomioida suunnitteluprosessin rakenteessakin. [Wang, 2000]

Lista on siis kuusikohtainen, syklinen suunnitteluprosessin avuksi tehty kaavio.

1) Rakenna toiminnallisuusjärjestelmä

2) Kerää tietoja ristiriidoista ja häiriöistä

3) Analysoi tiedot

4) Arvioi prototyyppiä

5) Muunna prototyyppiä ja yksityiskohtia

6) Jatka kohdasta kaksi, toista prosessi. [Wang, 2000]

Valitettavasti löytämässämme artikkelissa ei oltu vielä pantu käytäntöön Wangin listaa. Tästä saa kuitenkin selvän kuvan siitä, mitä toiminnan teorian soveltaminen tarkoittaa. Vaikka onkin olemassa toiminnan tarkistuslista, ei ole aina mielekästä soveltaa sitä orjallisesti. Toiminnan teoria tarjoaa tietyt perusperiaatteet ihmistoiminnan arviointiin ja kussakin tapauksessa näitä periaatteita sovelletaan järkeväksi koetulla tavalla.

5. Keskustelu

Toiminnan teoria on tietojenkäsittelytieteen näkökulmasta katsottuna vanha teoria. Toisaalta se on kuitenkin vasta tämän vuosisadan tuote - koko psykologiaa, jonka piirissä teoria on syntynyt, kutsutaan toisinaan "uudeksi tieteeksi". Meidän kannaltamme tämä onkin mielenkiintoinen asia pohdittavaksi, sillä onhan tärkeää huomioida teoriaa sovellettaessa, milloin se on luotu ja mihin tarkoitukseen.

Toiminnan teoria kehitettiin Neuvostoliitossa aikana, jolloin oli olemassa vain virallisia, hyväksyttyjä totuuksia. Tämä vaikutti varmasti teorian kehitykseen niin hyvässä kuin pahassa. Toisaalta Leontevin saatua Lenin-palkinto, toiminnan teoria alkoi saada varmasti aivan eri lailla positiivista julkisuutta kuin kilpailevat teoriat. Toisaalta joitain seikkoja saatettiin sensuroida, ainakin Vygotskylle on tiettävästi käynyt näin. Onhan myös huomattavaa kansainvälisen tieteenvaihdon hankaluus tuona aikana.

Psykologia on sen laatuinen tiede, ettei ehdottomia totuuksia juurikaan ole. Tämä onkin mielestämme toiminnan teorian vahvimpia puolia. Se on erittäin joustava eikä se ole soveltuva vain tiettyjen järjestelmien suunnitteluun. Joustavuutta todistaa myös sen monitieteelliset sovellukset. Paitsi joustava, toiminnan teoria on kattava. Se lähtee yksilöstä liikkeelle, mutta se ottaa huomioon ryhmän.

Vaikka toiminnan teoriasta voikin puhua ylisanoin, on tärkeää muistaa myös erehtyväisyys. Teoria itse saattaa ollakin varustautunut poikkeustilanteisiin, mutta vastuu on soveltajalla. Teorian joustavuus tarkoittaa myös sitä, että jokaista suunnittelukohdetta varten pitää arvioida tarpeet

uudelleen. Tämän arvioinnin laiminlyönnistä saattaa seurata suunnitteluvirhe. Teorian hyödyntäminen pitää suunnitella itse, eikä tätä suunnitelmaa pidä lukita lopullisena versiona. Parhaiten toiminnan teoriaa voi soveltaa siten, että pitää mielessä teorian käsitteet ja pääperiaatteet eikä väkisin yritä luoda valmiita kaikkialle sopivia malleja.

Teoriaan tutustuttuamme arvioimme teorian soveltamisen raskaaksi. Se ei ole mikään kevyt muotti, johon tutkimuksen voi sovittaa, vaan raskas ja tutkimustyötä dominoiva viitekehys. Jotta toiminnan teorian avulla tutkimuksen validiteetti kasvaisi, tarvitsee teoriaan paneutua huolellisesti. Tämä paneutuminen saattaa olla tutkimuksen kannalta liian raskasta, emmekä pidä teoriaa toimivana varsinkaan pienemmissä tutkimuksissa.

Toiminnan teorialla on kuitenkin varmasti vielä paljon annettavaa tulevaisuudessakin. Toisaalta ei pidä unohtaa, että teorian pohjalta on luotu myös uusia malleja (esimerkiksi [14], tilannetoimintateoria) ja taas uusia malleja tullaan luultavasti luomaan, niin mielenkiintoisen lähtökohdan toiminnan teoria tarjoaa. Joka tapauksessa teoria ja sen mahdollisuudet kannattaa pitää mielessä.

Lähteet

[Nardi & Kaptelinin, 2000] Bonnie A. Nardi and Victor Kaptelin, *Activity Theory: Basic Concepts and Applications, Abstract, Text Summary and Presentation*. 2000.

[Nardi & Kaptelinin, 1999] Bonnie A. Nardi and Victor Kaptelin, *The Activity Checklist: A Tool for Representing the "Space" of Context*. 1999.

[Nardi, 1996] Bonnie A. Nardi, Studying Context: A Comparison of Activity Theory, Situated Action Models, and Distributed Cognition. Teoksessa *Context and Consciousness: Activity Theory and Human-Computer Interaction*, Bonnie A. Nardi, MITT Press, Cambridge, 1996, 67-102.

[Center for Activity Theory and Developmental Work Research], http://www.edu.helsinki.fi/activity/1.htm.

[Ryder, 2001] Martin Ryder, *What is Activity Theory*, http://carbon.cudenver.edu/~mryder/itc_data/act_dff.html. 2001.

[http://www.marxists.org/glossary/people/v/y.htm#vygotsky-lev] http://www.marxists.org/glossary/people/v/y.htm#vygotsky-lev.

[Bannon, 1997] Liam Bannon, *Activity Theory*, http://www-sv.cict.fr/cotcos/pjs/TheoreticalApproaches/Actvity/ActivitypaperBannon.htm. 2000.

[Robbins, 2000] Dot Robbins, *Leontiev vs. Vygotsky*, http://lchc.ucsd.edu/MCA/Mail/xmcamail.2000_11.dir/0012.html. 2000.

[Suja, Pasquini & al., 1999] Mark-Alexander Suja, Alberto Pasquini & al., *Activity Theory as a Framework for Considering Human Affect in the Design*. IEEE 1999.

[Leiman, 1994] Mikael Leiman, *Integrating The Vygotskian Theory of Sign-mediated Activity and The British Object Relations Theory*. Joensuun yliopiston yhteiskuntatieteellisiä julkaisuja N:O 20, 1994.

[Sharpe, 1989] Pamela J. Sharpe, *The Theory of Activity: Perspectives in Practice.*University Microfilms International, 1989.

[Junefelt, 1995] Karin Junefelt (toim.), *Proceedings of The XIVth Scandinavian Conference of Linguistics and The VIIIth Conference of Nordic and General Linguistics August 16-21 1993*. Gothneburg Papers in Theoretical Linguistics 73, 1995.

[Wang, 2000] Rui Wang, *Designing Electronic Performance Support System in Organizational Context: An Active Theory Approach*. IEEE 2000.

[Iivari ja Linger, 1999] Juhani Iivari ja Henry Linger, *Knowledge Work as Collaborative Work: A Situated Activity Theory View*. IEEE 1999.

Kaksikätisyyden teorioita

Roope Raisamo

Tässä luvussa selvitetään molempien käsien roolia ihmisen ja tietokoneen vuorovaikutuksessa. Erityisen tarkastelun kohteena ovat ne psykologisen tutkimuksen tulokset, jotka selittävät käsien välisiä eroja. Lisäksi tarkastellaan käsien keskinäistä toimintaa kuvaavaa Yves Guiardin kinemaattisen ketjun mallia. Kahden käden käyttömahdollisuudet käyttöliittymissä ovat olleet aktiivisen kiinnostuksen kohteena vuodesta 1986 lähtien, jolloin Bill Buxton ja Brad Myers julkaisivat ensimmäisen artikkelin, jossa osoitettiin kaksikätisyydestä olevat kiistattomia hyötyjä, kun niiden tehtävät valitaan oikein.

1. Johdanto

Ihmiset käyttävät molempia käsiään useissa jokapäiväisissä toimissaan. Käsien ominaisuuksien välillä on yksilöllisiä eroja, mutta useimmilla ihmisillä toinen käsistä on selkeästi toista tarkempi ja hallitseva. Puhutaan dominoivasta ja ei-dominoivasta kädestä. On tunnettua, että oikea käsi on dominoivana kätenä vasenta selvästi yleisempi.

Aluksi tässä esityksessä motivoidaan lukijaa molempien käsien hyödyntämiseen käyttöliittymässä. Tämän jälkeen tarkastellaan teorioita, jotka selittävät käsien sopivuutta tietynlaisiin tehtäviin sekä käsien välistä työnjakoa ajan kuluessa. Psykologian tutkimuksessa on selvitetty käsien dominanssin suuruutta sekä sitä, miten dominoivan ja ei-dominoivan käden ominaisuudet eroavat toisistaan. Lisäksi on olemassa Guiardin kinemaattisen ketjun malli, joka kuvaa käsien välistä toimintaa ajan kuluessa.

2. Kahden käden käytön hyödyllisyys

Ensimmäinen empiirinen tutkimus kahden käden käytöstä suorakäyttöisissä ohjelmissa tehtiin viisitoista vuotta sitten [Buxton and Myers, 1986]. Jo tässä ensimmäisessä tutkimuksessa havaittiin kahden käden käytöstä selkeitä etuja johtuen toimintatavan luonnollisuudesta ja kahdesta rinnakkaisesta syötevirrasta. Toisaalta syötevirtojen jakaminen eri laitteille on yleensä parempi tapa kuin yhden syöttölaitteen kuormittaminen usealla rinnakkaisella ohjaimella [Zhai *et al.*, 1997].

2.1. Kahden käden käyttö on luonnollista

Buxton ja Myers [1986] kokeilivat molempien käsien käyttämistä piirto-ohjelmalla tehtävässä yhdistetyssä kappaleen siirto- ja koonmuutostehtävässä. Tehtävä voitiin suorittaa myös yhdellä kädellä siirtämällä kappale ensin oikeaan kohtaan ja muuttamalla sitten sen kokoa. Tutkijat myös rohkaisivat koehenkilöitä tekemään sen tällä tavalla harjoitteluvaiheessa. Syöttölaitteina olivat vasemman käden ulottuvilla kaksi liukusäädintä koon muuttamiseksi ja oikealla kädellä piirtotabletti.

Huomattavin tulos tässä tutkimuksessa oli se, että kun käyttäjät alkoivat suorittaa annettua tehtävää, kuusi koehenkilöä neljästätoista käytti molempia käsiä rinnakkain heti ensimmäisestä yrityksestä alkaen, vaikka tutkijat eivät mitenkään tähän kehoittaneet. Kaikkiaan koehenkilöt käyttivät molempia käsiään rinnakkain 40,9% tehtävien suoritukseen käytetystä ajasta. Rinnakkaisen toiminnan suuri määrä osoittaa, että molempien käsien käyttäminen on ihmiselle luonnollista toimintaa eikä kumpikaan suoritettavista

osatehtävistä aiheuttanut suurta kuormaa ihmisen motorisille kyvyille. Myös tehtävän suoritusnopeus korreloi selvästi rinnakkaisuuden määrän kanssa.

2.2. Kaksikätinen tekniikka tehostaa työskentelyä

Toisessa Buxtonin ja Myersin koehenkilöillään teettämässä tehtävässä pyrittiin tutkimaan, onko olemassa sellaisia yleisiä tehtävätyyppejä, joissa kahden käden käyttäminen toisi huomattavan suorituskyvyn parannuksen yksikätisiin tekniikoihin verrattuna.

Tehtävänä oli tekstinkäsittelyssä tapahtuva sanojen valitseminen siten, että käyttäjän täytyi myös vierittää dokumenttia, jotta valittava teksti tulisi näytölle. Tavoitteena oli, että hyvin suunniteltu navigoinnin ja valitsemisen jakaminen eri käsien suoritettavaksi olisi tavallisia tekstinkäsittelyjärjestelmiä helpompi oppia ja käyttää. Tässä tehtävässä liukusäätimet korvattiin toisella piirtotablettilla, jolloin molemmilla käsillä käytettiin siis piirtotablettia. Koehenkilöitä oli 24, jotka jakaantuivat tasan kokeneisiin ja aloitteleviin käyttäjiin. Kumpikin näistä ryhmistä jaettiin vielä kahteen osaan, joista toiset käyttivät vain yhtä kättä ja toiset kahta kättä annetun tehtävän suorittamiseen.

Tuloksina havaittiin, että kokeneista käyttäjistä kahta kättä käyttänyt ryhmä oli 15% nopeampi vain yhtä kättä käyttäneeseen ryhmään verrattuna. Aloittelijoiden osalta ero oli vielä selvempi: kahta kättä käyttänyt ryhmä oli 25% yhtä kättä käyttänyttä nopeampi. Kahden käden käytöllä oli myös selvä vaikutus kokeneiden ja aloittelevien käyttäjien keskinäisiin suorituseroihin: yhden käden käytössä kokeneet käyttäjät olivat 85% aloittelijoita nopeampia, mutta kahta kättä käytettäessä ero oli vain 32%. Lisäksi kahta kättä käyttäneet aloittelijat olivat vain 12% kokeneita yhden käden käyttäjiä hitaampia, eikä tällä erolla ollut tilastollista merkitsevyyttä. Oikein suunnitellun kaksikätisen tekniikan voidaan siis katsoa pienentävän aloittelijoiden ja kokeneiden käyttäjien suorituskyvyn eroja, ja näin nopeuttavan järjestelmän opetteluvaihetta.

Tulosten perusteella voidaan todeta, että valitsemis- ja navigointitehtävien jakaminen eri käsille paransi kaikkien käyttäjien suorituskykyä. Tässä tehtävässä kyse ei ollut kuitenkaan molempien käsien toiminnan rinnakkaisuudesta, vaan siitä, että kun molempien tablettien kohdistimet olivat aina lähellä oikeaa paikkaa, vähentynyt käsien liike nopeutti tehtävän suoritusta. Liikkeiden vähentymisellä on vielä suurempi vaikutus, kun käytetään suhteellista syöttölaitetta (esimerkiksi hiirtä), jolla ei voida välittömästi siirtyä mihin tahansa kohtaan näytöllä.

2.3. Yhtä kättä ei kannata kuormittaa liian monella toiminnolla

Zhai *et al*. [1997] toistivat Buxtonin ja Myersin jälkimmäisen tehtävän soveltaen sitä World Wide Webin selaamiseen. Tehtävässä oli tavoitteena selata joukko WWW-sivuja mahdollisimman nopeasti. Sivut oli rakennettu siten, että linkkien valitseminen vaati sivun vierittämistä. Kokeen tarkoituksena oli vertailla tavallisen hiiren, peukalopyörällä varustetun hiiren (Microsoft WheelMouse™), isometrisellä ohjaussauvalla (IBM TrackPoint III™) varustetun hiiren sekä samanaikaisten hiiren ja isometrisen ohjaussauvan käyttöä. Hiiri edusti yhtä syötevirtaa tuottavia laitteita. Peukalopyörällä ja isometrisellä ohjaussauvalla varustetut hiiret edustivat kahta syötevirtaa tuottavia yhdellä kädellä käytettäviä laitteita. Hiiren ja isometrisen ohjaussauvan rinnakkainen käyttö vastasi Buxtonin ja Myersin koetilannetta kaksikätisenä tekniikkana eroten kuitenkin siinä, että eri käsillä ohjattiin erilaisia syöttölaitteita kahden samanlaisen sijaan. Tehtävät suoritti 12 vapaaehtoista koehenkilöä.

Tuloksista havaittiin, että kaksikätinen tekniikka oli nopein tekniikka, mutta isometrinen ohjaussauva hiireen lisättynä ylti lähes samaan nopeuteen. Sen sijaan peukalopyörällä varustettu hiiri oli jopa hitaampi kuin tavallinen hiiri. Tämä tulos tuo esiin, että vaikka peukalopyörän käyttäminen on intuitiivista, kognitiivinen rasitus kasvoi niin paljon, että suoritusnopeus laski tavallisen hiiren käyttöön verrattuna merkittävästi. Isometrisellä ohjaussauvalla varustetun hiiren hyvä sijoitus johtuu sen hyvästä suunnittelusta, jolla päästiin lähes yhtä hyvään suorituskykyyn kuin kaksikätisellä tekniikalla.

Syöttötapojen paremmuus oli vastaavanlainen myös kysyttäessä niiden miellyttävyyttä käyttäjiltä: isometrinen ohjaussauva ja kaksikätinen tekniikka olivat selvästi parhaat, hiiri sai lievästi positiivisen arvion, mutta peukalopyörähiiri selvästi negatiivisen. Tämän tutkimuksen eräs yleinen havainto on se, että eri käsillä käytettävien syöttölaitteiden ei tarvitse olla keskenään samanlaisia saavuttaakseen parannusta suorituskyvyssä ja käyttäjien tyytyväisyydessä tavallisen hiiren käyttöön verrattuna.

3. Eri tapoja käyttää molempia käsiä käyttöliittymässä

Graafiset käyttöliittymät antavat suunnittelijoille paljon sekä hyviä että huonoja mahdollisuuksia, joihin kuuluu myös mahdollisuus rakentaa huonosti toimivia järjestelmiä. Kaksikätisissä käyttöliittymissä nämä molemmat puolet korostuvat vielä selvemmin. Huonosti suunniteltu kahta kättä käyttävä järjestelmä voi olla yhden käden järjestelmiä vaikeampi ja hitaampi käyttää.

Chatty [1994] esittää, että järjestelmässä ei saisi olla yhtäkään sellaista tehtävää, jossa kahden käden käyttäminen olisi ainoa mahdollinen tapa sen suorittamiseen. Tämä johtuu siitä, että toisella kädellä saattaa aina olla jotakin muuta

tehtävää, kuten puhelimen kuulokkeen tai paperin pitäminen. Näin kahta kättä käyttävien järjestelmien tulisi siis olla käytettävissä myös yhdellä kädellä. Graafisen käyttöliittymän yhtenäisyysperiaatteista seuraa, että näiden yhtä ja kahta kättä käyttävien toimintojen tulisi olla mahdollisimman samanlaisia. Tähän päästään kirjoittajan mukaan sillä, että käytetään todellista maailmaa kuvaavia metaforia. Näin kahden käden käyttö saadaan aikaan vain laajentamalla järjestelmää valitun metaforan mukaisesti.

Chatty tarkastelee esimerkkinä metaforan laajentamisesta raahaamisen kaksikätistä vastinetta: kun raahaamisessa tartumme kohteeseen yhdellä kädellä ja siirrämme sitä, samaan tapaan tulisi tapahtua jotakin ennustettavaa, kun tartumme kohteen reunoihin kahdella kädellä ja venytämme sitä. Fitzmaurice *et al.* [1995] ovat toteuttaneet tällaisen vuorovaikutustekniikan. Tämä venytystoiminto on kahta kättä käyttävä käyttöliittymän laajennus, joka on mahdollista suorittaa yhdellä kädellä useassa vaiheessa: joko kappaleen kokoa täytyy muuttaa kaksi kertaa eri suunnista tai kappale täytyy ensin siirtää uuteen paikkaan ja sen jälkeen muuttaa sen koko halutuksi.

Nigayn ja Coutazin [1993] luokittelu multimodaalisille käyttöliittymille tarjoaa hyvän viitekehyksen kaksikätisten käyttöliittymien tarkasteluun. Yksinkertaisin käyttötapa on käsien vuorottainen käyttö. Jos järjestelmä mahdollistaa useiden syöttölaitteiden rinnakkaisen toiminnan, näille voidaan määrätä toisistaan riippumattomia tai toisiaan tukevia tehtäviä. Kuvassa 1 on esitetty kahden käden käyttötavat näin luokiteltuna.

KÄSIEN KÄYTTÖ	**Peräkkäistä**	**Rinnakkaista**
Samaan tarkoitukseen	vuorottainen käyttö, sama ohjattava kohde	yhdistetty käyttö
Eri tarkoituksiin	vuorottainen käyttö, eri ohjattava kohde	rinnakkainen käyttö

Kuva 1. Kahden käden käyttötavat

3.1. Vuorottainen käyttö

Yksinkertaisin tapa laajentaa yksikätinen käyttöliittymä kahta kättä hyödyntäväksi on lisätä järjestelmään toinen paikannuslaite, jota voidaan käyttää samaan tapaan kuin alkuperäistä. Käytännössä tästä on hyötyä vasta sitten, kun molemmat näistä laitteista ohjaavat näytöllä omaa kohdistintaan. Esimerkiksi Apple Macintosh -koneiden NuBus-väylään ja PC-koneiden USB-väylään voidaan liittää useita rinnakkaisia paikannuslaitteita, mutta ilman lisäohjelmistoja järjestelmä sitoo ne kaikki samaan kohdistimeen, jolloin ainoat

uuden laitteen tuomat edut ovat käyttäjän vaihtoehtojen ja mahdollisesti käyttömukavuuden lisääntyminen.

Jos paikannuslaitteilla on omat kohdistimensa, minkä tahansa nykyisen ohjelman käyttö voi nopeutua Buxtonin ja Myersin [1986] havaitsemalla tavalla: kun molemmat kohdistimet ovat lähellä peräkkäin tarvittavia toimintoja, kohdistimen siirtämiseen kuluva aika pienenee ja näin työskentely nopeutuu. Tällainen kahden käden käyttö ei vaadi ohjelmoijilta erityisiä lisäpanostuksia eikä vaikuta ohjelman käyttämiseen yhdellä kädellä.

Toista paikannuslaitetta voitaisiin käyttää myös samaan toimintoon kuin alkuperäistä, esimerkiksi piirtämiseen tai kuvakkeiden siirtämiseen. Ihmiset ovat kuitenkin hitaita suorittamaan tarkkoja tehtäviä ei-dominoivalla kädellään, minkä vuoksi näitä tehtäviä ei kannata sille antaa. Tarkkuuden puutetta voidaan kompensoida tekemällä ei-dominoivan käden kohteista suuria ja siten helposti osoitettavia. Chatty [1994] esittää myös toisen vaihtoehdon, jossa pieniä kohteita käsitellään suurella kohdistimella, joka yhden pisteen sijasta olisi jokaisessa pisteessään aktiivinen. Vaikka järjestelmä suunniteltaisiinkin ottamaan huomioon ei-dominoivan käden epätarkkuus, toisen käden hyödyllisyys tulee paremmin esiin vasta sitten, kun molempia käsiä voidaan käyttää rinnakkain.

3.2. Rinnakkainen käyttö

Ihmiselle on luonnollista käyttää molempia käsiään rinnakkain. Vaikka sitä ei meille erikseen opeteta, käytämme ei-dominoivaa kättämme avustaviin tehtäviin, kuten työkalun tuomiseen dominoivalle kädelle. Kahta kättä hyödyntävän tietokoneohjelman käyttäminen olisi hyvin rajoittunutta, jos käsien toiminnan tulisi olla tiukasti peräkkäistä: eiväthän kätemme yleensäkään odota toisiaan tehtäviä suorittaessamme ja sen pakottaminen turhauttaisi käyttäjää. Näin voidaan todeta, että rinnakkaisuus on luonnollisen käyttäytymisemme mukaisesti kaksikätisen käyttöliittymän perusominaisuus.

Molempia käsiä voidaan käyttää myös erillisiin samanarvoisiin tehtäviin. Tällainen käyttö liittyy usein peleihin ja erityissovelluksiin. Simulaattoreissa molempia käsiä voidaan käyttää tällä tavalla silloin, kun niitä käytetään näin myös simuloitavassa tilanteessa, esimerkiksi auton ajamisessa tai lentokoneen ohjaamisessa. Tällä tavalla voidaan tehdä myös ihmisen motorisia kykyjä haastavia testejä tai pelejä. Chattyn [1994] mukaan todellinen hyöty rinnakkaisuudesta saadaan kuitenkin vasta silloin, kun eri käsillä tehtävät toiminnot tukevat toisiaan.

3.3. Yhdistetty käyttö

Hienostunein käyttötapa kahdelle paikannuslaitteelle on niiden toimintojen yhdistäminen niin, että ne tukevat toisiaan. Myös tämä tapa on ihmiselle täysin luonnollinen: esimerkiksi naulatessamme ei-dominoivalla kädellä pidetään naulaa paikallaan ja dominoivalla kädellä käytetään vasaraa. Toinen luonnollinen käyttötarkoitus on dominoivan käden vakauden tai voiman parantaminen tukemalla sitä ei-dominoivalla kädellä, kuten tapahtuu esimerkiksi golfin pelaamisessa tai liian täyteen kaadetun kahvikupin tukemisessa.

Perinteisissä käyttöliittymissä on usein korvattu toinen käsi eräänlaisella taikuudella: kun esimerkiksi piirto-ohjelmassa vedämme kappaleen alakulmasta sitä suuremmaksi, vastakkainen kulma pysyy paikallaan kuin näkymättömän käden pitämänä. Chatty [1994] esittää, että kun tällaisessa ohjelmassa siirrytään kahden käden käyttöön, tämä taikuus tulisi kytkeä pois päältä. Tällöin vastakkainen kulma pysyisi paikallaan, jos käyttäjä pitäisi sitä paikallaan toisella kohdistimella. Muutoin koko kappale tulisi mukana alakulmasta vetämällä eikä silloin kasvaisi. Tätä vuorovaikutustapaa kutsutaan nimellä *pidä ja vedä* (hold-and-pull).

Toinen yhdistetyn toiminnan muoto on kahden kohteen samanaikainen käyttäminen. Esimerkiksi joissakin videonauhureissa nauhoitustoiminnon käyttö vaatii kahden napin samanaikaista painamista. Vastaavasti tietokoneohjelmassa voitaisiin kriittisen toiminnon suorittamiseen vaatia kahden graafisen painikkeen samanaikaista valitsemista. Tällainen käyttötapa vaatisi kuitenkin pienien aikaerojen sallimisen samaan tapaan kuin kaksoisnapsautusta tehtäessä ja vaikeuttaisi ohjelman käyttöä.

4. Käsien välisistä eroista

Kahden käden käytöstä on olemassa kolme teoriaa, jotka selittävät käsien suorituskykyeroja nopeissa päämääräsuuntautuneissa liikkeissä [Annett *et al.*, 1979; Kabbash *et al*., 1993].

Ensimmäinen teorioista perustuu siihen, että kädet eroavat toisistaan pääsääntöisesti aistihavaintojen tarkkailun perusteella. Flowers [1975] tulkitsi tätä teoriaa erottelemalla ns. ballistiset ja ohjatut liikkeet toisistaan. Ballistiset liikkeet kestävät alle 300 ms ja niitä pidetään liian nopeina, jotta niiden suorittamisessa voitaisiin hyödyntää aistihavaintojen antamaa palautetta. Pidempään kestävät liikkeet ovat ohjattuja ja hyödyntävät aistien välittämää palautetta. Flowers huomasi, että molemmat kädet saavuttivat yhtä hyvät nopeudet ballistisessa yhden sormen naputtelutehtävässä. Sen sijaan dominoiva käsi oli ylivoimainen tehtävässä, jossa naputeltiin vuorotellen kahteen eri kohteeseen.

Toisen teorian [Schmidt *et al.*, 1979] mukaan dominoivan käden toiminnassa esiintyy vähemmän häiriöitä. Liikkeiden pituuden lisääminen tai niihin käytettävän ajan vähentäminen vaatii enemmän voimaa, johtaa suurempaan liikkeiden hajontaan ja näin lisää virheitä. Tämän teorian mukaan korjausliikkeillä on kiinteä tarkkuus ja niitä voidaan tehdä tarvittaessa noin 200 ms välein. Annett *et al.* [1979] havaitsivat, että suurin osa käsien välisistä eroista johtui siitä, että ei-dominoivalla kädellä jouduttiin tekemään noin 50% enemmän pieniä korjausliikkeitä ja nämä korjausliikkeet kestivät keskimäärin noin 120 ms. Tästä voidaan päätellä, että ei-dominoivan käden ohjaaminen on yksinkertaisesti epätarkempaa kuin dominoivan.

Todorin ja Doanen [1978] esittämä kolmas teoria ennustaa ei-dominoivan käden paremmuutta suuremmilla etäisyyksillä toimittaessa. Teoria perustuu Welfordin [1968] havaintoon, jonka mukaan nopeat päämääräsuuntautuneet liikkeet koostuvat kahdesta erillisestä osasta: nopeasta etäisyyden vähentämisliikkeestä ja hitaammasta kohteeseen hakeutumisvaiheesta. Ensimmäinen vaiheista vastaa nopeudeltaan Flowersin ballistista liikettä, mutta toinen vaatii näköhavainnon käyttöä. Teorian mukaan yhtä vaikeissa tehtävissä siirtämiseen kuluvan ajan tulisi dominoivalla kädellä kasvaa, kun kohteen etäisyys ja koko kasvavat, mutta ei-dominoivalla kädellä vaikutuksen tulisi olla päinvastainen tietyissä rajoissa. Kabbash *et al.* [1993] suorittivat jatkotutkimuksen, joka tukee Todorin and Doanen teoriaa eri vaikeusasteisissa tehtävissä. Keskenään yhtä vaikeissa tehtävissä he havaitsivat dominoivan (oikean) käden kohdalla nopeuden parantumisen kohteen koon kasvaessa, mutta ei-dominoivan (vasemman) käden kohdalla nopeusetu saavutettiin etäisyyden kasvaessa.

Mikään näistä teorioista ei ennusta tarkasti käsien välisiä eroja. Niitä voidaan kuitenkin käyttää apuna jaettaessa tehtäviä eri käsille. Ensimmäisen teorian perusteella ei-dominoivalle kädelle sopivat hyvin yksinkertaiset tehtävät, joissa ei vaadita paljonkaan aisteihin perustuvaa tarkkailua. Toinen teorioista korostaa ei-dominoivan käden epätarkkuutta ja siitä johtuvaa hitautta, minkä perusteella sille ei tulisi antaa suurta tarkkuutta vaativia tehtäviä. Kolmannen teorian perusteella on mahdollista, että joissakin tehtävissä ei-dominoiva käsi olisi dominoivaa parempi. Näiden tehtävien tulisi sisältää pitkiä etäisyyksiä ja suurten kohteiden käsittelyä.

5. Kinemaattisen ketjun malli

Guiard [1987] on tutkinut tarkemmin eri käsille sopivia tehtävätyyppejä. Hän osoitti, että suurta osaa jokapäiväisistä tehtävistämme voidaan kuvata asymmetrisiksi ja kaksikätisiksi siten, että kummallakin kädellä on selvästi toisesta poikkeava rooli ja nämä roolit riippuvat toisistaan kolmella tavalla:

1. Vasen käsi toimii oikean käden apuna ja asettaa sille viitekehyksen. Näin on esimerkiksi jo edellä kaksikätisen toiminnon esimerkkinä mainitussa naulaamisessa, jossa vasemmalla kädellä tuetaan naulaa, kun oikealla kädellä käytetään vasaraa.
2. Toiminnan suoritus siirtyy vasemmasta oikeaan käteen. Esimerkiksi vasemmalla kädellä tartutaan paperiin ja tämän jälkeen oikeassa kädessä olevalla kynällä kirjoitetaan siihen.
3. Vasemman käden toiminta on yksinkertaisempaa kuin oikean. Esimerkiksi taiteilija pitää palettia vasemmassa kädessään ja maalaa tarkasti oikealla kädellään.

Koska nämä toimintatavat ovat ihmiselle luonnollisia, niiden voidaan olettaa aiheuttavan muita pienemmän kognitiivisen taakan niitä käyttäville ihmisille, mistä voi seurata parempi tehtävien suorituskyky. Näitä kaksikätisten tehtävien ominaisuuksia kannattaa siis käyttää pohjana, kun mietitään, mitkä ohjelman toiminnot voisivat hyödyntää toista kättä.

5.1. Kinemaattisen ketjun malli

Kinemaattinen ketju on fysiikassa jono peräkkäisiä moottoreita. Kinemaattisen ketjun malli puolestaan on käsien toimintaa selittävä malli, jota ei sinänsä voi käyttää toiminnan ennustamiseen laskennallisesti. Se perustuu fyysiseen tarkasteluun, jossa ei ole mitattavia suureita. Mallin tarkoituksena on selittää käsien työnjaon logiikkaa monissa jokapäiväisissä tehtävissä.

Mallissa on kaksi perusoletusta:

1. Kädet voidaan esittää kahtena abstraktina moottorina. Näiden moottorien sisäinen toiminta ja niiden ohjaamiseen liittyvä kognitiivinen kuorma jätetään huomiotta.
2. Ihmisen toiminnassa nämä moottorit toimivat sarjassa ja muodostavat siten kinemaattisen ketjun, jossa edellisen tulokset vaikuttavat seuraavaan.

Malli siis jättää huomiotta kaiken käsien toimintaan liittyvän biomekaanisen monimutkaisuuden ja keskittyy sen sijaan käsien suorittamiin toimenpiteisiin ja niiden yhdistämiseen. Erityisesti näkökulma on siinä, *mitä* vasemmalla ja oikealla kädellä tehdään, ei siinä *miten* niveliä ja lihaksia pitää ohjata. Tärkeää on lopputulos ja tehtävä, eivät sen suorittamiseen tarvittavat anatomiset yksityiskohdat. Näin mallissa puhutaankin käsistä eikä yksittäisistä lihaksista.

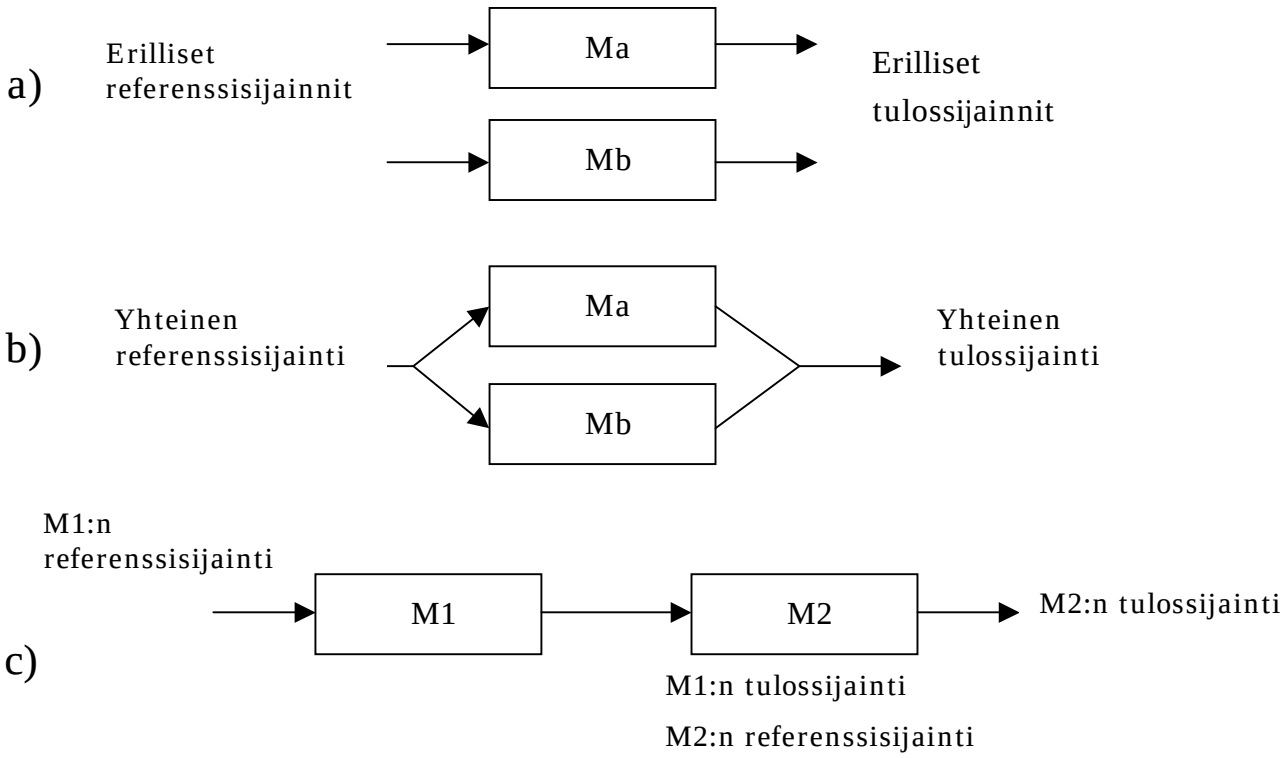
Kädet siis esitetään kahtena abstraktina moottorina, joiden tarkempi toiminta jätetään huomiotta. Käsiä voidaan ajatella mallissa ”mustina laatikkoina”. Tällaisen moottorin tarkoituksena on muuttaa sijaintia. Syötteenä

moottori saa alkuperäisen sijainnin ja tuloksena muutetun sijainnin. Tässä ei kuitenkaan tarvitse olettaa ajallista peräkkäisyyttä stimuluksen ja vasteen välillä. Moottori kontrolloi eroa kahden sijainnin välillä, mutta nämä sijainnit voivat jatkuvasti vaihdella, ja muuttua samanaikaisesti. Kaavioissa voimme kuitenkin piirtää yksisuuntaiset nuolet siksi, että näillä sijainneilla on yleensä selvästi eri status. Toinen on moottorin saama referenssisijainti ja toinen moottorin kontrolloima tulossijainti. Esimerkkinä ihmisen kaltaisen robotin kyynärpää toimii kämmenen ja kyynärpään välisen käden osan ohjaajana suhteessa referenssinä toimivan olkapään ja kyynärpään välisen käden osan asentoon, joka taas riippuu robotin rungon asennosta.

Näin siis tässä mallissa painotetaan tehtävän mekaniikkaa ja liiketta jäsenten biomekaniikan ja hermojärjestelmän sijaan. Perusteluna on se, että tehtävän suorittamisen kannalta näiden tehtävänä on pyrkiä mahdollisimman onnistuneesti tavoitteen toteutumiseen.

5.2. Moottorien yhdistäminen

Kaksi moottoria voidaan yhdistää kolmella tavalla, jotka on esitetty kuvassa 2.



Kuva 2. Kahden moottorin yhdistämistavat. (a) ortogonaalinen yhdistäminen, (b) rinnakkaismuotoinen yhdistäminen, (c) sarjamuotoinen yhdistäminen. Ma, Mb, M1 ja M2 kuvaavat moottoreita.

Ortogonaalisessa yhdistämisessä moottorit ovat täysin toisistaan riippumattomia. Lisäksi työnjako moottorien välillä on symmetrinen. Rinnakkaismuotoisessa yhdistämisessä moottorien toiminnassa on selkeä keskinäinen riippuvuus, mutta työnjako niiden välillä on edelleen symmetristä. Tällaisiin kaksikätisiin tehtäviin kuuluvat esimerkiksi painonnosto ja muut sellaiset tehtävät, joissa kädet vaikuttavat symmetrisesti lopputulosta parantavasti. Sarjamuotoisessa yhdistämisessä puolestaan moottorit riippuvat toisistaan, mutta niiden välinen työnjako on selvästi asymmetrinen. Suurin osa ihmisen luontaisista kaksikätisistä toiminnoista kuuluu tähän ryhmään. Esimerkiksi oikea käsi liikuttaa kynää suhteessa paperilehtiöön, joka on vasemmassa kädessä.

Näin pohjana kinemaattisen ketjun mallille on ajatella käsiä sarjamuotoisesti yhdistettyinä moottoreina. Mitkä tahansa kaksi toisiaan seuraavaa moottoria toimivat yhteistyössä samalla tavalla kuin ihmisen kädet toimivat yhdessä useimmissa tyypillisissä kaksikätisissä tehtävissä. Mallissa tarkastellaan erityisesti sellaista tilannetta, jossa oikea käsi toimii kinemaattisen ketjun viimeisenä lenkkinä ja vasen käsi toiseksi viimeisenä, ja ketjussa on vähintään kolme moottoria. Välttämättä oikea käsi ei kuitenkaan olekaan viimeinen lenkki, vaan esimerkiksi huilun soittajalla sellaisena toimii suu.

Guiard [1987] myös todistaa paperissaan, kuinka edellä esitetyt kolme kaksikätisiä toimintoja luonnehtivaa piirrettä voidaan selittää kinemaattisen mallin avulla. Kinemaattisten ketjujen peruspiirteenä on eri tasoilla toimivien moottorien hierarkkisuus. Esimerkiksi kädessä kauempana sormista olevat moottorit toimivat suuremmilla etäisyyksillä ja muuttavat käden sijaintia nopeasti. Kohti sormenpäitä moottorien tarkkuus ja ohjattavuus kasvaa, mutta ne toimivat paljon aikaisempia pienemmällä alueella. Vastaavasti ei-dominoivan käden tarkkuus on dominoivaa heikompi, mutta tutkimusten mukaan se toimii suurilla etäisyyksillä dominoivaa paremmin, mikä puoltaa sen asemaa ketjun aikaisempana lenkkinä.

6. Yhteenveto

Kinemaattisen ketjun malli selittää käsien välistä työnjakoa kuvaamalla kädet abstrakteina moottoreina, jotka on yhdistetty sarjamuotoisesti. Mallia voidaan käyttää apuna suunniteltaessa käsille sopivia tehtäviä ja niiden keskinäistä yhdistämistä. Koska malli on kuitenkin vain analogiaan perustuva eikä laskennallinen, sen perusteella ei voi tehdä tarkkoja ennusteita siitä, kuinka hyvin kädet juuri tiettyyn tehtävään sopivat. Malli myös jättää huomiotta biomekaaniset ja kognitiiviset tekijät, joilla on vaikutusta lopputulokseen.

Viiteluettelo

[Annett *et al*., 1979] J. Annett, M. Annett, P. T. W. Hudson and A. Turner, The control of movement in the preferred and non-preferred hands. *Quarterly Journal of Experimental Psychology, 31*, 641-652.

[Buxton and Myers, 1986] William Buxton and Brad A. Myers, A study in two-handed input. *Human Factors in Computing Systems, CHI '86 Conference Proceedings*, ACM Press, 1986, 321-326.

[Chatty, 1994] Stéphane Chatty, Extending a graphical toolkit for two-handed interaction. *ACM UIST '94 Symposium on User Interface Software and Technology*, ACM Press, 1994, 195-204.

[Fitzmaurice *et al*., 1995] George W. Fitzmaurice, Hiroshi Ishii and William Buxton, Bricks: Laying the foundations for graspable user interfaces. *Human Factors in Computing Systems, CHI '95 Conference Proceedings,* ACM Press, 1995, 442-449.

[Flowers, 1975] Kenneth Flowers, Handedness and controlled movement. *British Journal of Psychology*, *66*, 1975, 39-52.

[Guiard, 1987] Yves Guiard, Asymmetric division of labor in human skilled bimanual action: the kinematic chain as a model. *The Journal of Motor Behavior, 19 (4)*, 1987, 486-517.

[Kabbash *et al*., 1993] Paul Kabbash, I. Scott MacKenzie and William Buxton, Human performance using computer input devices in the preferred and non-preferred hands. *Human Factors in Computing Systems, INTERCHI '93 Conference Proceedings*, ACM Press, 1993, 474-481.

[Nigay and Coutaz, 1993] Laurence Nigay and Joëlle Coutaz, A design space for multimodal systems: concurrent processing and data fusion. *Human Factors in Computing Systems, INTERCHI '93 Conference Proceedings*, ACM Press, 1993, 172-178.

[Schmidt *et al.*, 1979] Richard A. Schmidt, Howard Zelaznik, Brian Hawkins, James S. Frank, and John T. Quinn Jr., Motor-output variability: a theory for the accuracy of rapid motor acts. *Psychological Review, 86 (5)*, 1979, 415-451.

[Todor and Doane, 1978] John I. Todor and Thomas Doane, Handedness and hemispheric asymmetry in the control of movements. *Journal of Motor Behavior, 10 (4)*, 1978, 295-300.

[Welford, 1968] A. Welford, *Fundamentals of Skill*. Methuen, London, 1968.

[Zhai *et al.*, 1997] Shumin Zhai, Barton A. Smith, and Ted Selker, Improving browsing performance: a study of four input devices for scrolling and pointing tasks. *Proceedings of INTERACT '97: The IFIP Conference on Human-Computer Interaction*, 1997, 286-292.

PAC-malli

Tero Kukola

Tässä luvussa keskittytään agenttipohjaisen PAC-mallin ja sen laajennuksen PAC-Amodeuksen esittelyyn. Lopussa vertaillaan PAC-malliperhettä komponenttipohjaiseen MVC++-malliin. Esitellyt mallit liittyvät käyttöliittymien käytännön toteutukseen tarjoamalla laajennettavissa olevan arkkitehtuurin, jonka avulla suuretkin käyttöliittymätoteutukset on mahdollista saada toimiviksi ja helpommin hallittaviksi.

1. Johdanto

Käyttäjän ja sovelluksen välinen vuorovaikutus pitää saada mahdollisimman helpoksi ja tehokkaaksi. Tätä on pyritty toteuttamaan esimerkiksi reaaliaikaisella vuorovaikutteisuudella, suoravaikutteisuudella ja multimodaalisuudella. Samalla käyttöliittymien toteuttamisen vaativuus on kasvanut.

Vaatimukset sovellusten toiminnallisuuden suhteen kasvavat jatkuvasti kasvattaen sovellusten monimutkaisuutta ja kokoa. Toisaalta ohjelmistokehitykseen tarkoitetut työkalut ovat kehittyneet oliopohjaisiksi ja siirtyneet askeleen kohti käsitteiden maailmaa.

Lisääntyneiden vaatimusten ongelmaa ratkaisemaan ja oliomenetelmiä hyödyntämään on kehitetty erilaisia arkkitehtuurimalleja, kuten PAC ja MVC++. Tämä paperi keskittyy PAC-mallin ja sen laajennuksen PAC-Amodeuksen esittelyyn. Lopussa vertaillaan PAC-malliperhettä MCV++-malliin.

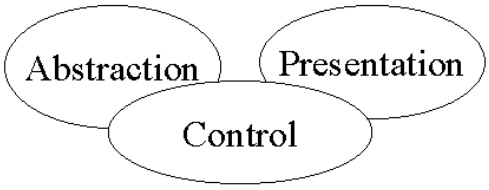
PAC-mallia koskevissa lähteissä käyttöliittymän ajatellaan koostuvan agenteista. Käyttöliittymän agentit on mallinnettu esimerkiksi PAC- tai MCV++-mallin mukaisesti ja käytössä on poikkeuksetta termi käyttöliittymäagentti. Useimmat todennäköisesti käyttäisivät tässä yhteydessä agentin sijasta yleisempiä termejä komponentti, olio tai objekti. Tässä esityksessä käyttöliittymä koostuu kuitenkin lähteiden tavoin agenteista.

2. PAC-malli

Tämän luvun esitys perustuu lähteeseen [Coutaz et al, 1993]. PAC on käsitetason malli käyttöliittymäagentista, joka koostuu kolmesta erillisestä komponentista: Presentation, Abstraction ja Model. Tämä jako antanut nimen PAC-mallille.

PAC-agentin toiminnallisuus jakautuu eri komponenttien välille seuraavasti:

1. *Presentation*: Presentation on PAC-agentin varsinainen käyttöliittymäosa. Presentation-komponentin tehtävänä on esittää käyttöliittymä käyttäjälle ja tulkita käyttäjän antamia syötteitä. Presentation ei ole koskaan suoraan yhteydessä abstraktioon tai muihin PAC-agentteihin.
2. *Control*: Control toteuttaa PAC-agentin Presentation- ja Abstraction-komponentin väliset riippuvuussuhteet ja kommunikaation muiten agenttien kanssa. Yleisesti PAC-mallissa kaikki riippuvuussuhteet eri komponenttien välillä ohjataan Control-komponentin kautta. Control-komponentit toimivat "liimana", jolla PAC-agenteista rakennetaan koko käyttöliittymähierarkia.
3. *Abstraction*: Abstraction sisältää sovelluksen kannalta olennaisen agentin tilan ja toiminnallisuuden. Abstraction voi käytännössä sisältää myös sovelluksen ytimen toiminnallisuutta.



Kuva 1: PAC-agentin komponentit

PAC-agenteista voidaan rakentaa hierarkia yhdistämällä niitä Control-komponenteista. Agenttien välinen kommunikaatio tapahtuu hierarkian kautta. Hierarkia on rakenteeltaan puumainen. Kommunikaatio pystysuuntaista, eli agentti kommunikoi ainoastan yliagenttinsa tai aliagenttinsa kanssa.

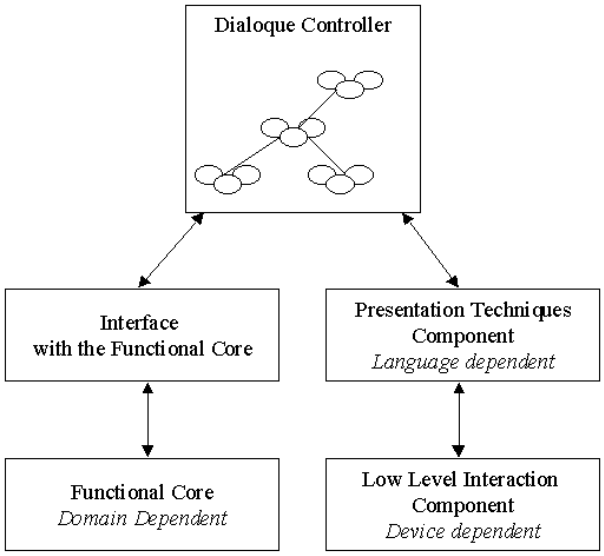
Koska PAC on puhtaasti käsitetason käyttöliittymäagenttimalli, ei se voi ottaa kovinkaan paljoa kantaa toteutuksen suhteen. PAC-mallin mukaisen agentin eri piirteet voivat vaihdella runsaasti valitun toteutusmenetelmän ja kohdesovelluksen asettamien vaatimusten seurauksina. Tämän seurauksena PAC-agentin toiminnasta on jätetty määrittelemättä toiminnallisuutta, joka pitää pitää ratkaista sovelluskohtaisesti:

1. PAC-malli ei ota kantaa siihen, miten Control-komponentissa toteutetaan Presentationin ja Abstractionin välinen kommunikaatio ja toisaalta miten agenttien Control-komponentit kommunikoivat keskenään. Eräitä mahdollisia toteutustason kommunikointimenetelmiä ovat suorat funktiokutsut tai viestipohjainen kommunikaatio. PAC-malli ei myöskään määrittele, miten Control-komponenttien välinen koordinaatio toteutetaan.
2. PAC-malli ei ota kantaa siihen, miten Presentation-komponentti toteutetaan. Tämä tarkoittaa sitä, että Presentation komponentin sisäinen rakenne ja vuorovaikutus käyttäjän kanssa (syötteiden ja tulosteiden käsittely) on jätetty avoimeksi. Monimutkaiset Presentation-komponentit voidaan tarvittaessa pilkkoa useammaksi PAC-agentiksi. Tällöin yhden PAC-agentin toiminnallisuus korvataan PAC-agenttiryhmän yhteisellä toiminnallisuudella.
3. PAC-malli ei määrittele mitään Abstraction-komponentin sisällöstä.
4. PAC-malli ei ota kantaa siitä, miten PAC-agenttien hierarkia luodaan tai miten PAC-mallin mukainen käyttöliittymä ylipäätään luodaan.

3. PAC-Amodeus

3.1. Rakenne ja toiminta

PAC-malli käyttää Arch-mallia toiminnallisen jakonsa perustana. Lähteessä [Coutaz] on esitelty Arch-malli selkeästi. Lähteissä on vaihtelua PAC-Amodeuksen mukaisen Arch-mallin osien nimeämisessä. Tämän luvun esitys perustuu lähteeseen [Coutaz et al, 1995] ja käyttää sen mukaista nimeämistä.



Kuva 2: PAC-Amodeuksen mukainen Arch-malli

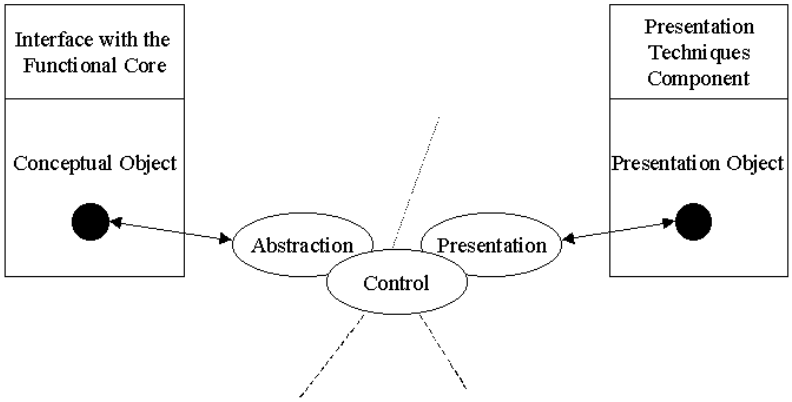
Jatkossa käytössä ovat seuraavat lyhenteet:

FC	Functional Core
IFC	Interface with the Functional Core
DC	Dialogue Controller
PTC	Presentation Techniques Component
LLIC	Low Level Interaction Component

PAC-Amodeus on rakenteeltaan alkuperäisen Arch-mallin kaltainen. PAC-Amodeus ei ota kantaa FC:n, IFC:n, PTC:n ja LLIC:n toteutukseen, mutta jalostaa Arch-mallia seuraavilla tavoilla:

1. PTC ja LLIC ovat molemmat toteutuskieliriippuvaisia. LLIC on tämän lisäksi laiteriippuvainen. Laitteet ovat tässä tapauksessa fyysisiä syöttö- tai tulostelaitteita, kuten hiiri, näppäimistö, näyttö ja niin edelleen. PTC ei ole laiteriippuvainen.
2. DC on toteutettu PAC-agenttien hierarkialla.

PAC-agenttien Abstraction-osa voi viitata suoraan IFC:hen tai FC:hen, mikäli IFC ei ole olemassa. Samalla tavalla PAC-agentin Presentation-osa voi viitata suoraan PTC:hen tai LLIC:hen mikäli PTC ei ole olemassa. Tällainen suora osoittaminen ei ole alkuperäisen PAC-mallin mukainen, jossa agentit voiva viitata ainoastaan hierarkiassa ylempänä tai alempana oleviin agentteihin. Ratkaisun motivaationa on tehdä PAC-Amodeuksesta tehokkaampi ja välttää tarpeetonta hierarkian läpikäyntiä.



Kuva 3: DC:n PAC-agentti. Agentin Abstraction- ja Presentation-komponentit voivat osoittaa IFC:n ja PTC:n komponentteja suoraan ilman hierarkian läpikäyntiä. Katkoviivat merkitsevät kommunikaatiota muiden agenttien kanssa.

Lähteessä [Coutaz et al, 1995] ehdotetaan PTC:n jakamista kahteen kerrokseen, jotka ovat Extension Layer (EL) ja Interaction Toolkit Adapter (ITA). ITA määrittelee sovelluksen käyttöön virtuaalisen käyttöliittymäkirjaston, jolla agenttien Presentation-komponentit voidaan toteuttaa. Virtuaalinen käyttöliittymäkirjasto käyttää toteutukseensa sopivaa olemassaolevaa kirjastoa (esimerkiksi Sun Microsystemsin Java Swing tai Trolltechin Qt) ja sen vaihtaminen vaatii osittaisen uudelleenkirjoittamisen. Agenttien Presentation-komponenttien toteutukset säilyvät ennallaan - tässä piilee koko PTC:n olemassaolon idea. Sellaiset komponentit, joita ei voida suoraan toteuttaa valmiilla työkalukokoelmaan sijoitetaan Extension Layeriin.

3.2 Slinky metamodel

Yksi arkkitehtuurimalli ei yleensä toimi jokaisen sovelluksen suhteen. Esimerkiksi IFC:n ja PTC:n tarkoituksena on vähentää riippuvuuksia komponenttien välillä, lisätä muokattavuutta ja parantaa siirrettävyyttä. Valitettavasti kyseisten komponenttien tai rajapintojen olemassaolo vähentää

sovelluksen tehokkuutta. Esimerkiksi reaaliaikainen suoravaikutteinen sovellus vaatii nopeaa syötteen ja datan käsittelyä. Tällaisessa sovelluksessa puhtaasti PAC-Amodeuksen mukainen arkkitehtuuri saattaa osoittautua liian hitaaksi tai vähintäänkin turhaksi. PAC-Amodeuksen täydellinen noudattaminen ei ole siis aina perusteltua.

PAC-Amodeuksen yhteydessä on käytössä Arch-mallista tuotu "Slinky metamodel", jonka ideana on soveltaa PAC-Amodeusta sovelluksen vaatimusten mukaisesti. PAC-Amodeus mallista luodaan vaatimuksiin sopiva malli, jossa mallin eri komponenttien merkitystä on punnittu erikseen. Esimerkiksi nopeutta vaativan reaaliaikasovelluksen arkkitehtuurissa IFC ja PC ovat suurella todennäköisyydellä yksinkertaisia funktiorajapintoja tai ne jätetään mallista kokonaan pois. Usealla eri laitealustalla ajettavan, hajautettuihin tietokantoihin ja transaktioihin perustuvan ja tuhansille käyttäjille tarkoitettun sovelluksen arkkitehtuurissa IFC ja PC lienevät melko suuria toiminnallisia kokonaisuuksia.

3.3 Amodeuksen heuristiikat

PAC-Amodeus mallia varten on kehitetty seuraava sääntöjen joukko, joilla on tarkoitus helpottaa PAC-Amodeus mallin mukaisen arkkitehtuurin suunnittelu- ja toteutusvaiheita.

Sääntö1: Mallinna pääikkuna agentiksi.

Selitys: PAC-agentteihin perustuvan käyttöliittymän juuriagentti muodostaa yleensä sovelluksen pääikkunan. Sovelluksen käyttöliittymä ei välttämättä aina ole yhden ikkunan sisällä, vaan voi jakautua esimerkiksi useampiin erillisiin, enemmän tai vähemmän tasa-arvoisiin ikkunoihin. Erilliset ikkunat toteutetaan agentteina samaan tapaan kuin pääikkuna ja tulkitaan pääikkunan lapsikkunoiksi, jotka ovat oman hierarkiansa pääikkunoita. Yksinkertaisia apuikkunoita, kuten viestidialogeja, ei tarvitse mallintaa agentteihna.

Sääntö 2: Käytä agenttia säilyttämään visuaalinen yhtenäisyys useampien näkymien välillä

Selitys: Mikäli yhdestä mallista on monta näkymää ja kyseiset näkymät on mallinnettu agentteina, luodaan hiearkiaan yliagentti "Multiple View", jonka aliagenteiksi näkymät asetetaan. Luotu agentti linkittää aliagentit. Kun jossakin aliagentissa tapahtuu jotakin, mikä vaatii muutoksia muiden aliagenttien näkymissä, ilmoitetaan siitä yliagentille. Yliagentti ilmoittaa muutoksesta muille aliagenteille. Menetelmällä siis vähennetään agenttien välisiä riippuvuuksia ja keskitetään näkymien kontrolli.

Sääntö 3: Mallinna työkalupaletti agenttina

Selitys: Agentin Abstraction-komponentti sisältää listan niistä käsitteiden luokista, jotka voidaan paletista valita. Presentation-komponentti käyttää käyttöliittymäkirjaston toiminnallisuutta rakentamaan itse paletin. Control-komponentti kuvaa käyttäjän valinnat Presentation- ja Abstraction-komponenttien välillä, muuntaa tapahtumat viesteiksi, joilla on korkeamman tason merkitys, ja lähettää viestit ylöspäin hierarkiassa.

Sääntö 4: Mallinna ikkunan muokattava työalue agenttina

Selitys: Sovelluksen ikkunassa voi olla alue, jossa käyttäjä voi muokata sovelluksen käsitteitä. Työalue on vastuussa käyttäjän syötteiden tulkitsemisesta, muokkaa käsitteitä syötteistä tulkitun toiminnan perusteella, mikäli muokkausta varten ei ole erikseen omaa agenttia, ja käsittelee viestit aliagenteista, mikäli niissä on muokattavia käsitteitä. Lisäksi työalue voi olla vastuussa muokattavien käsitteiden välisten yhteyksien graafisesta esittämisestä. Työalue on siis agentti, jolla on taito käsitellä käsitteiden joukkoa. Komponentit, joissa käsitteitä ei muokata, ei tarvitse mallintaa agentteina.

Sääntö 5: Mallinna monimutkainen käsite agenttina

Selitys: Käsite voi olla sisäiseltä rakenteeltaan monimutkainen, sen graafinen esitys voi olla rakenteinen tai sillä voi olla useita erilaisia graafisia esityksiä.

- Sisäiseltä rakenteeltaan monimutkainen käsitte muodostuu alikäsitteistä. Rakenteen pitäisi näkyä käyttäjälle.
- Käsitteellä voi olla useita erilaisia esityksiä eri paikoissa. Näkymät hallitaan siihen erikoistetulla agentilla.

Sääntö 6: Jos sovellus muodostuu useista pääikkunoista ja ikkunoiden välillä on riippuvuus, mallinetaan tämä riippuvuus agentin avulla.

Sääntö 7: Yhdistetyttyjä, usean agentin vastuulle hajautettuja toimintoja varten luodaan agentti.

Selitys: Agenttien välillä on syntaktinen riippuvuus, mikäli käyttäjän toiminta näissä ikkunoissa voidaan yhdistää korkeamman tason abstraktioksi (fuusio). Esimerkki: käyttäjä valitsee jonkin objektin piirto-ohjelman paletista ja piirtää työalueelle kyseisen objektin. Nämä kaksi toimintoa (valinta ja piirto) yhdistetään korkeammaksi abstraktioksi (piirrä valittu objekti) ns. sementtiagentilla.

Sääntö 8: Eri komponenttien välillä vallitseva merkitystason (semanttinen) riippuvuus mallinnetaan agentilla.

Selitys: Sääntö on samankaltainen kuin sääntö 7. Erona semanttinen agentti ylläpitää semanttista riippuvuutta eikä yhdistä käyttäjän toimintoja.

Sääntö 9: Jos LLIC:n tai PTC:n toteuttamiseen käytetyillä työkaluilla on mahdollista toteuttaa agentti suoraviivaisesti, niin tee agentista vuorovaikutusobjekti ja liitä se yliagentin Presentation-komponenttiin.

Sääntö 10: Jos yliagentilla on täsmälleen yksi aliagentti ja sovelluksen jatkokehityksessä ei ole odotettavissa muutoksia tilanteeseen ja agenttien välinen liikenne vie resursseja, voi olla hyödyllistä yhdistää agentit yhdeksi agentiksi.

Mainittujen sääntöjen perusteella PAC-mallissa voi olla agentteja, joista puuttuu Presentation- tai Abstraction-komponentti tai molemmat. Agentin Abstraction-komponentti voi olla yksinkertainen linkki IFC:hen tai suoraan FC:hen, mikäli IFC:tä ei ole olemassa. Agentit, jotka ovat puhtaita kontrollereita (ilman Presentation- ja Abstraction-komponenttia), eivät komponentteina kuulu arkkitehtuurin suunnitteluvaiheeseen. Ne ovat lähinnä toteutusvaiheen ratkaisuja eräisiin usein toistuviin ongelmiin.

4. Arviointia ja vertailua MVC++-malliin

Tässä luvussa oletetaan, että lukija tuntee MVC++-mallin. Lähteestä [Jaaksi, 1995] löytyy hyvä kuvaus MVC++-mallista.

MVC++-malli vs. PAC-malli:

- MVC++-mallin Model vastaa PAC-mallin Abstraction-komponenttia
- MVC++-mallin View ja Controller vastaavat PAC-mallin Presentation-komponenttia
- PAC-mallin Control-komponentin toiminnallisuus on MVC++-mallissa kontrollerin vastuulla

MVC++-malli voitaisiin helposti nostaa ulos C++-kehyksestään käsitteelliselle tasolle yhdistämällä se samalla tavalla Arch-malliin kuin PAC-Amodeus. MVC++-malli on monella tavalla PAC-Amodeuksen kaltainen. Molemmat järjestävät agentit puuhierarkiaksi, agenttien välinen kommunikaatio tapahtuu kontrollerin kautta ja näkymä ei saa olla suorassa yhteydessä malliin. MVC++-mallin abstraktit partnerit vastaavat osittain PAC-Amodeuksen PTC:tä ja IFC:tä. Suurin ero PAC-Amodeuksen ja MVC++-mallin

välillä on se, että MVC++-agenttien käyttöliittymän toimintalogiikka on toteutettu kontrollerin ja näkymän tiukalla yhteistyöllä. PAC-Amodeuksessa agentin kontrolleri ei ota kantaa näkymän sisäiseen toimintaan.

PAC-Amodeuksen tapa määritellä Controller-komponentti on toisaalta puhtaudessaan parempi kuin MVC++:n kontrolleri. Ajoittain MVC++-kontrolleria toteuttaessa tulee mieleen, että sillä on liikaa tehtäviä. On tapauksia, joissa olisi hyödyllistä jakaa kontrolleri useampaan pienempään kontrolleriin. Toisaalta, on osittain makuasia haluaako kontrollerin toiminnan kasautuvan yhdelle komponentille vai hajautetuksi useamman komponentin vastuulle - vaadittu toiminnallisuus on kuitenkin toteutettava jossakin.

Eräs kiinnostava vaihtoehto olisi sekoittaa malleja. Yhtenä esimerkkinä toimikoon asetusten tekoon tarkoitettu dialogi, joka on toteutettu PAC-agenttina. Dialogi sisältää useita näkymiä, joissa jokaisessa on valittavissa paljon erilaisia asetuksia. Presentation-komponentti voitaisiin toteuttaa MVC++-mallin mukaisesti. Näkymäkomponentteja olisi jokaista dialogin näkymää kohti yksi kappale, malli sisältäisi dialogissa tehdyt muutokset ja kontrolleri koordinoisi dialogin käyttämisen. Mikäli dialogissa olevat valinnat hyväksytään, välittää kontolleri tehdyt muutokset PAC-agentin kontrollerille. Mikäli tehtyjä valintoja ei hyväksytä, voidaan dialogissa olevan mallin tila hävittää tarpeettomana. Toinen vastaava esimerkki olisi wizard-dialogin toteuttaminen. Esimerkit voitaisiin toteuttaa myös pilkkomalla agentti useamman agentin ryhmäksi, joka koordinoitaisiin yhdellä puhtaalla kontrolleriagentilla.

5. Yhteenveto

PAC ja PAC-Amodeus ovat varsin kiinnostavia käsitemalleja mallintaa käyttöliittymäagentteja. Erot MVC++-mallin mukaiseen rakennelmaan ovat kuitenkin melko vähäisiä. Kiinnostavin vaihtoehto on tutkia PAC-Amodeuksen ja MVC++-mallin ominaisuuksien yhdistämistä tai vain hakea inspiraatiota sovellusten toteuttamiseen.

Huomattavaa on myös se, että molempien, PAC-Amodeuksen ja MVC++:n, painopiste on käyttöliittymän ja sovelluksen ytimen toimintalogiikan erottamisessa. Tämä jako on havaittu käytännössä toimivaksi. Suurimmat toteutuksen ongelmat syntyvät kuitenkin käyttöliittymän agenttihierarkian koordinoinnista. Sekä PAC-Amodeus että MVC++ tarjoavat hieman suuntaviivoja hierarkian koordinointiin antamalla sääntöjä hierarkian järjestämiseen kontrollereiden avulla. On totta, että kontrollerien ja hieararkian toteutus on vahvasti riippuvainen sovelluksesta. Hedelmällinen selvityksen aihe olisi kuitenkin etsiä mahdollisimman yleisiä kontrollereiden ja hierarkian piirteitä, jotka ovat sovellettavissa mahdollisimman moniin tilanteisiin.

Lähteet

[Coutaz et al, 1993] Coutaz J., Nigay L., Salber D., *Conceptual Software Architecture Models for Interactive Systems*, Amodeus Project Document: SystemModelling / WP11, LGI-IMAG, Grenoble, 1993

[Coutaz et al, 1995] Coutaz J., Nigay L., Salber D., *Agent-Based Architecture Modelling for Interactive Systems*, Amodeus Project Document: SystemModelling / WP53, LGI-IMAG, Grenoble, 1995

[Coutaz, 1997] J. Coutaz, *PAC-ing the Architecture of Your User Interface*, julkaistu teoksessa DSV-IS'97, Eurographics Workshop on Design, Specification and Verification on Interactive Systems, Springer Verlag Publ., pp. 15-32

[Coutaz] J. Coutaz, Software Architecture Modeling for User Interfaces, IMAG, Grenoble

[Jaaksi, 1995] Ari Jaaksi, *Implementing Interactive Applications in C++*, julkaisussa Software Practice & Experience, 1995

Hajautettu kognitio

Kimmo Koivunen

Tässä luvussa tarkastellaan hajautetun kognition teoriaa. Teorian esittelyn jälkeen syvennytään erityisesti sen merkitykseen ihmisen ja teknologian väliselle vuorovaikutukselle. Tarkastelussa käytetään viitekehystä, jonka ovat kehittäneet Hollan, Hutchins ja Kirsh [2000].

1. Johdanto

Tietoverkkojen myötä on siirrytty ajasta, jolloin työskenneltiin yksin omalla tietokoneella erillään muista, aikaan, jolloin tietokoneet ovat kytketty verkkojen kautta toisiinsa ja jolloin mahdollistuu monipuolinen ja tehokas yhteistyö tietokoneiden avulla muiden kanssa. Lisäksi tietoverkkoihin kytketyt teknologiset laitteet ovat tulleet entistä huomaamattomammin läsnäoleviksi kaikkialle. Tämä asettaa uusia haasteita ihmisen ja tietokoneen välisen vuorovaikutuksen ymmärtämiselle ja tutkimiselle. On siirryttävä ainoastaan työpöydällä olevan mikron huomioivasta näkökulmasta monimuotoisen, verkottuneen ja kaikkialla läsnäolevan teknologian huomioon ottavaan näkökulmaan, kun halutaan ymmärtää ihmisen ja teknologian välistä

vuorovaikutusta. Hajautettu kognitio (distributed cognition) tarjoaa tähän uuden, mielenkiintoisen mahdollisuuden. Teoriana se auttaa ymmärtämään tietyn ympäristön ja sen toimijoiden kogniitiivisia prosesseja. Tätä ymmärrystä voidaan hyödyntää tutkiessa ihmisen ja teknologian välistä vuorovaikutusta. Teorian avulla voidaan myös rakentaa viitekehys, joka soveltuu uusien ja teknologiaa hyödyntävien sovellusten suunnitteluun. [Hollan, Hutchins and Kirsh, 2000].

Hajautetun kognition teoriassa kognitio nähdään tapahtuvan ihmisten ja resurssien välillä, ei pelkästään yksilön omassa päässä. Hajautettu kognitio nousee mielekkääksi näkökulmaksi ihmisten ja teknologian vuorovaikutuksen ymmärtämiseksi teknologian nopeasti kehittyessä, koska inhimilliselle toiminnalle on tyypillistä siirtää kogniitiviista kuormaa ympäristöön aina kun se on mielekästä. Tällöin sen avulla voidaan paremmin ymmärtää kognitiivisten järjestelmien monimuotoista ja joustavaa rakennetta.

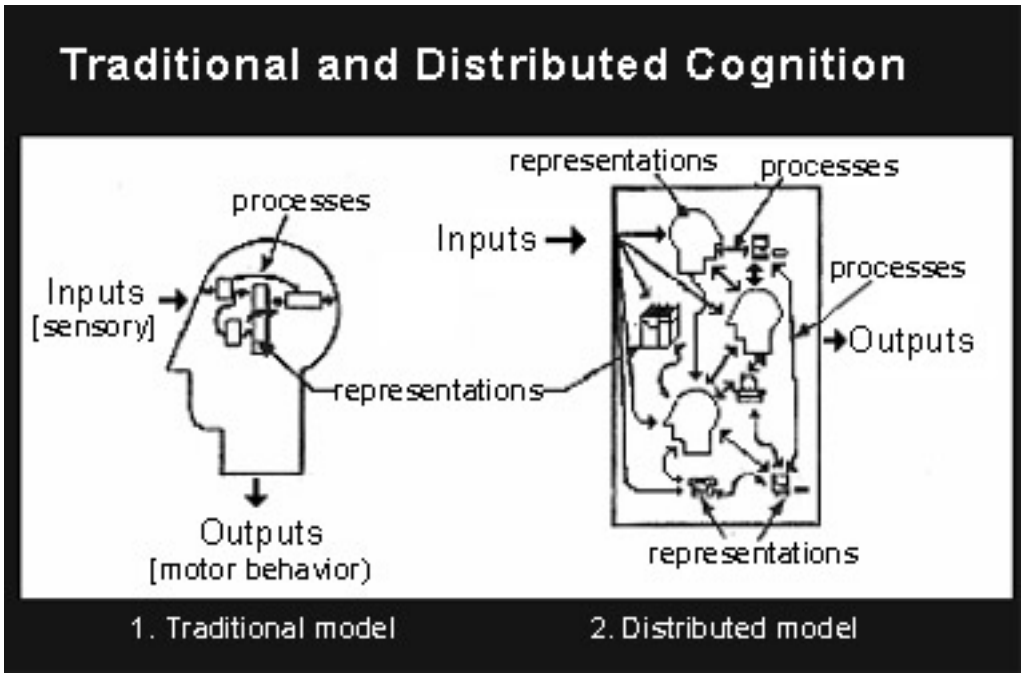
Hajautettu kognition teoria on kehitetty Etelä-Kalifornian yliopistossa Edwin Hutchinsin johdolla 1980-luvun puolivälissä [Rogers, 2000]. Teorian tekee mielenkiintoiseksi näkemys yhteistoiminnassa tapahtuvasta kognitiosta. Huomattavaa on myös, että hajautetussa kognitiossa erilaiset artefaktit voivat olla osallisena kognitiivisissa prosesseissa. Teorian taustalta löytyy teoreettisia ja metodologisia näkemyksiä kognitiotieteestä, antropologiasta ja sosiaalitieteistä [Rogers, 1997]. Sitä voidaankin pitää hyvin monitieteisenä näkemyksenä.

Tässä esityksessä tarkastellaan hajautetun kognition teoriaa ja selvitetään sen teoreettisia taustoja. Tämän jälkeen syvennytään erityisesti sen merkitykseen ihmisen ja teknologian vuorovaikutuksen kannalta.

2. Teoria

2.1. Yleistä

Hajautetun kognition teoria, kuten muutkin kognitiiviset teoriat, pyrkii kognitiivisten prosessien ymmärtämiseen. Perinteisesti kognitiiviset prosessit ja representaatiot nähdään tapahtuvan pelkästään yhden yksilön omassa päässä. Hajautetun kognition teoriassa laajennetaan näkemystä kognitiosta vuorovaikutukseksi tietyssä ympäristössä toimivien ihmisten, teknologian ja materiaalien väliseksi (Kuva 1).



Kuva 1: Perinteinen ja hajatettu kognitio [Rogers 2000]

Hajautettu kognitio etsii kognitiivisia prosesseja kaikkialta missä niitä saattaa tapahtua. Hajautetussa kognitiossa odotetaan löydettävän järjestelmä, joka voi joustavasti muotoilla itseänsä alijärjestelmiin, jotka rinnakkaisesti toimien voivat saattaa päätökseen hyvinkin moninaisia tehtäviä. Kognitiivinen prosessointi on rajattu siihen osallistuvien elementtien toiminnallisen suhteen mukaan, eikä niinkään elementtien tilallisen tai fyysisen sijainnin suhteen [Hollan, Hutchins and Kirsh, 2000]. Huomion arvoista on myös se, että hajautetussa mallissa representaatiot ovat sekä sisäisiä että ulkoisia ja niitä voivat olla kaikenlaiset artefaktit, kuten kirjat tai tietokoneelle tallennettu tieto.

Ensimmäisenä periaatteellisena erona vanhoihin teorioihin onkin analysoitavan yksikön rajaus. Yksiköksi ei rajata yksilöä vaan tapauskohtaisesti harkitaan mitkä kaikki toimijat kuuluvat yhteen.

Toinen periaate, joka poikkeaa voimakkaasti perinteisestä näkemyksestä, on niiden mekanismien rajaus, mitkä voivat osallistua kognitiiviseen prosessiin. Perinteisesti kognitiivinen prosessointi nähdään täysin yksittäisen ihmisen ominaisuudeksi tai kyvyksi, mutta hajautetun kognition teoriassa siihen voidaan nähdä osallistuvan fyysinen ympäristö erilaisine artefakteineen. Huomattavaa on myös se, ettei ihmisen tarvitse olla se toimija, joka vastaanottaa kognitiiviseen järjestemään saapuvaa uutta informaatioita. Erilaiset artefaktit voivat tehdä tämän yhtä hyvin kuin ihminen. Järjestelmässä toimivat ihmiset saattavat vasta myöhemmin prosessoida tätä informaatioita

artefaktien avulla. Vastaavasti järjestelmän reaktiot ulkopuolelle voivat varsinaisesti olla artefaktien toimintaa.

2.2. Kognitiiviseen prosessiin osallistujat

Ihmisten muodostama ryhmä voi osallistua kognitiiviseen prosessiin, sillä se voidaan hajauttaa ryhmän jäsenten kesken. Tällöin ryhmä voi esimerkiksi suoriutua vaikeammista tehtävistä kuin mihin sen yksittäiset jäsenet pystyisivät, koska osaprosesseihin jaettuna kokonaisuus on helpommin hallittavissa. Lisäksi eri yksilöiden ulkoistamat tiedot voivat ohjata muiden tietoja ja taitoja ja näin saadaan aikaiseksi jotakin uutta tietämytsä, jota ryhmässä toimijoilla ei ollut ennestään. Esimerkkinä voidaan ajatella suurta ohjelmistoprojektia. Nykyiset ohjelmistot sisältävät hyvin moninaisia ominaisuuksia, joita tuskin enää yksi henkilö pystyisi hallitsemaan vaan sen toteutuksessa käytetään suunnittelijoita, ohjemoijia, käyttöliittymä-asiantuntijoita, vain muutamia mainiten. Jokainen heistä hallitsee jonkin tietyn osa-alueen ohjelmiston toteutuksessa, mutta yhdessä toimien he voivat saada aikaiseksi enemmän kuin yksin pystyisivät.

Samoin erilaiset artefaktit voivat olla tärkeä osa kognitiivista järjestelmää. Niitä voidaan käyttää apuna asioiden hahmottamisessa tai niiden avulla voidaan keventää kognitiivista kuormaa. Esimerkiksi sokean keppi tai mikrobiologin mikroskooppi ovat tärkeitä välineitä heidän tavalleen havainnoida ympäristöään [Hollan, Hutchins and Kirsh, 2000]. Järjestelmän artefaktit voivat olla myöskin reprensaatioita, joiden avulla hahmotetaan ratkottavaa ongelmaa. Niihin myös ulkoistetaan tietoja muiden toimijoiden avuksi ja tueksi.

Kolmas tärkeä ulottuvuus on kulttuuri. Jokaisessa tomintaympäristössä on oma, sille tyypillinen kulttuurinsa, joka vaikuttaa kokonaisvaltaisesti kaikeen kognitiivisen järjestelmän toimintaan. toisaalta järjestelmä toimii aina jossakin kulttuurisidonnaisessa ympäristössä, jolloin koko kultturin historia vaikuttaa järjestelmään. Toisaalta erityisesti järjestelmässä käytetyt representaatiot ovat siinä vallitsevan kulttuurin tuotoksia, joita voi olla vaikea ymmärtää ilman kulttuurin tuntemusta. Kultturi myös saattaa olla esteenä uusien toimintatapojen näkemiselle tai hyväksynnälle ja se voimakkaasti ohjaa vakiintuneita tapoja. Kieli on myös tärkeä osa kulttuuria ja ilman sen tuntemista voi olla mahdotonta ymmärtää järjestelmässä käytettyjä representaatioita. Esimerkiksi kirjoitetuissa dokumenteissa ja täytetyissä kaavakkeissa näkyy vallitseva kulttuuri, joka ohjaa mm. niihin suhtautumista mutta myös niiden muotoa.

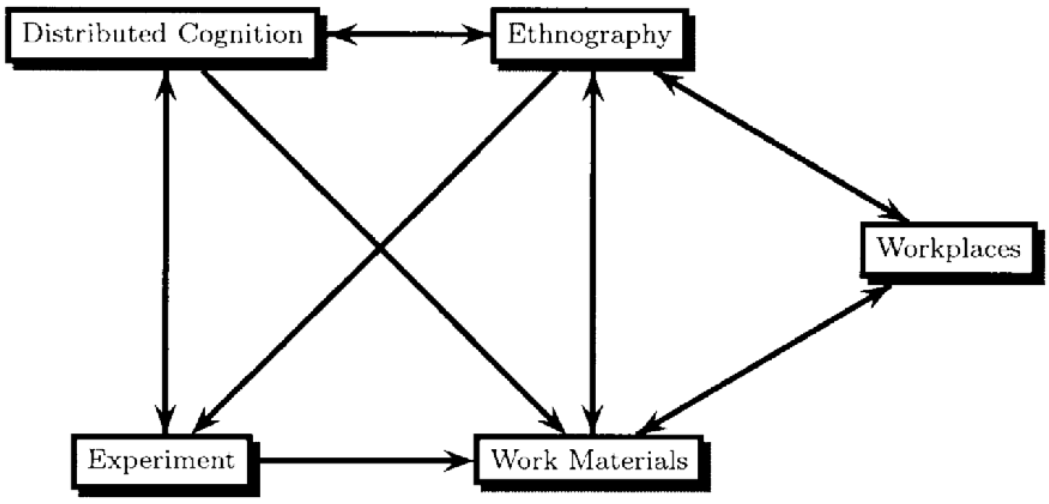
2.3. Tutkimus

Haasteeksi hajautetun kognition teoriassa nouseekin se, miten sitä voidaan tutkia tai havainnoida. Millaisin menetelmin pystytään nostamaan esille sen erilaiset ilmentymät? Jotta voidaan havainnoida artefaktien ja ihmisten muodostamia kognitiivisia prosesseja, tulee tutkimus suorittaa autenttisessa ympäristössä. Tällaiseen tutkimukseen tarvitaan etnografisia menetelmiä. Tutkijan on myös tunnettava riittävän hyvin tutkimuksen kohteena olevaa sovellusaluetta, jotta hän voisi ymmärtää miten ihmiset käyttävät ympäristöään kognitiivisten prosessien tukena. Tutkijan tulisinkin olla hyvin perillä alalla käytetystä kielistä ja muusta vallitsevasta kulttuurista. Ilman niiden ymmärrystä tutkija ei voi tunnistaa tapoja, joilla ihmiset hyödyntävät erilaisia artefakteja. Tutkimusalueen ammattikielen ymmärrys korostuu siinä miten ympäristössä toimivat siirtävät tietoa toisilleeen. Kielen ymmärtäminen on tärkeää myös reprensentaatioden tulkinnassa ja ymmärtämisessä. Näiden moninaisten vaikutussuhteiden selvittäminen ja saadun tutkimustiedon suodattaminen vaatii aikaa ja pitkäjänteisyyttä.

3. Teorian soveltaminen

3.1. Viitekehys

Hollan, Hutchins ja Kirsh [2000] ehdottavat viitekehystä, jota voidaan käyttää apuna ihmisen ja teknologian välisen vuorovaikutuksen tutkimuksessa (kuva 2). Viitekehyksellä tarjotaan suunnitelijolle paremmat mahdollisuudet erilaisten uusien digitaalisten materiaalien suunnitteluun.



Kuva 2: Viitekehys HCI-alan tutkimukseen [Hollan, Hutchins and Kirsh, 2000]

Viitekehyksessä hajautettu kognitio (kuvassa 2 Distributed Cognition) tarjoaa pääperiaatteita, jotka pätevät yleisesti. Nämä periaatteet auttavat tnnistamaan tutkittavasta työympäristöstä (Workplaces) ilmiöitä, jotka ansaitsevat tarkempaa tarkastelua. Etnografia (Ethnography) tarjoaa menetelmiä näiden ilmiöiden tarkkailuun ja havainnointiin. Tällä tavoin saatuja tuloksia tarkastellaan hajautetun kognition periaatteiden mukaan ja näin saadaan uusia, tarkempia päätelmiä. Nämä päätelmät saattavat vaatia rajattuja kokeita (Experiment) tuloksien vahvistamiseksi. Kokeilla voidaan havainnoida rajatussa ympäristössä jotakin tiettyä luonnollisessa ympäristössä tapahtuvaa ilmiötä ja näin saada tarkempaa informaatioita, jolloin muun kerätyn informaation kanssa pystytään rakentamaan kokonaisvaltaisempi ja tarkempi näkemys todellisuudesta. Koetilanteella voidaan siis varmistaa todellisessa ympäristössä saatuja havaintoja, koska kokeessa pystytään rajaamaan ympäristön monimuotoisuutta ja keskitymään juuri tiettyihin seikkoihin ilman häiriötekijöitä. Vahvistettujen päätelmien jälkeen voidaan ympäristöön tuoda uusia materiaaleja (Work Materials). Tässä vaiheessa viitekehyksen mukainen työ ei ole vielä valmis vaan nyt tulee jälleen etnografisin menetelmin varmistaa, että uusi työmateriaali tai –tapa todella sopii ympäristöön. Tämä varmistetaan jälleen hajautetun kognition periaatteiden avulla. Jos muutos ei onnsitunut, on palattava takaisin vanhaan ja periaatteita tarkentamalla voidaan päästä uudella yrityksellä parempiin tuloksiin.

Lisäksi tulee huomioida, että viitekehyksen eri osat ovat jatkuvassa vuorovaikutuksessa keskenään. Hajautetun kognition tarjoamia periaatteita tulee ja voidaan tarkistaa kaikkissa viitekehyksen vaiheissa. Toisaalta raja materiaalien ja ympäristöjen välillä on epäselvä, sillä esimerkiksi tekniset laitteet voivat olla kiinteä osa ympäristöä ja samalla myös materiaaleja. Lisäksi viitekehyksen käytön tuloksena saadut uudet materiaalit tulisi testata koetilanteessa ennen siirtämistään todelliseen työympäristöön.

3.2. Hutchinsin tutkimukset

Edwin Huchinsia pidetään koko hajautetun kognition teorian kehittäjänä [Rogers, 2000]. Hän monet tutkimukset ovatkin luoneet teoreettisen perustan koko ajattelulle. Hän tutkinut mm. laivan komentosillalla taphtuvaa navigointia ja nykyaikaisen matkustajalentokoneen ohjaamoa miehistöineen.

Laivan navigointia koskevissa tukimuksissaan hän on pystynyt osoittamaan kuinka eri henkilöt jakavat tietoja keskenään ja samalla järjestelmällisesti hajauttavat kognitiivisen prosessin. Lentokoneen ohjaamoa seuraavissa tutkimuksissaan Hutchins on nostanut esille ympäristön

osallistumista kognitiiviseen prosessiin. Esimerkiksi kokeneilla lentäjillä on vakiintunut tapa käyttää säätutkan testiohjelmaa muistuttamassa siitä, että koneen tankkaus on vielä kesken [Hollan, Hutchins and Kirsh, 2000]. Tällöin tutkaa käytetään normaaleissa rutiineissa täysin muuhun kuin siihen mihin se on suunniteltu. Tällaisia havaintoja on vaikea tehdä ilman etnografista tutkimusta, sillä lentäjät kokevat sen niin tavallisena toimintana, että eivät todennäköisesti muistaisi mainita asiaa pelkkiin haastatteluihin perustuvissa tutkimuksissa. Hutchins totesi myös, että siirtyminen vanhasta viisarimittarista digitaaliseen lentonopeusmittarin muutti ja vaikeutti pilottien työtä [Hollan, Hutchins and Kirsh, 2000]. Tutkimuksessaan hän sai selville, etteivät pilotit juurikaan ajatelleet nopeutta numeroina vaan viisarin suhteellinen asento antoi heille paremmin tarvitun informaation. Erityisesti tämä korostui tilanteissa, joissa nopeuden muutoksen jälkeen tarvittiin muita toimenpiteitä [Hollan, Hutchins and Kirsh, 2000]. Näiden esimerkkien kautta Hutchins pystyi päättelemään, että pilotit siirsivät kognittivista kuormaa työympäristöön aina kun se oli mahdollista. Samalla hän osoitti, että erilaiset laitteet ovat tärkeä osa kognitiivista prosessia ja niiden poistaminen ta muuttaminen voi olla kriittistä suoritettavan tehtävän kannalta.

3.3. Tietokoneavusteinen ryhmätyö ja oppimisympäristöt

Erityisesti hajautettu kognitio tarjoaa mielekkään näkökulman erilaisten ryhmätyöohjelmien suunnttelijoille. Sitä onkin esitelty tutorialeissa alan konferensseissa (ECSCW'95 ja CSCW'96) [Rogers, 1997]. Ryhmätyöohjelmistossa työskentely on selkeästi hajautettua ja muiden kanssa toimimisessa tärkeimmiksi elementeiksi nousevat tiedon ulkoistetut representaatiot, kuten erilaiset dokumentit ja keskustelukommentit. Hajautettu kognitiota voidaan siis käyttää apuna ymmärtämään ryhmätyöskentelyn monimuotoisuutta ja tunnistamaan siinä esiintyviä ongelmia. Erityisesti se avustaa ongelmissa, jotka liittyvät hajautettuun työskentelyyn ja hajatetussa ympäristössä käytettyihin materiaaleihin.

Tietoverkkojen kautta käytettävät oppimisympäristöt ovat yksi ryhmäohjelmistojen erityisalue. Näissä järjestelmissä oppijat rakentavat yhteistoiminnassa muiden oppijoiden kanssa uutta tietämystä. Tärkeäksi nousee järjestelmässä työskentelevien oppijoiden työskentelyprosessi, sillä se opettaa oppijoille enemmän kuin varsinainen tehtävän ratkaisu tai tulos [Oshima, Bereiter and Scardamalia, 1995]. Oppijoiden ulkoistamat representaatiot toimivat siis uuden tiedon rakentamisen pohjana. Samalla oppijoiden on helppo tutkia ja vertailla omaa tietämystään muiden tietämykseen, jolloin oppijan on helppo myös syventää tietojaan.

Ulkoistaminen auttaa oppijaa myös selkeyttämään ja jäsentämään omaa tietämystään, jolloin uuden tiedon omaksuminen ja vanhojen tietojen tarkentaminen helpottuu. Hajautetun kognition mukaisesti oppija aktiivisella työskentelyllä oppimisympäristössä toimii kognmitiivisen järjestelmän osana ja onnistuu saamaan paljon oppimisen kannalta merkittävää omaa ajattelua aktivoivaa järjestelmästä. Tällöin voidaan myös päätellä, että oppimista on tapahtunut [Oshima, Bereiter and Scardamalia, 1995].

4. Kritiikki

Hajautettu kognitio vaatii aikaa vievää etnografista tutkimusta, jonka jälkeen voidaan vasta saada selville jotakin [Rogers 2000]. Tällainen tutkimustapa asettaa haasteita sen soveltajalle. Lisäksi saadun raakadatan analysoinnissa tarvitaan taitoa ja kokemusta. Monet perinteisempiin suuntauksiin kyllästyneet ovatkin aluksi innostuneet hajatetusta kognitiosta, sillä siitä on helppo päästä perille Hutchinsin julkaisujen avulla. Osa heistä on joutunut kuitenkin pettymään, koska se ei tarjoa lopulta mitään suoria ratkaisuja [Rogers 2000]. Hajautetun kognition teoriasta on kuitenkin apua suuremman järjestelmän tai yhteisön toiminnan kokonaisvaltaiseen hahmottamiseen. Tutkijan on tunnettava tutkimusympäristö käytäntöineen, jotta hän voisi onnistua kokoamaan etnografisesti tietoja ja analysoimaan sitä [Rogers 2000].

Hajautettu kognition ajatus on virkistävän erilainen, mutta äärimmilleen vietynä se putoaa samoihin sudenkuoppiin kuin sen vastakohtana pidettävä yksilöä korostava ajattelu. Jos korostetaan, että kognitio on täysin hajautettua, törmätään samaan rajoitetun näkemykseen kuin, jos kognitiota pidetään täysin yksilön sisäisenä prosessina. Jos taas näkemys on näiden väliltä, voidaan helposti yhdistää molempien näkemysten vahvuuksia ja päästä parempaan tulokseen kognitiivisten järjestelmien ymmärtämisessä. Tätä voidaan perustella kysymyksellä, miten yksilön sisäinen kognitio ja hajatettu kognitio ovat vuorovaikutuksessa [Salomon 1993]. Tuskin voidaankaan nähdä kognition tapahtuvan täysin ulkoisesti vaan myös yksilön sisäinen ajattelu on erittäin tärkeää. Kuitenkin on selvää, että ihmiselle on ominaista pyrkiä keventämään kognitiivista kuormaansa aina kun ympäristö tarjoaa siihen mahdollisuuden ja täten selkeästi käyttää ympäristöä kognitiivisten prosessien tukena.

Hajautetun kognition teoriaa voidaan kritisoida myös siitä, että se keskittyy täysin pelkkän kognitiivisen toiminnan tarkasteluun. Se jättää huomioimatta ihmisen kokonaisvaltaisesti, kuten muutkin kognitiiviset teoriat. Toisaalta hajautetun kognitiossa pyritään järjestelmää havainnoimaan myös antropologiasta ja sosiaalitieteistä saaduin menetelmin. Tällöin järjestelmässä

toimivia ihmisiä tarkkaillaan kokonaisvaltaisemmin ja huomiodaan myös ympäristössä vallitseva kulttuuri.

5. Yhteenveto

Hajautettu kognitio tarjoaa uuden, tuoreen näkökulman ihmisten ja teknologian välisen vuorovaikutuksen ymmärtämiseeen. Sen perusajatukset voidaan tiivistää seuraavasti:

- kognitio voidaan hajauttaa ryhmässä yksilöiden kesken
- kognitiiviseen prosessiin voi osallistua työskentely-ympäristö erilaisine artefakteineen
- kognitioon osallistuvien elementtien toiminallinen suhde on tärkeämpää kuin niiden ajallinen tai tilallinen suhde

Hajautettu kognitio tarjoaa siis perinteisestä poikkeavan näkemyksen kognitiivisten prosessien hahmottamiseen. Erityisesti kognitiivisten prosessien nähdään tapahtuvan muuallakin kuin yksilön päässä. Lisäksi erilaiset artefaktit, kuten kirjat tai tietokoneet, voivat olla olennainen osa kognitiivista prosessia. Ne voivat muodostaa tärkeän osan kognitiivista järjestelmää, joka ei toimi ilman niitä. Artefaktit voivat olla myös reprensentaatioita, joita kognitiivisessa järjestelmässä toimivat ihmiset ovat tuottaneet ja käyttävät hyödykseen kognitiivisissa prosesseissa. Kognitiivisten prosessien havainnoinnissa onkin tärkeää osata rajata järjestelmään kuuluvat elementit mukaan ja siihen kuulumattomat ulkopuolelle. Tämä jako ei ole aikaan tai tilaan sidottu, vaan tärkeää on eri elementtien muodostama toiminnallinen kokonaisuus.

Hajautettu kognitio ei kuitenkaan suoranaisesti tarjoa mitään tiettyjä ratkaisuja. Se auttaa enemmämkin ymmärtämään eri tilanteisiin liittyviä seikkoja, jotka ovat ominaista vain niille tilanteille ja joita ei muilla tavoin ehkä pystyisi lainkaan huomioimaan. Hajatetun kognition suurin etu onkin sen tuoma selitys ihmisten ja heidän työssään käyttämien artefaktien joustavasta ja monimuotoista riippuvuudesta [Rogers, 2000]. Tällöin voidaan havannoida kuinka ihmiset käyttävät ympäristöään ja sen artefakteja osana kognitiivista prosessia.

Hollan, Hutchins ja Kirsh [2000] ehdottavat digitaalisten materiaalien suunnitteluun viitekehystä, jossa hajautetulla kognitiolla on merkittävä rooli. Viitekehys pyrkii todellisessa työskentely-ympäristössä esiintyvien erilaisten vuorovaikutustilanteiden analyysiin, jonka perusteella voidaan syvällisemmin ymmärtää tilanteen tapahtumia. Tällä tavoin saadaan selville miten erilaiset

ympäristöt ja välineet ovat osa kognitiivista prosessia ja sitten kerättyä tietoa voidaan soveltaa parempien digitaalisten materiaalien suunnitteluun.

Hajautetulle kognitiolle ei toisaalta löydy käyttöä esimerkiksi perinteisen ohjelmisto- tai käyttöliittymäsuunnitteluun. Tietotekniikan kehittyessä entistä huomaattomammaksi ja kaikkialla läsnäolevaksi se kuitenkin auttaa ymmärtämään miten ihmiset toimivat näissä ympäristöissä ja tilanteissa vuorovaikutuksessa teknologian kanssa. Tällöin pystytään selvemmin ymmärtämään vourovaikutuksen rikkautta ja monimuotoisuutta ja tämän ymmäryksen kautta suunnitelemaan entistä parempia teknologiaa hyödyntäviä sovelluksia. Hajautetun kognition käyttäminen tässä tarkoituksessa on kuitenkin hyvin työlästä ja paljon aikaa vaativaa.

Viiteluettelo

[Hollan, Hutchins and Kirsh, 2000] James Hollan, Edwin Hutchins and David Kirsh, Distributed cognition: toward a new foundation for human-computer interaction research. ACM Transactions on Computer-Human Interaction Volume 7, Issue 2 (2000), 174-196

[Oshima, Bereiter and Scardamalia, 1995] Jun Oshima, Carl Bereiter and Marlene Scardamalia, Information-Access Characteristics for High Conceptual Progress in a Computer- Networked Learning Environment Proceedings of Computer Support for Collaborative Learning '95, 1995

[Rogers, 1997] Yvonne Rogers, A brief introduction to distributed cognition. August, 1997. Available as *http://www.cogs.susx.ac.uk/users/yvonner/dcog.html*.

[Rogers, 2000] Yvonne Rogers, Recent theoretical developments in HCI: their value for informing system design. 2000. Available as *http://www.cogs.susx.ac.uk/users/yvonner/hci-theory.pdf*

[Salomon, 1993] Gavriel Salomon, No distribution without individuals' cognition: a dynamic interactional view. In: Gavriel Salomon (ed.) Distributed Cognition. Cambridge, UK: Cambridge University Press, 1993, 111-138.

Puhekäyttöliittymien vuorovaikutusmallit ja -teoriat sovellusarkkitehtuurin näkökulmasta: Speech Interaction Theory

Markku Turunen

Tässä työssä tarkastellaan puhesovelluksiin liittyviä vuorovaikutusmalleja ja –teorioita sovellusarkkitehtuurien näkökulmasta. Erityistä huomioita kiinnitetään ratkaisujen sovellettavuuteen käytännössä. Esityksen kehykseksi on valittu Speech Interaction Theory (puhepohjaisen vuorovaikutuksen teoria), joka toimii sateenvarjoteoriana käytännön puhesovelluksiin. Tässä esitellään teorian tärkeimmät piirteet ja sen käyttökelpoisuutta tutkimaan soveltamalla teoriaa Tampereen yliopistolla kehitettyyn Postimies puhejärjestelmään ja sen pohjana olevaan vuorovaikutusmalliin.

1. Johdanto

Puhekäyttöliittymien, tai yleisemmin puhesovelluksien taustalla on aina malleja ja teorioita siitä, kuinka luonnolliseen kieleen pohjautuva vuorovaikutus ihmisen ja tietokoneen vuorovaikutuksesta tulisi toteuttaa. Monitieteisellä alueella työskenneltäessä osa näistä malleista tulee esimerkiksi ihmisten välisen kommunikoinnin tutkimuksesta eri alueilta kuten filosofiasta tai kielitieteistä. Esimerkkinä mainittakoon useasti käytetty Speech Acts Theory

[Searle, 1969]. Monet näistä esityksistä keskittyvät ihmisten väliseen luonnolliseen vuorovaikutukseen eivätkä ota kantaa siihen, kuinka tuloksia voitaisiin soveltaa ihmisen ja tietokoneen välisessä (puhuttuun) kieleen pohjautuvassa vuorovaikutuksessa. Erityisesti tarvetta olisi teorioille ja malleille, jotka lähestyvät käytännön järjestelmiin liittyvää problematiikkaa ja ottavat huomioon sekä nykyisen että lähitulevaisuuden järjestelmien teknisten lähtökohtien asettamat mahdollisuudet ja rajoitukset vuorovaikutuksen luonteelle.

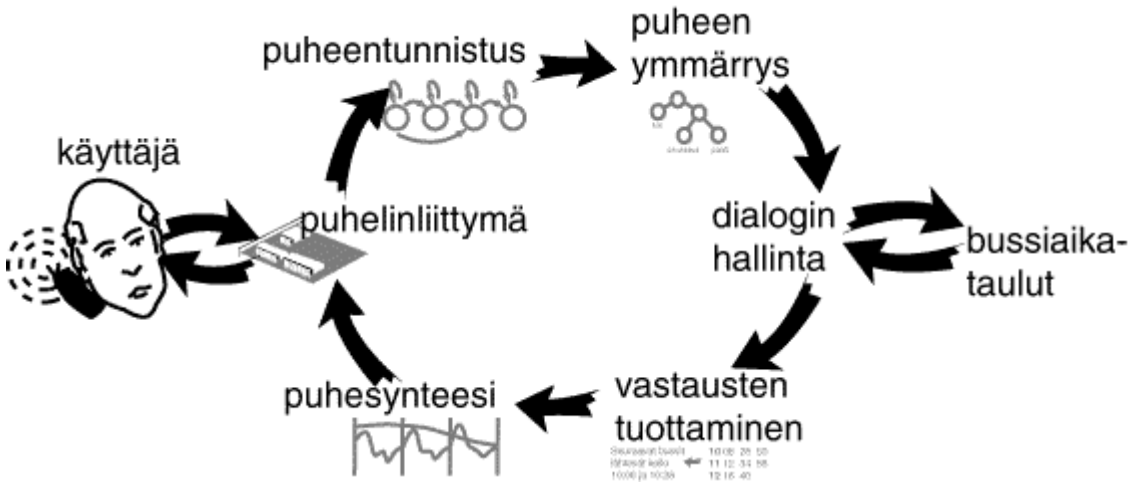
Tässä työssä keskitytään tarkastelemaan vuorovaikutuksen pohjalla olevia *vuorovaikutusmalleja* ja *–teorioita* sovelluskehityksen näkökulmasta. Päähuomio kiinnitetään toteutettujen *puhesovellusarkkitehtuurien* pohjana oleviin malleihin ja teorioihin. Tämä määritelmä kattaa sinällään erittäin laajan joukon alueita, aina yleisestä kielitieteestä dialoginmallinnukseen ja tietojenkäsittelytieteeseen. Varsinaista omaa yhtenäistä teoreettista pohjaa ei puhesovelluksille ole, lähimmäksi tulee *puhepohjaisen vuorovaikutuksen teoria* (Speech Interaction Theory, [Bernsen et. al., 1998]). Tässä tarkastellaan kyseisen teorian tärkeimpiä piirteitä ja soveltuvuutta kuvaamaan käytännön järjestelmiä ja erityisesti niiden taustalla olevia arkkitehtuureja. Kyseessä on laaja sateenvarjokäsite, joka sinällään kattaa useita muita teorioita ja malleja, erityisesti dialoginhallintaan liittyen. Tässä työssä tarkastellaan, kuinka käytännön järjestelmien kannalta tärkeimmät teoriat ja mallit suhteutuvat puhepohjaisen vuorovaikutuksen teoriaan.

Eräs teorian tärkeimpiä kriteerejä on sen sovellettavuus. Teorian tulee vastata ja tarjota tukea sen aihealueeseen kuuluvaan problematiikkaan. Puhepohjaisen vuorovaikutuksen teorian käsitellessä käytännöllisten puhejärjestelmien kehittämiseen liittyvää ongelmakenttää sen tulisi pystyä tarjoamaan (käsitteellisiä) työkaluja alueen tutkimukselle. Puhepohjaisen vuorovaikutuksen teorian hyödyllisyyttä ja käyttökelpoisuutta tarkastellaan tässä työssä suhteessa Tampereen yliopistolla kehitettyyn Postimies sovellukseen [Turunen ja Hakulinen, 2000b] ja sen takaa löytyvään vuorovaikutusmalliin [Turunen ja Hakulinen, 2001] ja arkkitehtuuriin [Turunen ja Hakulinen, 2000a].

Aloitan tarkastelun käymällä toisessa luvussa läpi tärkeimmät puhesovelluksiin liittyvät käsitteet sekä tarkastelemalla yleisesti arkkitehtuureista löytyviä piirteitä ja niiden vuorovaikutusmalleja. Kolmannessa luvussa käyn läpi puhepohjaisen vuorovaikutuksen teorian. Neljännessä luvussa tarkastelen puhepohjaisen vuorovaikutuksen teoriaa soveltamalla sitä Jaspis arkkitehtuurin, siihen liittyviin vuorovaikutusmalleihin ja sen pohjalta tehtyihin sovelluksiin. Viidennessä luvussa esitän yhteenvedon, johtopäätökset ja katsauksen tulevaan työhön.

2. Puheeseen pohjautuva vuorovaikutus

Tässä luvussa käydään läpi puheeseen pohjautuvan vuorovaikutuksen peruskäsitteet ja puhesovellusten perusrakenne. Kuvassa 1 on esitelty yleinen puhesovelluksen rakenne käyttäen esimerkkinä puhelinpohjaista aikataulujärjestelmää. Keskeiset komponentit, jotka löytyvät useimmista puhesovelluksista ovat *dialoginhallinta, tietokantaliittymä, luonnollisen kielen ymmärtäminen, puheentunnistus, liittymä syöte- ja tulostelaitteisiin, puheentuotto* ja *luonnollisen kielen tuottaminen*.



Kuva 1. Yleinen puhesovelluksen rakenne.

Dialoginhallinta ohjaa järjestelmän toimintaa yleisellä tasolla. Dialoginhallinta pohjautuu aina jonkinlaiseen *dialogimalliin*. Tästä voidaan erottaa kaksi keskeistä tekijää puhesovellusten suunnittelun ja toteutuksen kannalta: *dialogistrategia* tai *–tyyli* sekä *dialogin kontrollointimenetelmä*. Ensimmäinen ottaa kantaa siihen, millaista vuorovaikutus käyttäjän ja järjestelmän välillä on yleiseltä luonteeltaan ja jälkimmäinen kertoo millä tavalla dialoginhallinta käytännössä toteutetaan (esimerkiksi lomakkeena tai tilakoneena).

Tietokantaliittymä, tai yleisemmin yhteys varsinaiseen sovellukseen, nähdään usein selkeästi erillisenä komponenttina. Järjestelmän tulee sisältää malli siitä, kuinka tätä sovelluskohtaista tietoa hallitaan. Varsinkin ihmisen ja tietokoneen vuorovaikutuksen tutkimuksessa on käsitelty sovelluskohtaisen tiedon ja käyttöliittymän (joksi dialoginhallinta voidaan puhesovelluksen yhteydessä käsittää) suhdetta arkkitehtuurin kannalta. Esim. [Nigay & Coutaz, 1995] käsittelevät tätä suhdetta monipuolisesti. Usein alueen tarkastelu on kuitenkin vaikeasti sovellettavissa luonnollisen kielen dialogijärjestelmiin, jotka pohjautuvat hyvin erilaiseen malliin kuin esim. graafiset käyttöliittymät ja multimodaaliset järjestelmät.

Luonnollisen kielen generointia lähestytään useilla tasoilla. Yksinkertaisimmillaan luonnollisen kielen generointi on upotettu täysin dialoginhallintaan, jolloin järjestelmän vastaukset kiinnitetään jo järjestelmän suunnittelun aikana. Suorituksen aikana ei tällöin generoida mitään vaan käytetään valmiita puheenvuoroja. Toisessa ääripäässä järjestelmä muodostaa tilanteeseen soveltuvat puheenvuorot dialoginhallinnassa käsitteellisellä tasolla ja käyttää luonnollisen kielen generointia tuottaakseen näistä lopulta mahdollisimman luonnollisia ja miellyttäviä puhetulosteita. Tätä lähestymistapaa edustaa mm. Jaspis-arkkitehtuuriin sisällytetty *Presentation Management* alijärjestelmä [Hakulinen ja Turunen, 1999].

Puheen tuottamismoduuli on käytännössä melko triviaali osa puhesovelluksen toteuttajan kannalta: kysymys on useimmiten joko valmiiksi nauhoitettujen äänitteiden toistamisesta tai synteettisen puheen tuottamisesta luonnollisen kielen generoinnin pohjalta. Teknisessä mielessä järjestelmiin liittyy useita ei-triviaaleja asioita esim. ääninäytteiden yhdistämiseen sekä prosodian hallintaan.

Liittymä syöte- ja tulostelaitteisiin käsittää useimmiten puhelinkortin, mikrofonin ja kaiuttimien hallinnan. Monissa tapauksissa tämä on melko suoraviivaista, mutta esim. hajautetuissa läsnäolevan tietotekniikan sovelluksissa vakiintuneita malleja syöte- ja tulostelaitteiden hallitsemiseksi ei vielä ole.

Puheentunnistus nähdään joskus plug-and-play komponenttina, jonka tuottamia tuloksia prosessoidaan edelleen kielen ymmärrysmoduulissa. Tämä näkemys ei kuitenkaan tuota parasta mahdollista tulosta: puheentunnistusteknologia on rajoittunutta monilta osin ja tarvitaan malleja sekä suunnittelun että toteutuksen tueksi. Esimerkiksi kieliopin ja sanavaraston valinta ovat keskeisessä osassa sekä suunnittelussa että suorituksen aikana.

Luonnollisen kielen ymmärtäminen nähdään usein dialoginhallinnan ohessa keskeisenä komponenttina dialogijärjestelmissä. Kielen käsittelyssä on runsaasti perinteitä erityisesti tekstipohjaisten järjestelmien tutkimuksesta ja tällä alueella on tuotettu valtavat määrät niin teoreettista kuin käytännöllistäkin työtä. Vasta viime aikoina on kuitenkin alettu tuottamaan erityisesti puheeseen pohjautuvan kielen käsittelyyn tarkoitettuja komponentteja.

Oheinen suppea tarkastelu osoittaa millaisena puhejärjestelmät yleisesti nähdään ja esitetään. Useisiin järjestelmiin kuuluu kuitenkin paljon muitakin piirteitä, esim. käyttäjien ja aihealueen mallintamista. Lisäksi niiden vuorovaikutusmalli poikkeaa suuresti kuvassa 1 esitetystä hyvin lineaarisesta näkemyksestä. Seuraavassa luvussa esitellään yleinen puhejärjestelmiä ja

niiden elementtejä kuvaamaan pyrkivä puhepohjaisen vuorovaikutuksen teoria.

3. Puhepohjaisen vuorovaikutuksen teoriaa

Tässä luvussa tarkastellaan puhepohjaisen vuorovaikutuksen teoriaa (Speech Interaction Theory). Esittely pohjautuu pitkälti lähteeseen [Bernsen et al., 1998]. Tekijät eivät väitä teoriaansa täydelliseksi tai valmiiksi. He pikemminkin pitävät sitä ”käsitteellisenä työkalupakkina”, jota voidaan hyödyntää käytännön puhejärjestelmien suunnittelussa ja toteutuksessa. Heidän mukaansa teoria on ohjelmistosuuntautunut ja keskittyy kuvamaan puhejärjestelmiin kuuluvia elementtejä. Teoria onkin esitetty sellaisessa muodossa, jota voidaan (tietyiltä osin) pitää eräänlaisena käytännön puhejärjestelmien arkkitehtuurin sulautumana.

Ensimmäisessä aliluvussa esitellään puhepohjaisen vuorovaikutuksen teorian lähtökohdat. Toisessa aliluvussa käydään läpi teorian keskeisen osan muodostava *puhepohjaisen vuorovaikutuksen perusmalli* (the basic interaction model) ja viimeisessä aliluvussa tarkastellaan esimerkin muodossa teorian kykyä kuvata käytännön puhejärjestelmiä.

3.1 Teorian lähtökohdat

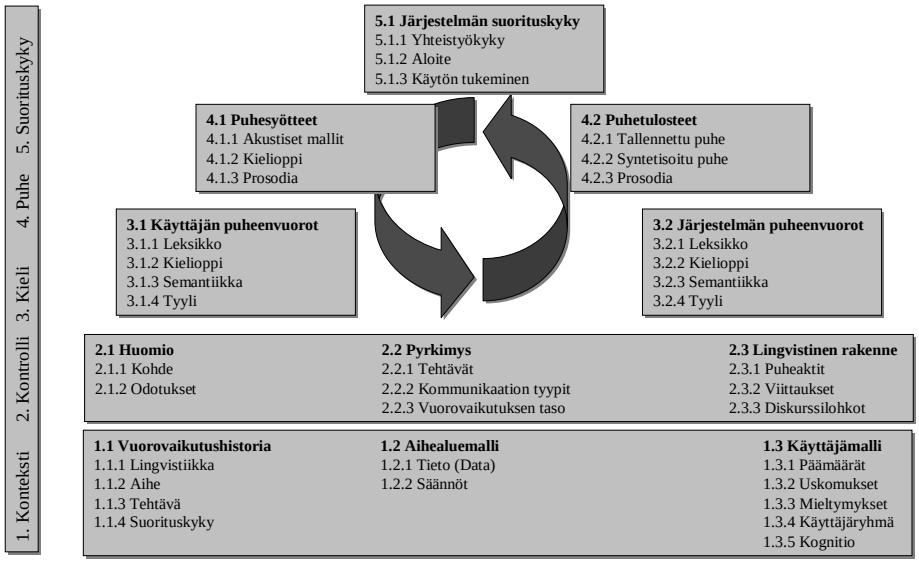
Speech Interaction Theory lähtee siitä perusajatuksesta, että ollakseen käyttökelpoisia tai ylipäätään toteutettavissa olevia puhesovelluksiin liittyvän vuorovaikutuksen tulee olla tehtäväkeskeistä ja tarkoin rajattua vastakohtana rajoittamattomalle vuorovaikutukselle joka on luonteenomaista ihmisten väliselle kommunikaatiolle. Tämä puolestaan tarkoittaa, että myös vuorovaikutuksen taustalla vaikuttavan teorian tulisi lähestyä asiaa tehtäväkeskeisyyden näkökulmasta ja ottaa huomioon järjestelmien rajoitukset. Johtuen teknisistä rajoituksista, joiden ei voida perustellusti odottaa katoavan monilta osin tulevaisuudessakaan, sekä ottaen huomioon sen tosiseikan, että ihmiset puhuvat tietokoneille eri tavalla kuin toisille ihmisille, voidaan perusteellisesti lähteä siitä olettamuksesta, että ihmisen ja tietokoneen puhepohjainen kommunikaatio on luonteeltaan erilaista kuin ihmisten välinen kommunikaatio. Tämä puolestaan tarkoittaa, että tunnetut ihmisten välistä kommunikaatioita koskevat teoriat, kuten Speech Acts Theory [Searle, 1969] eivät ole suoraan sovellettavissa ihmisen ja tietokoneen väliseen puhepohjaiseen vuorovaikutukseen.

Edellä esitetty ei tarkoita sitä, etteikö ihmisten välisen vuorovaikutuksen teoriat (ja tutkimus ylipäätäänkin) olisi soveltuvaa ja sovellettavissa myös ihmisen ja tietokoneen väliseen kielelliseen vuorovaikutukseen. Eräs vuorovaikutusteorian kriteeri onkin teorian kehittäjien mukaan sen kyky

auttaa ihmisten välisten vuorovaikutuksen tutkimustulosten siirtämisessä ihmisen ja tietokoneen väliseen vuorovaikutukseen. He esittävät myös joukon muita kriteerejä, joita tehtäväorientoituneen puhepohjaisen järjestelmän taustalla olevan vuorovaikutusteorian tulisi sisältää. Näihin kuuluvat järjestelmien ominaisuuksien kuvaaminen (sisältäen järjestelmän rajoitukset), erilaisten järjestelmien ominaisuuksien vertaileminen, sovellusten suunnittelun ja toteutuksen tukeminen sekä niiden tarpeiden kuvaaminen, joihin teknisen kehityksen tulisi vastata. Tekijät kutsuvat näillä ominaisuuksilla varustettua teoriaa *käytännölliseksi yksityisestä yleiseen eteneväksi puhepohjaisen vuorovaikutuksen teoriaksi* (a practical bottom-up theory of interactive speech systems).

3.2 Puhepohjaisen vuorovaikutuksen perusmalli

Bernsen et al. kutsuvat kuvassa 2 esitettyä vuorovaikutusmallia *puhepohjaisen vuorovaikutuksen perusmalliksi* (the basic speech interaction model). Mallissa on kaksi organisointitapaa: ensinnäkin malli on jaettu viiteen eri tasoon, jotka ovat *konteksti*, *kontrolli*, *kieli*, *puhe* ja *suorituskyky*. Toista organisointitapaa visualisoi kuvassa esitetyt nuolet, jotka havainnollistavat puhesovellusten loogista rakennetta.



Kuva 2. Puhepohjaisen vuorovaikutuksen perusmalli [Bernsen *et al.*, 1998]

Konteksti (1)

Kontekstilla on puhejärjestelmissä erityisen suuri merkitys, sillä puhejärjestelmissä kieli- ja kulttuurialuiden sekä yksilöiden väliset erot tulevat selvemmin esille kuin graafisissa käyttöliittymissä. Kontekstia tarvitaan useillla eri tasoilla jotta järjestelmä voidaan sopeuttaa kulloiseenkin käyttäjään. Bersen et al. jaottelevat kontekstin vuorovaikutuahistoriaan, aihealuemalliin ja käyttäjämalliin.

Vuorovaikutushistoria (1.1)

Vuorovaikutushistoria käsittää vuorovaikutuksen aikana tietyin kriteerein taltioitua informaatiota. *Lingvistinen historia* (1.1.1) käsittää kielen pintatason, semantiikan sekä mahdollisesti myös muita lingvisisiä piirteitä kuten puheakteja. Historiatietoutta kielen lingvistisestä rakenteesta voidaan hyödyntää useilla alueilla esim. viiitteiden tulkinnassa, puhesyötteiden ja tulosteiden mukauttamisessa käyttäjään jne.

Aihehistoria (1.1.2) käsittää sen järjestyksen, missä (ali)tehtävät ovat tulleet aktiivisiksi, ts. huomion kohteeksi. Aihehistoriaa hyödynnetään lähinnä dialoginhallinnassa, esim. meta-kommunikaatiossa virheidenhallinnassa.

Tehtävään liittyvä historia (1.1.3) käsittää vuorovaikutuksen aikana vaihdetun aihealueeseen liittyvän informaation. Tehtävähistoria käsittää kaiken sen tiedon, joka liittyy järjestelmän aihealuueeseen sinällään, esimerkiksi sääpalvelusovelluksessa lämpötilan kysyttyinä päivinä tai sähköpostisovelluksissa sähköpostien sisällön ja postilaatikon tilan kulloisenakin hetkenä. Rajanveto alkaa tosin jo jälkimmäisissä tapauksissa olla vaikeaa, sillä miten meidän tulisi tulkita esim. muunnellun sähköpostilaatikon tila – onko se pikemminkin dialogiin kuin aihelueeseen liittyvää tietoa?

Suorituskykyhistoria (1.1.4) kertoo, kuinka hyvin ihmisen ja tietokoneen välinen vuorovaikutus on edennyt. Suorituskykyhistorian avulla voidaan vuorovaikutus sopeuttaa käyttäjään esim. reagoimalla ongelmiin vaihtamalla dialoginhallintastrategiaa (esim. käyttäjän aloitteeseen perustuvasta järjestelmän aloitteesen perustuvaan). Ongelmana tässä kuitenkin usein on se, kuinka vuorovaikutuksen onnistuneisuutta voidaan ja tulisi mitata.

Aihealuemalli (1.2)

Aihealuemalli käsittelee sovelluksen aihealueeseen liittyviä konsepteja eli sitä tietoutta aihealueesta, mitä vuorovaikutusjärjestelmässä käsitellään. Aihealueeseen liittyvää informaatiota käsitellään *tiedon* (1.2.1) ja *sääntöjen* (1.2.2) avulla. Säännöt määrittävät kuinka tietoa voidaan hyödyntää sovelluksen alueella. Yksinkertaisena esimerkkinä toimii relaatiot lähtö- ja tulopaikkojen välillä aikatauluja käsittelevässä sovelluksessa.

Aihealuemallin voidaan katsoa käsittävän myös maailmantietous, erimerkiksi kuinka ajanilmaukset tulee tulkita.

Käyttäjämalli (1.3)

Tieto käyttäjästä mahdollistaa järjestelmän sopeutumisen kulloiseenkin käyttäjään. Käyttäjän *päämäärät* (1.3.1) käsittää ne tehtävät, joita järjestelmä sisältää. Käytännössä päämääriä on yleensä melko rajoitettu joukko, esim. aikataulutiedon saaminen tai postin lukeminen. Toisaalta joissakin keskustyyppisissä palveluissa päämääriä voi olla runsaastikin, mutta tällöin kyseessä on joukko erilaisia sovelluksia, kuten integroitu sähköposti- ja kalenterisovellus.

Käyttäjän *uskomukset* (1.3.2) käsittelevät uskomuksia järjestelmästä, aihealueesta ja laajasti ottaen maailman tilasta ylipäätäänkin suhteessa järjestelmään. Käyttäjän *mieltymykset* (1.3.3) käsittelevät erilaisia toiveita, joita voidaan yleistää joko yksittäisiin käyttäjiin tai käyttäjäryhmiin. Mieltymykset voivat vaihdella aihealueeseen sidotuista hyvinkin yleisiin.

Käyttäjäryhmät (1.3.4) ilmaisevat vuorovaikutuksen kannalta relevantteja käyttäjäluokitteluja. Perinteinen noviisi-ekspertti jako käsittää yleensä kaksi ulottuvuutta, aihealuetietämyksen sekä sovellustietämyksen. Puhejärjestelmien kannalta muita erityisen tärkeitä käyttäjiä ryhmittäviä tekijöitä ovat kielialueet ja joissakin tapauksissa myös murrealueet, opitut vuorovaikutusmenetelmät jne.

Viimeisenä käyttäjän mallintamiseen kuuluvana asiana Bernsen et al. mainitsevat *kognitiivisen prosesoinnin* (1.3.5). Tarkemmin he eivät tähän aiheeseen ota kantaa.

Kontrolli (2)

Vuorovaikutuksen kontrolloinnilla Bernsen et al. tarkoittavat kaikkea vuorovaikutukseen liittyvää käsitteellistä prosessointia ts. sitä aluetta, mistä puhutaan useimmiten dialoginhallintana. He jaottelevat vuorovaikutuksen hallinnan Groszin ja Sidnerin [1986] mukaisesti kolmeen osa-alueeseen: *huomion kohteeseen*, *käyttäjän pyrkimyksiin* ja *lingvistiseen rakenteeseen*.

Huomion kohde (2.1)

Huomion kohde käsittää tiettynä vuorovaikutustapahtuman ajankohtana käsiteltävänä olevat asiat. *Kohdejoukko* (2.1.1) käsittää kaikki ne vuorovaikutuksen alitehtävät, jotka ovat mahdollisia vuorovaikutuksen alitehtäviä järjestelmässä. Usein kohdejoukko käsittää kaikki tehtävät, ts. järjestelmässä on milloin tahansa mahdollisuus siirtyä mihin tahansa toiseen tehtävään (meta-kommunikaatio muodostaa tästä poikkeuksen). Toisissa järjestelmissä vuorovaikutus on rajoitettu kulloinkin tehtäväjoukon aitoihin

osajoukkoihin. Erillisten kohdejoukkojen käyttäminen jakaa paitsi tehtävät eri osa-alueisiin niin se mahdollistaa myös tehtäväkohtaisten tunnistimien ja niihin liittyvien kielimallien käyttämisen.

Mikäli vuorovaikutuksessa käytetään hyväksi kohdejoukkoja, täytyy pitää yllä listaa *odotuksista* (2.1.2), jotka määrittelevät kuinka kohdejoukko tulisi valita kussakin vuorovaikutustilanteessa. Mikäli kohdejoukko valitaan väärin joudutaan helposti tilanteeseen, jossa ei synny yhteisymmärrystä käyttäjän ja järjestelmän välille, sillä käytännössähän esim. järjestelmän ymmärtämä sanavarasto ei tällöin sisällä käyttäjän puheenvuorossaan ilmaisevia sanoja.

Erilaisia osatehtävien joukkoja voidaan hyödyntää parhaiten käyttämällä järjestelmän aloitteeseen pohjautuvaa dialoginhallintastrategiaa, jolloin järjestelmä valitsee kulloiseenkin vuorovaikutustilanteen odotusten pohjalta soveltuvimman tehtävien osajoukon, joka voi olla joko staattisesti tai dynaamisesti mallinnettu. Joustavaan vuoronvaihtoon perustuvassa vuorovaikutuksen hallinnassa joudutaan usein käyttämään kohdejoukkona koko tehtävien joukkoa, sillä sopivien kohdejoukkojen muodostaminen dynaamisesti on käytännössä erittäin vaikeaa lukuun ottamatta erilaisia erikoistilanteita kuten meta-dialogeja.

Intentionaalinen rakenne (2.2)

Intentionaalisella rakenteella Bernsen et al. tarkoittavat (Groszia ja Sidneriä mukaillen) kaikkia niitä asioita, jotka liittyvät vuorovaikutuksessa ilmeneviin tehtäviin ja vuorovaikutuksen muotoon. He erottavat tästä kolme osatekijää: tehtävät, vuorovaikutuksen tyypit ja vuorovaikutuksen tasot.

Käyttäjän pyrkimyksenä (2.2.1) vuorovaikutustilanteessa voi olla yleisesti ottaen melkein mitä tahansa eikä suoraa yhteyttä pyrkimyksien ja tehtävien välille voida johtaa. Puhuttaessa vuorovaikutuksesta tietokoneen ja ihmisen välillä voidaan perustellusti kuitenkin olettaa tehtäväkeskeinen lähestymistapa, jolloin molemmilla kommunikaation osapuolilla on yhteinen päämäärä jolloin pyrkimykset ja tehtävät eivät ole erillisiä.

Pyrkimykset ja tehtävät muodostavat rakenteita, joissa erilaiset pyrkimykset ja tehtävät edeltävät toisiaan, ovat toisilleen alisteisia jne. Usein nämä näkyvät vuorovaikutusjärjestelmissä alidialogeina, jolloin esim. hotellivarauksen tekemiseksi tarvitaan käynnistää erilaisia alidialogeja, jotka käsittelevät tulo- ja lähtöpäivämääriä jne. Tehtävät voivat olla joko hyvin muotoiltuja tai huonosti muotoiltuja, joskaan jälkimmäisiä ei käytännön järjestelmissä esiinny (poikkeuksena voidaan pitää erilaisia tekstipohjaisia kysely- ja keskustelujärjestelmiä). Hyvin muotoillussa tehtävässä on rakenne, joka määrittelee vuorovaikutuksessa tarvittavan informaation vaihdon sekä monissa tapauksissa myös vuoronvaihdon järjestyksen. Käytännössä tämä

ilmenee esim. lomakepohjaisissa dialogeissa kenttien välisinä riippuvuussuhteina.

Bernsen et al. jakavat *tehtävien tyypit* (2.2.2) kolmeen luokkaan. Perustyyppinä on vuorovaikutusjärjestelmän *kohdealueeseen liittyvä dialogi*. Omana alueenaan on vuorovaikutukseen sinällään liittyvä *meta-dialogi*, joka tyypillisimmin käsittelee virheiden korjausta eri muodoissa. Meta-dialogi voi käsitellä tehtäväalueeseen liittyvää tietoa, ja monesti sen on onnistuakseen näin tehtäväkin. Bernsen et al. näkevät näiden kahden luokan lisäksi *muiden vuorovaikutusmuotojen* luokan, joka käsittää yksinkertaisesti kaiken sen mikä ei sisälly edellä mainittuihin luokkiin. Esimerkkinä he käyttävät aloitus- ja lopetuspuheenvuoroja. Ei kuitenkaan ole lainkaan selvää, kuinka esim. lopetuspuheenvuorojen kaltaisiin osadialogit tulisi suhtautua: tietyssä mielessä ne voidaan nähdä metadialogina tai kokonaan omana luokkanaan.

Vuorovaikutuksen taso (2.2.3) on eräs puhepohjaisen vuorovaikutuksen teorian selkeimmin kuvattuja osia. Vuorovaikutuksen taso määrittää tietyllä hetkellä käytettävissä olevat rajoitukset käyttäjän kommunikaatiolle ts. määrittelee sen tason millä käyttäjä pystyy kommunikoimaan järjestelmän kanssa puhepohjaisesti. Viisi kuvattua tasoa ovat *tavaus*, *binääriset vastaukset*, *monivalintavastaukset*, *kohdistetut vastaukset* ja *kohdistamattomat vastaukset*. Bernsen et al. huomauttavat, että suoraa vastaavuutta järjestelmän ja käyttäjän puheenvuoroille ei voida aina luoda, sillä esimerkiksi binääriseksi kysymykseksi tarkoitettu järjestelmän puheenvuoro voidaan tulkita avoimeksi kysymykseksi, johon saadaan kohdistamaton vastaus. Tämä implikoi myös järjestelmän puheenvuorojen huolellisen suunnittelun merkitystä, sillä esim. binääriset kyllä / ei kysymykset voivat huonosti muotoiltuna olla ongelmallisia.

Lingvistinen rakenne (2.3)

Vuorovaikutuksen lingvistinen rakenne käsittää puheaktit, viittaukset sekä diskurssisegmentit. *Puheaktit* (2.3.1) käsittelevät Searlea [1969] mukaillen keskusteluteorian perusyksiköitä, joskin usein järjestelmien yhteydessä käytetään muitakin sukulaiskäsitteitä kuten dialogiakteja (Bunt, 1994) tai dialogisiirtoja [Carletta et al., 1997]. Käytännössä näitä termejä käytetään hyvin epämääräisesti määrittelemään erilaisia puhevuorojen tyyppejä.

Bernsen et al. esittävät lyhyesti puheaktien tyypit [Searle, 1979] ja näin tuovat ne osaksi puhepohjaisen vuorovaikutuksen teoriaa, mutta huomauttavat samalla, että toisaalta nämä viisi eri kategoriaa tuskin ovat riittäviä puhepohjaisten järjestelmien tarpeisiin ja toisaalta erilaisista yrityksistä huolimatta ei ole onnistuttu luomaan kattavaa tarkempaa puheaktien valikoimaa. Puheaktit ovat muutenkin ongelmallisia hyödyntää, esimerkiksi epäsuorat puheaktit ovat viittausten tavoin puhejärjestelmille erittäin vaikeasti

tunnistettavia. Ongelmista huolimatta on olemassa erilaisia puheakteihin, ja dialogiakteihin sekä dialogisiirtoihin pohjautuvia järjestelmiä.

Viittaukset (2.3.2) muodostavat perinteisen lingvistisen ongelma, joka puhejärjestelmissä ei näyttele niin keskeistä osaa kuin esim. tekstipohjaisissa dialogijärjestelmissä lähinnä kommunikaatiossa käytetyn kielen rajoitusten vuoksi. Toisaalta ongelmaa pyritään lähestymään myös praktiselta näkökannalta eli käyttäjän puheenvuoroja ei pyritäkään tulkitsemaan kokonaisuudessaan - tämä luonnollisesti saattaa aiheuttaa virheitä itsessään. Yleisesti ottaen viittausten selvittäminen on kuitenkin vaikea ongelma, jonka merkitys kasvaa mitä lähemmäksi luonnollista puhuttua kieltä edetään.

Diskurssilohkot (2.3.3) ovat lingvistisiä vastineita tehtäville (2.2.2). Käyttäjän pyrkimysten (2.2.1) kohteina voi olla ainoastaan sellaiset tehtävät, joilla on suora suhde diskurssilohkoihin. Nämä määrittelevät diskurssilohkojen merkityksen. Jotta diskurssilohkon pyrkimys olisi ymmärrettävissä, sen täytyy tulla vastaanotetuksi puheenvuoron vastaanottajan taholta (ts. osapuolten välillä täytyy olla yhteisymmärrys).

Diskurssilohkojen osilla voidaan nähdä olevan erilaisia rooleja, kuten esimerkiksi pyyntö-informointi-varmistus, esitys-hyväksyntä ja aloitus-vastaus rakenteet osoittavat. Esimerkki kehittyneemmistä diskurssilohkoja hyödyntävistä teorioista on Rhetorical Structure Theory [Mann ja Thompson, 1987], jossa diskurssilohkot ryhmitellään hierarkisesti.

Diskurssilohkoja on hyödynnetty eri tavoin vuorovaikutusjärjestelmissä. Esimerkiksi ruotsalainen LINLIN dialogijärjestelmä [Jönsson, 1993] hyödyntävää diskurssilohkorakenteita dialoginhallinnassa ja Fisher et al. [1994] käsittelevät Rhetorical Structure Theoryn hyödyntämistä dialogijärjestelmissä. Laajaa levinneisyyttä diskurssilohkoihin perustuvat puhejärjestelmät eivät kuitenkaan ole saaneet.

Kieli (3)

Kielen käsittely lähtee, tai sen tulisi lähteä puhepohjaisissa vuorovaikutusjärjestelmissä siitä seikasta, että kirjoitetun ja puhutun kielen välillä on suuria eroja eikä puhejärjestelmissä voida käyttää suoraan samoja menetelmiä kuin mitä kirjoitetun kielen kanssa on totuttu käyttämään. Käytännössä puhutun kielen käsittelyssä onkin mittavia ongelmia, jotka tekstipohjaisissa järjestelmissä on ratkaistu ainakin tyydyttävästi.

Bernsen et al. määrittelevät usein epäselvästi käytetyt termit *vuoro, lausahdus* (utterance) ja lausahdusyksikkö (utterance unit) seuraavasti: vuoro käsittää yhden puhujan puhuman asian siihen saakka, kunnes vuoro siirtyy kokonaisuudessaan seuraavalle puhujalle. Vuoro voi käsittää yhden tai useamman lausahdusyksikön. Lausahdus voi olla vuoro tai lausahdusyksikkö.

Kielen tasolla vuorovaikutusta käsitellään *leksikon, kieliopin, semantiikan* ja *tyylin kannalta.* Kaikkia tarkastellaan sekä käyttäjän että järjestelmän näkökulmasta.

Käyttäjän puheenvuorot (3.1)

Leksikolla (3.1.1) tarkoitetaan sanalistaa (sanastoa) varutettuna syntaktisilla ja semanttisilla piirteillä. Sanasto määrittelee yhdessä kieliopin kanssa vuorovaikutuksessa käytettävän kielen. Tässä prosessi on *konvergenssillä* suuri merkitys. Tällä tarkoitetaan iteratiivista lähestymistapaa, jossa sanastoa laajennetaan kunnes se kattaa sovelluksen tarpeet. Tässä ei luonnollisesti tarkoita kaiken kattavaa sanastoa vaan pragmaattista sanavarastoa, jonka avulla vuorovaikutus on mahdollista tai pikemminkin sujuvaa.

Kielioppi (3.1.2) määrittelee kuinka sanoja voidaan yhdistää suuremmiksi kokonaisuuksiksi. Käyttäjän puheenvuorojen kohdalla kielioppi voidaan ensinnäkin määritellä puheentunnistuksen kannalta, ts. määritellä millaisia kieltä puheentunnistin tulkitsee. Toisaalta puheentunnistimelta saatujen tuloksien käsittelyyn voidaan käyttää lingvistisiä työkaluja. Puhutun kielen kohdalla nk. robusti parsinta on erityisen tärkeää

Semantiikalla (3.1.3) käsitetään sanojen ja sitä suurempien kokonaisuuksien abstraktia merkitystä. Semantiikan käsittelyyn käytetään erilaisia menetelmiä aina yksinkertaisista kuvauksista kehittyneisiin formalismeihin. Monissa järjestelmissä yksinkertaiset kuvaukset syntaktisen rakenteen (sanojen tai fraasien) ja semanttisten abstrahointien (merkitysten ja näitä seuraavien toimenpiteiden) välillä toimivat yllättävän hyvin.

Puheen *tyyli* (3.1.4) käsittelee erilaisia asioita kuten sanojen valintaa, lausepituutta, viittauksien ja analogioiden käyttöä jne. Korkealla tasolla tyylistä puhutaan määrittelevillä käsitteillä kuten lyhyttä, vuolasta, töksähtelevää jne. Järjestelmän kannalta käyttäjien puheen tyyli tulisi ohjata (järjestelmän omien puheenvuorojen ja esimerkkien kautta) mahdollisimman yksinkertaiseksi.

Järjestelmän puheenvuorot (3.2)

Puhepohjaisen vuorovaikutuksen teoria ei käsittele järjestelmän puheenvuorojen *leksikoa* (3.2.1), *kielioppia* (3.2.2), *semantiikkaa* (3.2.3) ja *tyyliä* (3.2.4) samalla tavoin erillisinä kuin vastaavia kohtia käyttäjän osalta. Pikemminkin *puhetyyli* käsitetään tässä yleiseksi ominaisuudeksi, joka muodostuu näiden neljän osatekijän yhteisvaikutuksesta ja jolla on suuri vaikutus myös siihen, kuinka käyttäjä puhuu järjestelmälle. Tällöin tulosteiden muodostamisessa käytetyn leksikon ei tulisi sisältää syötteiden tulkintaan käytetyn leksikon kannalta ongelmallisia sanoja ja tulosteiden muodostamiseen käytetyn kieliopin ei tulisi antaa esimerkkejä syötekieliopin kannalta vaikeista rakenteista.

Puhetyylillä voidaan ohjata vuorovaikutuksen suuntaa myös sellaisten asioiden kuin avoimien tai suljettujen kysymysten osalta. Avoimet kysymykset on perinteisesti koettu hyvin ongelmallisiksi. Myös varmistuksiin ja palautteisiin liittyy kielen kannalta paljon mielenkiintoisia kysymyksiä.

Puhe (4)

Puhepohjaisen vuorovaikutuksen teorian neljäs osa käsittelee akustisen signaalin ja tekstiesityksen välistä suhdetta ja tähän liittyviä näkökohtia. Tähän suhteeseen liittyy monia hyvin ongelmallisia kohtia, kuten puheen osalta prosodiset elementit ja tekstin osalta sellaiset ilmaisut, joita on vaikea ilmaista yksikäsitteisesti puheen avulla.

Puhesyötteet (4.1)

Käyttäjän puhesyötteiden osalta keskeisiä asioita ovat tunnistamisessa käytetyt *akustiset yksiköt* (4.1.1) ja niiden ominaisuudet (tyyppi, määrä jne). *Kieliopin* (4.1.2) suhteen määritellään ne menetelmät ja ominaisuudet, joiden avulla akustiset yksiköt koodaan suuremmiksi kokonaisuuksiksi. Esimerkiksi word-spotting tekniikkaa käytettäessä kaikkia akustisia yksiköitä ei pyritäkään tunnistaman vaan näiden osajoukot pyritään tunnistamaan sanoiksi tai fraaseiksi.

Prosodian (4.1.3) hyödyntäminen puheentunnistuksessa paljon huomiota herättänyt alue. Ongelmana on kuitenkin se, että vaikka joitakin puheen prosodisia piirteitä tunnetaan (esim. intonaation laskun ja nousun merkitys kysymyslauseissa), ei läheskään kaikkia piirteitä tunneta eikä pystytä automaattisesti tunnistamaan.

Edellä esitettyjen kolmen kohdan lisäksi puhepohjaisen vuorovaikutuksen teoria käsittelee erilaisia puheentunnistuksen liittyviä tekijöitä kuten puheen tyyliä (sana kerrallaan, jatkuvaa), puhujariippuvuutta jne. Nämä eivät kuitenkaan jakaudu selkeästi esitettyihin luokkiin eivätkä esitetyt luokat toisaalta tue näitä piirteitä kunnolla. Tämä on merkittävä puute, sillä kyseessä on yksi tärkeimpien puhesovelluksia karakterisoivien ominaisuuksien joukko.

Puhetulosteet (4.2)

Puhetulosteita puhepohjaisen vuorovaikutuksen teorian käsittelee hyvin suppeasti tehden lähinnä kahtiajaon *koodatun puheen* (4.2.1) eli luonnollisen nauhoitetun puheen ja *parametrisen puheen* (4.2.2) eli puhesynteesin välillä. Puheeseen liittyviä ominaisuuksia, kuten esim. prosodisia ominaisuuksia ei käsitellä sen tarkemmin. Myöskään monissa järjestelmissä (esim. [Turunen ja Hakulinen 2000b]) tärkeää monikielisyyttä ei tässä käsitellä.

Järjestelmän suorituskyky (5 ja 5.1)

Suorituskyky on puhepohjaisen vuorovaikutuksen teorian ehkä mielenkiintoisin, mutta samalla myös ongelmallisin osa. Käsitteen avulla pyritään kuvaamaan järjestelmien käyttäytymistä ja erityisesti sen kokonaisvaltaista suorituskykyä käyttäjän näkökulmasta. Suorituskykyä ei tässä käsitellä tehokkuuteen viittaavana käsitteenä. Suorituskyvyn osa-alueita ovat *yhteistyö*, *aloite* ja *käyttäjän opastaminen*.

Yhteistyökyky (5.1.1)

Järjestelmän kannalta käyttäjän odotetaan olevan yhteistyökykyinen eli halukas yhteistyöhön. Tämä tehtäväkeskeisen vuorovaikutusjärjestelmän perusolettamus on sinällään oikeutettu. Toisaalta voitaisiin kuitenkin miettiä myös käyttäjien osalta, tulisiko käyttäjän halukkuutta ja mahdollisuuksia jaotella hienovaraisemmin esim. osatehtävien osalta.

Käyttäjän kannalta järjestelmän yhteistyökyky on samoin erittäin tärkeää, minkä vuoksi järjestelmien suunnittelussa tulee kiinnittää erityistä huomiota niihin konkreettisiin seikkoihin, joiden avulla järjestelmän yhteistyökykyä saadaan parannettua.

Aloite (5.1.2)

Vuorovaikutuksessa käsiteltävän aiheen valintaa kontrolloivaa keskustelun osapuolta kutsutaan aloitteentekijäksi. Aloitteen ja puheaktien (2.3.1) välillä vaikuttaa vuorovaikutussuhde, joka ei kuitenkaan ole millään tapaa triviaali. Aihetta ovat tarkemmin käsitelleet mm. Whittaker ja Stenton [1988] erilaisten yleistyksien muodossa.

Aloitteellisuus voidaan jakaa kolmeen pääluokkaan: *järjestelmän aloitteellisuuteen*, *käyttäjän aloitteellisuuteen* ja *joustavaan aloitteellisuuteen*. Joustava aloitteellisuus voidaan edelleen jakaa *rajoitettuun joustavaan aloitteellisuuteen* sekä *vapaaseen joustavaan aloitteellisuuteen*. Käytännössä täysin vapaa joustava aloitteellisuus, jolloin kumpi tahansa osapuoli voi olla aloitteellinen missä tilanteessa tahansa on erittäin harvinaista. Yksinkertaisissa järjestelmissä tämä voi kuitenkin olla mahdollinen ja toimiva ratkaisu.

Käyttäjän opastaminen (5.1.3)

Puhepohjaisten järjestelmien kannalta etenkin noviisikäyttäjien kohdalla on erityisen tärkeää pyrkiä opastamaan ja ohjaamaan käyttäjää omaksumaan sellainen vuorovaikutusmalli joka mahdollistaa tehtävän onnistuneen suorittamisen. Käyttäjälle tulisikin tarjota malli siitä, kuinka järjestelmän kanssa toimitaan.

Puhepohjaisen vuorovaikutuksen teoria mainitsee kolme lähdettä joiden avulla voidaan helpottaa käyttäjän työtä vuorovaikutusmallin muodostamisessa: eksplisiittiset ohjeet järjestelmän opastusosuudessa, implisiittiset järjestelmän opasteet sekä eksplisiittiset suunnitteluohjeet.

Eniten hyödyntämättömiä mahdollisuuksia sisältävät järjestelmän implisiittisesti ilmaisemat opasteet. Perusolettamuksina on, että käyttäjät adaptoituvat järjestelmään sen perusteella kuinka järjestelmä käyttäytyy. Toinen vuorovaikutuksessa huomioitava perusasia on se, että ihmiset käyttäytyvät toisin koneille kuin toisille puhuessaan. Käyttäjien vuorovaikutusmallin muodostamisessa auttaa myös käytävä metakommunikaatio, joskin esimerkiksi varmistuksien kanssa ajaudutaan helposti ongelmiin.

3.3 Järjestelmien ominaisuuksien kuvaaminen

Eräs puhepohjaisten vuorovaikutusjärjestelmien teorian hyödyntämismahdollisuus on sen käyttäminen puhejärjestelmien kuvaamiseen siten, että järjestelmän keskeiset piirteet olisi ilmaistavissa jäsennellysti ja korkealla tasolla. Käytännössä järjestelmien ominaisuudet voidaan kuvata kuvassa 2 esitettyjen ja luvussa 3.2 kuvattujen ominaisuuksien suhteen yhdellä kaksipuolisella A4-sivulla.

Sovellan seuraavilla kahdella sivulla teoriaan pohjautuvaa lomaketta Tampereen yliopistolla kehitettyyn sähköpostien lukujärjestelmään, Postimieheen [Turunen ja Hakulinen 2000b].

4. Postimiehen vuorovaikutusmalli

Postimies (Mailman) on puhelinpohjainen monikielinen sähköpostien lukujärjestelmän prototyyppi.

Vuorovaikutushistoria (1.1)

Lingvistinen historia (1.1.1): semanttinen merkitys
Aihehistoria (1.1.2): tehtävät
Tehtävään liittyvä historia (1.1.3): postilaatikkoon ja posteihin liittyvä tieto
Suorituskykyhistoria (1.1.4): tunnistamattomien vuorojen peräkkäinen määrä

Aihealuemalli (1.2)

tieto (1.2.1): käyttäjän postilaatikko
säännöt (1.2.2): -

Käyttäjämalli (1.3)

Käyttäjän päämäärät (1.3.1): tiettyjen postien lukeminen ja / tai yleiskuva
Käyttäjän uskomukset (1.3.2): -
Käyttäjän mieltymykset (1.3.3): synteesin ääni, nopeus, postien suodatus

Käyttäjäryhmät (1.3.4): -
kognitiivinen prosesointi (1.3.5): listojen jakaminen lyhyisiin kokonaisuuksiin

Huomion kohde (2.1)

Kohdejoukko (2.1.1): kaikki mahdolliset tehtävät ja metakommunikaatio
Odotukset (2.1.2): -

Intentionaalinen rakenne (2.2)

Käyttäjän pyrkimykset eli *tehtävät* (2.2.1): postilaatikon selaaminen, yksittäisten postin lukeminen, luetuksi merkitseminen ja poistaminen
Kommunikaation tyypit (2.2.2): käyttäjän aloite, meta-kommunikaatiossa järjestelmä aloitteellinen
Vuorovaikutuksen taso (2.2.3): komentoja hyvin rajoitetulta alueelta (jotka ovat luonteeltaan valintoja listalta), binäärisiä kysymyksiä

Lingvistinen rakenne (2.3)

Puheaktit (2.3.1): -
Viittaukset (2.3.2): -
Diskurssilohkot (2.3.3): -

Käyttäjän puheenvuorot (3.1)

Leksikko (3.1.1): 20 sanaa
Kielioppi (3.1.2): kiinnitetty
Semantiikka (3.1.3): yksinkertaiset kuvaukset sanoista käsitteiksi
tyyli (3.1.4): lyhyttä, komentokieltä muistuttavaa

Järjestelmän puheenvuorot (3.2)

leksikko (3.2.1): agenttien mallien pohjalta muodostamat lauseet
kielioppi (3.2.2): kiinnitetty
semantiikka (3.2.3): yksinkertaiset aihealueeseen sidottu kuvaus
tyyli (3.2.4): vaihteleva eri agenteilla

Puhesyötteet (4.1)

Akustiset yksiköt: sanapohjaiset HMM-mallit, 20 kappaletta
Kielioppi (4.1.2): sanaverkko
Prosodia (4.1.3): -

Puhetulosteet (4.2)

Koodattu puhe (4.2.1): -
Parametrinen puhe (4.2.2): Mikropuhe 4.2 SAPI

Järjestelmän suorituskyky (5.1)

Yhteistyökyky (5.1.1): -

Aloite (5.1.2): yleisesti ottaen käyttäjällä, järjestelmä tarjoaa apua tietyissä tilanteissa ja ottaa aloitteen joissakin meta-kommunikaatiota käsittelevissä tilanteissa

Käyttäjän opastaminen (5.1.3): eksplisiittiset ja implisiittiset ohjeet (mm. avainsanojen ilmaisemista järjestelmän puheenvuoroissa)

5. Kritiikki ja johtopäätökset

Puhepohjaisen vuorovaikutuksen teoriaa ei voi lähteä arvioimaan minään varsinaisena teoriana. Kuten sen kehittäjät osuvasti ilmaisevat, kyseessä on pikemminkin eräänlainen käsitteellinen työkalupakki, kokoelma käsitteitä ja kriteerejä joiden avulla käytännön puhesovellusten suunnittelua, toteuttamista ja karakterisointia voidaan lähestyä. Tässä suhteessa teoria on onnistunut ja ennen kaikkea myös hyödyllinen, sillä se onnistuu monilta osin tuomaan teoreettista näkökulmaa hyvin käytännönläheiseen aiheeseen ja toisaalta pystyy myös vastaaman käytännön haasteisiin joissakin kohdissa, kuten esimerkiksi järjestelmien kuvaamisessa.

Toinen näkökulma teoriaan on sen alustavuus ja keskeneräisyys. Teoria ei pyri olemaan millään tapaa kattava ja se pikemminkin tarjoaa osittain järjestetyn ja aloitetun työmaan kuin lopulliseen ja muotoonsa hioutuneen kokonaisuuden. Tämänkin huomioiden osa sen piirteistä, kuten esim. järjestelmän puhetulosteisiin (4.2) liittyvät kohdat ovat liian ylimalkaisia ja osat, esimerkiksi käyttäjien kognitiiviseen kuormitukseen (1.3.4) liittyvät kohdat suorastaan päälle liimattuja. Suurimmaksi puutteeksi näen kuitenkin eri kategorioiden varsinaisen sisällön eli niitä kuvaavien attribuuttien heikon esittämisen ja useissa tapauksissa puuttumisen. Näitä on ainoastaan hyvin harvoissa kohdissa - kohta 5.1.2 eli aloitteellisuus toimii hyvänä esimerkkinä siitä, millaista sisällön kuvausta teorian osa-alueet kaipaisivat.

Teorian tapa jaotella vuorovaikutusjärjestelmien elementit viiteen osaan (tai kerrokseen) on myös hieman ongelmallinen. Viides osa eli järjestelmän suorituskyky on muista, konkreettisiksi järjestelmän komponenteiksi kuvautuvista kokonaisuuksista selkeästi erillinen ja itse asiassa näiden rinnakkainen dimensio, kokonaisuutta kuvaava tekijä.

Elementtien eri kerroksien lisäksi tekijät ovat sisällyttäneet teoriaan myös toisen tason eli loogisen arkkitehtuurin, jota kuvataan kuvassa 2 nuolella ja laatikoilla. Tämän merkitys jäi käytännössä vähäiseksi. Järjestelmien loogista rakennetta kuvaavan osuuden informaatioarvo on paitsi vähäinen myös joissakin tapauksissa harhaanjohtava. Useiden järjestelmien, esimerkkeinä

Galaxy-II [Seneff et al., 1998] ja Jaspis [Turunen ja Hakulinen 2000a] looginen rakenne on vaikeata kuvata tai sovittaa tämän esityksen piiriin.

Teorian vahvimpia alueita ovat useiden unohdettujen, tai hyvin epämääräisesti esitettyjen osa-alueiden kuten vuorovaikutushistorian ja käyttäjämallin tuominen mukaan järjestelmien eksplisiittiseksi osaksi. Jos sitä verrataan luvussa 1 esitetyn kaltaisiin luonnehdintoihin ollaan saavutettu jo paljon. Toisaalta se ei kata kaikkia luvussa 1 esittämiäni piirteitä, esim. erilaiset dialogin toteutusmallit ovat käytännössä hyvin keskeisiä järjestelmiä karakterisoivia piirteitä.

Toinen vahva alue on teorian kyky kuvata järjestelmiä yleisellä tasolla. Esimerkiksi luvussa 3.3. Postimies-järjestelmään kuvauksen toivoisi yleistyvän, sillä vastaavien tietojen saaminen järjestelmistä erillisistä tutkimusartikkeleista, www-sivuilta yms. on monesti mahdotonta.

Teorian yleistettävyys koskemaan multimodaalisia tai perinteisistä puhesovelluksista poikkeavia järjestelmiä on avoin kysymys. Esimerkiksi läsnäolevan tietotekniikan sovellukset, mobiilit puhesovellukset ja vahvasti puheeseen pohjautuvat moltimodaaliset järjestelmät saattavat taipua huonosti esitettyyn teoriaan ja etenkin sen loogisen mallin kuvaukseen. Näissäkin järjestelmissä voidaan monin paikoin käyttää puhepohjaisen vuorovaikutuksen teoriaa soveltaen sitä puheeseen, mutta toisaalta silloin menetetään ehkä teorian vahvin voimavara eli sen kyky kuvata kokonaisia järjestelmiä.

Kokonaisuutena puhepohjaisen vuorovaikutuksen teoria tarjoaa käyttökelpoisen pohjan käytännön puhejärjestelmien suunnitteluun, toteuttamiseen ja kuvaamiseen. Käyttökelpoisuuden nostamiseksi siihen tulisi sisällyttää enemmän järjestelmien eri elementtejä kuvaavaa materiaalia. Toisaalta järjestelmän laajentaminen muidenkin kuin perinteisten puhesovellusten alueelle olisi tärkeä jatkokehityksen aihe.

Viiteluettelo

[Bernsen *et al.*, 1998] Niels Ole Bernsen, Hans Dybkjær, Laila Dybkjær. Designing Interactive Speech Systems. Springer-Verlag, London, 1999.

[Carletta et al., 1997] Carletta, J., Dahlbäck, N., Reithinger, N., Walker, M. (Eds.). Standards for Dialogue Coding in Natural Language Processing. Dagstuhl Seminar Report 167. Schloss Dagstuhl: IBFI, 1997.

[Fischet et al., 1994] Fischet M., Maier, E. Stein,A. Generating cooperative system responses in information retrieval dialogue. In proceedings of the International Workshop on Natural Language Generation, 1994: 207-216.

[Gorsz ja Sidner, 1986] Grosz, B., Sidner C. Attention, intentions, and the structure of discourse. Computational Linguistics, 12(3), 1986: 175-204.

[Hakulinen ja Turunen, 1999] Jaakko Hakulinen ja Markku Turunen. Presentation Agents for Speech User Interfaces . In CHI 2000 Extended Abstracts, Den Haag, The Netherlands, April 1-6, 2000, 115-116.

[Jönsson, 1993] Jönssön A. Dialogue Management for Natural Language Interfaces. An Empirical Approach. Pd.D. thesis, Linköping studies in Science and Technology no. 312, 1993.

[Nigay & Coutaz, 1995] Laurence Nigay ja Joelle Coutaz, A Generic Platform for Addressing the Multimodal Challenge. In Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems: 98-105, 1995.

[Searle, 1969] John Searle. SpeechActs: An Essay in the Philosophy of Language. Cambridge University Press, 1969.

[Searle, 1979] John Searle. Expression and Meaning. Studies in the theory of Speech Acts. New York: Cambridge University Press, 1979.

[Seneff et al, 1998] S. Seneff, E. Hurley, R. Lau, C. Pao P. Schmid, V. Zue, Galaxy-II: a Reference Architecture for Conversational System Development, in *Proceedings of ICSLP98*, 1998.

[Turunen & Hakulinen, 2000] Markku Turunen ja Jaakko Hakulinen, Jaspis - A Framework for Multilingual Adaptive Speech Applications. In Proceedings of 6th International Conference of Spoken Language Processing (ICSLP 2000), 2000.

[Turunen ja Hakulinen, 2001] Markku Turunen ja Jaakko Hakulinen. Jaspis - A Framework for Multilingual Adaptive Speech Applications. In Proceedings of 6th International Conference of Spoken Language Processing (ICSLP 2000), 2000..

[Turunen ja Hakulinen, 2001] Markku Turunen ja Jaakko Hakulinen. Mailman - a Multilingual Speech-only E-mail Client Based on an Adaptive Speech Application Framework. In Proceedings of Workshop on Multi-Lingual Speech Communication (MSC 2000), 2000: 7-12.

[Turunen ja Hakulinen, 2001] Markku Turunen ja Jaakko Hakulinen, Agent-based Adaptive Interaction and Dialogue Management Architecture for Speech Applications. In Text, Speech and Dialogue. Proceedings of the Fourth International Conference TSD 2001 (to appear).

[Whittaker ja Stenton, 1988] Whittaker, S. ja Stenton, P. Cues and control in expert-client dialogues. In Proceedings of the 26th Annual Meeting of the ACL. Morristown, NJ: Association for Computational Linguistics (ACL), 1988: 123-130.

Puheaktien teoria ja niiden anti puhejärjestelmille

Jaakko Hakulinen

Puheaktien teoria on alun perin kielifilosofian puolella syntynyt näkemys, jonka mukaan kieli ei ole vain väline tosiasioiden ilmaisemiseen, vaan puhuminen on tekoja. Teoriaa on kehitetty filosofian lisäksi lingvistiikassa ja nykyään sitä on käytetty monella tavoin dialogijärjestelmien kehityksessä tietojen–käsittelytieteenkin puolella. Puheaktien teoria tarjoaa dialogijärjestelmien kehittäjien käyttöön joukon abstraktioita vuorovaikutuksen mallinnukseen sekä myös valmiita sääntöjä, joiden avulla esimerkiksi luonnollisen kielen generointia voidaan kehittää.

1. Johdatus puheaktien teoriaan

Puheaktit, englanniksi speech acts, on kielifilosofiasta lähtöisin oleva teoria kielen tarkoituksesta. Se on reaktio perinteiselle filosofiselle näkemykselle, jonka mukaan kielen tehtävänä on esittää tosiasioita koskevia väitelauseita. Puheaktien teoria nostaa kielen toiminnan muodoksi ja esittää, että on joukko asioita, joita voimme kielen avulla tehdä.

Puheaktien teorian kehitti Oxfordin kielifilosofeihin kuulunut J. L. Austin. Hän ei koskaan itse kirjoittanut aiheesta varsinaista teosta, mutta luennoi

aiheesta paljonkin. Hänen William James luennot koottiin jälkikäteen kirjaksi "How to Do Things With Words" [Austin 1961]. Tämä teos esittää puheaktien teorian ja selvityksen siitä, miten Austin teoriaansa päätyi.

Lähtökohtana Austinilla oli ongelma, joka nousi esille sellaisissa ilmauksissa, kuin "Lyön markan vetoa, että huomenna sataa", "Julistan sodan Zanzibarille" tai "Tuomitsen teidän kymmeneksi vuodeksi pakkotyöhön". Nämä lauseet eivät vaikuta kuvaavan maailmaa, tai siis ne eivät ole väitelauseita. Sen sijaan nämä lauseet saavat, oikeissa olosuhteissa ja oikeiden henkilöiden sanomina, aikaan muutoksen maailmassa. Nämä lauseet siis *tekevät* jotain. Näin ollen kielellä näyttää olevan muitakin tehtäviä, kuin väitelauseiden muodostaminen.

Austin jakoi näin lauseet väitelauseisiin (constantives) ja näihin toimintalauseisiin (performatives). Eräs merkittävä ero näiden kahden ryhmän välillä on niiden suhtautuminen totuusarvoihin. Väitelauseilla voidaan periaatteellisesti aina määrittää totuusarvo sen mukaan, pitääkö väitelause paikkansa ts. vastaako väite maailmassa vallitsevaa tosiasiallista tilaa. Toisaalta toimintalauseille voidaan kullekin määrittää onnistuneisuusehdot (felicity conditions), ts. ehdot, joiden voimassa ollessa se muutos, jonka lause voi saada aikaan, todella tapahtuu. Esimerkiksi vankilatuomion julistamislauseen onnistuneisuusehdot ovat varsin tiukat ja vaativat, että julistajalla on vaadittu asema, l. hän on tuomari, puhuteltava on tuomittavana oikeudessa ja puheenvuoro tapahtuu oikeudenkäynnissä oikealla hetkellä jne. Nämä ehdot luonnollisesti riippuvat kulloisestakin toimesta, jota puheenvuorolla ollaan tekemässä, ja myös kulttuurista, sillä kielen avulla aikaansaatavat muutokset vaativat kielen ulkopuolisia kulttuurisia instituutioita.

Kuitenkin, jatkettuaan tutkimustaan eteenpäin Austin havaitsi, että tämä kahtiajako ei ole toimiva. Voimme löytää lauseita, jotka ovat yhtäaikaa totuuslauseita ja lauseita, jotka tekevät jotain. Samoin onnistuneisuusehdot rinnastuvat vastaamaan totuusarvoja. Näin Austin jatkoi teoriaa eteenpäin ajatukseen, että kaikki lauseet, joita käytämme, tosiasiassa tekevät jotain. Näin maailman tilaa muuttavat lauseet ja maailmasta totuuksia raportoivat lauseet ovatkin vain erikoistapauksia, ne tekevät vain jotain niistä monista asioista, joita kielellä voidaan tehdä.

Puheaktien teoria siis muotoutuu nyt muotoon, jossa kaikki lauseet ovat jonkinasteisia tekoja ja näitä tekoja on erilaisia. Toisaalta Austin huomasi, että jokainen puheenvuoro jonka käytämme, sisältää kolme tekoa. Siis sen lisäksi, että tekoja on erityyppisiä ja siis vaihtoehtoisia, niitä on eri tasoisia. Näitä eritasoisia tekoja Austin havaitsi kolme ja nimesi ne *locutionary*, *illocutionary* ja *perlocutionary act* nimillä. Näistä ensimmäinen tarkoittaa varsinaista sanomista,

toinen sitä tekoa, minkä käyttäjä saa kielen avulla tehtyä ja viimeinen sitä vaikutusta, jonka puhe saa kuulijassaan aikaan.

Tarkemmin nämä voidaan määritellä esimerkiksi, Allania [1998] seuraten, seuraavasti:

> Suorittaessaan LOCUTIONARY ACT tason toimen, puhuja käyttää tunnistettavaa ilmausta *e*, joka sisältää lauseen tai sen osan kielestä L, puhuttuna prosodialla π.
>
> Puheenvuorossa U, puhuja suorittaa ILLOCUTIONARY ACT toimen, käyttämällä jotain loctionary act tointa, viitatakseen siten, että U:lla on ilmauksen ILLOCUTIONAARINEN VOIMA, kuten väitelause, vahvistus, kielto, ennustus, lupaus, pyyntö ja niin edelleen.
>
> Puhujan PERLOCUTIONARY ACT toimi on saada aikaan jokin vaikutus kuuntelijaan puheenvuoron avulla siten, että kuulija tunnistaa puheenvuorosta (jotkin) locutionary ja illocutionary voimat.

Yksittäinen, onnistunut, puheakti siis suorittaa yleensä kaikki nämä kolme tointa (toki yritys puhua voi epäonnistua toteuttamaan mitään näistä). Erityisesti huomio puheaktien teoriassa kuitenkin kohdistuu illocutionary act toimiin sillä juuri näiden toimien luokitteluunhan Austin pyrki.

Vieläkään ei kuitenkaan ole päästy yksimielisyyteen näiden illocutionary tason toimien joukosta, ja tämä asia onkin yksi suurimmista ongelmista puheaktien teorian kohdalla [Traum 1999]. Monia ehdotuksia ja ryhmittelyjä on kuitenkin tehty. Yksi vaikutusvaltaisimmista ja perustelluimmista on Searlen [1979] tekemä ryhmittely.

2. Searlen puheaktien ryhmittely

Searlen työ puheaktien teorian parissa on erittäin merkityksellistä. Hän systematisoi Austinin teoriaa ja kehitti sitä eteenpäin monella rintamalla. Yksi osa tätä systematisointia on juuri puheakti tyyppien ryhmittely. Austin ajatteli, että puheaktit voitaisiin ryhmitellä tarkastelemalla puheella tapahtuvia toimia kuvaavia verbejä. Searle puolestaan pyrki löytämään korkeammalla abstraktiotasolla olevien käsitteiden avulla sopivan ryhmittelyn. Hän listaa suuren joukon erilaisia, puheakteja erottelevia periaatteita ja luo näiden perusteella oman puheaktien joukkonsa. Lopullisessa ryhmittelyssä Searlella on jäljellä viisi asiaa, joita kielellä voi tehdä. Nämä viisi puheaktien tyyppiä ovat:

representatives, sitovat puhujan lausutun proposition totuusarvoon, ts. puhuja sitoutuu väitteeseen, että hän tietää kuinka kyseinen asia on. Kyseessä ovat siis väitelauseet.

directives, puhuja yrittää saada kuulijan tekemään jotain. Kyseessä ovat mm. pyynnöt ja kysymykset.

commissives, sitovat puhujan johonkin tulevaan tapahtumaan. Kyseessä ovat lupaukset, uhkaukset, tarjoukset jne.

expressives, kuvaavat puhujan psykologista tilaa. Kyseessä ovat esim. kiittäminen, anteeksipyyntö, onnittelu jne.

declarations, saavat aikaan välittömiä muutoksia ihmisten määräämien instituutioiden tilassa. Esimerkiksi sodan julistus, työstä erottaminen ja avioliittoon vihkiminen kuuluvat tähän kategoriaan.

Vaikka Searlen ryhmittely onkin vahvemmalla pohjalla, kuin Austinin, suhtautuu mm. Levinson [1983] siihen erittäin kriittisesti. Monia muitakin ryhmittelyjä on esitetty. Usein, etenkin käytännön sovelluksissa kuten dialogijärjestelmissä [esim. Hagen and Popowich 2001] nämä ryhmittelyt eivät perustu mihinkään teoreettiseen pohdintaan, vaan puheakteja on määritelty käyttöön enemmänkin tarpeen tullen.

Searlen puheaktien määritelmää on käyttänyt mm. Hutchins (1989) pyrkiessään luokittelemaan erilaisia käyttöliittymämetaforia. Hän nostaa esille, toki jo Austinillakin esiintyneen, Searlen viimeisen puheaktityypin, deklaraatiot. Todellisessa maailmassa mahdollisten deklaraatioiden joukko on melko pieni, mutta tietokonetta käytettäessä voidaan käyttäjän toimet nähdä deklaraatioina, jotka muuttavat koneen sisäistä maailmaa. Hutchins nostaa rinnakkain asetteluksi normaalin, komentopohjaisen (komentorivi) käyttöliittymän ja ajatuksen tasolla muodostetun deklaraatioliittymän. Normaalissa komentorivikäyttöliittymässä käyttäjän voidaan nähdä antavan komennon koneen toteutettavaksi, puheakti on siis direktiivi. Esimerkiksi käyttäjä tuhoaa tiedoston sanomalla *delete foo*. Tämä voidaan ymmärtää olevan muotoa *käsken sinut tuhoamaan foon*. Deklaratiivisen metaforan kautta taas komennon muoto olisi *määrään tiedoston foo tuhotuksi*. Näin käyttäjää muuttuu eräänlaiseksi kaikkivoivaksi taikuriksi, jonka puhe muuttaa maailman tilaa eikä hänen ja maailman välissä ole enää minkäänlaista välittäjää. Deklaratiivisen metaforan ongelmana Hutchins näkee virhetilanteet, eli tilanteet joissa käyttäjä antaa komennon, jota ei voi suorittaa. Tällöin, hänen mielestään ei voitaisi

antaa virheilmoitusta, koska ei ole ketään, joka voisi sanoa, että näin ei voi tehdä. Hänen mielestään asian vain ei pitäisi tapahtua, mikä taas ei tietenkään ole käytettävyyden kannalta toivottavaa. Ainoana ratkaisuna Hutchins näkee, että käyttöliittymä olisi sellainen, että sillä ei voisi antaa komentoja, joita ei voida toteuttaa. Kuitenkin voitaisiin nähdä, että virheilmoitukset voisivat toimia deklaratiivisessakin käyttöliittymässä. Voisimme ottaa esimerkiksi vaikka tilanteen, jossa kokouksen jäsen sanoo "nimitän sinut puheenjohtajaksi", ja vieressä ollut vastaa siihen, että "meillä on ollut tapana, että pääjohtaja nimittää puheenjohtajan". Näin deklaraation antajalle ilmoitetaan, että vaadittavat ehdot deklaraation toteutumiselle eivät ole voimassa. Samalla periaatteella virheilmoitukset voitaisiin jonkinlaisen virheagentin avulla tuoda melkein suoraan komentokäyttöliittymästä deklaratiiviseen käyttöliittymään.

3. Onnistuneisuusehdot

Jo aiemmin mainitut onnistuneisuusehdot (felicity conditions) ovat puheaktien tyyppien lisäksi tärkeä termi puhe aktien teoriassa. Onnistuneisuusehdon idea on siis, että kun kielelliset ilmaisut ovat myös tekoja, niin teot voivat olla onnistuneita tai epäonnistuneita. Samalla tavalla kuin ajajan yritys ajaa autoa epäonnistuu jos hänellä ei ole ajoneuvoa käytettävissään, voi lupaus epäonnistua, jos puhujalla ei ole mahdollisuuksia sen toteuttamiseen. Esimerkiksi kuun lupaaminen taivaalta voidaan nähdä epäonnistuneena puheaktina, jos sitä tarkastellaan ilman metaforista tulkintaa. Searle [1979] käyttää onnistuneisuusehtojen jakoa neljään ryhmään mm. vertaillessaan eri puheaktien tyyppejä näiden ehtojen avulla. Nämä neljä onnistuneisuusehtojen tyyppiä ovat *preparatory condition*, *sincerity condition*, *propositional content condition* ja *essential condition*. Ensimmäinen näistä on ehto, joka vaatii, että kyseisen puheaktin propositionaalinen sisältö on todellakin mahdollista suorittaa. Toinen ehto vaatii, että puheaktin tekijä todellakin sitoutuu aktiin, ts. on tosissaan lausahduksensa kanssa. Kolmas ehto vaatii, että lausahdus todella tarkoittaa oikeaa asiaa ja viimeinen ehto, että koko lausahdus muodostaa hyväksyttävän, halutun kaltaisen puheaktin. Onnistuneisuusehdot muodostavat siis käsitteen, jota voidaan soveltaa puheaktien teorian erilaisissa sovelluksissa ja analyyseissä lähtein jo puheaktien tyyppien etsimisestä.

Kun puheakteja käytetään perustana dialogijärjestelmässä, joudutaan pian määrittelemään puheakteille täsmällisempää teoriaa, joka kertoo, milloin mikäkin puheakti toteutuu. Tällöin terveeseen järkeen pohjautuvat onnistuneisuusehdot osoittautuvat pian liian yleisiksi. Esimerkiksi väitelauseen toteutumista tutkiessa voidaan esittää esimerkiksi seuraavanlaisia kysymyksiä: jos puhuja esittää väitelauseen, mutta kuulija ymmärtää sen kysymyksenä, mikä puheakti on toteutunut. Tai jos puhuja esittää väitelauseen, mutta kuulija

on sitä mieltä, että puhuja valehtelee, mikä puheakti toteutuu. Näitä pohdiskeluja formalisoitaessa joudutaan ottamaan kantaa mm. väitteiden suhteesta maailman tilaan ja törmätään mm. yhteisen uskomuksen (mutual belief) käsitteeseen. Yhteinen uskomuskin on sangen monimutkainen käsite, koska sitä käsitellessä joudutaan tutkimaan mm. ajatusta, mitä keskustelukumppanit uskovat toistensa uskovan. Tällaista teoriaa logiikan formalismin keinoin on puheakteille kehittänyt mm. Cohen [1990] tekoälytutkimuksen piirissä. Näin monitasoiset abstraktiot eivät toki ole välttämättömiä, jotta puheakteja voitaisiin dialogijärjestelmissä hyödyntää, vaan ne tulevat eteen vasta, kun järjestelmän perusta toteutetaan mallintaen keskustelukumppaneita ja näiden sisäisiä tiloja sekä niiden muutoksia dialogiaktien tuloksina.

4. Epäsuorat puheaktit

Yksi mielenkiintoisimmista puheaktien teorian yhteydessä usein käsitellyistä kielen ominaisuuksista ovat epäsuorat puheaktit. Termillä tarkoitetaan lausahdusta, jonka varsinainen merkitys on jotain muuta kuin sen pintamuoto tai lauseen varsinainen merkitys on pintamuodon lisäksi jotain muutakin. Puheaktien teorian termeillä siis lausahduksen illocuationary act on jotain muuta (tai jotain muutakin) kuin sen locutionary act antaisi ymmärtää. Esimerkiksi ihminen sanoo jotain, joka on muodoltaan vaikkapa kysymys, mutta todellisuudessa kyseessä onkin komento, siis vaikkapa "Voisitko poistua huoneesta?" on pintamuodoltaan kysymys vaikka lauseen todellinen merkitys onkin käsky poistua huoneesta. Tietenkin lauseella on myös kysymyksen illocutionaarinen voima, koska ainakin periaatteessa kuulija voi vastata kysymykseen "En voisi.". Näin siis lause toteuttaa kaksi puheaktia.

Epäsuoria puheakteja käytetään normaalissa puheessa sangen usein, joten kyseessä ei ole asia, jonka voisi luonnollista kieltä tutkiva teoria ohittaa erikoistapauksena. Erityisesti epäsuoraa puheaktia käytetään monissa kielissä pyyntöjen kohdalla. Kun ihminen pyytää toista tekemään jotain, pyynnön muoto on usein "Voisitko…", eli pintamuodoltaan kyseessä on kysymys. Yleinen näkemys on, että epäsuoran puheaktin käyttö liittyy kohteliaisuussääntöihin [Levinson 1983]. Suorat pyynnöt ja käskyt kuulostavat karkeilta ja niitä loivennetaan asettamalla ne epäsuoraan muotoon. Tätä muotoilua voidaan tehdä tiettyyn rajaan asti yhä vahvempana, ts. pyynnöistä tehdä kohteliaampia ja kohteliaampia. Esimerkiksi "Avaa ovi", "Avaisitko oven?", "Voisitko avata oven?", "Olisiko mahdollista, että avaisit oven?", "Olisitko niin ystävällinen, että avaisit oven?".

Epäsuorat puheaktit muodostava haasteen puheaktien teorialle. Jotenkin pitäisi pystyä, selittämään, miten ihmiset pystyvät ymmärtämään ja

käyttämään epäsuoria puheakteja tai yleensä miten epäsuorien puheaktein ymmärtäminen ja käyttäminen, mm. dialogijärjestelmissä on ylipäätään mahdollista sekä tietenkin myös miten epäsuoria puheakteja tulisi käyttää. Tähän ongelmaan on esitetty monia ehdotuksia ja mm. tällä alalla Searle on puheaktien teoriaa käsitellyt. Hänen [1979] hypoteesinsa on, että, puhujan on mahdollista kommunikoida puheensa avulla pintamuotoa enemmän, koska kommunikaatio perustuu keskustelukumppaneiden jakamaan taustatietouteen, loogiseen päättelykykyyn, keskustelun yleisten periaatteiden [ks. Grice 1989] noudattamiseen sekä puheaktien ominaisuuksiin.

Kun siis joku sanoo toiselle "Voisitko avata oven?", voidaan ajatella kuulijan päättelevän, että puhuja puhuu normaalien keskusteluun sääntöjen ja tapojen mukaan ja että puhuja luultavasti tietää, että kuulija on kykeneväinen avaamaan oven. Näin kuulija voi päätellä, että puhuja todennäköisesti tarkoittaa lausahduksellaan jotain muutakin. Jotta puhuja voisi järjellisesti käskeä kuulijaa avaamaan oven, pitää kuulijan olla kykeneväinen tekemään tämän. Koska oven avaaminen olisi luonnollinen teko kyseisessä tilanteessa, voi kuulija päätellä, että puhuja haluaa hänen avaavan oven.

Edellisessä päättelyssä vedottiin puheaktien ominaisuuksiin kohdassa "Jotta puhuja voisi järjellisesti käskeä kuulijaa avaamaan oven, pitää kuulijan olla kykeneväinen tekemään tämän.". Tässä siis vedottiin käsky tyyppisen puheaktin *preparatory condition* onnistuneisuusehtoon. Muuten päättely on erikoistapaus nk. *conversational implicature* [Grice 1989, myös Levinson 1983] teoriasta, joka selittää, miten voimme ymmärtää erilaisia epäsuoria ilmaisuja kuten ironiaa yms.

Tietenkään oikeasti emme tee tällaisia päätelmiä päässämme tietoisella tasolla mutta juuri dialogijärjestelmiä rakennettaessa tämän tasoiset yksityiskohtaiset päättelyketjut on tarpeen ymmärtää tarkkaan. Päättelystä kannattaa myös panna merkille, että siihen sisältyy todennäköisyys, eli kuulija ei voi olla täysin varman, että puhujan lausahdus oli pyyntö avata ovi.

5. Pragmatiikka

Kielitiede ja kielen käsittely jaetaan yleisesti kolmeen osaan, syntaksiin, semantiikkaan ja pragmatiikkaan. Syntaksi siis tutkii kielen rakennetta riippumatta sen merkityksestä tai sisällöstä. Semantiikka taas liittää kielen todellisuuteen ja, erään määritelmän mukaan, tutkii lauseiden totuusarvoja, siis vastaavatko lauseet todellisuutta tai niille annettuja määritelmiä. Pragmatiikka taas on seuraava taso ja tutkii kieltä käyttötilanteessa. Puheaktit sijoittuvat tässä ryhmittelyssä osaksi pragmatiikkaa. Esimerkiksi puheen puheaktin tyyppi (illocationary act) voi muodoltaan (locationary act) täsmälleen samalla lauseella vaihtua käyttötilanteen vaihtuessa.

Tämän kolmijaon lisäksi voidaan toki nähdä vielä syntaksia edeltävinä tutkimuksen alueina fonetiikka ja morfologia. Lisäksi useimmat pragmatiikan määritelmät jättävät ulkopuolelleen psykolingvistiikan, sosiolingvistiikan, neurolingvistiikan ja vastaavat tutkimusalat.

Pragmatiikalla ei siis ole yhtä, hyväksyttyä määritelmää. Seuraavassa käyn kuitenkin läpi joitakin Levinsonin [1983] esittelemiä Pragmatiikan määritelmiä, jotta pragmatiikan suhde syntaksiin ja semantiikkaan ja tätä kautta sen tarpeellisuus juuri dialogijärjestelmien tasolla tulee esille.

Pragmatiikkaa käytetään siis joissain yhteyksissä hyvin laajan määrittelyn omaavana terminä, jolloin se sisältää kaiken, mikä syntaksin ja semantiikan ulkopuolelle jää, sisältäen siis mm. psykologian alaan sijoittuvat neurologiset tutkimukset. Toisaalla pragmatiikalle on kuitenkin kehittynyt varsin täsmällinen ja kapeampi määritelmä. Levinson [1983] nostaa tämän merkityksen lähtökohdaksi loogikko Carnapin antaman määritelmä, jonka kuuluu kirjoittajan suomentamana seuraavasti:

> Jos tutkimuksessa tehdään eksplisiittinen viittaus puhujaan, tai laajemmin sanottuna kielen käyttäjään, silloin tutkimus kuuluu pragmatiikan alueeseen. … Jos abstrahoimme pois kielen käyttäjän ja analysoimme vain ilmaisuja ja niiden merkityksiä, olemme semantiikassa. Ja lopulta jos abstrahoimme pois merkitykset ja analysoimme vain ilmaisujen välisiä suhteita, olemme syntaksin tutkimuksessa.

Tämä määritelmä on käytännössä kuitenkin liian tiukka, koska se ei ota tutkimukseen mukaan muuta kuin käyttäjät. Levinson tekeekin määritelmään lisäyksen, jolla määritelmässä ollut kielen käyttäjä laajennetaankin termiksi konteksti. Pragmatiikka on siis kielen tutkimusta, jossa viitataan siihen kontekstiin, jossa kieltä käytetään. Tämä konteksti sisältää tiedon keskusteluun osallistujista, ajallisesta ja tilallisesta tilanteesta, jossa kieltä käytetään ja tiedoista, uskomuksista ja aikomuksista, joita kielen käyttäjillä on.

Voimme nyt nähdä, että pystyäksemme rakentamaan toimivan puhepohjaisen järjestelmän, meidän on joissain muodossa otettava mukaan myös pragmatiikan alaan kuuluvia asioita. Järjestelmä, jossa on vain perinteiset kielenkäsittelyn komponentit syntaksin ja semantiikan aloilta, siis jäsennys ja lauseiden totuusarvojen evaluointi, on hyvin konemainen. Se ei periaatteessa osaa viitata itseensä eikä käyttäjään, ja se vastaa jokaiseen kysymykseen samalla vastauksella tilanteesta riippumatta. Toki pragmatiikan osa-alueisiin kuuluvia asioita voidaan järjestelmiin sijoittaa ilman niiden eksplisiittistä identifiointiakin, mutta pragmatiikan teorioiden avulla järjestelmiä voidaan toki kehittää jo valmiiksi kehitettyjen teorioiden avulla. Käyttäjän mallin mukaan ottaminen on yksi viimeaikoina puhejärjestelmissä lisääntyvää kiinnostusta nauttinut alue. Esimerkiksi nk. grounding (ks. esim. Heeman et. al.

1998), eli huolen pitäminen siitä, että keskustelukumppani on ymmärtänyt hänelle esitetyt asiat oikein, eli molemmat puhuvat vielä samasta asiasta, kuuluu selkeästi pragmatiikan alaan.

Puheaktien teoria sijoittuu siis pragmatiikan osaksi. Se tutkii kieltä nimenomaan kontekstissa. Puheaktit antavat teoreettisen työkalun päästä käsiksi kielen kontekstin myötä vaihtuviin merkityksiin. Nimenomaan eksplisiittinen erottelu puheen tason toimintaan (locutionary) ja toisaalta tarkoitetun merkityksen tasoiseen kielelliseen toimintaan (illocutionary) antavat selkeän tavan hahmottaa mm. epäsuoria puheakteja. Puhejärjestelmä voisi, vahvaa käyttäjän mallinnusta pohjanaan käyttäen käyttää hyödyksi myös perlocutionary tason toimintoja, siis rakentaa tulosteensa sen pohjalta, miten järjestelmä haluaisi saada käyttäjän tilan muuttumaan. On kuitenkin ymmärrettävää, että ajatus tällaisesta järjestelmästä ei ole eettisesti kovin mielenkiintoisen kuuloinen. Moni tuskin haluaisi käyttää järjestelmää, jonka toimintaperiaate on käyttäjän manipuloiminen.

6. Puheaktit ja dialogi-, puhe- ym. järjestelmät

Puheaktien teoria on toiminut pohjana monenlaisissa tietokonejärjestelmissä ulottuen ainakin puhejärjestelmistä ohjelmistoagenttien välisiin kommunikointikieliin. Puheaktien teoria on ollut tietojenkäsittelyn puolella voimakkaasti käytössä juuri agenttien välistä kommunikaatiota mallinnettaessa. Tässä työssä puheaktien teoriasta on löydetty teoreettinen perusta vuorovaikutuksen mallinnukselle. Agenttien välinen kommunikointi nähdään siis koostuvan puhekateista, tai yleistettynä kommunikaatioakteista. Esimerkkinä toimii KQML (Knowledge query and manipulation language) [Labrou and Finnin 1996]. Tämä kieli pohjaa teoreettisella tasolla joihinkin puheaktien teorian käsitteisiin. Se sisältää joukon perus toimintoja, joita kutsutaan nimellä *performatives.* Nämä siis vastaavat juuri illocutionary tason erilaisia toimintoja.

Varsinaisissa dialogijärjestelmissä puheakteja voidaan käyttää dialogin mallinnukseen. Puheaktit voivat toimia käsitteellisen mallinnuksen välineenä, kun käyttäjän vuoropuhelua tulkitaan siihen muotoon, jonka perusteella järjestelmä päättää, mitä se seuraavaksi tekee. Kun käyttäjän syötteet tulkitaan puheakteiksi, voidaan dialogia ajatella mallinnettavan puheaktien sarjana. Järjestelmä siis ohjelmoidaan vastaamaan kuhunkin puheaktiin tietyllä tavalla, tietenkin keskustelun tilanteesta riippuen. Tällaista puheakteihin pohjautuvaa "dialogin kielioppia", joka kulkee mm. nimellä *speech act grammar* on kehitelty monissa järjestelmissä [ks. esim. Hagen and Popowich 2001]. Tällaisen lähestymistavan ongelmana on kuitenkin, että yksi käyttäjän puheenvuoro voi sisältää monia puheakteja. Paitsi että yksi puheenvuoro voi sisältää useita

lauseita, joilla jokaisella voi siis olla oma puheaktinsa, voi, kuten olemme huomanneet, yksikin lausahdus toteuttaa monta puheaktia. Vastaavasti tietenkin järjestelmän tuottamat vastaukset voivat sisältää monia puheakteja. Näin siis suoraviivainen puheaktista toiseen etenevä dialogin mallinnus ei toimi luonnollisen kielen dialogeissa. Tätä ongelmaa voidaan ratkoa [Hagen and Popowich 2001] tulkitsemalla kukin käyttäjän puheenvuoro kaikiksi niiksi puheakteiksi, joita siitä kyetään tunnistamaan ja rakentamalla dialogin mallinnus siten, että se ottaa syötteekseen tällaisen puheaktien joukon.

Teoreettisesti tarkasteltuna tällainen mallinnus ottaa pohjakseen puheaktien teorian varsinaisen ytimen, eli idean, että puhe on toiminnan muoto [Bunt 1989]. Tämän toiminnan vaikutukset ovat pääasiassa muutoksia keskustelukumppaneiden mielentiloissa, tai yleisemmin keskustelun kontekstissa, termin laajassa merkityksessä. Puheaktit näin ollen aiheuttavat siirtymiä kontekstista toiseen. Näin järjestelmä, pitääkseen yllä keskustelun tilannetta ja voidakseen jatkaa keskustelua, tarvitsee mallin tästä kontekstista, käytännössä siis keskustelunkumppanin "mielentilasta".

Järjestelmä siis pyrkii selvittämään mitkä ovat käyttäjän (relevantit) uskomukset ja aikomukset. Näistä aikomuksista käyttäjä viestii antamiensa syötteiden kautta. Lisäksi järjestelmän täytyy pitää yllä eksplisiittistä mallia omista tiedoistaan, jotka vaikuttavat keskustelun kulkuun. Kaiken kaikkiaan mallinnus siis sisältää tiedot siitä, mitä järjestelmä tietää ja mitä se tietää käyttäjän tietävän sekä mitä käyttäjä haluaa tai aikoo. Varman tiedon lisäksi Bunt [1989] ehdottaa, että mallinnukseen otetaan mukaan mahdollisuus epävarmaan tietoon. Järjestelmä siis voi arvella, että käyttäjä tietää tai tahtoo jotakin. Tämä tiedon epävarmuus voitaisiin myös määritellä jatkuvana välinä täydestä epätietoisuudesta täyteen tiedon varmuuteen. Bunt kuitenkin hylkää tällaisen "sumean" arvotuksen, ja tyytyy varmaan tietoon ja arveluun ja pystyy näin käsittelemään dialogin mallinnusta perinteisen logiikan keinoin. Dialogijärjestelmä siis rakennetaan mallintamalla, miten käyttäjän syöte muuttaa järjestelmässä olevaa kontekstin mallia ja toisaalta, minkälaisen puheaktin järjestelmä muodostaa kunkin kontekstin perusteella, ja miten tämä puheakti muuttaa järjestelmän kontekstimallia.

7. Puheaktit ja kielen generointi

Kuten edelläkin on tullut ilmi, voidaan puheakteja käyttää mallinnuksen yksiköinä myös kielen generointia ohjattaessa. Etenkin, jos dialogin mallinnus on tehty puheakteihin perustuen, on myös järjestelmän tulosteiden muodostus luonnollista hoitaa puheaktien kautta. Käytännössä tämä siis tarkoittaisi, että kielen generointimoduuli saisi syötteenään joukon tai listan puheakteja.

Tällainen syötteenä tuleva puheakti siis sisältäisi sekä puheaktin tyypin, että sen propositionaalisen sisällön.

Kun syöte tulee puheaktien muodossa, voi kielen generointi käyttää hyväkseen puheaktien tutkimuksen tuloksia. Muun muassa epäsuorien puheaktien generointi voisi olla mahdollista. Näin saataisiin aikaan esimerkiksi kohteliaita kysymyksiä ja pyyntöjä, kuten edellä esiteltiin. Searle [1979] luettelee hyvin selkeitä yleistyksiä epäsuorien puheaktien mahdollisista muodoista tyyliin *S voi tehdä epäsuoran pyynnön joko kysymällä josko tai sanomalla että pyydetyn teon preparatory condition on voimassa.* Tämä sääntö siis antaisi mahdollisuuden järjestelmälle asettaa pyyntö "sano ikäsi" muotoon "voitko sanoa ikäsi" tai "voit sanoa ikäsi". Ottamalla huomioon, että näitä sääntöjä voidaan yhdistellä, voidaan järjestelmän kohteliaisuuden tasoa nyt hallita esimerkiksi käyttäjän mieltymysten mukaan enemmän tai vähemmän kohteliaaksi.

8. Yhteenveto

Puheaktien teoria on kaiken kaikkiaan hyvin yleinen kokonaisuus. Teorian perusidea, siis ajatus kielen käytöstä toimintana, ei ole sinänsä edes teoria. Kun siihen lisätään loctionary, illocutionary, perlocutionary jaottelu sekä onnistuneisuusehtojen ajatus sekä idea illuctionary tason toimintojen ryhmittelystä, päästään tilanteeseen, jossa puheaktit voidaan käsittää Kuhnin tieteen teorian [1962] tasoiseksi teoriaksi, jonka piirissä tutkimusta sitten tehdään. Tämän perusteorian puitteissa sitten voidaan kehittää yksityiskohtaisempaa teoriaa puheakteista ja niiden toimintaperiaatteista. Tämä tutkimus on jatkunut alkuperäisten kielifilosofien töiden jälkeen kielitieteiden lisäksi mm. tekoälytutkimuksen piirissä [Cohen 1990].

Johtuen juuri perusteorian laajuudesta ja yleisyydestä, on puheaktien teoriaa sovellettu monilla suunnilla niin, että eri suunnilla ei välttämättä ole mitään tekemistä toistensa kanssa. Dialogijärjestelmien puolella puheakteja voidaan hyödyntää esimerkiksi dialogin etenemisen mallinnukseen, jolloin puheaktit toimivat käsitteellisinä kokonaisuuksina, joiden avulla dialogi saadaan jaettuna mallinnettaviin yksiköihin. Toisaalta voidaan hyödyntää puheaktien teorian parissa tehtyä teoreettista työtä tuomalla tuloksia dialogijärjestelmissä käytännön sovelluksiin. Esimerkiksi Searlen työ epäsuorien puheaktien parissa on, kuten esitettiin, sovellettavissa kielen generointiin.

Kaiken kaikkiaan puheaktien teoria on siis laaja kokonaisuus, jonka parista löytyy moninaisia hyödyllisiä tuloksia myös toteutuspainotteiseen puhekäyttöliittymätutkimukseen. Kuitenkaan se ei tarjoa suoraa mallia

minkään dialogijärjestelmän osan rakentamiseen vaan tarjoaa lähinnä käsitteellistä välineistöä erilaisten ratkaisujen etsimiseen.

Viiteluettelo

[Allan, 1998] Allan, K. (1998). *Meaning and speech acts*: http://www.arts.monash.edu.au/ling/speech_acts_allan.shtml.

[Austin, 1961] Austin, J.L. (1961). *How to Do Things With Words*: Harvard University Press.

[Bunt, 1989] Bunt, H. C. (1989). "Information Dialogues as Communicative Action", in Taylor, M. M., Néel F. and Bouwhuis, D.G. ed., *The Structure of Multimodal Dialogue:* Elsevier, pp 47-74.

[Cohen, 1990] Cohen, P. R. (1990). "Performatices in a Rationally Based Speech Act Teory", in Meeting of the Association for Computational Linguistics 1990, pp 79-88.

[Grice, 1989] Grice, H. P. (1989). *Studies in the Way of Words*: Harvard University Press.

[Hagen and Popowich, 2001] Hagen, E. and Popowich, F. (2001). "Flexible Speech Act Based Dialogue Management" in *1st SIGdial Workshop on Discourse and Dialogue*.

[Heeman et al., 1998] Heeman, A. P., Johnstron, M., Denney, J. and Kaiser, E. (1998). "Beyond Structured Dialogues: Factoring out Grounding" in *Proceedings of the International Conference on Spoken Language Processing* (ICSLP-98) Sydney, Australia, December 1998 pp. 863-866

[Hutchins, 1989] Hutchins, E. (1989). "Methaphors for Interface Design", in Taylor, M. M., Néel F. and Bouwhuis, D.G. ed., *The Structure of Multimodal Dialogue:* Elsevier, pp 11-28.

[Kuhn, 1962] Kuhn, T. S. (1962). *Tieteellisten vallankumousten rakenne.* Art House 1994.

[Labrou and Finin, 1996] Labrou, Y. ja Finin, T. (1996). *Semantics for an Agent Communication Language*: in Intelligent Agents IV: Agent Theories, Architectures and Languages" , Michael Wooldridge, Munindar Singh and Anand Rao (editors.), Springer-Verlag, Lecture Notes in Computer Science, Volume 1365, 1998.

[Levinson, 1983] Levinson, S. (1983). *Pragmatics.* Cambridge: Cambridge University Press.

[Searle, 1969] Searle, J.R. (1969). *Speech Acts*: Cambridge University Press.

[Searle, 1979] Searle, J.R. (1979): *Expression and Meaning, studies in the theory of speech act*: Cambridge University Press.

[Traum, 1999] Traum, D. R. (1999). "Speech acts for dialogue agents" in Woolridge, M. and Rao, A. ed., *Foundations of Rational Agency*. Kluwer, pp 169-201.