1 Käytettävyyden arviointi ilman käyttäjiä

Sirpa Riihiaho

Käytettävyyttä voidaan arvioida jo hyvin aikaisessa vaiheessa, kun ensimmäiset tuoteideat syntyvät. Asiantuntijat voivat arvioida tuotteen käytettävyyttä esimerkiksi toiminnallisuutta kuvaavien kaavioiden perusteella. Niiden pohjalta voidaan tarkastella työtehtävien etenemistä ja toiminnan loogisuutta. Käyttäjien kommentointia helpottavat erilaiset mallit ja prototyypit. Esimerkiksi luonnokset ohjelman näytöistä toimivat hyvänä pohjana keskusteluille ja läpikäynneille, joissa arvioidaan uuden järjestelmän sopivuutta käyttäjän työtehtäviin. Käyttöliittymästä voidaan myös irrottaa pieniä osia arvioitavaksi. Esimerkiksi järjestelmän valikkoa, ikoneita ja termistöä voidaan arvioida käyttäjien kanssa irrallaan muusta järjestelmästä muistaen kuitenkin konteksti, johon ne kuuluvat.

Suunnitteluvaiheessa työn tukena ovat *yleiset käytettävyyssäännöt*, eri järjestelmäympäristöihin kehitetyt *käyttöliittymäohjeet* ja mahdolliset yritys- tai tuotekohtaiset *tyylioppaat*. Vaikka ohjeita pyritään noudattamaan jo suunnittelussa, välillä on syytä pitää katselmuksia, joissa tarkistetaan järjestelmällisesti, rikkooko järjestelmä joitakin sääntöjä. Omalle työlleen on helposti sokea, joten tarkistukseen on hyvä pyytää joko käytettävyysasiantuntijoita tai kollegoja kyseisen projektin ulkopuolelta. Asiantuntijoilla on kokemusta erilaisista järjestelmistä, joten he ovat tehokkaita löytämään yleisiä käytettävyysongelmia.

Kun järjestelmästä valmistuu jotakin edes osittain toimivaa tai näkyvää - erilaisia malleja tai prototyyppejä, arviointiin kannattaa ottaa mukaan todellisia käyttäjiä, sillä heillä on tuotteen käyttötarkoituksesta paras tietämys. Käyttäjien kanssa tuotetta tai ohjelmistoa voidaan arvioida *käytettävyystestein* ja sen eri muunnelmin, kuten *vapaalla läpikäynnillä* tai *ryhmäläpikäynnillä*.

1.1 Eri menetelmät eri tarpeisiin

Eri arviointimenetelmät pureutuvat erityyppisiin käytettävyysongelmiin. *Heuristisella arvioinnilla* löydetään yleisiä käytettävyysongelmia, kuten vieraat termit, epäyhtenäisyydet järjestelmän sanastossa ja näyttöjen sommittelussa tai painikkeiden ja tekstikenttien epäjohdonmukainen ryhmittely ja järjestys. *Standardikatselmuksissa* puolestaan löydetään poikkeamat tietyn käyttöympäristön normaalista käytännöstä. Järjestelmän opetteluun liittyviä ongelmia voidaan kartoittaa esimerkiksi *kognitiivisella läpikäynnillä* ja käytön tehokkuuteen käytettävyystestein. Vuorovaikutuksen sujuvuutta ja soveltuvuutta aiottuihin tehtäviin arvioidaan parhaiten käyttäjien avustuksella erilaisissa käytettävyystesteissä tai ryhmäläpikäynnissä.

1.2 Heuristinen arviointi

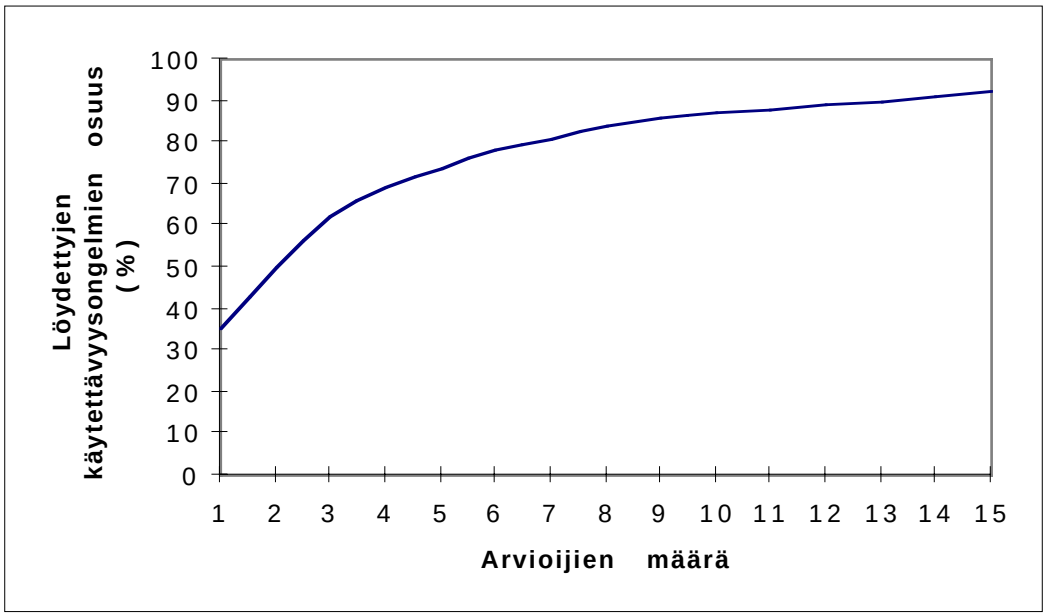
Heuristisessa arvioinnissa tarkastetaan käyttöliittymän osat erilaisten käytettävyysperiaatteiden eli heuristiikkojen avulla. Muistilistat voivat sisältää yleisiä suunnitteluohjeita tai tietylle tuoteperheelle räätälöityjä käytettävyysohjeita. Arvioinnissa ei tarvitse käyttää tutkittavaa järjestelmää, vaan arviointiin riittävät järjestelmän ja sen käyttöliittymän suunnitelmat tai varhaisen vaiheen prototyypit. Näin arviointi pystytään tekemään varhaisessa vaiheessa tuotekehitystä ja arvioinnin tulokset voidaan ottaa huomioon kehitystyössä.

Heuristinen arviointi on hyvä tapa etsiä käyttöliittymän ongelmakohtia yksinkertaisella, nopealla ja edullisella tavalla. Arvioinnissa ei juurikaan oteta kantaa järjestelmän hyödyllisyyteen eli sen sopivuuteen aiottuun tehtävään, koska arvioijilla ei useinkaan ole tähän tarvittavaa sovellusalatietoa. Heuristinen arviointi ei siis korvaa käyttäjien kanssa tehtäviä testejä, vaan tarjoaa kehitystyön tueksi menetelmän, jolla käytettävyysvirheitä voidaan karsia mahdollisimman aikaisessa vaiheessa. Osa arvioinneissa löydetyistä ongelmista saattaa olla sellaisia, että niihin ei lyhyissä käytettävyystesteissä jouduta, mutta pidemmällä aikavälillä nämäkin saattaisivat aiheuttaa turhaa hämmennystä.

1.2.1 Ketkä sopivat arvioijiksi?

Heuristisen arvioinnin tehokkuus riippuu siitä, kuinka paljon arvioijalla on kokemusta käytettävyydestä ja tutkittavasta järjestelmästä sekä sen käyttötarkoituksesta. Käytettävyyden ja sovellusalueen hallitseva asiantuntija on tehokkain yhdistelmä, mutta tällaisia henkilöitä on harvoin saatavilla. Seuraavaksi paras vaihtoehto on pyytää käytettävyysasiantuntijaa arvioimaan järjestelmää. Hyödyllisiä tuloksia saadaan silloinkin, kun käytettävyysasioita tuntematon sovelluksen suunnittelija arvioi järjestelmää muistilistojen avulla, sillä jokainen ajoissa havaittu ja korjattu ongelma helpottaa jatkotyötä. Käyttäjiä ei juurikaan kannata käyttää arvioijina, koska heillä on harvoin hyvää käsitystä käytettävyyden osatekijöistä tai tietojärjestelmien toimintaperiaatteista. Käyttäjien kanssa järjestelmää on parempi tarkastella heidän työtehtäviensä avulla esimerkiksi käytettävyystesteissä.

Eri arvioijat löytävät järjestelmästä erilaisia ongelmakohtia. Tämän vuoksi arvioijia tulisi olla useampi. Nielsen (1993) on tutkimuksissaan todennut, että yksi käytettävyyteen perehtynyt arvioija löytää noin 35 % käytettävyysongelmista, kolme arvioijaa noin 60 % ja viisi arvioijaa noin 75 % ongelmista. Suositeltava arvioijien määrä on siis 3-5. Tätä useammat arvioijat tuovat mukanaan melko vähän lisätuloksia.



Kuva 1: Kolmesta viiteen käytettävyysasiantuntijaa on suositeltava määrä heuristiseen arviointiin

1.2.2 Arvioinnin vaiheet

Heuristinen arviointi voidaan jakaa neljään vaiheeseen:
1. Järjestelmän läpikäynti yksittäin
2. Ongelmien keruu
3. Ongelmien vakavuuden arviointi
4. Keskustelu ja ideointi

Kunkin arvioijan tulisi käydä käyttöliittymä läpi yksinään muutaman tunnin istuntona. Jos järjestelmä on niin laaja, että sitä ei ehdi muutamassa tunnissa käymään läpi, siitä on syytä valita vain tietty osa kerrallaan arvioitavaksi. Käyttöliittymä tulisi käydä heuristisessa arvioinnissa läpi vähintään kahdesti. Ensimmäisellä kerralla luodaan yleiskuva järjestelmästä ja sen rakenteesta. Toisella kerralla keskitytään tiettyihin käyttöliittymän osiin ja niiden mahdollisiin käytettävyysongelmiin.

Heuristinen arviointi voidaan tehdä joko siten, että katsotaan järjestelmä näyttö näytöltä läpi ja kunkin näytön kohdalla tarkistetaan näytön osat heuristiikkojen avulla, tai siten, että valitaan yksi heuristiikka tarkasteltavaksi ja käydään kaikki näytöt läpi kyseistä sääntöä silmällä pitäen.

Jos arvioitavan järjestelmän käyttö vaatii tietyn sovellusalan syvällistä tuntemusta tai ammattitermistön hallintaa eikä arvioija tätä hallitse, arvioinnissa on hyvä olla mukana sovellusalan tunteva tarkkailija. Tämä tarkkailija kirjaa havaitut ongelmat, neuvoo tarvittaessa ongelmatilanteissa ja vastaa sovellusalaan liittyviin kysymyksiin.

Kunkin arviointi-istunnon tuloksena syntyy lista havaituista ongelmista sekä perustelut, miksi nämä ovat ongelmia. Kukin arvioija voi laatia oman listansa tai

arvioinneissa voi olla mukana erillinen havainnoija, joka kirjaa arvioijan löytämät ongelmat.

Erillisten läpikäyntien jälkeen ongelmalistoista kootaan yksi lista. Jotta listaa voitaisiin hyödyntää mahdollisimman tehokkaasti kehitystyössä, ongelmat tulisi asettaa tärkeysjärjestykseen niiden vakavuuden suhteen. Tässä voidaan käyttää esimerkiksi seuraavaa jaottelua:

- katastrofaalinen
- vakava
- häiritsevä
- vähäinen
- kosmeettinen

Ongelman vakavuuteen vaikuttaa se, kuinka usein ongelma ilmenee, kuinka vaikeaa siitä on selvitä ja kuinka helposti ongelma opitaan välttämään.

Ongelmien ryhmittely kannattaa tehdä varsinaisten arviointien jälkeen, jotta arvioinneissa voidaan keskittyä ongelmien löytämiseen ja jotta kaikilta arvioijilta saataisiin mielipide kunkin ongelman vakavuudesta. Tämän arvioinnin voi tehdä joko ryhmässä tai jälleen kukin arvioija yksinään.

Ongelman vakavuuden lisäksi jatkotoimenpiteisiin vaikuttaa ongelmien korjaamiseen vaadittavat panostukset. Arvioijat osaavat esittää vain suuntaa-antavia arvioita korjaamiseen tarvittavasta työmäärästä, mutta nekin auttavat suuntaamaan työpanostusta. Suunnitteluryhmältä voidaan kerätä tarkempia arvioita tarvittaessa.

Kun priorisoitu ongelmalista on saatu valmiiksi, kannattaa vielä kokoontua ryhmässä keskustelemaan tuloksista ja luomaan parannusehdotuksia. Ryhmässä on hyvä olla mukana arvioijat, mahdollinen tarkkailija ja muutama suunnitteluryhmän jäsen. Ongelmien lisäksi ryhmässä tulisi keskustella käyttöliittymän hyvistä puolista, koska heuristinen arviointi pyrkii nimenomaan löytämään ongelmia ja hyvät puolet jäävät usein huomiotta.

1.2.3 Arvioinnissa käytettävät muistilistat

Kirjallisuudessa esitetään käyttöliittymien yleisistä periaatteista lukuisia erilaisia listoja, jotka soveltuvat muistilistoiksi niin käyttöliittymän suunnitteluun kuin sen arviointiin. Ohjeet voidaan jakaa komeen eri tasoon:

- yleiset käytettävyyssäännöt
- yksityiskohtaiset käytettävyysohjeet
- tietyn ympäristön käyttöliittymäohjeisto

Yleisiä käytettävyysohjeita esittävät muun muassa Shneiderman (1992) sekä Nielsen ja Molich (1990). Nämä ohjeet ovat melko yleisiä ja vaativat tarkennusta tutkittavan tuotteen mukaan. Ohjeet on kuitenkin helppo muistaa ja täten mahdollista ottaa huomioon koko suunnittelun ajan.

Yksityiskohtaisia käytettävyysohjeita löytyy muun muassa Smithin ja Mosierin (1986) raportista. Siinä annetaan ohjeita ohjelmiston käyttöliittymän suunnitteluun tiedon syöttämisen, tiedon esittämisen, tehtävien vaiheistamisen, käyttäjän opastamisen, tiedon välittämisen ja tiedon suojaamisen osalta.

Ympäristökohtaisia käyttöliittymäohjeita esitetään eri käyttöjärjestelmien oppaissa.

1.2.4 Heuristiset säännöt

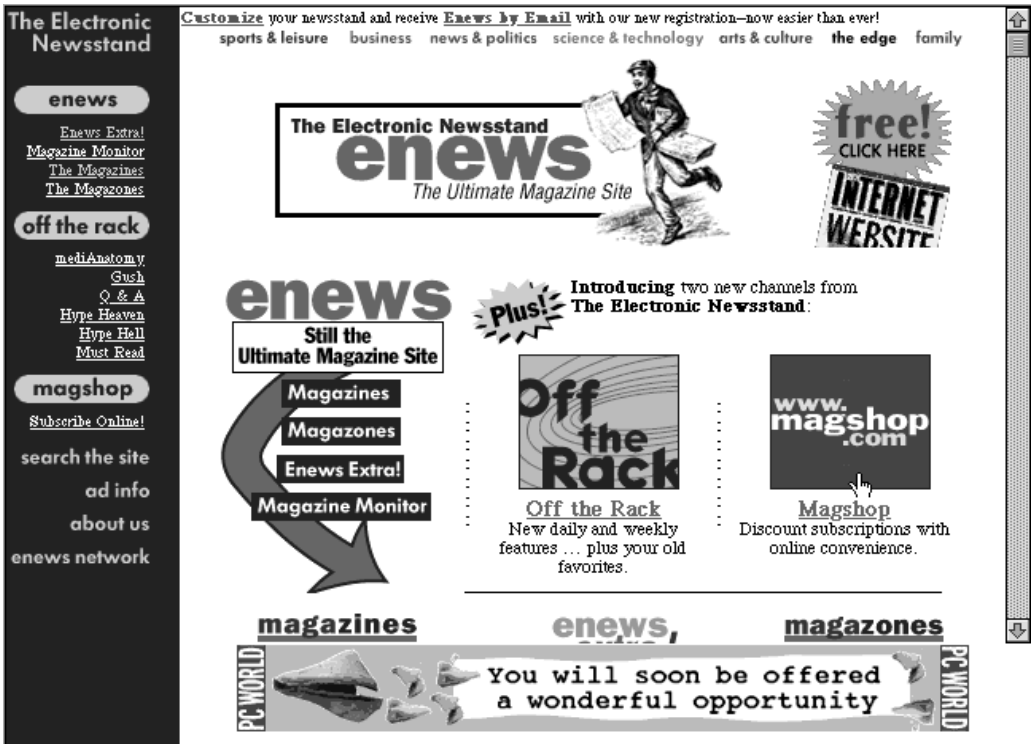
Heuristisessa arvioinnissa käytetään usein apuna Nielsenin ja Molichin kymmenen säännön listaa:

1	Käytä yksinkertaista ja luonnollista dialogia
2	Käytä käyttäjien omaa kieltä
3	Minimoi käyttäjän muistikuorma
4	Tee käyttöliittymästä kauttaaltaan yhdenmukainen
5	Anna käyttäjälle palautetta toiminnoista
6	Anna selkeä poistumistapa eri tiloista ja toiminnoista
7	Anna käyttäjälle mahdollisuus käyttää oikopolkuja
8	Anna virhetilanteista selkeät virheilmoitukset
9	Vältä virhetilanteita
10	Anna riittävä ja selkeä apu ja dokumentaatio

Nämä säännöt esitetään myös viitteessä (Nielsen 1993), johon tämä esitys paljolti perustuu. Koska ohjeet ovat yleisiä, niistä kannattaa luoda omaan järjestelmään räätälöity versio. Esimerkiksi aiemmissa järjestelmäversioissa havaitut ongelmat auttavat tarkentamaan sääntöjä. Nielsen (1994) on myöhemmin muokannut listaa, mutta kokonaisuudeltaan sisältö on vastaava.

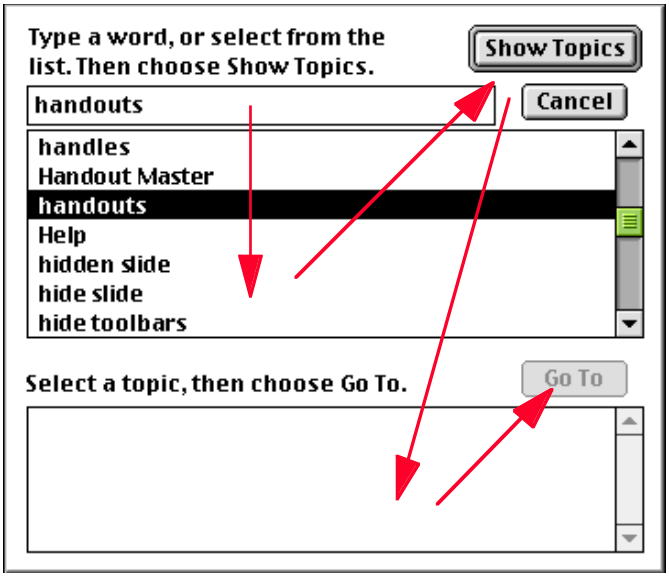
Yksinkertainen ja luonnollinen dialogi

Käyttöliittymän tulisi olla mahdollisimman yksinkertainen: siinä ei saisi olla mitään turhaa vaan ainoastaan senhetkiseen tehtävään tarvittava tieto. Kaikki tieto ja toiminnot, joita tarvitaan vain harvoin, pitäisi piilottaa käyttäjältä esimerkiksi erilliseen ikkunaan, alivalikkoon tai kaukosäätimessä pienen kannen alle. Näin käyttäjä löytää helposti ja nopeasti haluamansa perusasiat, koska hänen ei joka kerta tarvitse selata lukuisia yksityiskohtia läpi.



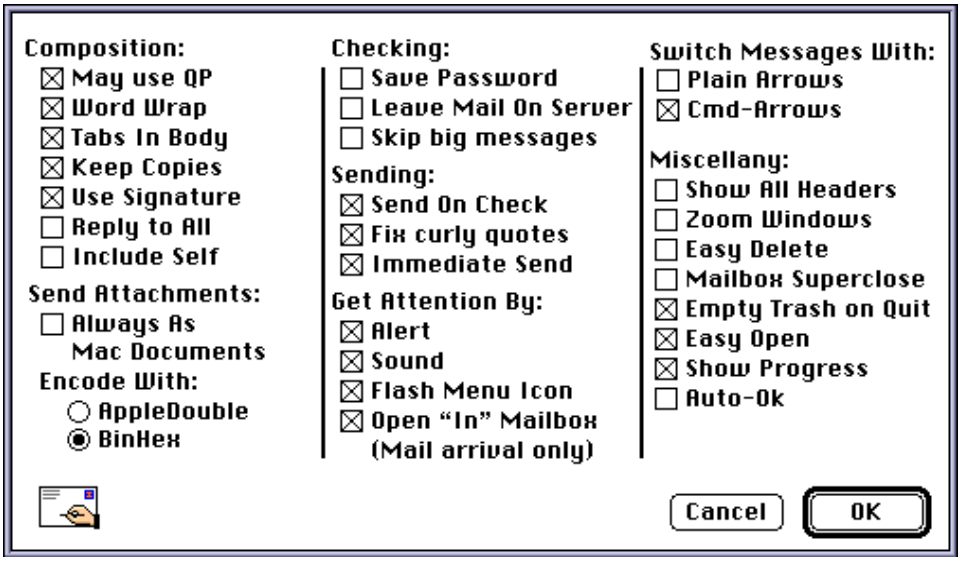
Kuva 2: Ikkunassa on niin paljon tavaraa, että olennaista tietoa ei löydy.

Käyttöliittymän luonnollisuus tarkoittaa muun muassa sitä, että esille tuodut asiat, niiden ryhmittely ja järjestys tukevat käyttäjän työtä. Ongelmia luultavasti ilmenee, mikäli käyttäjän oletetaan täyttävän lomakkeen kentät tietyssä järjestyksessä ja painavan tietyissä väleissä sopivia painikkeita. Käyttäjälle tulisi siis tarjota selkeä etenemistapa tehtäviensä tekoon, mutta mahdollistaa myös muita etenemistapoja.



Kuva 3: Painikkeiden sijoittelu ei tue tehtävän etenemistä.

Asioiden löytämistä helpottaa osien graafinen sommittelu. Osia voidaan ryhmitellä sijoittelulla, väreillä, muodoilla, koolla sekä tekstityypeillä . Mikäli tiettyyn tehtävään tarvittavat osat on ryhmitelty hyvin, käyttäjä löytää nopeasti tarvitsemansa osat.



Kuva 4: Tiedot on periaatteessa ryhmitelty, mutta ryhmät ja eri vaihtoehdot ovat niin samanlaisia, että lähes kaikki vaihtoehdot on käytävä läpi ennen kuin löytää haluamansa.

Käyttäjän huomiota voidaan ohjata esimerkiksi liikkuvilla, välkkyvillä, isoilla tai värikkäillä osilla. Niitä pitääkin käyttää harkiten. Jos käyttäjä näkee näytöllä vilkkuvan valon, hän yleensä olettaa, että häneltä odotetaan jonkinlaista reaktiota. Mikäli käyttäjän ei tarvitse tehdä mitään tai huomata esimerkiksi hälytystä, hänen huomionsa on käännetty turhaan pois itse asiasta ja keskittyminen herpaantuu. Myös värejä tulisi käyttää maltillisesti ja tyytyä muutamaan perusväriin koko käyttöliittymässä varsinkin normaaliin työskentelyyn tarkoitetuissa järjestelmissä.

Käyttäjien oma kieli

Käyttöliittymän kielen tulisi olla järjestelmän käyttäjän omaa äidinkieltä ja hänen omaa ammattisanastoaan. Käyttöliittymän viestit tulisi esittää aina käyttäjän kannalta ajateltuna, selkeästi ja turhaa tietokonesanastoa välttäen. Esimerkiksi termit "konfigurointi" ja "moodi" kuuluvat vain harvojen käyttäjien sanastoon ja kulutuselektroniikan puolella jopa sana "ohjelmointi" aiheuttaa hämmennystä ja tietynlaista vastahakoisuutta. Tekstin lisäksi kieli tarkoittaa käyttöliittymän grafiikkaa, kuten ikoneita.

Metaforien eli eräänlaisten vertauskuvien avulla käyttöliittymään voidaan luoda arkimaailmasta tuttuja piirteitä, jotka auttavat käyttäjiä soveltamaan olemassaolevia tietojaan ja taitojaan uuden järjestelmän käytössä. Komennot "Leikkaa" ja "Liitä" ovat esimerkkejä sanallisista metaforista ja kuva saksista esimerkki visuaalisesta metaforasta. Metaforat tulee valita huolella, sillä harhaan johtavat vertauskuvat vaikeuttavat järjestelmän oppimista ja käytön logiikan ymmärtämistä.

Käyttäjille tuttua sanastoa on vaikea kartoittaa kysymällä käyttäjiltä suoraan sopivia termejä, koska sopivin sana tulee harvoin käyttäjän mieleen. Hyväksi todettu tapa luoda sanasto on laatia järjestelmän kehittäjien, käytettävyysasiantuntijoiden ja muutaman käyttäjän ehdotusten pohjalta lista vaihtoehtoja, joista käyttäjät voivat äänestämällä valita sopivimmat (Nielsen 1993). Hyvä tapa tutustua käytettyyn sanastoon on myös haastatella tulevia käyttäjiä, nauhoittaa haastattelut ja poimia sieltä heidän käyttämänsä termit.

Käyttöliittymän perusosissa, kuten valikossa ja painikkeissa, on hyvä käyttää kyseisen käyttöjärjestelmän vakiintunutta sanastoa, jos poikkeuksiin ei ole erityisen hyvää syytä.

Käyttäjän muistikuorman minimointi

Tietokone on hyvä tallentamaan asioita, joten erilaisten tietojen muistaminen tulisi jättää koneelle eikä käyttäjälle. Käyttäjän muistikuormaa on esimerkiksi turhaa kuormittaa vaatimalla jo kertaalleen annetun tiedon muistamista järjestelmän muissa osissa.

Koska ihminen tunnistaa asioita paljon helpommin kuin muistaa ulkoa ilman vihjeitä, käyttäjälle pitäisi antaa vaihtoehtoja, joista hän voi valita. Esimerkiksi oikea tiedosto on helppo valita annetuista vaihtoehdoista, mutta sen nimen muistaminen ulkoa voi olla vaikeaa. Vastaavasti valikkojen käytöllä vähennetään käyttäjän tarvetta muistaa lukuisia eri komentoja syntakseineen.

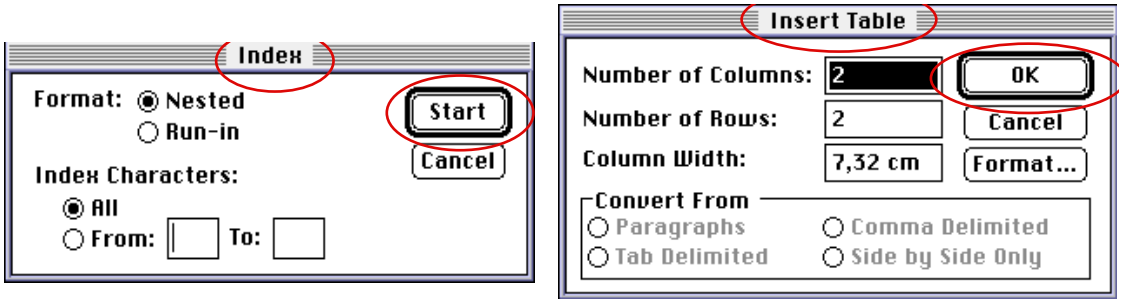
Kuva 5: Kuvatiedostot on helppo tunnistaa pienien kuvakkeiden ja nimien avulla.

Mikäli vaihtoehtoja ei pystytä rajaamaan, käyttäjän muistikuormaa voidaan vähentää antamalla käyttäjälle mallivastaus tai mahdollisuuksien mukaan oletusarvoinen vastaus. Näin käyttäjä näkee, missä muodossa vastaus tulisi antaa. Numeerisen vastauksen sallitut raja-arvot tulisi myös näyttää, samoin käytetty yksikkö.

Käyttäjän muistikuormaa voidaan vähentää myös noudattamalla tiettyjä sääntöjä johdonmukaisesti koko käyttöliittymässä. Tästä kerrotaan tarkemmin seuraavassa kohdassa.

Yhdenmukaisuus

Tietyn komennon tai valinnan tulisi toimia yhdenmukaisesti käyttöliittymän kaikissa osissa. Laitteissa esimerkiksi on tärkeää, että käyttäjällä on tietty painike tai toiminto, jolla hän pystyy palaamaan mistä tahansa tilasta alkutilaan. Tietojärjestelmissä puolestaan korostuu totuttujen käytäntöjen noudattaminen. Esimerkiksi OK-painikkeella käyttäjät ovat tottuneet tallentamaan muutokset ja sulkemaan ikkunan. Jos OK-painikkeen vieressä onkin erillinen Tallenna-painike, käyttäjä saattaa ihmetellä, eikö OK-painike tallennakaan tietoa. Hämmennystä aiheuttaa sekin, jos OK- ja Peruuta-painikkeilla palataan tilanteesta riippuen eri tiloihin: edelliseen tilaan, perustilaan tai johonkin niiden väliin.



Kuva 6: Erään tekstinkäsittelyohjelman komennoilla “Insert Index” ja “Insert Table” aukeavat ikkunat on nimetty eri tavoin, samoin ikkunoiden painikkeet, mikä aiheuttaa turhaa hämmennystä.

Käyttöliittymän yhdenmukaisuus tarkoittaa myös käyttöliittymän osien sijoittelua. Esimerkiksi useissa eri ikkunoissa esiintyvät osaset pitäisi sijoitella kaikissa ikkunoissa vastaaviin paikkoihin, samaan järjestykseen.

Kun samat toimintasäännöt pätevät koko käyttöliittymässä, käyttäjä osaa ja uskaltaa kokeilla järjestelmän uusiakin piirteitä oppimansa perusteella. Pidemmälle vietynä yhdenmukaisuus koskee järjestelmän sisäisen yhdenmukaisuuden lisäksi eri versioiden, saman käyttöjärjestelmän tai sovellusalan ja tietyn valmistajan eri tuotteiden välistä yhdenmukaisuutta. Jos tietyt perussäännöt pätevät järjestelmästä toiseen, järjestelmän oppiminen helpottuu ja käyttäjä suoriutuu yksinkertaisista perustehtävistä jo lyhyen opettelun jälkeen.

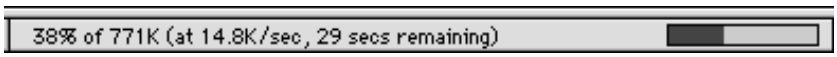
Yhdenmukaisuus on niin tärkeä ominaisuus, että joissain tapauksissa se päihittää jopa käytettävyyden. Mikäli käyttäjät ovat vuosikausia tottuneet tekmään asian sujuvasti tietyllä tavalla, toteutustapaa ei välttämättä kannata muuttaa esimerkiksi eri versioiden välillä, jos ongelma ei ole huomattava.

Riittävä palaute

Käyttäjän tulisi saada palautetta kaikista tekemistään valinnoista ja komennoista, jotta hän voisi olla varma, että järjestelmä on hyväksynyt annetun tiedon, käsitellyt sitä ja tuottanut halutun tuloksen. Mikäli käyttäjä ei mitenkään havaitse, että järjestelmä tekee jotain tai sai jo jotain valmiiksi, hän luultavasti olettaa, että järjestelmä ei toimi tai jotain meni pieleen. Tällaisissa tapauksissa tulee usein toistettua sama toiminto useaan kertaan - aivan turhaan. Järjestelmän tulisi siis kertoa käyttäjälle virhetilanteiden lisäksi kaikista muistakin toiminnoistaan.

Välitön palaute antaa käyttäjälle tunteen järjestelmän hallinnasta ja ohjauksesta. Myös oppiminen tehostuu välittömällä palautteella, koska käyttäjä osaa yhdistää tekemänsä valinnat ja toiminnot niiden aiheuttamiin tapahtumiin, jos niiden välinen viive on kyllin pieni.

Mikäli järjestelmä tuottaa halutun tuloksen alle 0,1 sekunnissa, käyttäjällä säilyy tunne välittömästä vuorovaikutuksesta. Tällöin erillistä palautetta ei tarvita, vaan pelkkä tuloksen esittäminen riittää palautteeksi. Erillistä palautetta ei välttämättä tarvita vielä yhden sekunnin viiveelläkään. Käyttäjä kylläkin huomaa tällaisen viiveen, mutta ajatus pysyy yleensä katkeamattomana. Tätä pidemmillä viiveillä käyttäjälle pitäisi antaa erillinen palaute merkiksi siitä, että jotain on tapahtumassa ja vähintään yli kymmenen sekuntia kestävästä toiminnosta antaa arvio sen kestosta (Nielsen 1993). Toiminnon kestoa on usein vaikea arvioida minuutteina ja sekunteina, joten se voidaan esittää suhteellisesti esimerkiksi toiminnon edistyessä täyttyvällä palkilla.



Kuva 7: Toiminnon arvioitu kesto on ilmoitettu hyvin ikkunan alareunassa.

Selkeä poistumistapa eri tiloista ja toiminnoista

Jotta käyttäjä uskaltaisi kokeilla käyttöliittymän eri toimintoja, hänelle pitäisi antaa joka toiminnosta keino palata takaisin edelliseen tilaan tai peräti pois koko järjestelmästä. Palaamisella edelliseen tilaan tarkoitetaan esimerkiksi toiminnon perumista "Kumoa"-komennon avulla tai poistumista dialogi-ikkunasta "Peruuta"-painikkeella. Jos jollekin toiminnolle tai huomattavalle muutokselle ei pystytä tarjoamaan vastaavaa kumoavaa toimintoa, tämä on hyvä ilmoittaa käyttäjälle ennen toiminnon aloitusta. Tiedoston poistossa tai vastaavissa toiminnoissa käyttäjältä on syytä pyytää vahvistus toiminnolleen.

Oikopolut

Kun käyttäjä on oppinut käyttämään järjestelmää, hän haluaa usein hyödyntää sitä mahdollisimman nopeasti ja tehokkaasti. Tällaisille asiantuntijakäyttäjille tulisi tarjota oikopolkuja eri toimintoihin ja valintoihin. Tyypillisiä oikopolkuja ovat komentojen lyhenteet, hiiren eri nappuloiden ja kaksoisnäpäytyksen käyttö sekä toimintonäppäimet ja erilaiset näppäinyhdistelmät.

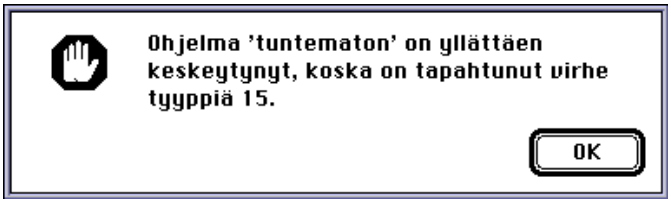
Kuva 8: Valikossa esitetään vastaavat näppäinkomennot oikopoluksi.

Oikopolut eivät saa olla ainoa keino tehdä jokin toiminto, vaan niiden tulee olla vaihtoehtona jollekin helppokäyttöiselle ja selkeästi merkitylle käyttötavalle. Järjestelmässä tarvittavat komennot tulisi esittää näkyvästi esimerkiksi valikossa, jossa voidaan antaa myös komentoa vastaava pikanäppäin. Valittuaan komennon tarpeeksi usein valikosta käyttäjä hiljalleen oppii vastaavan pikanäppäimen ja saattaa ruveta käyttämään sitä.

Järjestelmän käyttöä voidaan tehostaa myös tarjoamalla käyttäjälle oletusarvoja, kuten kuluvaa päivää päiväykseksi. Koska käyttäjä tekee usein samantyyppisiä toimintoja peräkkäin, hänen toimintaansa voidaan tehostaa antamalla aiemmin käytetyt komennot uudelleen käytettäväksi. Vastaavasti käyttäjälle voidaan tarjota valikossa muutama hänen aiemmin käsittelemänsä tiedosto valmiina vaihtoehtoina, koska käyttäjä käsittelee usein samoja tiedostoja lyhyen ajan sisällä.

Selkeät virheilmoitukset

Järjestelmän tulee antaa käyttäjälle virhetilanteista mahdollisimman selkeä ja tarkka ilmoitus käyttäjän ymmärtämällä kielellä ja sanastolla. Ilmoituksen tulee olla kohtelias eikä käyttäjää syyttävä. Ilmoituksen tulisi myös ohjata käyttäjää korjaamaan virheen.



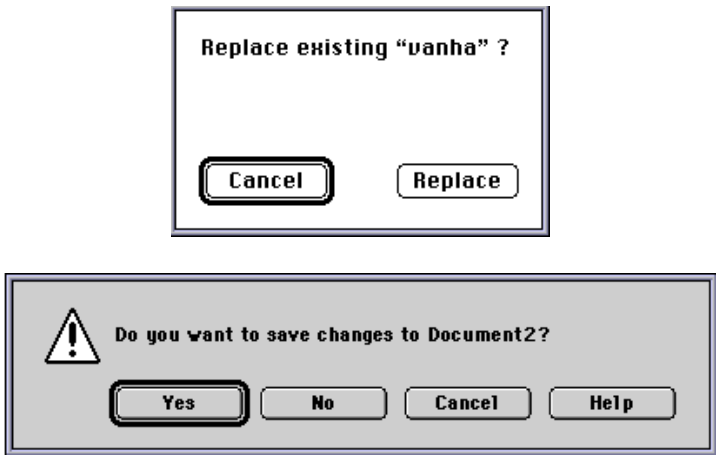
Kuva 9: Tämä virheilmoitus ei anna tavalliselle käyttäjälle mitään tietoa.

Koska eri käyttäjät kaipaavat eritasoisia virheilmoituksia, virheilmoitukset on hyvä esittää tasoittain tarkentuen. Oletusarvoisesti näytetään esimerkiksi vain virheen syy ja korjausehdotus selkokielellä ja ylläpitäjien ja huoltajien kaipaamat koodit vain erillisestä pyynnöstä.

Virhetilanteiden välttäminen

Hyviä virheilmoituksia parempi vaihtoehto on järjestelmä, jossa virhetilanteet vältetään. Kirjoitusvirheet komentojen ja tiedostonimien osalta ovat esimerkiksi vältettävissä antamalla käyttäjälle valmiit vaihtoehdot valikossa ja listoissa.

Virhetilanteita vältetään myös pyytämällä käyttäjältä varmistus sellaisiin toimintoihin, joita ei pystytä kumoamaan, kuten tiedoston tuhoamiseen.



Kuva 10: Käyttäjältä on hyvä varmistaa, haluaako hän todella kirjoittaa uuden tiedoston vanhan päälle tai poistua tiedostosta tallentamatta muutoksia.

Myös järjestelmän eri toimintatilat ovat virheille alttiita. Jos käytössä oleva toimintatila on huonosti esitetty, käyttäjä voi tehdä toimintoja, jotka ovat jossain toisessa tilassa hyväksyttäviä, mutta eivät aiheuta senhetkisessä tilassa ainakaan haluttua toimintoa. Toimintatila voidaan esittää vaikkapa kohdistimen vaihtumisella, värikoodein tai tilan mukaan vaihtuvilla äänillä.

Riittävä ja selkeä apu ja dokumentaatio

Järjestelmän tulisi olla sellainen, että sitä pystyy käyttämään ilman erillisiä ohjeita. Tähän päästään vain harvoin, joten järjestelmän mukana tulisi tarjota erityyppistä tukea. Järjestelmän mukana kulkeutuu helpoimmin järjestelmän oma opasteohjelma. Eräs tapa toteuttaa kontekstisidonnainen eli tilanteen mukaan vaihtuva ohje on puhekuplaohjeiden käyttö. Niiden avulla käyttäjälle pystytään antamaan tietoa juuri siitä käyttöliittymän osasta, jota käyttäjä sillä hetkellä osoittaa.

Sekä järjestelmän opasteohjelma että paperiversio tulisi olla jäsennelty siten, että asiat löytyvät niistä helposti ja nopeasti. Hyvä sisällysluettelo ja kattava hakemisto, jossa samaan asiaan on useita eri hakusanoja, ovat tässä eräitä perustekijöitä. Ohjeet tulisi esittää käyttäjän tehtävien mukaan edeten kuvaavia esimerkkejä käyttäen. Lisäksi ohjeiden eri osat tulisi laatia mahdollisimman itsenäisiksi eli ohjeet eivät saisi juurikaan nojautua toisaalla esitettyyn tietoon.

Ohjeissa tulisi muistaa myös sääntö: “Käytä käyttäjien omaa kieltä.” Esimerkiksi erään toimistopuhelimen käyttöohje pikavalinnan tallentamiselle: “`Paina haluamaasi pikavalintanäppäintä. Jos lisänäppäimistön 2. taso on aktivoitu, paina ensin “Shift”-näppäintä`”, ei tue käyttäjää, kun hän yrittää selvittää, miksi pikavalinnan tallennus ei onnistu. “Lisänäppäimistö”, “2.taso”, “aktivoitu” ja “Shift-

näppäin” ovat outoja termejä käyttäjälle, joka ei välttämättä edes tiedä, että hänen puhelimeensa voitaisiin liittää lisänäppäimistö.

1.2.5 Millaisia ongelmia heuristisella arvioinnilla löytyy?

Heuristisella arvioinnilla löydetään sekä vakavia että pienempiä käytettävyysongelmia. Sen avulla löytyy suhteellisesti enemmän pieniä ongelmia kuin esimerkiksi käytettävyystesteissä. Vaikka osa löydetyistä ongelmista on vain kosmeettisia, nekin vaikuttavat järjestelmän yleisilmeeseen ja sen käytön miellyttävyyteen.

Eri osien vertailu, kuten eri ikkunoiden yhdenmukaisuuden tarkistus, on helpompaa paperiversioilla kuin toimivalla versiolla. Sen sijaan puutteita, kuten toimintoja, joita järjestelmässä olettaisi olevan, mutta ei ole, on vaikeampi huomata paperiversioita tarkastellessa kuin toimivaa versiota käytettäessä.

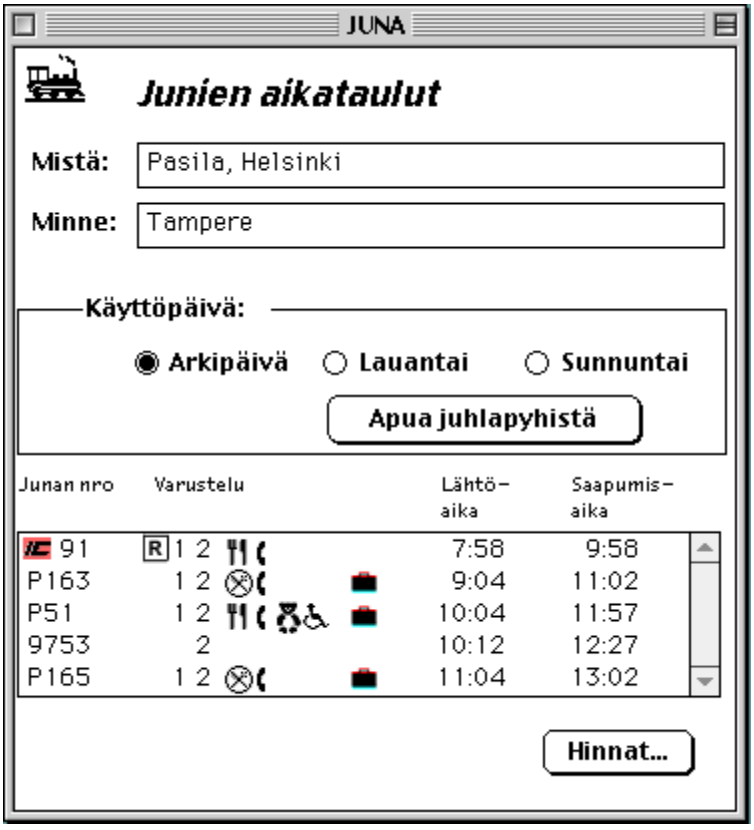
1.2.6 Esimerkki

Tarkastellaan esimerkkinä suunnitelmaa junien aikataulun selausärjestelmästä, joka on tarkoitettu toimisto- ja kotikäyttöön.

Kun järjestelmä käynnistetään, kuvassa esitetty pääikkuna aukeaa. Kentät "Mistä" ja "Minne" ovat tällöin tyhjiä. Kenttiin tulee kirjoittaa aseman nimi ja kaupunki. Jos kaupungissa on vain yksi asema kyseisellä reitillä, pelkkä kaupungin nimi riittää. Kun käyttäjä on täyttänyt kentät ja painaa joko Enteriä tai Returnia tai klikkaa hiirellä kenttien ulkopuolella, järjestelmä etsii valintalistaan kyseisen reitin aikataulun. Tätä ennen lista on ollut tyhjä. Jos aseman tai kaupungin nimeä ei löydy järjestelmästä, käyttäjälle annetaan dialogi-ikkunassa virheilmoitus: "Tuntematon asema" tai "Tuntematon kaupunki".

Käyttöpäiväksi on oletusarvoisesti valittu arkipäivä. Painikkeesta "Apua juhlapyhistä" aukeaa dialogi-ikkuna, jossa on lueteltu juhlapyhät ja kerrottu, noudattavatko junat tällöin lauantain vai sunnuntain aikatauluja. Dialogi-ikkuna on suljettava OK-painikkeella ennen kuin järjestelmän käyttöä voidaan jatkaa.

"Hinnat…"-painikkeesta avautuu ikkuna, jossa kerrotaan lippujen hinnoista.



Kuva 11: Heuristisessa arvioinnissa tutkittavan ohjelman käyttöliittymä

Tarkastellaan järjestelmää heuristiikka kerrallaan:

1. Yksinkertainen ja luonnollinen dialogi

- "Hae"-painike puuttuu
- Käyttöpäivä vie suhteettoman ison tilan dialogista
- “Junien aikataulut” voisi olla ikkunan otsikkona eikä erillisenä otsikkona
- Junan kuva on tarpeeton
- Aikatauluissa esitetään melko epäoleellista tietoa junista, kuten junan nro, ja oleellisin tieto eli lähtö- ja saapumisaika on viimeisenä.
- Osien sommittelu on viimeistelemätön

2. Käyttäjän kieli

- Termit “Käyttöpäivä” ja “Varustelu” ovat melko outoja.
- Termien “Mistä” ja “Minne” sijaan voisi puhua esimerkiksi lähtö- ja pääteasemasta
- Varustelutasoa esittävät symbolit ovat osittain epäselviä

3. Muistikuorman minimointi

- Asemien nimet voisi tarjota valintalistana, jotta käyttäjän ei tarvitse muistaa niitä. Jos valintalistaa ei voida tehdä, olisi tarjottava edes mallivastaus, josta näkee, missä järjestyksessä kaupunki ja asema on annettava.
- “Apua juhlapyhistä” -ikkuna on suljettava ennen kuin tehtävää voi jatkaa. Käyttäjän pitää siis itse muistaa, millä aikataululla juna kulkee kyseisenä päivänä, ja tehtävä tämän mukainen valinta käyttöpäivään.

4. Yhdenmukaisuus

- Ikkunasta puuttuvat normaalit hyväksymis- ja poistumispainikkeet sekä aputoiminto. Hakuun olisi myös syytä olla erillinen painike, kuten monissa muissa sovelluksissa.

5. Palaute

- Käyttäjä ei saa palautetta asemien kirjoittamisen jälkeen, jos hän ei paina Enteriä, Returnia tai aktivoi hiirellä jotain tekstikenttien ulkopuolista osaa.

6. Selkeä poistuminen

- Ikkunasta puuttuu erillinen poistumispainike

7. Oikopolut

- Toiminnoille ei tarjota oikopolkuja
- Matkustuspäivän voisi antaa myös suoraan päiväyksenä, jolloin järjestelmä voisi itse tutkia, mitä aikataulua junat tällöin noudattavat

8. Selkeät virheilmoitukset

- Virheilmoitukseen “Tuntematon asema” tai “Tuntematon kaupunki” tulisi liittää käyttäjän kirjoittama nimi. Näin mahdolliset kirjoitusvirhet olisi helppo huomata.

9. Virhetilanteiden välttäminen

- Valintalistat asemista ja kaupungeista poistaisivat kirjoitusvirheet
- Päiväyksen antaminen vähentäisi väärinkäsityksiä kulloinkin noudatetusta aikataulusta

10. Selkeät ohjeet

- Ohjeita ei ole

Löydetyt ongelmat on hyvä koota yhdeksi listaksi ja priorisoida ne. Yksi selkeä tapa on esittää tutkittu ikkuna ja havaitut ongelmat taulukkona samalla sivulla mielellään vielä parannusehdotusten kanssa. Näin ongelmat on helppo ymmärtää, samoin parannusehdotukset. Taulukko 1 esittää osan edellä mainituista ongelmista.

Taulukko 1: Ongelmien yhteenveto ja priorisointi (vakava/ häiritsevä/ kosmeettinen)

Ongelma	Heuristiikat	Vakavuus	Parannusehdotus
Käyttäjän pitää kirjoittaa aseman ja kaupungin nimi	Muistikuorma, virheiden välttäminen	Vakava	Valintalistat
Ohjeita ei ole	Selkeät ohjeet	Vakava	Lisätään ohjeet ja “Ohjeet”-painike
Termi“Käyttöpäivä” on melko outo	Käyttäjien kieli	Häiritsevä	“Lähtöpäivä” tai “Matkustuspäivä”
Osien sommittelu on viimeistelemätön	Yksinkertainen ja luonnollinen dialogi	Kosmeettinen	Kenttien reunat tulisi tasata ja painikkeet sijoitella yhtenäisemmin

1.3 Kognitiivinen läpikäynti

Kognitiivisessa läpikäynnissä (Wharton et al. 1994) pyritään ottamaan huomioon ihmisen ajattelutapa ja tapa opetella uusia asioita. Menetelmällä arvioidaan järjestelmän opittavuutta eli sitä, kuinka helppoa järjestelmää on käyttää ensimmäisillä kerroilla. Koska monet käyttäjät opettelevat käyttämään järjestelmiä ilman käyttöohjeita, kognitiivinen läpikäynti keskittyy opetteluun kokeilemalla. Menetelmä keskittyy oppimisen arviointiin siinä määrin, että muut käytettävyystekijät, kuten tehokkuus ja miellyttävyys, jäävät huomiotta. Menetelmää ei olekaan tarkoitus käyttää ainoana arviointikeinona vaan täydentämään muita menetelmiä.

Kognitiivisessa läpikäynnissä käydään esimerkiksi paperiversioiden ja järjestelmäkuvauksen tai prototyypin avulla sovelluksen keskeisimmät tehtävät läpi vaihe vaiheelta. Menetelmää voidaan soveltaa siis jo tuotekehityksen alkuvaiheessa.

1.3.1 Kognitiivisen läpikäynnin vaiheet

Kognitiivinen läpikäynti jakaantuu viiteen vaiheeseen, joista kaksi ensimmäistä kuuluu alkuvalmisteluihin ja loput itse analyysiin:

1. Esiselvitys
2. Ryhmän kokoaminen
3. Kognitiivinen läpikäynti istunnossa
4. Havaintojen tallentaminen
5. Havaittujen virheiden ja puutteiden korjaaminen

Esiselvitys

Ennen läpikäyntiä tulee selvittää, ketkä ovat järjestelmän tyypillisimpiä käyttäjiä. Käyttäjistä tulisi pohtia, millainen kokemus heillä on vastaavien järjestelmien käytöstä ja millainen tekninen osaaminen heillä on.

Läpikäynnissä tutkittavat tehtävät tulisi valita markkinatutkimusten, vaatimusmäärittelyjen ja aiemmin havaittujen ongelmakohtien perusteella. Järjestelmän keskeisimmät perustehtävät tulisi käydä läpi, samoin kuin jokin hieman monimutkaisempi tehtävä, joka vaatii näiden perustehtävien yhdistelyä. Tehtävien ratkaisu pitää miettiä vaihe vaiheelta käyttäjien osaamistason vaatimalla tarkkuudella: kokeneille käyttäjille ratkaisuun ei tarvitse sisällyttää, miten komento valitaan valikosta, mutta aloitteleville käyttäjille tämäkin voi olla tarpeen.

Kognitiiviseen läpikäyntiin tarvitaan melko tarkka kuvaus järjestelmän käyttöliittymästä. Prototyyppi tai järjestelmäkuvaus käyttöliittymästä tehtyjen suunnitelmien kanssa soveltuvat hyvin läpikäyntiin.

Ryhmän kokoaminen

Arvioinnin voi tehdä ryhmässsä tai yksinään. Parhaat tulokset saadaan, jos työ tehdään ryhmässä, johon kuuluu vähintään yksi kognitiotieteen perustiedot hallitseva henkilö. Kognitiotieteen hallinta ei kuitenkaan ole välttämättömyys, sillä hyviä tuloksia on saatu ohjelmistosuunnittelijan yksinäänkin suorittamilla läpikäynneillä.

Ryhmään on hyvä kerätä henkilökuntaa eri osastoilta: markkinointi-, suunnittelu-, ohjelmisto-, dokumentointi- ja koulutuspuolelta. Yksi käyttöliittymien arviointiin erikoistunut henkilö täydentää ryhmän osaamista sopivasti.

Tehtävien läpikäynti istunnossa

Läpikäynnissä tutkitaan valitut tehtävät vaihe vaiheelta. Tehtävien suoritusta tarkastellaan ns. oikean ratkaisun mukaisesti eli sen mukaan, miten järjestelmän suunnittelija on tehtävän ajatellut suoritettavaksi. Havaitut ongelmat kirjataan muistiin, mutta tehtävää jatketaan aivan kuin käyttäjä olisi toiminut siihen asti oikein.

Kunkin tehtävän osalta pohditaan seuraavia kysymyksiä:

1	Onko käyttäjällä järjestelmän kannalta oikea tavoite?
2	Löytääkö hän järjestelmästä oikean toiminnon?
3	Yhdistääkö hän kyseisen toiminnon tavoitteeseensa?
4	Mikäli oikea toiminto on suoritettu, saako käyttäjä riittävästi palautetta tehtävän etenemisestä?

Käyttäjällä voi olla väärä tavoite siksi, että kyseinen vaihe ei ole luonnollinen osa hänen tehtäväänsä. Esimerkiksi oikean toimintotilan valitseminen ennen varsinaista tehtävää saattaa aiheuttaa ongelmia.

Oikea toiminto löytyy helposti, jos sille on oma komento valikossa tai esimerkiksi ikoni tai painike näytöllä. Näppäinyhdistelmät, joita ei ole selkeästi kerrottu käyttöliittymässä, ovat tyypillisiä ongelmia oikean toiminnon havaitsemisessa.

Hyvin näkyvillä olevaa valintaa tai komentoa ei välttämättä yhdistä tavoitteeseensa, mikäli siinä käytetyt sanat tai ikonit eivät ole käyttäjälle tuttuja. Tässä vaiheessa pitäisi siis tuntea käyttäjien oma sanasto ja toimintaympäristö.

Kun käyttäjä on saanut suoritettua oikean toiminnon, hän kaipaa vielä jonkinlaista varmistusta toiminnon onnistumisesta. Selkeä palaute tai edes jokin merkki toiminnon suorituksesta on tässä vaiheessa avainasemassa.

Havaintojen tallentaminen

Läpikäynnin aikana koko ryhmän näkyvillä olevat taulut, lehtiöt tai kalvot ovat parhaita tallennusvälineitä. Jälkitarkastelua varten istunto on hyvä myös videoida.

Läpikäynnistä tulisi tallentaa ainakin seuraavat tiedot:

- oletukset käyttäjistä ja heidän taidoistaan
- uskottava tarina, joka kertoo, miten käyttäjä suoriutuu tehtävästä vaihe vaiheelta
- käyttäjän tarvitsemat tiedot: mitä tulee tietää ennen tehtävää ja mitä opittava sen aikana
- läpikäynnissä esiin tulleet ideat ja parannusehdotukset

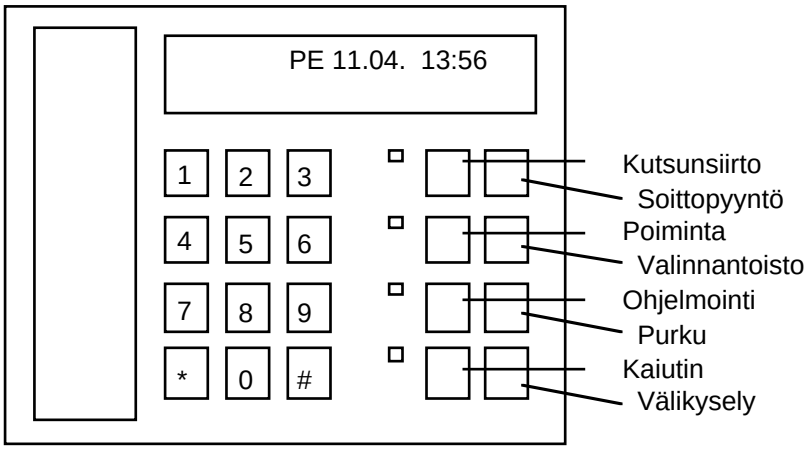
Ongelmien korjaaminen

Havaittuja puutteita ja ongelmia voidaan korjata esimerkiksi poistamalla käyttäjän näkökulmasta ylimääräiset vaiheet tai antamalla vaiheisiin paremmat ohjeet. Huonosti löydetyt toiminnot pitää tuoda paremmin esille esimerkiksi omiksi painikkeikseen tai valinnaksi valikkoon. Harhaanjohtavat sanat ja ikonit sekä vieraat termit on muutettava käyttäjien sanaston mukaisiksi, heidän tehtäviään tukeviksi. Palautetta on lisättävä tai täsmennettävä, jos tehtävän etenemistä ei havaittu tarpeeksi selkeästi.

1.3.2 Esimerkki

Kognitiiviseen läpikäyntiin on hyvä valita tarpeeksi pieniä tehtäviä. Erikoistoiminteiden teko toimistopuhelimella on sopivan kokoinen tehtävä läpikäyntiin. Tässä käydään läpi kutsunsiirron tekeminen.

Tutkittava järjestelmä:	Toimistopuhelin
Käyttäjä:	Tutkija Teknillisessä korkeakoulussa
Taidot:	Osaa soittaa puhelimella ja vastata puheluihin, mutta ei juurikaan muuta
Skenaario:	Tutkija siirtyy työskentelemään muutamaksi tunniksi toiseen huoneeseen, jonka puhelinnumero on 451 3397
Tehtävä:	Ohjata saapuvat puhelut toiseen huoneeseen
Käyttöliittymä:	Kuva 12



Kuva 12: Kognitiivisessa läpikäynnissä tarkasteltava puhelin

- kahden rivin LCD-näyttö, jossa musta teksti rusehtavalla pohjalla
- painikkeet:
  `0-9`, `*`, #, '`Soittopyyntö`', '`Valinnantoisto`', '`Purku`' ja '`Välikysely`'
- painikkeet, joiden vieressä punainen merkkivalo:
  '`Kutsunsiirto`', '`Poiminta`', '`Ohjelmointi`' ja '`Kaiutin`'

Oikea toimintosarja:

1. paina '`Ohjelmointi`'-painiketta
2. paina '`Kutsunsiirto`'-painiketta
3. anna alanumero `3397`
4. paina lopuksi '`Ohjelmointi`'-painiketta

Tehtävä käydään läpi vaihe vaiheelta kysyen joka välissä:

?1 onko käyttäjällä oikea tavoite?

?2 löytääkö hän järjestelmästä oikean toiminnon?

?3 yhdistääkö hän kyseisen toiminnon tavoitteseensa?

?4 mikäli oikea toiminto on suoritettu, saako käyttäjä riittävästi palautetta tehtävän etenemisestä?

Vaiheeseen kirjataan myös, mitä käyttöliittymässä tapahtuu vaiheen aikana.

1. Paina '`Ohjelmointi`'-painiketta

Puhelimen palaute:
'`Ohjelmointi`'-merkkivalo syttyy ja näyttöön tulee teksti: "`Valitse painike tai anna tunnus`"

Tavoite?	Ongelma	ohjelmointi ei ole luonnollinen osa käyttäjän tehtävää
Löytyykö?	OK	'`Ohjelmointi`'-painike on hyvin näkyvillä
Ymmärtääkö?	OK?	Termi '`Ohjelmointi`' on OK, jos käyttäjä ymmärtää tehtävän vaativan ohjelmointia
Palaute?	OK	puhelin antaa palautteena ohjeen, miten edetä

2. Paina ‘`Kutsunsiirto`’-painiketta
Puhelimen palaute:
‘`Ohjelmointi`’-merkkivalo palaa edelleen ja näyttöön tulee teksti:
“`Kutsunsiirto`”

Tavoite?	OK	puhelimen näyttö neuvoo valitsemaan jonkin painikkeen
Löytyykö?	OK	`Kutsunsiirto`’-painike hyvin näkyvillä
Ymmärtääkö?	Ongelma	kutsunsiirto ei välttämättä ole käyttäjälle selkeä termi. Muut näkyvät vaihtoehdot ovat kuitenkin huonompia.
Palaute?	Kenties ongelma	ainoa palaute toiminnosta on melko tummasävyisessä näytössämuuttuva teksti ja huomiota enemmän kiinnittävät valot eivät vaihdu

3. Anna alanumero
Puhelimen palaute:
‘`Ohjelmointi`’-merkkivalo palaa edelleen ja näyttöön tulee teksti:
“`Kutsunsiirto 3397 Käytettävyyslab`”

Tavoite?	Kenties ongelma	puhelin ei kehoita antamaan numeroa
Löytyykö?	OK	numeropainikkeet hyvin näkyvillä
Ymmärtääkö?	OK?	numeron näppäily luonnollista, mutta ei välttämättä selvää, että annetaan vain alanumero
Palaute?	OK	näyttöön tuli annettu numero ja kyseisen numeron haltijan nimi

4. Paina ‘`Ohjelmointi`’-painiketta
Puhelimen palaute:
‘`Ohjelmointi`’-merkkivalo sammuu, ‘`Kutsunsiirto`’ syttyy ja näyttöön tulee teksti: “`Kutsunsiirto tallennettu`”. Noin 5 sekunnin päästä teksti muuttuu: “`Kutsunsiirto 3397 Käytettävyyslab`”

Tavoite?	Ongelma	käyttäjä luulee jo päättäneensä tehtävän
Löytyykö?	OK	‘`Ohjelmointi`’-painike on hyvin näkyvillä ja sitä on käytetty jo aiemmin
Ymmärtääkö?	Kenties ongelma	‘`Ohjelmointi`’-painikkeen valinta ei välttämättä luonnollista, mutta painikkeen vieressä palava valo ohjaa melko hyvin
Palaute?	OK	merkkivalot ja näytön teksti antavat riittävän palautteen

Taulukko 2: Kognitiivisen läpikäynnin tulokset tiivistettynä.

Tehtävä	Oikea tavoite?	Löytyykö?	Yhdistääkö tehtäväänsä?	Riittävä palaute?
Paina '`Ohjelmointi`'-painiketta	Ongelma: ohjelmointi ei kuulu käyttäjän tehtävään	OK	OK, jos ymmärtää ohjelmoinnin tehtävän osaksi	OK
Paina '`Kutsunsiirto`'-painiketta	OK	OK	Ongelma: termi on käyttäjälle vieras	Ongelma? vain teksti muuttuu, valot eivät
Anna alanumero	Ongelma? Puhelin ei pyydä numeroa	OK	Ongelma? Huomaako käyttäjä antaa pelkän alanumeron?	OK
Paina '`Ohjelmointi`'-painiketta	Ongelma: käyttäjä luulee jo päättäneensä tehtävän	OK	Ongelma? `Ohjelmointi` ei ole luonnollinen tapa päättää tehtävä, mutta valot ohjaavat valintaan	OK

Kognitiivisen läpikäynnin tuloksena voidaan suositella tehtävän vaiheiden vähentämistä. Suositeltavin tapa olisi aloittaa tehtävä suoraan '`Kutsunsiirto`'-painikkeesta. Tällöin vaihe 1 jäisi pois.

Vaiheen 2 palautetta voitaisiin parantaa antamalla näyttöön teksti: "`Anna puhelinnumero`". Tällöin vaiheen 3 suurin ongelma tulisi myös korjattua.

Vaihe 4 olisi hyvä saada kokonaan pois tehtävästä, sillä se on käyttäjän kannalta ylimääräinen. Mikäli järjestelmä kuitenkin tarvitsee numeron päätteeksi jonkin merkin, se voidaan mainita jo vaiheen 2 palautteessa, kuten: "`Anna numero ja lopuksi *`".

Ongelmana tässä ehdotuksessa on se, että pelkästä 'Kutsunsiirto'-painikkeen painalluksesta käynnistyy puhelujen siirto ennalta annettuun numeroon. Tällaista toiminnetta kaipaa silloin, jos haluaa toistuvasti kääntää puhelunsa samaan numeroon. Kognitiivisen läpikäynnin tarkoitus ei kuitenkaan ole arvioida näiden toiminteiden tärkeysjärjestystä, vaan todeta mahdolliset käytön ongelmat eri toteutustavoissa.

Lähteet

Nielsen, J., Molich, R. 1990, Improving a Human-Computer Dialogue. Communications of the ACM, 33 (1990) 3 (March), s. 338-348.

Nielsen, J. 1993, Usability Engineering. Academic Press, 1993, ISBN 0-12-518405-0, 358 s.

Nielsen, J. 1994, Heuristic Evaluation. Teoksessa Usability Inspection Methods, toim. Nielsen, J. & Mack, R.L. John Wiley & Sons, Inc., 1994, ISBN 0-471-01877-5, s. 25-62.

Shneiderman, B. 1992, Designing the user interface: Strategies for effective human-computer interaction, 2. painos, Addison-Wesley, 1992, ISBN 0-201-57286-9, s. 72-74.

Smith, S. L., Mosier, J. N. 1986, Design Guidelines for Designing User Interface Software. Technical Report MTR-10090, The MITRE Corporation, Bedford, MA 01730, USA.
Saatavilla myös osoitteesta ftp://ftp.cis.ohio-state.edu/pub/hci/Guidelines/

Wharton, C., Rieman, J., Lewis, C. & Polson 1994, P. The Cognitive Walkthrough Method: A Practitioner's Guide. Teoksessa Usability Inspection Methods, toim. Nielsen, J. & Mack, R.L. John Wiley & Sons, Inc., 1994, ISBN 0-471-01877-5, s. 105-140.