Käyttöliittymämallit

Käyttöliittymätutkimus-seminaari

Minna Majuri

mmajuri@cs.helsinki.fi

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

16.11.2001

Käyttöliittymämallit

Minna Majuri
mmajuri@cs.helsinki.fi

Käyttöliittymätutkimus-seminaari
Tietojenkäsittelytieteen laitos
Helsingin yliopisto
16.11.2001, 13 sivua.

Tiivistelmä

Tässä esitelmässä perehdytään käyttöliittymämalleihin ja käyttöliittymämallikieliin. Käyttöliittymämallit ovat suunnittelumalleja, jotka kuvaavat käyttöliittymäsuunnittelussa hyviksi havaittujen ratkaisujen periaatteita. Toisiinsa yhteydessä olevat käyttöliittymämallit muodostavat käyttöliittymämallikielen. Mallikieli toimii suunnittelijan apuvälineenä ongelmien ratkaisussa, suunnittelijoiden välisen viestinnän parantajana ja tiedon välittäjänä kokeneemmilta suunnittelijoilta aloittelijoille. Suunnitteluprosessin aikana malleja voidaan käyttää sovellusalueen dokumentointiin ja käyttäjien tyypillisimpien tavoitteiden selvittämiseen, suunnitteluongelmien ratkaisuapuna tai projektikohtaisten tärkeiden käsitteiden ja termien kuvaamiseen.

Aiheluokat: H.5.2 *user interfaces*
Avainsanat: käyttöliittymät, suunnittelumallit, suunnittelumenetelmät

Sisältö

1 Johdanto

Käyttöliittymämallit (user interface design pattern, interaction pattern) ovat suunnittelumalleja, jotka kuvaavat käyttöliittymäsuunnittelussa hyviksi havaittujen ratkaisujen periaatteita.

Suunnittelumalli (design pattern) kuvaa usein ja eri yhteyksissä toistuvan suunnitteluongelman ratkaisuperiaatteet. Mallien käytön oletetaan helpottavan suunnitteluprosessia ja parantavan lopputuloksen laatua, sillä niiden avulla pystytään välittämään kokemuksen synnyttämää tietoa eteenpäin. Mallien toivotaan myös vähentävän viestinnällistä kitkaa suunnittelijoiden välillä sekä auttavan maallikkoja osallistumaan suunnitteluun ammattilaisten kanssa.

Suunnittelumallikieli (pattern language) on hierarkisesti jäsennelty kokoelma toisiinsa yhteydessä olevia suunnittelumalleja. *Käyttöliittymämallikieleen* (HCI pattern language, UID pattern language) kuuluu sekä yleisluontoisia, käyttäjän koko tehtävän kattavia ratkaisuja että konkreettisempia, vaikkapa yksittäisiä käyttöliittymäohjekteja koskevia ratkaisuja.

Suunnittelumalleja voidaan käyttää eri tarkoituksiin käyttöliittymäsuunnittelun eri vaiheissa. Alussa malleja voidaan käyttää sovellusalueen dokumentointiin siten, että havaitut käytännöt ja toiminta kirjataan muistiin *toimintamalleina* (activity patterns). Suunnitteluvaiheessa käyttöliittymämallikielestä voi olla apua ongelmien ratkaisussa. Suunnittelumalleja voidaan käyttää myös projektikohtaisten tärkeiden käsitteiden ja termien kuvaamiseen, jotta kaikkilla osapuolilla olisi käsitys niiden merkityksistä. Useinhan käyttöliittymäsuunnitteluun osallistuu monen alan ammattilaisia, jotka eivät oeltusarvoisesti ymmärrä toistensa käyttämää terminologiaa.

Suunnittelumalleista kerrotaan enemmän luvussa 2 ja suunnittelumallikielistä luvussa 3. Luvussa 4 perehdytään tarkemmin käyttöliittymämalleihin ja esitetään joitakin näkökulmia käyttöliittymämallikieliin. Yhteenveto löytyy luvusta 5.

2 Suunnittelumallit

Suunnittelumalli (design pattern) kuvaa usein ja eri yhteyksissä toistuvan suunnitteluongelman ratkaisuperiaatteet. Mallit eivät ole keksimällä keksittyjä vaan ne on löydetty havainnoimalla, tutkimalla ja analysoimalla olemassaolevia ratkaisuja.

Suunnittelumallien käytön oletetaan helpottavan suunnitteluprosessia ja parantavan lopputuloksen laatua, sillä niiden avulla pystytään välittämään kokemuksen synnyttämää tietoa eteenpäin. Mallien toivotaan myös vähentävän viestinnällistä kitkaa suunnittelijoiden välillä sekä auttavan maallikkoja osallistumaan suunnitteluun ammattilaisten kanssa.

Suunnittelumalleilla kuvataan enimmäkseen onnistuneita ja toivottuja ratkaisuja. Huonoja ratkaisuja kuvataan *antimallien* (antipatterns) avulla [7]. Antimalli esittää asiayhteyden ja ongelmallisen ratkaisun, ja mikä tärkeintä, myös kuvauksen paremmasta ratkaisusta.

2.1 Alexanderin arkkitehtuurimallit

Arkkitehti Christopher Alexander työtovereineen kehitteli 1970–luvulla ajatuksen *suunnittelumallikielestä* (pattern language) [1], jolla kuvataan onnistuneita rakennus- ja ympäristösuunnitteluratkaisuja. Arkkitehtuurimallit kuvaavat fyysisen ympäristön, kuten kaupunkien, kortteleiden, talojen ja huoneiden suunnittelussa hyviksi havaittuja ratkaisuja. Eritasoiset mallit muodostavat yhdessä ketjun, joka johtaa suurista linjoista pieniin yksityiskohtiin. Mallit tarvitsevat ja tukevat toisiaan: todella laadukkaaseen kokonaisuuteen päästään vain, jos sekä suuret että pienet asiat on suunniteltu hyvin.

Tavoitteena oli välittää tietoa laadukkaista ratkaisuista, mutta myös luoda yhteinen kieli arkkitehdeille ja tavallisille ihmisille. Kun osapuolet ymmärtäisivät toisiaan, niin ihmiset pääsisivät paremmin vaikuttamaan oman asuinympäristönsä suunnitteluun ja ehkä kokisivat sen ansiosta ympäristönsä viihtyisämmäksi ja toimivammaksi.

2.2 Miten suunnittelumallit syntyvät?

Suunnittelumalleja ei keksitä vaan löydetään. Arkkitehtuurimallit löytyivät havainnoimalla paikkoja sekä niissä liikkuvia ja toimivia ihmisiä, ja tiivistämällä toistuvasti havaittujen ratkaisujen periaatteet ja vaikutukset.

Sekä arkkitehtuuri- että käyttöliittymämallien löytäminen perustuu ihmisten toiminnan havainnointiin ja toiminnan arviointiin. Tämä hieman kyseenalaistaa mallien ohjektiivisuuden — mikä takaa, ettei havainnointi ole vaikuttanut havainnoitaviin, ja millä perusteella joku ratkaisu on laadukkaampi kuin joku toinen? Hhavainnointiprosessia lienee mahdotonta ja tarpeetontakin standardoida, ja laadun määritelmää taas on niin mahdoton tavoittaa, että Alexanderkin tyytyi vain intuitiiviseen *quality without a name* - käsitteeseen [2]. Paras tapa suhjektiivisuuden vähentämiseen lieneekin se, ettei päätöksiä perusteta vain yhden ihmisen mielipiteisiin ja muutamiin havaintoihin.

2.3 Miten suunnittelumallit esitetään?

Suunnittelumallit esitetään useimmiten epäformaaleina sanallisina selityksinä. Tärkeimmät kohdat mallin kuvauksessa ovat nimi, asiayhteys, ongelma ja sen ratkaisuperiaatteet. Ratkaisua havainnollistetaan asiaankuuluvin tavoin, esimerkiksi valokuvilla tai kaavioilla. Lisäksi saatetaan kertoa mallin soveltamisen eduista ja haitoista sekä antaa viitteitä muihin malleihin, joita kyseisen mallin yhteydessä usein tarvitaan.

3 Suunnittelumallikielet

Suunnittelumallikieli (pattern language) on hierarkisesti jäsennelty, toisiinsa yhteydessä olevien suunnittelumallien kokoelma. Kieleen kuuluu sekä yleisluontoisia, suuren mittakaavan ratkaisuja, että pienten, konkreettisempien ongelmien ratkaisuja. Yksi suunnittelumalli ei ratkaise kaikkia suunnitteluongelmia, joten malleista viitataan muihin malleihin, joita tilanteessa myös kannattaa käyttää.

Suunnittelumallien muodostamaa järjestelmää voidaan kuvata formaalisti hierarkisella verkolla, jossa solmut kuvaavat suunnittelumalleja ja kaaret niiden välisiä yhteyksiä [4].

- *Suunnittelumallikieli* on suunnattu, syklitön verkko $\mathbf{PL} = (\mathbf{P},\mathbf{R})$, jossa on solmut $\mathbf{P} = \{P_1, \ldots, P_n\}$ ja kaaret $\mathbf{R} = \{R_1, \ldots, R_m\}$.
- Jokainen verkon solmu $P \in \mathbf{P}$ on *suunnittelumalli.*
- Jos suunnittelumallista viitataan toiseen, niin verkossa on kaari kyseisten solmujen välillä, siis kaikille $P, Q \in \mathbf{P}$ *: P viittaa Q:hun* $\iff \exists R = (P, Q) \in \mathbf{R}$.
- Solmusta $P \in \mathbf{P}$ lähtevät kaaret edustavat *viitteitä* kyseisestä mallista muihin malleihin. Solmuun tulevat kaaret ilmentävät mallin *asiayhteyttä* (context).
- Jokainen solmu $P \in \mathbf{P}$ on itsessään joukko $P = \{n, r, i, p, f_1 \ldots f_i, e_1 \ldots e_j, s, d\}$, missä n on mallin nimi, r sen luokitus (ranking), i kuvallinen selitys, p ongelma, $f_1 \ldots f_i$ tilanteeseen liittyvät muut ehdot (forces), $e_1 \ldots e_j$ esimerkit, s ratkaisu ja d kaavioesitys ratkaisusta.

Edellä esitetty formalismi korostaa suunnittelumallien välisten yhteyksien tärkeyttä. Suunnittelumallikielen rinnastaminen verkkoon on yritys osoittaa, kuinka yhteydet mallien välillä muuttavat kokoelman irrallisia niksejä suuremmaksi, mielekkääksi kokonaisuudeksi.

4 Käyttöliittymämallit

Käyttöliittymämallit (user interface design pattern, interaction pattern) ovat suunnittelumalleja, jotka kuvaavat käyttöliittymäsuunnittelussa hyviksi havaittujen ratkaisujen periaatteita. Ne muistuttavat lähtökohdiltaan alkuperäisiä arkkitehtuurimalleja, onhan molemmissa suunnittelun lähtökohtana ihmisen toiminnan huomioonottaminen. Siinä missä arkkitehtuurimallien tarkoitus on parantaa ympäristön laatua, käyttöliittymämalleilla autetaan tekemään (ohjelmien) käyttöliittymistä tarkoituksenmukaisia ja miellyttäviä.

Käyttöliittymäsuunnitteluun osallistuu usein monen alan asiantuntijoita. Käyttöliittymäsuunnittelijoiden ja ohjelmoijien lisäksi prosessissa on mukana myös sovellusalan asiantuntijoita tai ' tavallisia käyttäjiä". Ryhmät eivät yleensä tunne toistensa terminologiaa, joten keskustelu ja ideoiden välittäminen muille on vaikeaa. Toimiva yhteistyö olisi kuitenkin oleellisen tärkeää, jotta päästäisiin hyvään lopputulokseen.

Suunnittelumalleja voidaan käyttää eri tarkoituksiin käyttöliittymän suunnittelun eri vaiheissa. Suunnittelun alussa malleja voidaan käyttää sovellusalueen dokumentointiin. Mallien avulla kirjataan muistiin havaittuja käytäntöjä ja toimintaa, joista pyritään tekemään yleistyksiä — selvittämään siis käyttäjien tyypillisimpiä tavoitteita. Suunnitteluvaiheessa käyttöliittymämallikielestä voi olla apua ongelmien ratkaisussa. Suunnittelumalleja voidaan käyttää myös projektikohtaisten tärkeiden käsitteiden ja termien kuvaamiseen, jotta kaikkilla osapuolilla olisi käsitys niiden merkityksistä.

4.1 Sovellusalan kuvaaminen malleilla

Hyvän käyttöliittymäsuunnittelijan tulee ymmärtää sitä sovellusalaa, jonka työvälineeseen käyttöliittymää ollaan tekemässä. Sovellusalaan tutustuminen tapahtuu alan asiantuntijoiden kanssa keskustelemalla ja heidän toimiaan seuraamalla. Usein toistuvat tilanteet voidaan kuvata *toimintamalleina* (activity patterns) [3]. Malleja käytetään ainoastaan asioiden ja tapahtumien kuvailuun, niihin ei liitty arviointia

tai arvostelua. Toimintamallit antavat suunnittelijoille käsityksen siitä, mitkä ovat työkalun käyttäjien tyypillisimpiä tavoitteita. Tietoa tavoitteita puolestaan tarvitaan käyttöliittymäsuunnittelun lähtökohdaksi.

Muuan esimerkki toimintamallista, joka ei tosin ole sidottu mihinkään erityiseen toimialaan, on jonotuksen järjestäminen hajota ja hallitse -periaatteella [3].

> **Nimi:** Divide and conquer queue
>
> **Ongelma ja asiayhteys:** Konferenssiluennolle saapuu paljon ihmisiä yhdellä kertaa. Kaikki tulevat viime tingassa ja heidän pitää ilmoittautua ennen luentoa.
>
> **Vaikuttavat seikat:** Jonotus on ihmisistä ärsyttävää, varsinkin jos heillä on kiire. Kukaan ei myöskään pidä siitä, että hänen jälkeensä tulevat pääsevät hoitamaan asiansa ensin ilman hyvää syytä.
>
> **Ratkaisu:** Jaetaan ilmoittautumisjono useaksi pienemmäksi jonoksi jollain järkevällä tavalla, vaikkapa aakkosjärjestyksen perusteella. Lisää jonoja voidaan perustaa tai jonoja voidaan yhdistää tilanteen mukaan.

4.2 Käyttöliittymämallikielet

Useista eri tasoisista ja toisiinsa sidoksissa olevista käyttöliittymämalleista muodostuu käyttöliittymämallikieli (HCI pattern language, UID pattern language). Käyttöliittymämallikieli sisältää ratkaisuja niin laajoihin, yleisluontoisiin ongelmiin kuin pieniin, tarkasti rajattuihin ongelmatilanteisiinkin. Mallikielet voidaan ajatella olevan tavallaan 'korkeamman tason suunnitteluohjeita' ja kirjastoja, lisäksi ne ovat välineitä suunnittelijoiden väliseen kommunikaatioon ja tiedon välittämiseen kokeneemmilta suunnittelijoilta aloittelijoille.

Eräs käyttöliittymämallikieli kuvaa vuorovaikutteisten 'näyttelypömpeleiden' tai informaatiokioskien käyttöliittymäratkaisuja [4]. Ratkaisuja sopivat sellaisten sovellusten suunnitteluun, joita käytetään pääasiassa vain kerran ja silloinkin vain lyhyen aikaa, eikä käyttäjien oleteta vaivautuvan opettelemaan mitään. Yksi tähän kieleen

kuuluvista käyttöliittymämalleista on **Incremental Revealing**. Seuraavassa on siitä lyhyt kuvaus, josta kylläkin puuttuu tieto siitä, mitä muita malleja tämä malli kaipaa seurakseen.

> **Nimi:** Incremental Revealing
>
> **Asiayhteys:** Suunniteltavana on tietokoneavusteinen näyttelypömpeli, joka houkuttelee kävijöitä.
>
> **Ongelma:** Yksinkertainen ulkoasu ja järjestelmän tiedollisen laajuuden esittäminen ovat vastakohtaisia tavoitteita.
>
> **Ratkaisu:** Esitetään aluksi vain hyvin suppea ja yksinkertainen yleiskuva järjestelmän toiminnasta. Jos käyttäjä kiinnostuu yleiskuvan jostakin osasta, siitä tarjotaan lisätietoa ja näytetään, mitä johdatteluesityksen 'takana' on.

Jotta suunnittelija voi käyttää mallikieltä apuvälineenään, hänen pitää jollain tavalla pystyä arvioimaan kieleen kuuluvia malleja ja saamaan tietoa niiden ominaisuuksista ja vaikutuksista lyhyesti ja ytimekkäästi. Käyttöliittymämalleja voidaan luokitella esimerkiksi sen perusteella, minkä laajuiseen ongelman ne ratkaisevat tai mihin käyttöliittymän osa- alueeseen ne liittyvät. Käytännönläheisempi lähtökohta on luokitella mallit sen perusteella, kuinka ne parantavat ohjelman käytettävyyttä.

4.2.1 Abstraktiotaso, toiminta, fyysinen ulottuvuus

Käyttöliittymämalleja voidaan luonnehtia *abstraktiotason* (level of abstraction), *toiminta* (function) ja *fyysisen ulottuvuuden* (physical dimension) perusteella [5]. Abstraktiotaso on näistä tärkein. Abstrakteimmat mallit keskittyvät käyttäjän koko tehtävän (task) käsittäviin suuren mittakaavan ongelmiin. Konkreettisimmat mallit puolestaan käsittelevät käyttöliittymäratkaisuja yksittäisten käyttöliittymäobjektien tasolla.

Toiminta-attribuutti kertoo, mihin käyttöliittymän osa-alueeseen malli liittyy. Ratkaisu voi liittyä *havaitsemiseen* (perception) eli siihen, miten ohjelman tulosteet

esitetään käyttäjälle. Jos ratkaisu liittyy *tiedonkäsittelyyn* (manipulation), malli esittää tavan ohjelmalle annettavan syötteen käsittelyyn. Ratkaisu voi myös liittyä käyttöliittymässä liikkumiseen (navigation). Käyttöliittymämallien fyysisellä ulottuvuudella tarkoitetaan sitä, ratkaiseeko malli *tilaan* (spatial layout), tapahtumien sarjallisuuteen vai aikaan liittyvän ongelman.

Alussa esitelty Incremental revealing -malli on korkeimman abstraktiotason malli, sillä se koskee käyttäjän koko tehtävää 'Mitähän tässä pömpelissä esitetään?' . Mallin toiminta liittyy havaitsemiseen, sillä ratkaisu esittää tavan näyttää paljon tietoa hellävaraisesti ja houkuttelevasti käyttäjälle. Fyysinen ulottuvuus on sarjallisuus, sillä ratkaisun perusidea on informaation paljastaminen vähitellen peräkkäisinä tapahtumina.

4.2.2 Käytettävyysvaikutus

Edellä esitetty tapa luonnehtia käyttöliittymämalleja perustuu täysin suunnittelijan näkökulmaan, koska mallien luokitus vastaa suunnittelijan kohtaamia ongelmia. Käyttöliittymämallien ominaisuuksia voidaan tarkastella myös *käyttäjänäkökulmasta* (user perspective). Silloin keskitytään siihen, millaisiin käyttäjän tehtäviin mallit esittävät ratkaisuja ja kuinka ratkaisut parantavat käytettävyyttä [10].

Käytettävyyttä voidaan luonnehtia käytettävyysperiaatteilla (user interface principles) [9]. Periaatteita on kahdeksan:

- **Näkyvyys** (visibility): Käyttäjän pitäisi pystyä selvittämään, miten jokin toimii vain katsomalla sitä.
- **Mahdollisuudet** (affordances): Objektin havaittavien käyttömahdollisuuksien tulisi vastata sen todellisia käyttömahdollisuuksia.
- **Luonteva vastaavuus** (natural mapping): Sen, *miten* tehdään, tulisi luontevasti vastata sitä, *mitä* tehdään.
- **Rajoitteet** (constraints): Tehtävän suorittamiseen tarvittavien tietomäärän tulisi olla mahdollisimman pieni.

- **Käsitteellinen malli** (conceptual model): Käyttäjän sisäinen malli ei saa olla ristiriidassa sen kanssa, miten asia todella toimii. Tekojen seurausten tulisi siis olla ennakoitavissa.
- **Palaute** (feedback): Käyttäjän pitää saada tietää, onko jotakin tapahtumassa ja etenevätkö asiat oikein.
- **Turvallisuus** (safety) [10]: Käyttäjän tekemät vahingot tai virheet eivät saa aiheuttaa peruuttamatonta tuhoa.
- **Joustavuus** (flexibility) [10]: Järjestelmän pitää sietää käyttäjän epäröintiä ja mielenmuutoksia.

On vaikea arvioida, mihin käytettävyysperiaatteisiin aiemmin esitellyn Incremental Revealing -mallin soveltaminen vaikuttaisi. Mistä se kertoo? Eikö Incremental revealing olekaan käyttöliittymämalli, vai onko niin, että on olemassa hyviä käyttöliittymäratkaisuja, vaikkei niiden hyvyyttä pystytäkään osoittamaan käytettävyysperiaatteiden avulla?

4.2.3 Käyttöliittymämallikokoelmia

Seuraavassa on muutamia linkkejä WWW:stä löytyviin käyttöliittymämallikokoelmiin.

- Jennifer Tidwell.
  Common Ground: A Pattern Language for Human-Computer Interaction.
  `www.mit.edu/~jtidwell/common_ground.html` [26.10.2001].
- The Usability Group at the University of Brighton,.
  The Brighton Usability Pattern Collection.
  `www.it.bton.ac.uk/cil/usability/patterns/` [26.10.2001].
- Martijn van Welie.
  The Amsterdam Collection of Patterns in User Interface Design.
  `www.cs.vu.nl/~martijn/patterns/index.html` [6.11.2001].

4.3 Projektikohtaiset mallikielet

Kuten alussa todettiin, käyttöliittymäsuunnitteluun osallistuu moninainen joukko ihmisiä, joilla on hyvin erilaiset taustat. Osallistujat eivät välttämättä tunne toistensa käyttämien termien ja käsitteiden merkityksiä, joten keskustelu ja ideointi saattaa olla vaikeaa. Pojektikohtaiset tärkeät käsitteet, käyttäjän tavoitteita kuvaavat toimintamalli ja suunnittelun apuna käytetyt käyttöliittymämallit voitaisiin kenties koota projektin 'yhteiseksi kieleksi". Tämä voisi parantaa osapuolten välistä ymmärrystä ja helpottaa ns. tavallisten käyttäjien osallistumista suunnitteluun [8].

4.4 Avoimia kysymyksiä

Seuraavassa on muutamia käyttöliittymämalleihin liittyviä kysymyksiä. Vastauksia ei varmasti kukaan osaa antaa, mutta ehkäpä tästä syntyy hyvää keskustelua.

- Suunnittelumalleista on innostuttu erityisesti ohjelmistotekniikan puolella, mutta arkkitehtuurissa ja kaupunkisuunnittelussa ne eivät ole lyöneet itseään läpi. Myöskään käyttöliittymä- tai käytettävyyssuunnittelijat eivät ole olleet ensimmäisten joukossa innostumaan asiasta. Miksi?
- Onko mahdollista luoda sellaisia käyttöliittymäsuunnittelumalleja, jotka ovat riittävän ammattimaisia suunnittelijoiden väliseen viestintään ja samalla riittävän helppotajuisia suunnittelijan ja käyttäjän väliseen viestintään? Onko tällainen 'universaali käyttöliittymämallikieli' edes tarpeen?
- Miten käyttöliittymämalleja parhaiten löydetään, millaista tietoa niiden pitäisi sisältää ja millaisella abstraktiotasolla? Mikä on paras tapa esittää käyttöliittymämalli: jokin formaali menetelmä, epämuodollinen tekstikuvaus, UML-kaaviot, screen-shotit?
- Miten käyttöliittymämallien laatua arvioidaan?
- Mitä ongelmia käyttöliittymämalleihin liittyy?

5 Yhteenveto

Käyttöliittymämallit ovat suunnittelumalleja, jotka kuvaavat käyttöliittymäsuunnittelussa hyviksi havaittujen ratkaisujen periaatteita. Malleja voidaan käyttää eri tarkoituksiin suunnitteluprosessin eri vaiheissa. Suunnittelun alussa malleja voidaan käyttää sovellusalueen dokumentointiin ja käyttäjien tyypillisimpien tavoitteiden selvittämiseen. Suunnitteluvaiheessa käyttöliittymämallikielet tai -kokoelmat ovat avuksi ongelmien ratkaisussa. Projektikohtaisten tärkeiden käsitteiden ja termien, käyttäjän tavoitteita kuvaavien toimintamallien ja suunnittelussa käytettyjen käyttöliittymämallien kokoaminen projektin 'yhteiseksi kieleksi" voisi parantaa osapuolten välistä ymmärrystä, ja parantaa siten sekä suunnitteluprosessin että lopputuloksen laatua.

Käyttöliittymämallien olemuksesta ei olla asiantuntijapiireissä päästy varsinaisesti yksimielisyyteen. On jopa esitetty, että ala on niin nuori ja tekniikka niin nopeasti kehittyvää, että löydettyjen käyttöliittymämallien laatu on vain arvailujen varassa [6]. Kaiken kaikkiaan käyttöliittymämallien katsotaan kuitenkin olevan erittäin mielenkiintoinen ja monipuolinen tutkimsukohde — avoimia kysymyksiä tuntuu ainakin olevan enemmän kuin vastauksia.

Viitteet

[1] Alexander C. et al.

A Pattern Language: Towns, Buildings, Constructions. Oxford University Press, New York, USA, 1977.

[2] Alexander C.

The timeless way of building. Oxford University Press, New York (NY), USA, 1979.

[3] Bayle E. et al.

Putting it all together: towards a pattern language for interaction design. *SIGCHI Bulletin*, 30(1), january 1998, 17-23.

[4] Borchers J. O.

A pattern approach to interaction design. Teoksessa: *Proc. Designing Interactive Systems: Processes, Practices, Methods and Techniques* (DIS2000), New York City, USA, August 2000. ACM, New York, 2000, 369-377.

[5] Borchers J. O.

CHI meets PloP: An interaction patterns workshop. (Workshop at ChiliPloP'99). *SIGCHI Bulletin*, 32(1), january 2000, 9-12.

[6] Borchers J. O., Thomas J. C.

Patterns: What's In It For HCI? (Panel). Teoksessa: *Extended Abstracts of the CHI 2001 Conference on Human Factors in Computing Systems.* Seattle, March 2001. ACM Press, New York, 2001.
`www.stanford.edu/~borchers/publications/` [8.11.2001].

[7] Brown W. et al.

Antipatterns: Refactoring Software, Architectures and Projects in Crisis. Wiley Computer Publishing, 1998.

[8] Erickson T.

Lingua Francas for design: sacred places and pattern languages. Teoksessa: *Proc. Designing Interactive Systems: Processes, Practices, Methods and*

Techniques (DIS2000), New York City, USA, August 2000. ACM, New York, 2000, 357-367.

[9] Norman D.

The Design of Everyday Things. Basic Books, 1988.

[10] van Welie M., Traetteberg H.

Interaction patterns in user interfaces. *Proc. 7th Conference on Pattern Languages of Programs* (PloP 2000), Monticello, Illinois, USA, 2000. `jerry.cs.uiuc.edu/~plop/plop2k/proceedings/proceedings.html` [6.11.2001].