Esa Hiiva

# MATKAPUHELINSOVELLUKSEN MUUNTAMINEN WINDOWS-PÄÄTELAITTEELLE

Opinnäytetyö

Tietotekniikan koulutusohjelma

Huhtikuu 2009

MIKKELIN AMMATTIKORKEAKOULU

Mikkeli University of Applied Sciences

# KUVAILULEHTI

| | |
|---|---|
| MIKKELIN AMMATTIKORKEAKOULU<br>Mikkeli University of Applied Sciences | **Opinnäytetyön päivämäärä**<br>**28.4.2009** |
| **Tekijä(t)**<br>Esa Hiiva | **Koulutusohjelma ja suuntautuminen**<br>**Tietotekniikan koulutusohjelma**<br>Tietokone- ja ohjelmistotekniikan suuntautuminen |

**Nimeke**
Matkapuhelinsovelluksen muuntaminen Windows-päätelaitteelle

**Tiivistelmä**


Tässä opinnäytetyössä tutkittiin ongelmia joita ilmenee matkapuhelinsovelluksen muuntamisessa Windows-päätelaitteelle. Tutkimuksessa keskityttiin käyttöliittymänäkökulmaan. Opinnäytetyön toimeksiantajana oli MHG Systems Oy. Opinnäytetyöhön kuului MHG Systems Oy:n matkapuhelinasiakasohjelman muuntaminen Windows-päätelaitteessa toimivaksi asiakasohjelmaksi. Toteutetulle Windows-asiakasohjelmalle kirjoitettiin myös käyttöohje ja suunniteltiin asennuspaketti jakelua varten. Windows-asiakasohjelman toteutuksessa keskityttiin käyttöliittymän helppokäyttöisyyteen erilaisilla työasemilla.

Muunnettava matkapuhelinsovellus on ohjelmoitu Java Micro Edition (Java ME) ohjelmointikielellä ja siitä muunnettavan Windows-asiakasohjelman ohjelmointikieleksi valittiin Java Standard Edition (Java SE). Yrityksen palvelinohjelmisto on ohjelmoitu käyttäen Java Enterprise Edition (Java EE) ohjelmointikieltä. Windows-asiakasohjelman käyttöliittymä toteutettiin käyttäen Swing-käyttöliittymäkirjastoa ja hyödyntäen Swing Application Frameworkin tarjoamaa valmista sovellusrunkoa.

Opinnäytetyön osana suunniteltu ja toteutettu Windows-asiakasohjelma täytti sovellukselle asetetut vaatimukset ja julkaistiin laajentamaan MHG Systems Oy:n palveluvalikoimaa. MHG Systems Oy hyötyy sovelluksesta voidessaan tarjota uutta palvelua asiakkailleen ja asiakkaat hyötyvät sovelluksesta voidessaan käyttää asiakasohjelmaa matkapuhelinsovellusta helpommin ja useammilla erilaisilla päätelaitteilla.



**Asiasanat (avainsanat)**
ohjelmointi, Java, J2SE, J2ME, käyttöliittymät

| **Sivumäärä** | **Kieli** | **URN** |
|---|---|---|
| 37 s. | Suomi | URN:NBN:fi:mamk-opinn200935758 |


**Huomautus (huomautukset liitteistä)**

| **Ohjaavan opettajan nimi** | **Opinnäytetyön toimeksiantaja** |
|---|---|
| Jukka Selin | MHG Systems Oy |

# DESCRIPTION

| | |
|---|---|
| MIKKELIN AMMATTIKORKEAKOULU<br>Mikkeli University of Applied Sciences | **Date of the bachelor's thesis**<br>April 28. 2009 |
| **Author(s)**<br>Esa Hiiva | **Degree programme and option**<br>Information technology<br>Hardware and programming technology orientation |

**Name of the bachelor's thesis**

Converting a mobile phone application to Windows platform.


**Abstract**

This thesis concentrates to the problems involved when a mobile phone application was converted to Windows platform. Focus of the thesis is in the user interface of the application. Thesis was assigned by MHG Systems Ltd. MHG Systems Ltd have a mobile client application which had to be converted to a Windows platform client application. Windows client application was complemented with an operating manual. Installer package for the distribution had to be planned. Creating easy to use user interface was the main focus in the development process.

Java Micro Edition (Java ME) is used in the source mobile client application. Java Standard Edition (Java SE) was chosen for the Windows client application. Software on the company's servers is programmed using Java Enterprise Edition (Java EE). User interface of the Windows client application was programmed using Swing toolkit. Swing Application Framework was used for the structure of the application.

Implemented Windows client application fulfilled the requirements given to the application. The Windows client application was launched to complement MHG Systems Ltd service. MHG System Ltd will get benefits for the new service. Customers can use new Windows client software easier than the old mobile application. The Windows application can be used with much larger variety of the hardware than the mobile application.



**Subject headings, (keywords)**
Programming, Java, J2SE, J2ME, User interfaces

| **Pages** | **Language** | **URN** |
|---|---|---|
| 37 p. | Finnish | URN:NBN:fi:mamk-opinn200935758 |

**Remarks, notes on appendices**


| | |
|---|---|
| **Tutor**<br>Jukka Selin | **Bachelor´s thesis assigned by**<br>MHG Systems Ltd |

**SISÄLTÖ**

# 1 JOHDANTO

Nykypäivänä erilaisten päätelaitteiden määrä on kasvanut ja sovelluksia käytetään monissa erilaisissa käyttöliittymissä. Samalla on ilmaantunut tarve käyttää samoja sovelluksia eri päätelaitteilla. Sovelluksen muuntaminen toiselle päätelaitteelle on useimmiten helpompaa kuin uuden sovelluksen ohjelmoiminen. Käyttäjien on myös helpompaa käyttää tutuksi tullutta sovellusta eri päätelaitteella, kuin toista aivan erilaista sovellusta. Näistä syistä sovelluksen muuntamista kannattaa harkita uuden sovelluksen suunnittelemisen sijaan.

Matkapuhelinten yleistymisen jälkeen on tullut yleiseksi muuntaa Windows-sovelluksia toimimaan matkapuhelimissa. Tämän insinöörityön tutkimusongelmana on täysin päinvastainen idea, muuntaa olemassa olevasta matkapuhelinsovelluksesta Windows-sovellus. Insinöörityön tavoitteena on tuottaa toimeksiantajalle MHG Systems Oy:lle julkaisukelpoinen Windows-alustalla toimiva asiakassovellus, perustuen yrityksen tarjoaman matkapuhelimessa toimivan asiakassovelluksen toiminnallisuuteen. Sovelluksen tulee hyödyntää Windows-päätelaitteen suurempaa näyttöä ja parempia hallintalaitteita. Sovelluksen tulee myös toimia mahdollisimman monissa erilaisissa Windows-käyttöjärjestelmää käyttävissä laitteissa, riippumatta laitteiden teknisestä toteutuksesta.

Tämä insinöörityö koostuu tämän kirjallisen dokumentaation lisäksi Windows-asiakasohjelman suunnittelusta ja toteutuksesta sekä sovelluksen suomenkielisestä käyttöohjeesta ja sovelluksen asennuspaketin suunnittelusta. Tämän insinöörityön tavoitteena on selvittää tarvittava tietotaito matkapuhelinsovelluksen muuntamisesta Windows-sovellukseksi ja toteuttaa muutos käytännössä. Työn tuloksena tulisi olla toimiva ja julkaisukelpoinen Windows-sovellus.

Tässä työssä esitellään aluksi toimeksiantaja sekä asiakastarve, joka käynnisti tämän suunnitteluprosessin. Tämän jälkeen käydään läpi käyttöliittymän suunnitteluun liittyviä asioita, koska toteutettavan sovelluksen käyttöliittymä täytyy suunnitella alusta alkaen. Seuraavaksi käydään läpi käyttöliittymän suunnittelua ja toteutusta Java SE ohjelmointikielellä käyttäen Swing-käyttöliittymäkirjastoa. Teoriaosuuden jälkeen

käydään läpi Windows-asiakasohjelman suunnittelu ja toteutus, keskittyen sovelluksen käyttöliittymään.

Kaikki Java luokkien nimet on kirjoitettu *kursiivilla* tekstillä, oli kyseessä Javan oma sisäinen luokka tai itse toteutettu luokka. Osa termeistä on kirjoitettu myös englanniksi ensimmäisellä käyttökerralla. Pyrkimyksenä on ollut käyttää suomenkielisiä termejä aina kun mahdollista. Sovellukselle tehtyä 38-sivuista suomenkielistä käyttöohjetta ei ole mahdollista julkaista julkisessa asiakirjassa, joten se ei ole liitteenä. Myöskään ohjelmakoodia tai tarkkaa kuvausta sovelluksen toiminnallisuudesta ei ole sisällytetty tähän työhön, käyttöliittymää lukuun ottamatta.

# 2 TARVE UUDELLE ASIAKASOHJELMALLE

Tässä luvussa kerrotaan taustatietoa toimeksiantajasta MHG Systems Oy:stä. MHG Systems Oy:n tarjoama toiminnanohjausjärjestelmä kuvataan lyhyesti ja selvitetään ilmentynyt tarve uudelle asiakasohjelmalle. Insinöörityön osana tehtävän asiakasohjelman rooli järjestelmässä tulee esille kuvauksesta.

## 2.1 Toimeksiantaja MHG Systems Oy

MHG Systems Oy tarjoaa asiakkailleen verkkopalveluna Bioenergy ERP -toiminnanohjausjärjestelmää sekä siihen kuuluvaa matkapuhelimella käytettävää MHG Mobile asiakasohjelmaa. Palvelut tarjoavat asiakasyrityksille mahdollisuuden parantaa toiminnan tehokkuutta ja mahdollistavat tarkan kustannusten seurannan. Toimintaketjun reaaliaikainen seuranta antaa mahdollisuuden tarkempaan koneiden ja henkilöstön hallintaan. MHG Systems Oy:llä on laaja tietotaito bio- ja metsäenergialiiketoimista, jonka hyöty näkyy suoraan asiakasyrityksille. Asiakkaille on tarjolla paketti, joka kattaa verkkopalvelun lisäksi konsultoinnin, laitteet ja koulutuksen.

Toiminnanohjausjärjestelmää käyttävät asiakasyritykset ovat energiayhtiöitä, metsänhoitoyhdistyksiä, metsäyhtiöitä sekä yrityksiä jotka käsittelevät energiapuuta, haketta tai muuta biomateriaalia toimitusketjun eri vaiheissa. Suurin hyöty järjestelmästä saadaan, kun toiminnanohjausjärjestelmä kattaa koko toimitusketjun. Asiakasyrityksille

tarjotaan mahdollisuus seurata biomateriaalin kulkua, jolloin voidaan varmistaa tuotteen alkuperä. Biomassojen energiasisältöä voidaan seurata ja toiminnan hiilidioksidipäästöjä voidaan myös seurata.

### 2.2 MHG Bioenergy ERP -toiminnanohjausjärjestelmä

Järjestelmä koostuu moduuleista, joista kootaan asiakkaalle parhaiten toimintaa tehostava kokoonpano. Moduuleita ovat: Power (energia), Forest (metsä), Plantation (viljelmä), Recycling (kierrätys), Pellet (pelletti), Terminal (terminaali) ja Invoicing (laskutus). Palveluna tarjotaan erikseen myös MHG Platfom sovellusrunkoa, josta voidaan räätälöidä täysin asiakkaan tarpeiden mukainen toiminnanohjausjärjestelmä toimialasta riippumatta. MHG Bioenergy ERP järjestelmä rakentuu MHG Platformin päälle.

Bioenergy ERP -järjestelmää voidaan käyttää bioenergian toimitusketjun joka vaiheessa. Konekuski voi merkata syntyneet bioenergiaksi kelpaavat raaka-ainekasat järjestelmään MHG Mobilen avulla. Lähikuljettaja siirtää raaka-aineen tien varteen odottamaan noutoa ja merkkaa kasojen paikat Bioenergy ERP -järjestelmään käyttäen MHG Mobilea. Raaka-aine haketetaan ja hakkuri merkkaa MHG Mobilea käyttäen toimenpiteen suoritetuksi Bioenergy ERP -järjestelmään, jonka jälkeen kaukokuljettaja kuljettaa hakkeen terminaalin, tehden kuljetuksesta kuormakirjan Bioenergy ERP -järjestelmään, käyttäen MHG Mobilea. Terminaalista raaka-ainemassa välitetään voimalaitokselle, molempien käyttäessä raaka-aineen seurantaan Bioenergy ERP -järjestelmää Internetin välityksellä.

### 2.3 MHG Mobile -asiakasohjelma

Järjestelmän etäkäyttöä varten on toteutettu MHG Mobile -asiakasohjelma matkapuhelimille. MHG Mobile -palvelun avulla voidaan työntekijöille lähettää tehtäviä suoraan työmaalle ja työntekijä voi kuitata ja raportoida työn edistymisestä matkapuhelimilla. Palvelu osaa hyödyntää matkapuhelimessa olevaa kameraa liitetiedostojen lisäämiseen ja GPS-vastaanotinta paikannukseen.

### 2.4 Uusi asiakasohjelma Windows-päätelaitteille

Tietotekniikka on jo vallannut työkoneiden ohjaamotkin ja monissa työkoneissa ja ajoneuvoissa on PC-yhteensopiva tietokone työnteon apuna, osassa vakiovarusteena. Internet-yhteyttä ei välttämättä ole saatavilla, joten ei ole mahdollista käyttää Internet-selaimella käytettävää Bioenergy ERP -järjestelmää suoraan työkoneesta käsin. Tehtävien lähettäminen matkapuhelimeen ja tehtyjen töiden kuittaaminen järjestelmään on mahdollista käyttäen matkapuhelinverkkoa. Tässä tilanteessa tuleekin vastaan käytettävyysongelma, koska työkoneen tai ajoneuvon tietokonetta ei voida hyödyntää. Tehtävien kuittaamiseen on ollut pakko käyttää matkapuhelinta. MHG Systems Oy:llä ei ollut PC-asiakasohjelmaa, joten oli syntynyt tarve uudelle asiakasohjelmalle. Uuden sovelluksen toiminnallisuuden täytyisi vastata MHG Mobile -sovellusta, mutta toimia Windows-käyttöjärjestelmää käyttävissä PC-tietokoneissa. Nimeksi uudelle sovellukselle valittiin MHG Mobile PC ja ohjelmointikieleksi Java SE.

MHG Mobile PC:n toteutuksessa täytyy keskittyä toiminnallisuuteen ja testata toimintaa mahdollisimman monilla eri laitteilla. Sovellusta tullaan käyttämään pöytätietokoneissa, kannettavissa tietokoneissa, minikokoisissa kannettavissa tietokoneissa ja erilaisissa työkoneisiin integroiduissa tietokoneissa. Ohjelmaa pitää voida käyttää helposti kosketusnäytön kanssa tai hiirellä ja näppäimistöllä.

## 3 GRAAFISEN SOVELLUKSEN KÄYTTÖLIITTYMÄN SUUNNITTELU

Nykyään käytetään paljon erilaisia käyttöympäristöjä. Mikäli sovellus joudutaan muuntamaan erilaiseen käyttöympäristöön, voi ilmentyä tarve suunnitella käyttöliittymä kokonaan uudelleen. Insinöörityön osana suunniteltava ja toteutettava asiakasohjelma on tästä hyvä esimerkki, koska matkapuhelimen käyttöliittymä poikkeaa täysin Windows-sovelluksen käyttöliittymästä. Tässä luvussa käydään läpi huomioonotettavia asioita graafisen käyttöliittymän suunnittelussa Windows-tietokoneelle.

### 3.1 Yleistä huomioitavaa graafisen sovelluksen suunnittelussa

Mikäli sovellusta voidaan käyttää vain yhdessä tietyssä käyttöympäristössä, se rajoittaa mahdollista käyttäjäkuntaa merkittävästi. Asiakasmäärän suhteen onkin parasta, mikäli samaa sovellusta voidaan käyttää mahdollisimman monissa hyvin erilaisissa käyttöympäristöissä. Ellei saman sovelluksen käyttö ole mahdollista, tulisi sovelluksen olla helposti muokattavissa toimimaan uudessa käyttöympäristössä.

Sovelluksen tunnistettavuuden kannalta eräs tärkeimmistä tunnisteista on sovelluksen ikoni. Sovelluksen ikoni on näkyvillä Windows -käyttöjärjestelmässä monessa eri paikassa. Ikoni voi olla työpöydällä, tehtäväpalkissa, sovelluksen otsikkopalkissa sekä monissa muissa paikoissa. Ikonin täytyy olla selkä ja helposti tunnistettava, varsinkin tilanteissa joissa ikonin koko on erittäin pieni. Tärkeintä on, että käyttäjä pystyy erottamaan sovelluksen ikonin muiden sovelluksien ikoneista, eikä tulee sekaannuksia. (Cooper ja Reimann 2003, 458.)

Graafisen käyttöliittymän suunnittelussa täytyy huomioida monia seikkoja. Kalinen (1996, 142) painottaa kokonaisuuden tärkeyttä ja yhdenmukaista linjaa ulkoasussa. Yhdenmukainen linja voi jatkua myös koko tuoteperheeseen. Kalinen (1996, 142) suosittelee kirjaamaan ylös ulkoasuun liittyvät tyylivalinnat ohjeistoksi suunnittelijoille, joita noudattamalla voidaan säilyttää samanlainen ulkoasu. Yrityksellä saattaa olla omat tunnusvärit ja kuviot, joista yrityksen tuotteet on helppo tunnistaa. Käyttämällä samoja värejä ja kuvioita, käyttäjän on helpompi siirtyä tuoteperheen tuotteesta toiseen, koska visuaalinen ilme pysyy samankaltaisena.

### 3.2 Sovelluksen rakenne ja navigointi

Sovelluksen sisällä navigointi tuo haasteita käyttöliittymän toteutukseen. Navigointia tapahtuu eri näyttöjen välillä sekä eri paneelien välillä samalla näytöllä. Eri näytöillä tapahtuvat toiminnot täytyy erotella tarkasti, käyttäjä sekaantuu helposti, mikäli joutuu vaihtamaan näyttöjä useasti. Käyttäjälle tulee myös helposti mielikuva näytöltä poistuttaessa, että toimenpide on suoritettu loppuun. Samalla näytöllä tapahtuvia toimintoja saa selkeytettyä jakamalla sisältöä eri paneeleihin, mutta jakamisessa täytyy noudattaa loogisuutta. Paneeleita ei saa olla näytöllä liikaa ja paneelien olisi suotavaa olla

siinä järjestyksessä, jossa käyttäjä tulee niitä käyttämään. (Cooper ja Reimann 2003, 144.)

Valintaikkunoilla eli dialogeilla voidaan suorittaa erinäisiä toimintoja, kuten esimerkiksi virheilmoitusten näyttämistä käyttäjälle tai vahvistuksen pyytämistä käyttäjän aloittamalle toiminnolle. Valintaikkunaa voidaan käyttää myös esimerkiksi tiedostojen tai värin valitsemiseen. Valintaikkunoita on valmiissa ohjelmakirjastoissa ja niitä käyttämällä saadaan sovellukseen helposti toteutettua toimenpiteitä, jotka vaativat käyttäjän huomiota. (Zukowski 2002, 430-450.) Valintaikkunan sisällön täytyy olla selkeä ja erottua helposti pääikkunasta, jonka päälle valintaikkuna avataan. Valintaikkunan tulee kuitenkin olla mahdollisimman pieni, jottei se peitä liian suurta osaa pääikkunasta. (Shneiderman 1998, 268-270.) Cooperin ja Reimannin (2003, 393) mukaan valintaikkunoita tulee käyttää ainoastaan lyhyisiin toissijaisiin toimenpiteisiin sovelluksen sisällä.

### 3.3 Toimintojen hallinta graafisessa käyttöliittymässä

Sovelluksen on hyvä muistaa käyttäjän syötteitä ja suorittaa mahdollisimman paljon toimintoja käyttäjää häiritsemättä. Toimenpiteiden automatisointi nopeuttaa työskentelyä ja parantaa työtehoa, joka puolestaan kasvattaa tulosta. Käyttäjän syöttämä tieto pitää saada mahdollisimman vähäiseksi ja täytyy estää käyttäjän syötteen katoaminen ohjelman muistista. Cooper ja Reimann (2003, 193) toteaakin hyvin, että mikäli tieto on käyttäjän syöttämisen arvoista, on se myös sovelluksen tallentamisen arvoista.

Käyttäjän syötteen tulkitseminen on haastava toimenpide toteuttaa sovellukseen. Sovelluksen täytyy tarkastaa syöte ja hienovaraisesti ilmoittaa virheistä käyttäjälle, mutta vain tarvittaessa. Sovellus voi myös tehdä korjauksia käyttäjän syötteeseen, jolloin käyttäjää ei välttämättä tarvitse häiritä virheilmoituksella. Hyvä tapa olisikin tarkastaa käyttäjän syöte ja tarjota käyttäjälle mahdollisuutta korjata virheellinen syöte halutessaan, muttei pakottaa käyttäjää korjaamaan syötettä välittömästi. Cooper ja Reimann (2003, 211-213.)

# 4 KÄYTTÖLIITTYMÄOHJELMOINTI JAVA-OHJELMOINTIKIELELLÄ

Tämän luvun aiheena on esitellä hieman tarvittavaa teoriaa graafisen käyttöliittymän suunnitteluun Java-ohjelmointikielellä. Käyttöliittymän suunnittelu esitellään suppeasti, ja lähinnä vain niiltä osin, joille on käyttöä tämän insinöörityön osalta. Aluksi esitellään Swing-käyttöliittymäkirjasto ja kerrotaan tarkemmin yleisimmistä komponenteista, ulkoasusta ja komponenttien asettelusta. Valintaikkunoiden käytöstä kerrotaan myös lyhyesti. Tämän jälkeen tutustutaan Swing Application Frameworkiin, jonka avulla on helppo luoda graafisia sovelluksia. Lyhyesti tutustutaan myös Javan Preferences API:iin sekä NetBeansin käyttöliittymänrakennustyökaluun.

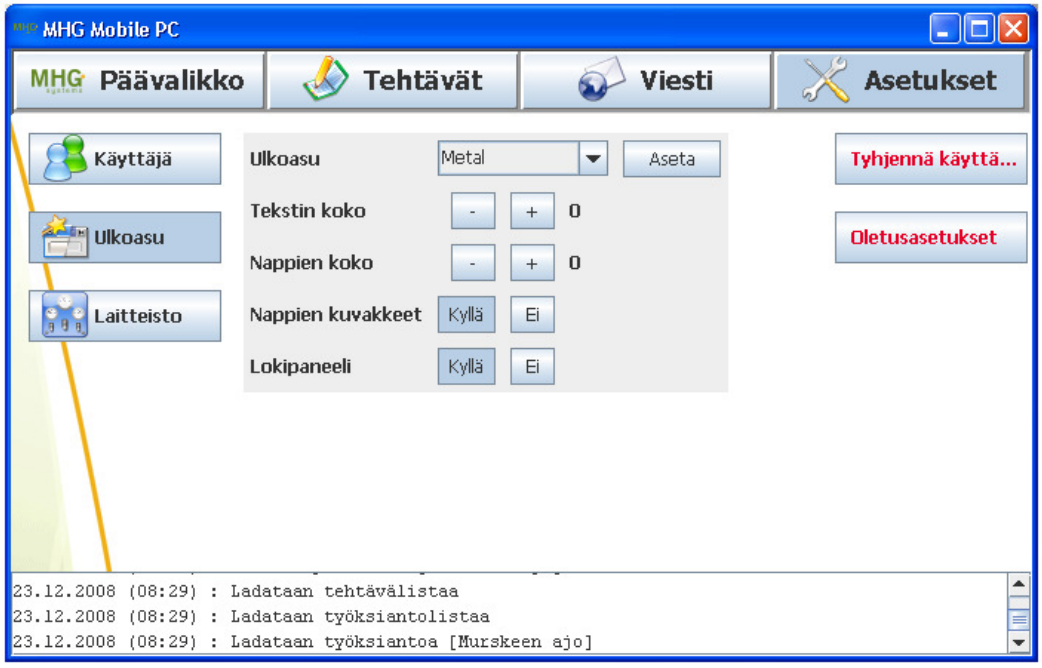
## 4.1 Swing-käyttöliittymäkirjasto

Java ohjelmointikieli sisältää sisäänrakennettuna käyttöliittymäkirjaston. Ensimmäinen Javan sisältämä käyttöliittymäkirjasto tunnetaan nimellä AWT eli *Abstract Window Toolkit*. AWT-kirjasto korvattiin myöhemmin Swing-käyttöliittymäkirjastolla, joka laajentaa AWT-kirjastoa ja korvaa osan sen toiminnoista. AWT-kirjasto on yhä olemassa, mutta sovellukset rakennetaan nykyään käyttäen Swing-kirjastoa. Swing-luokat löytyvät paketista *javax.swing*. (Horstmann ja Cornell 2003, 235-239.)

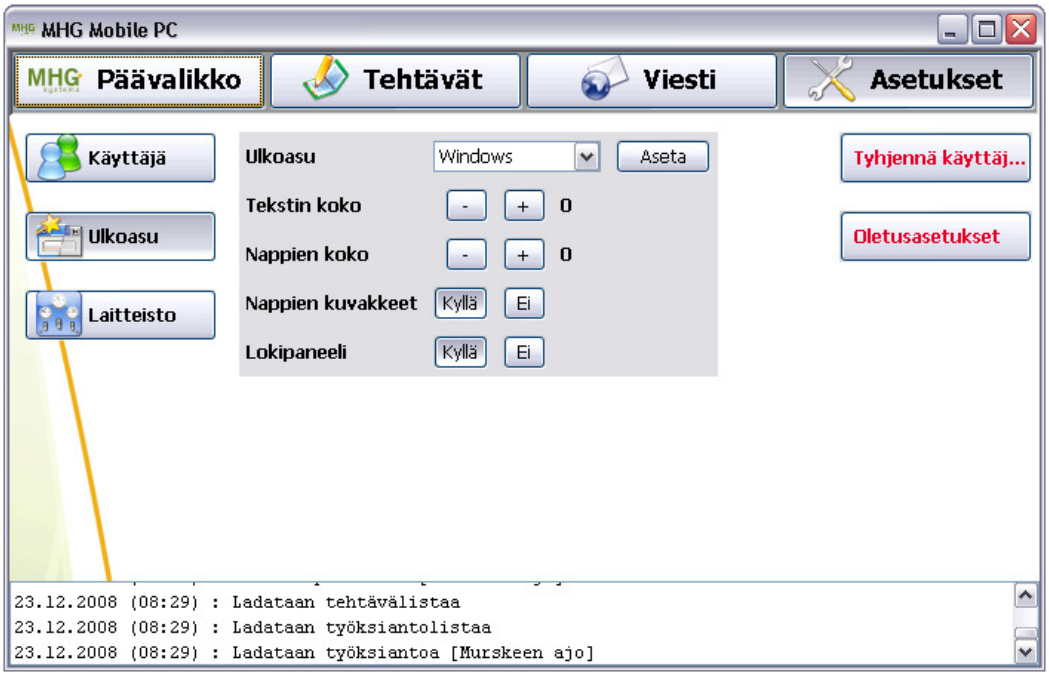
AWT-kirjastoa käytettäessä komponentit näyttävät ulkoisesti samoille kuin käytettävän käyttöjärjestelmän omat komponentit. Tämä antaa AWT-kirjastoa käyttäen toteutellulle sovelluksella ulkonäön joka vastaa kyseisen käyttöjärjestelmän sovelluksia, mutta tuo mukanaan myös haittavaikutuksia. Komponenttien sijoittelua ei ole mahdollista määrätä etukäteen kovinkaan tarkasti, koska komponenttien koko riippuu käytetystä käyttöjärjestelmästä. Swing -komponentit piirretään Javan sisällä, jolloin Java hallitsee komponenttien ulkonäköä. Swing -komponenttien ulkonäköä hallitaan *pluggable look and feel* (PLAF) arkkitehtuurin avulla. (Zukowski 2002, 269-270.)

### 4.1.1 Käyttöliittymän ulkoasu

Swing komponenttien oletusulkoasu on Metal, jonka Sun on kehittänyt käyttöympäristöriippumattomaksi. Käyttämällä Metal-ulkoasua sovellus näyttää täysin samalla jokaisella käyttöjärjestelmällä. Metal-ulkoasu kuitenkin poikkeaa käyttöjärjestelmien omasta tyylistä ja sovellus voi näyttää käyttäjästä oudolle. Sunin Metal ulkoasu esitellään kuvassa 1, ja Windowsin harmaa ulkoasu esitellään seuraavalla sivulla kuvassa 2. Ulkoasua voikin vaihtaa muuttamatta komponentin sisältöä tai toimintaa. Ulkoasuna voi käyttää myös käyttöjärjestelmän omaa ulkoasua. Tyylien käytössä on kumminkin eräitä rajoituksia, esimerkiksi Macintoshin ulkoasua voidaan käyttää vain Macintoshin omaa käyttöjärjestelmää käytettäessä ja Windowsin ulkoasua voidaan käyttää vain Windows-käyttöjärjestelmää käytettäessä. (Horstmann ja Cornell 2003, 236-238.)




**KUVA 1 Sunin Metal-ulkoasu**




**KUVA 2 Windowsin harmaa ulkoasu**

### 4.1.2 Käyttöliittymän komponentit

Käyttöliittymäsuunnittelun kannalta käyttöliittymän komponentit voidaan jakaa asettelukomponentteihin ja toimintokomponentteihin sekä tiedonsyöttöön ja -näyttöön tarkoitettuihin komponentteihin. Asettelukomponenttien avulla voidaan sijoitella muita komponentteja ja luoda käyttöliittymän ulkoasu väreillä ja muodoilla. Toimintokomponenteilla käyttäjä voi käynnistää toimintoja sovelluksessa ja tehdä valintoja. Sovelluksessa tarvitaan usein tapa näyttää tietoa käyttäjälle sekä kuvin että tekstimuodossa. Käyttäjän syötteen taltioimiseen tarvitaan komponentteja ja yleensä samoilla komponenteilla voidaan myös näyttää tietoa käyttäjälle. Lyhyt kuvaus muutamista yleisimmistä ja tarpeellisimmista komponenteista on seuraavana.

Paneeleilla (*JPanel*) voidaan ryhmitellä komponentteja eri toimintojen mukaan ja paneelit mahdollistavat komponenttien sijainnin tarkan asettelun. *JPanel*-luokan paneelille voidaan piirtää ja se voi sisältää komponentteja. Paneelille voi määrittää värin ja paneelin reunoille on valittavissa erilaisia tyylejä. (Horstmann ja Cornell 2003, 247-248 ja 344.)

Toimintokomponentteja Javassa on laaja valikoima. Napit ovat eräs keino käynnistää toimintoja. *AbstractButton*-luokasta periytetyt *JButton*-luokka toteuttaa tavallisen painonapin ja *JToggleButton*-luokka valinnan säilyttävän päälle/pois-napin. *JToggleButton*-nappi jää valituksi ensimmäisellä painalluksella ja palautuu alkuperäiseen tilaansa toisella painalluksella. Napeille voi määrittää tekstin lisäksi kuvakkeet (icon) jokaiselle eri tilalle. Tiloja on *JButton*-napilla kolme, jotka ovat normaalitila ja tila nappia painettaessa sekä tila jolloin hiiren kohdistin on napin päällä. *JToggleButton*-napilla on lisäksi valittu-tila. (Zukowski 2002, 323-341.) Valintaruuduilla (*JCheckBox*) saadaan helposti toteutettua kyllä/ei-tyyppisiä valintoja ja valintanapeilla (*JRadioButton*) valintoja useammasta vaihtoehdosta. Valintanappeja voidaan ryhmitellä *ButtonGroup*-luokan avulla ryhmiksi, joista vain yksi nappi kerrallaan on valittuna. (Horstmann ja Cornell 2003, 378-382.)

Monipuolisimmista komponenteista hyödyllisimpiä on mm. yhdistelmäruutu (*JComboBox*). Komponenttiin voi kirjoittaa suoraan uuden arvon, mikäli muokkaaminen on sallittua, mutta samalla komponentti tarjoaa myös valmiita vaihtoehtoja. Liukusäätimet (*JSlider*) sopivat hyvin käyttötarkoituksiin joissa halutaan käyttäjän asettaa arvo annettujen raja-arvojen välille. Liukusäätimeen on mahdollista saada näkyviin jakoviivoja, arvoja jakoviivojen kohdalla ja jopa kuvakkeita. (Horstmann ja Cornell 2003, 389-394.)

### 4.1.3 Komponenttien asettelu

Komponenttien sijoittelua hallitaan asettelunhallintaluokkien avulla. Jokainen säilö (container) käyttää jotakin asetteluhallintaan sisältämiensä komponenttien asetteluun näytölle. Seuraavaksi esitellään tarkemmin muutama yleinen ja hyödyllinen asettelunhallintaluokka.

Liukuva asettelunhallinta (*FlowLayout*) on oletusasettelunhallinta *JPanel*-luokan säilölle. Liukuva asettelunhallinta sijoittaa komponentteja peräkkäin säilön sisään, kunnes uusi komponentti ei mahdu säilöön. Tällöin asettelunhallinta jatkaa komponenttien lisäämistä seuraavalle riville, ensiksi lisättyjen komponenttien alapuolelle. Komponenttien sijainti saattaa muuttua, mikäli säilön leveyttä muutetaan. Komponentti saattaa jopa vaihtaa paikkaa eri riville. (Horstmann ja Cornell 2003, 342.)

Reuna-asettelu (*BorderLayout*) on *JFrame*-luokan sisältöruudun oletusasettelu. Reuna-asettelu jakaa säilön maksimissaan viiteen osaan, joista käytetään nimityksiä ilmansuuntien mukaan. Säilön yläosassa (North) ja alaosassa (South) on koko säilön levyiset alueet ja säilön vasemmalla sivulla oleva alue (West) sekä oikealla sivulla oleva alue (East) rajoittuvat yläosassa olevan alueen alareunan ja alaosassa olevan alueen yläreunan väliin. Nämä neljä reunalla olevaa aluetta pysyvät aina saman levyisinä suhteessa vastaavaan reunaan, vaikka säilön kokoa muutettaisiin. Reuna-alueiden väliin jäävä alue (Center) täyttää aina reuna-alueiden väliin jäävän tilan. Mikä tahansa alueista voidaan jättää pois asettelusta. (Horstmann ja Cornell 2003, 343-344.)

Mikäli komponenttien sijaintia halutaan hallita tarkemmin suhteessa toisiin komponentteihin, tarvitaan monipuolisempi asettelunhallinta. Ruudukkoasettelulle (*GridLayout*) ilmoitetaan haluttu rivi- ja sarakemäärä asettelua luodessa ja ruudukkoasettelu asettelee säilön komponentit yhden jokaiseen ruudukon ruutuun. Säilön kokoa muutettaessa ruudukon solut pysyvät aina samankokoisina, kaikkien solujen koko pienenee tai suurenee samanaikaisesti. Ruudukkoasettelulla on helppo toteuttaa ryhmä samankokoisia komponentteja. (Horstmann ja Cornell 2003, 346.)

Yksi monipuolisimmista ja samalla hyödyllisimmistä asetteluista on ruudukkoperiaatteella toimiva *GridBagLayout*. *GridBagLayout* asetteluhallinta poikkeaa muista tavallisista asettelunhallinnoista sillä, että se käyttää apuluokkaa *GridBagConstraints* säilyttääkseen asetustenhallinnan sijoitteluparametrejä. Yleensä *GridBagConstraints*-luokan ilmentymiä on yhtä monta kuin komponentteja säilössä. Merkittävin etu *GridBagLayout*-asettelulla verrattuna *GridLayout*-asetteluun, on mahdollisuus käyttää komponentteja jotka käyttävät useamman kuin yhden ruudun ruudukosta. (Zukowski 2002, 296-297.)

### 4.1.4 Valintaikkunat

Valintaikkunoita käyttämällä voidaan toteuttaa helposti toimenpiteitä, jotka ovat näyttäviä, mutta niiden toteuttamiseen ei tarvitse käyttää paljoa resursseja. Ohjelmakoodin kannalta valmiiden valintaikkunoiden käyttö on helppoja ja omaa ohjelmakoodia ei tarvitse kirjoittaa kovinkaan paljon. Kolme tärkeintä komponenttia Java-kielessä valintaikkunoiden toteutukseen ovat *JOptionPane*, *JFileChooser* ja *JColorChooser*. (Zu-

kowski 2002, 430.) Seuraavassa lyhyesti kuvattuna minkälaisia valintaikkunoita voidaan toteuttaa käyttäen *JOptionPane* ja *JFileChooser*-luokkia. Tässä opinnäytetyössä ei ollut tarvetta *JColorChooser*-valintaikkunan käyttöön, joten sivuutan sen myös teoriaosuudessa.

**JOptionPane**

Käyttäjälle voidaan näyttää erilaisia viestejä käyttäen *JOptionPane*-luokkaa. Yksinkertaiset viesti-ilmoitukset sisältävät ilmoituksen otsikon, kuvakkeen, tekstin sekä hyväksymisnapin. Tämän tyyppinen ilmoitus on paras vaihtoehto virheilmoituksille ja varoituksille. Ilmoituksen otsikon täytyy olla kuvaava, jotta käyttäjä saa heti selville miksi virheilmoitus on ilmaantunut näytölle. Ilmoituksen kuvake määräytyy ilmoituksen tyypin mukaan. Tyyppi voi olla virheviesti, varoitusviesti, tietoa sisältävä viesti tai tyypitön viesti, jolla ei ole kuvaketta. Näiden tyyppien lisäksi on myös tyyppinä kysymysviesti, jolla ei ole oikeaa käyttöä, koska viestiin ei voi vastata muuten kuin painamalla hyväksymisnappia. (Zukowski 2002, 434-435.)

Mikäli käyttäjältä tarvitaan lisätietoja tai vahvistusta toimenpiteelle, tarvitaan monipuolisempi valintaikkuna. *JOptionPane*-luokkaa käyttämällä voidaan toteuttaa valintaikkuna jossa yksittäisen hyväksymisnapin tilalla sijaitsee napit toimenpiteen hyväksymiseen, hylkäämiseen ja peruuttamiseen. Tämän tyylinen viesti on erittäin yleinen sovelluksissa. Mikäli käyttäjältä halutaan syötetietoa, on mahdollista *JOptionPane*-luokan avulla toteuttaa myös valintaikkuna jossa on tekstikenttä syötteelle tai pudotusvalikko vaihtoehdon valitsemiseen. (Zukowski 2002, 435-438.)

**JFileChooser**

Tiedostovalintaikkuna on monimutkainen valintaikkuna, jolla käyttäjä voi valita tiedoston käyttämältään tietokoneelta. Tiedostovalintaikkuna voi olla tiedoston avaamiseen tai tallentamiseen tarkoitettu, jolloin toimintonäppäimen nimi muuttuu toimintoa vastaavaksi. Toimintonäppäimen nimen voi myös muuttaa haluamakseen. Tiedostovalintaikkuna on muokattavissa käyttötarkoitukseen sopivaksi. Muun muassa oletushakemisto ja oletustiedosto voidaan valita, samoin voidaan suodattimia käyttämällä määrittää tiedostovalintaikkuna näyttämään ainoastaan halutun tyyppisiä tiedostoja. Tie-

dostonvalintaikkunalla voidaan valita yksittäisiä tiedostoja, tai voidaan sallia käyttäjän valita useita tiedostoja. (Horstmann ja Cornell 2003, 474-475.)

### 4.2 Swing Application Framework

Swing Application Framework on graafisten käyttöliittymien rakentamiseen Java-kielelle kehitetty valmis runko, jonka päälle on helppo ja nopea kasata oma sovellus. Swing Application Frameworkistä käytän jatkossa nimitystä framework. Graafista käyttöliittymää käyttävissä Swing-sovelluksissa joudutaan usein toteuttamaan samoja toimenpiteitä, jotka on valmiiksi toteutettu frameworkissa. Ohjelmoijan ei tarvitse huolehtia perusasioista, joten hän voi keskittyä täysin sovelluksen sisällön tuottamiseen.

Frameworkin avulla voidaan hallita helposti sovelluksen elinkaarta, joka jakaantuu kuuteen vaiheeseen. Sovelluksen elinkaaren vaiheet toimintajärjestyksessä ovat laukaisu (launch), alustus (initialize), käynnistys (startup), valmis (ready), lopetus (exit) ja sammutus (shutdown). Vaiheita vastaavat metodit voidaan ylikirjoittaa, jolloin voidaan lisätä toimintoja eri vaiheisiin. Sovelluksen saamiseksi toimimaan frameworkin avulla on kutsuttava launch-metodia sekä startup-metodi täytyy ylikirjoittaa. Sovelluksen sammutus on suoritettava käyttäen exit-metodia. Elinkaaren initialize, ready ja shutdown-metodit ovat valinnaisia, ja ne voidaan ylikirjoittaa vain tarpeen niin vaatiessa. (O'Connor 2007.)

Toinen merkittävä hyöty frameworkista on resurssienhallinta. *ResourceBundle*-luokkaa käyttämällä voidaan sovelluksesta tehdä monikielinen ja hallita kaikkia sovelluksen kieliä yhdestä ja samasta sijainnista. Resurssien hallinta ei rajoitu vain tekstimuotoiseen tietoon, vaan on mahdollista käyttää resurssienhallinnan kautta muun muassa sovelluksen kuvia ja värejä. (O'Connor 2007.)

Toiminnot (Action) korvaavat tapahtumakäsittelijöiden käytön toiminnon laukaisemiseksi esimerkiksi nappia painettaessa. Toiminnon suorittama koodi voidaan sijoittaa joko samaan luokkaan toiminnon laukaisseen napin kanssa tai toiminnot voidaan kaikki kasata sovelluksen pääluokkaan. Kasaamalla kaikki toiminnot sovelluksen pääluokkaan, saadaan helppo hallittavuus ja samalla kaikki toiminnot jakavat samat re-

surssit. Toiminnot voidaan myös suorittaa taustalla erillisessä säikeessä. (O'Connor 2007.)

### 4.3 Asetusten hallinta käyttäen Java Preferences API:a

Sovelluksen asetuksien ja käyttäjän asetuksien tallentamiseen on Java-kieleen kehitetty Preferences API, jota käyttäen voidaan haluttuja asetuksia tallentaa Windows-käyttöjärjestelmän rekisteriin. Rekisteriin tallennettaessa asetukset ovat paremmassa turvassa vahingossa tapahtuvalta muokkaukselta, kuin tiedostoihin tallennetut asetukset. Preferences API:n käyttö on yksikertaista, tieto tallennetaan rekisteriin tyypin mukaisella put-metodilla ja tietoa haetaan tyypin mukaisella get-metodilla. (Travis 2003.)

Preferences API:a käyttämällä tieto tallentuu rekisteriin joko järjestelmäkohtaisesti tai käyttäjäkohtaisesti, riippuen käytetäänkö juurena *systemRoot*:ia vai *userRoot*:ia. Asetukset tallennetaan puurakenteisesti valitun juuren alle. Mikäli asetukset ovat järjestelmäkohtaisia, voivat kaikki tietokoneen käyttäjät käyttää samoja asetuksia. Mikäli asetukset ovat käyttäjäkohtaisia, jokainen käyttäjä voi käyttää vain omia asetuksiaan. Pienemmät sovellukset käyttävät yleensä vain toista juurista. Toiminnallisuuden kannalta ei ole väliä kumpaa juurta käytetään, koska molemmat juuret käyttäytyvät identtisesti. (Travis 2003.)

### 4.4 NetBeans IDE ja käyttöliittymän rakentaminen

Käyttöliittymän rakentaminen on helppoa NetBeans IDE -sovelluskehittimellä, koska sovelluskehitin sisältää sisäänrakennetun työkalun graafisten käyttöliittymien rakentamiseen. NetBeansin käyttöliittymien rakennustyökalu (GUI Builder) tunnettiin aiemmin myös nimellä Project Matisse. Javalle on myös monia muita GUI Buildereita, NetBeansin sisäänrakennetun lisäksi. GUI Builder mahdollistaa käyttöliittymän rakentamisen ilman tuntemusta käyttöliittymän ohjelmoinnista. GUI Builderin avulla voidaan toteuttaa käyttöliittymiä piirtämällä näytöt ja raahaamalla komponentit näytölle. Vain näyttöjen toiminta täytyy ohjelmoida, komponenttien asettelun ja värityksen sekä monia muita toimintoja voidaan toteuttaa suoraan GUI Builderin avulla kätevästi hiirellä klikkailemalla. GUI Builder tarjoaa kokemattomille ohjelmoitsijoille mahdollisuuden ohjelmoida helposti ja nopeasti näyttäviä graafisia sovelluksia, mutta mahdol-

listaa samalla myös kaupallisten sovellusten toteuttamisen edullisesti käyttöliittymän ohjelmoinnissa säästyneen ajan ansiosta.

# 5 MHG MOBILE PC -ASIAKASOHJELMA

Tämä luku kertoo MHG Mobile PC asiakasohjelman suunnittelusta ja toteutuksesta. Aluksi käydään läpi sovelluksen suunnittelu keskittyen käyttöliittymään, unohtamatta kuitenkaan sovelluksen toiminnallista osuutta. Tämän jälkeen tarkastellaan toteutusta, sitä kuinka sovellus on valmistunut ja minkälainen siitä on tullut. Toteutuksessakin painotus on käyttöliittymällä. Toteutuksessa esitellään myös GPS:n käyttöä sekä sovelluksen ja palvelimen välistä toimintaa.

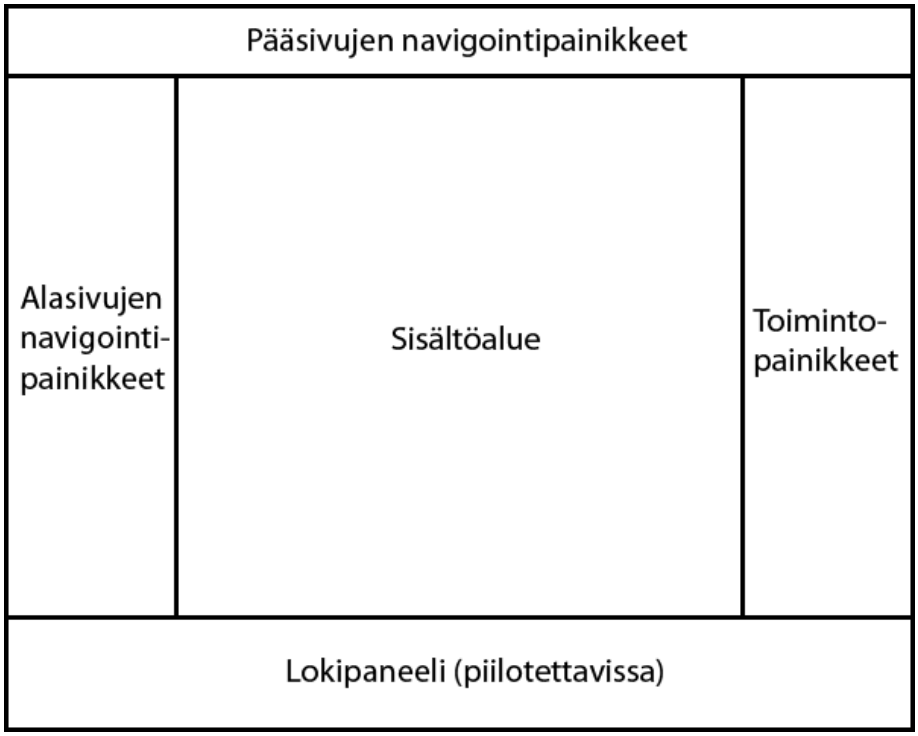
## 5.1 Suunnittelu

Suunnittelussa keskitytään helppokäyttöisyyden lisäksi erityisesti sovelluksen toimivuuteen erilaisilla alustoilla. Käyttöliittymän täytyy olla helppokäyttöinen, koska sovelluksen käyttäjät eivät ole IT-alan ammattilaisia ja käyttöliittymän täytyy toimia erilaisilla ohjauslaitteilla. Ohjauslaitteina voivat muun muassa olla hiiri ja näppäimistö tai kosketusnäyttö, jota voidaan käyttää sormilla tai stylus-kynällä. Erityisesti täytyy ottaa huomioon sovelluksen helppo käytettävyys kosketusnäyttöä käytettäessä. Toimintojen täytyy olla selkeitä ja kaikki toimenpiteet jotka eivät vaadi välttämättä käyttäjän huomiota on automatisoitava käytön helpottamiseksi. Sovelluksessa täytyy olla myös suojamekanismeja virhetilanteiden varalle, jottei tallennettu tieto katoa virhetoiminnosta tai virheellisestä käytöstä johtuen. Sovelluksen rungon täytyy olla helposti muokattavissa sovelluksen kehittyessä ja lisättäessä uusia ominaisuuksia sovellukseen.

### 5.1.1 Käyttöliittymän rakenteen suunnittelu

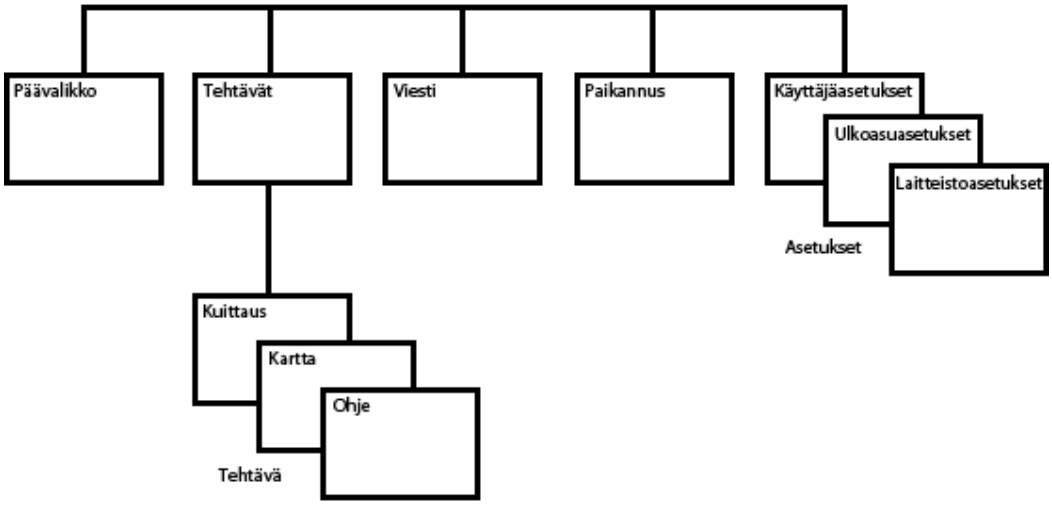
Sovellusta täytyy voida käyttää monissa erilaisissa laitteissa, joten käyttöliittymän täytyy olla mahdollisimman hyvin skaalautuva ja helposti muokattavissa. Suunnittelussa parhaaksi ratkaisuksi oli valikoitunut rakenne, jossa sovelluksen toiminnot jakautuvat näyttöihin ja näytöt koostuvat erillisistä paneeleista. Paneelirakenteen pohjana toimii reuna-asettelu, jolloin näyttö jaetaan viiteen paneeliin. Jokaisessa näytössä ylimmäisen

paneelin muodostavat pääsivujen navigointipainikkeet. Alimmainen paneeli on lokipaneeli, johon tulee näkyville jokainen käyttäjän suorittama toimenpide aikaleimalla varustettuna. Lokipaneelin avulla sovellus myös ilmoittaa käyttäjälle sovelluksessa tapahtuneista toimenpiteistä ja virhetilanteista. Pääsivujen navigointipainikkeet ja lokipaneeli ovat samanlaiset jokaisessa näytössä. Lokipaneeli tulee voida piilottaa näkyvistä, jolloin saadaan enemmin tilaa muuhun käyttöön. Vaikka lokipaneeli on piilotettuna, tulee sen ilmaantua näkyviin lyhyeksi aikaa uuden viestin ilmaantuessa. Mikäli lokipaneeli piilotetaan, venyvät alasivujen navigointipainikkeiden alue, sisältöalue sekä toimintopainikkeiden alue näytön alareunaan asti. Näyttöä vaihdettaessa vaihtuvat sisältöalue sekä toimintopainikkeet näytön mukaan. Alasivujen navigointipainikkeet tulevat näkyviin mikäli näytöllä on alasivuja. Vain tehtävä ja asetukset -näytöillä on alasivuja. MHG Bioenergy ERP Internet -palvelussa näytön vasemmassa laidassa sijaitseva kuva tuodaan MHG Mobile PC:n alasivujen navigointipainikkeiden paneelin taustakuvaksi. Käyttämällä samaa kuvaa saadaan luotua mielikuvaa yhteenkuuluvuudesta Internet-palvelun kanssa. Kuvassa 3 on kuvattuna reuna-asettelun mukainen näytön jakaminen paneeleihin.




**KUVA 3 Näytön rakenne**

Käyttöliittymä perustuu näyttöihin, kaikkiaan näyttöjä on kymmenen erilaista, alasivut mukaan lukien. Päänäyttöjä ovat Päävalikko, Tehtävät, Viesti, Paikannus sekä kolmeen alasivuun jakautuva asetukset. Päänäytöille pystyy siirtymään suoraan navigointipainikkeista yhdellä painalluksella. Tehtävät-näytöltä avattava tehtävä avautuu tehtävä-näytölle, jossa on kolme alasivua. Tehtävä-näytölle voi siirtyä vain tehtävälistan kautta. Alasivuja tehtävälle ovat kuittaus, kartta ja ohje. Asetukset-näyttö jakaantuu kolmeen alasivuun, joita ovat käyttäjäasetukset, ulkonäköasetukset sekä laitteistoasetukset. Näytöt ovat esitelty kuvassa 4. Näyttöjen lisäksi sisään kirjautumiseen tullaan käyttämään erillistä ponnahdusikkunaa. Seuraavaksi kerrotaan lyhyt kuvaus näytöille suunnitelluista toiminnoista.




**KUVA 4 Sovelluksen näytöt**

### Päävalikko-näyttö

Päävalikko tulee sisältämään lyhyen kuvauksen sovelluksen toiminnoista sekä painikkeen tai linkin sovelluksen käyttöohjeen avaamiseen. Päävalikkoon ei tule minkäänlaista toiminnallisuutta. Päävalikon tärkein tehtävä on luoda sovellukselle selvä aloitussivu, jotta sovellus ei vaikuttaisi vain kokoelmalle erillisiä näyttöjä.

### Tehtävät-näyttö

Tehtävien hallinta tulee tapahtumaan Tehtävät-näytöltä. Näytöllä on tarkoitus näyttää listat yleisistä tehtävistä ja työksiannoista. Tehtäviä tulee voida ladata, lähettää ja pois-

taa helposti useita tehtäviä kerrallaan. Tehtävälistojen tulee olla havainnolliset ja niistä täytyy saada helposti selville tehtävien tila sekä tieto tehtävälle tallennettujen kuittausten määrästä. Tehtäviä täytyy voida hallita Tehtävät-näytöltä avaamatta jokaista tehtävää erikseen tarkastaakseen tietoja tehtävistä.

**Tehtävä-näyttö**

Erillistä Tehtävä-näyttöä ei tule olemaan, vaan Tehtävä-näyttö koostuu kolmesta näytöstä, joille tehtävän tiedot on jaettu. Alasivut ovat kuittaus, kartta ja ohje. Käytän nimitystä alasivu näistä näytöistä, jotka ovat täysin itsenäisiä näyttöjä, mutta joiden välillä voidaan liikkua näytön vasemmassa sivussa olevien alasivun navigointinappien avulla. Alasivun navigointinapit on samat jokaisessa näytössä ja paneelirakenteen viidestä paneelista vain sisältöalue ja toimintonäppäimet paneelien (KUVA 3 Näytön rakenne) sisältö vaihtuu eri alasivuilla.

Tehtävä-näytön kuittaus-alasivu tulee olemaan koko sovelluksen laajin näyttö. Näytölle tulevat kentät tehtävän kuittaamiseen, joita tulee olemaan lukuisia erilaisia. Kuittaustiedot voivat olla numero- tai tekstimuotoisia sekä osa voi olla valmiista vaihtoehdoista valittavia. Kuittaukseen tulee voida lisätä myös GPS -sijainti ja liitetiedostoja sekä huomio, joka on vapaamuotoista tekstiä.

Tehtävän mukana voidaan toimittaa kartta, joka tullaan näyttämään kartta -alasivulla. Kartta on palvelimen luoma kiinteä kuvatiedosto. Kartta-alasivulle tulee myös näkyville GPS-laitteelta sijainti ja suunta. Kartan päälle piirretään käyttäjän sijaintia kuvaava nuoli, mikäli GPS-laitteelta saatava sijainti sijaitsee kartan alueella.

Internet-palveluun syötetyistä tehtävän tiedoista kasataan automaattisesti ohje-alasivulla näytettävät tiedot. Nykyisessä järjestelmässä tehtävän ohje sisältää ainoastaan toisarvoista tietoa, joka toimitetaan kartan mukana. Ohje toimitetaan yhtenä merkkijonona, joka näytetään sellaisenaan näytöllä.

**Viesti-näyttö**

Järjestelmään täytyy voida lähettää maksimissaan 1000 merkkiä pitkä viesti Viesti-näytöltä Internet-yhteyttä käyttäen. Näytölle tulee tekstikenttä viestin kirjoittamiseen, laskuri näyttämään käytettyjen merkkien määrän sekä GPS-sijainnin lisäysominaisuus.

**Paikannus-näyttö**

MHG Mobilessa oleva paikannustoiminto toteutetaan samoin toiminnoin MHG Mobile PC:ssä, vaikkei toiminto ole kovinkaan käytettyä MHG Mobilessa. Paikannus-näytölle on mahdollista saada GPS-sijainti ja tarvittaessa muita tietoja GPS-laitteelta. Näytön tarpeellisuutta tutkitaan tarkemmin sovellusta kehitettäessä.

**Asetukset-näyttö**

Kuten ei tule olemaan erillistä Tehtävä-näyttöä, ei myöskään tule olemaan erillistä Asetukset-näyttöä. Asetukset jakaantuvat kolmelle alasivulle, joita ovat käyttäjäasetukset, ulkoasuasetukset ja laitteistoasetukset. Tehtävä-näytön alasivuista poiketen kaikilla asetukset-alasivuilla on samat toimintonapit, jolloin vain sisältöalue (KUVA 3 Näytön rakenne) vaihtuu alasivua vaihdettaessa. Toimintonappeja tarvitaan käyttäjän tietojen tyhjentämiseen sovelluksesta ja käytettävästä tietokoneesta sekä sovelluksen ulkoasu- ja laitteistoasetusten palauttamiseen oletusarvoihin.

Käyttäjäasetukset-alasivulla täytyy näkyä sisään kirjautuneen käyttäjän tiedot, eli verkostokoodi ja käyttäjätunnus. Alasivulta täytyy voida käynnistää käyttäjänvaihto, joka suoritetaan erillisessä ponnahdusikkunassa. Sovelluksen kieltä täytyy voida vaihtaa käyttäjäasetuksista ilman uudelleenkirjautumista.

Kaikki sovelluksen ulkoasuun vaikuttavat asetukset tulevat olemaan ulkoasuasetukset-alasivulla. Mikään tälle alasivulle tuleva asetus ei saa vaikuttaa sovelluksen toimintaan, ainoastaan ulkoasuun. Ulkoasun tyyliä (Look and Feel) täytyy voida vaihtaa, sekä erikseen tekstin ja nappien kokoa. Alasivulla täytyy lisäksi olla mahdollisuus poistaa nappien kuvakkeet käytöstä, jotta on mahdollista käyttää käyttäjän halutessa joko suu-

rempaa tekstinkokoa napeissa tai pienempiä nappeja. Lokipaneelin piilottaminen tulee myös ulkoasuasetukset-alasivulle.

GPS-asetukset ovat ainoat asetukset joita on suunniteltu tässä vaiheessa laitteistoasetukset alasivulle. GPS-laitteen COM-portti ja käytettävä sarjaliikennenopeus täytyy voida asettaa alasivulla. Mahdolliset tulevat laitteisiin liittyvät asetukset lisätään tälle alasivulle.

**Ponnahdusikkuna sisään kirjautumiseen**

Sisään kirjautuminen tullaan suorittamaan erillisessä ponnahdusikkunassa, joka avataan sovelluksen pääikkunan päälle. Sisään kirjautuminen on pakollista ja sovellusta ei tule voida käyttää ennen sisään kirjautumista. Sisään kirjautuminen tapahtuu käyttäen Internet-yhteyttä. Mikäli käyttäjää vaihdettaessa Internet-yhteyttä ei ole saatavilla, on käyttäjänvaihdosta pystyttävä palaamaan takaisin sovellukseen aiemmin käytössä olleella käyttäjätunnuksella kirjautuneena.

### 5.1.2 Sovelluksen toiminnallisuuden suunnittelu

Painikkeista käynnistyvät tapahtumat (Action) kasataan pääohjelmaluokkaan, josta niitä on helppo hallita ja kaikilla toiminnoilla on samat resurssit käytettävissä. Näitä tapahtumia ovat kaikki toiminnot, jotka vaativat sovelluksessa useamman luokan palveluita. Paikalliset vain yhteen näyttöön vaikuttavat toiminnot voidaan jättää paneeliluokkaan. Näihin toimintoihin kuuluu mm. tekstikentän tyhjennysnappi, joka ei vaikuta mihinkään muuhun kuin samalla näytöllä olevaan tekstikenttään.

Sovelluksen asetukset ovat omissa luokissaan jaoteltuna asetusten tyypin mukaan ja kaikkia asetuksia hallitaan erillisen hallintaluokan *SettingsContainer* kautta. Asetusluokkia ovat *SystemSettings*, *UserSettings* ja *HardwareSettings*. Asetusten hallintaluokka sisältää kaikkien asetusluokkien ilmentymät. Hallintaluokan kautta kaikki paneelit pystyvät käyttämään samoja asetusluokkien ilmentymiä. Asetusluokkien tehtävänä on sovelluksen käynnistyessä ladata tallennetut asetukset tietokoneen rekisteristä käyttäen Java Preferences API:a, palvella muita luokkia tarjoamalla asetustietoja sekä tallentaa muutetut asetukset rekisteriin. Asetusluokissa suoritetaan myös muutamia

loogisia toimintoja, kuten esimerkiksi salasanan salaaminen MD5-algoritmin mukaiseksi tunnisteeksi muuntamalla. Asetusluokkia täydentää *Config*-luokka, joka sisältää sovelluksen pysyviä asetustietoja vakioina. Asetuksia joita *Config*-luokka sisältää ei voida sovelluksesta käsin muuttaa, mutta niitä voidaan helposti muuttaa sovelluksen eri julkaisuversioihin muuttamalla ohjelmakoodista ennen kääntämistä. Asetukset sisältävät muun muassa palvelimen yhteysosoitteen, johon MHG Mobile PC ottaa yhteyttä.

## 5.2 Toteutus

Toteutuksen ensimmäisessä vaiheessa rakennettiin sovelluksen runko, sisältäen näyttöluokat, paneeliluokat ja näyttöjen vaihtomekanismin. Rakenteen toimittua muodostettiin paneelien sisällöt komponenteista. Tässä vaiheessa keskityttiin saamaan oikeat komponentit oikeille paikoilleen. Toiminnallisuutta ei vielä toteutettu. Ensimmäisen vaiheen jälkeen sovellus näytti ulkoisesti valmiilta sovellukselta, mutta ei sisältänyt vielä minkäänlaista toiminnallisuutta, pois lukien näyttöjen vaihtomekanismin. Sovelluksen ollessa jo lähes lopullisessa ulkoasussaan oli aika tehdä välitarkastus ja kerätä mielipiteitä ulkoasusta ja asettelusta MHG Systems Oy:n työntekijöiltä kehityspalaverissa. Palautteen perusteella muutamaa toteutustapaa muuntamalla sovelluksen käyttöä oli entisestään mahdollista helpottaa.

Toisessa vaiheessa toteutettiin sovelluksen tärkeimmät toiminnallisuudet. Toiminnallisuuden toteutuksessa edettiin samassa järjestyksessä kuin toimintojen käyttämisessä. Ensimmäiseksi toiminnallisuudesta toteutettiin yhteydenotto palvelimelle ja kirjautuminen, jonka yhteydessä toteutettiin myös sovelluksen kielen vaihtaminen. Palvelin palauttaa listan kielivaihtoehdoista, joita sovelluksessa voi käyttää kyseisellä käyttäjätilillä. Kirjautumisen jälkeen seuraava toteutettava toiminto oli tehtävienhallinta. Tehtävien lataus muodostuu tehtävälistan lataamisesta, joka sisältää vain tehtävien nimet ja koodit, sekä tehtäväpohjien lataamisesta koodien perusteella. Tehtävälistalle toteutettiin samalla myös tehtävien poisto listalta ja tietokoneelta. Samalla toteutettiin asetusnäytöille toiminnallisuus käyttäjätilin vaihtoon ja ulkonäköasetusten muuttamiseen.

Sovelluksen toiminnallisuudesta rakennettiin seuraavaksi Tehtävä-näytön kolme alasivua. Tehtävä-näytön kuittaus-alasivu on laajin näyttö ja toiminnallisuuden raken-

taminen vei aikaa huomattavasti. Kuittaus-alasivu muodostuu tehtäväpohjan mukaan ja sisältää kyseisen tehtävätyypin mukaiset syöttötietopaneelit. Kartta-alasivulle lisättiin tehtäväpohjan mukana toimitettavan kartan piirto ja ohje-alasivulle lisättiin tekstikenttä ohjeelle.

Muita näyttöjä täydennettiin tärkeimpien toimintojen toteuttamisen jälkeen. Toiminnallisuutta lisättiin GPS-paikannuksella. Tehtävä-näytön kartta-alasivulle lisättiin GPS paikkatietojen näyttö ja oman sijainnin osoittavan nuolen piirtäminen karttakuvan päälle. Viesti-näytön toiminnallisuus rakennettiin sisältäen GPS-paikkatietojen näytön.

Paikannus-näyttö toteutettiin, vaikka näytön hyödyllisyys lopullisessa sovelluksessa ei ollut varmaa. Paikannus-näytölle oli tiedossa merkittävä muutoksia myöhemmin sovelluksen kehittyessä, mutta tässä vaiheessa näytön sisällöksi jätettiin ainoastaan GPS-sijainti. Julkaisuversiosta Paikannus-näyttö piilotettiin kokonaan.

Sovelluksessa käytetyt MHG Systems Oy:n ja MHG Mobile PC:n logot sekä näyttöjen vasemman laidan taustakuva tulivat yrityksen sisältä, eivätkä ole omaa tuotosta. Kaikki muut näyttöjen nappien ikonit ovat Everaldo Coelhon Linux-käyttöjärjestelmän KDE-työpöytäympäristölle luomasta Crystal ikonipaketista. Crystal ikonipaketti on *Lesser Gnu Public License* (LGPL) -lisensoitu ja näin vapaasti käytettävissä sovelluksissa. Tarvittavan kuvamateriaalin tullessa muualta, oli mahdollista keskittyä täysin sovelluksen suunnitteluun ja ohjelmointiin

Sovelluksen asennuspaketin tekoon käytettiin Caphyonin valmistamaa Advanced Installer sovellusta. Merkittävin tekijä valintaan oli, että sovelluksesta on saatavilla ilmaisversio jota voi käyttää kaupallisissa sovelluksissa. Ilmaisversio on varsin yksinkertainen ja sillä voi tehdä vain englanninkielisiä asennuspaketteja, mutta ajaa asiansa ja MHG Mobile PC:n jar-tiedosto ja muut oheistiedostot saadaan asennettua helposti pikakuvakkeineen. Sovelluksen kaupallisella versiolla on mahdollista kääntää jar-tiedostosta suoraan ajettava exe-tiedosto sekä asennusohjelman kielen voi vaihtaa. Käyttämällä aluksi ilmaista yksinkertaistettua versiota, tarjoaa helpon tavan siirtyä myöhemmin käyttämään saman sovelluksen kaupallista versiota, mikäli sille tulee tarvetta.

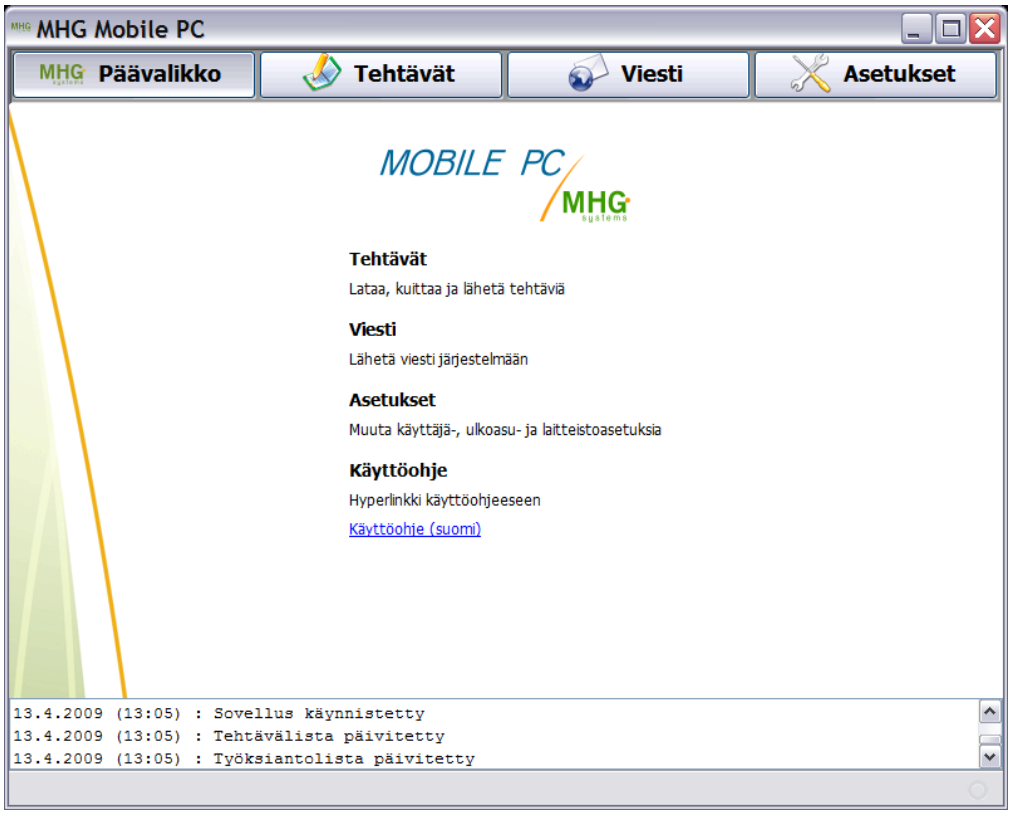
### 5.2.1 Käyttöliittymän toteutus

Paneelirakenteen avulla saadaan helppo muokattavuus ja paneelien hallitseminen näyttöinä toteutettiin ei-visuaalisilla näyttöluokilla, jotka toimivat paneelikokoelmina, sisältäen myös hieman toiminnallisuutta. Näyttöjen vaihtamisesta huolehtii *ScreenController*-luokka, jonka ilmentymä ladataan sovelluksen käynnistyessä ja samaa ilmentymää käytetään, kunnes sovellus suljetaan. Näyttöjen hallinnassa käytetään liityntäluokkaa (interface) *IScreen*, joka määrittelee näyttöluokkien sisältämät metodit. *IScreen*-liityntäluokka mahdollistaa näyttöjen hallinnan ilman tarkkaa tietoa näytön tarkasta luokasta, koska jokainen näyttöluokka toteuttaa *IScreen*-liitynnän. Näyttöluokkien hallinta toteutetaan julkisilla vakioilla (public static final IScreen), joihin näyttöluokat ladataan, kun *ScreenController*-luokan ilmentymä otetaan käyttöön. Näyttöluokkien ilmentymät pysyvät samoina, mutta näyttöjen sisältämiä paneeleita voidaan ladata uudelleen. Näytöt pysyvät taustalla muistissa ja näytöstä toiseen vaihtaminen on nopeaa. Etuna saadaan myös näyttöä vaihdettaessa syötettyjen tietojen säilyminen näytöllä, ilman ylimääräistä ohjelmakoodia.

Paneelien avulla näytön toiminnot voidaan sijoittaa helposti samaan paikkaan joka näytöllä. Sovelluksen käyttö nopeutuu kun mm. navigointi- ja toimintopainikkeet ovat aina samoilla paikoilla. Samalla voidaan käyttää samoja paneeleita eri näytöissä, jolloin sovelluksen ohjelmointi on nopeampaa ja ohjelmakoodin hallinta helpompaa. Paneelirakenne mahdollistaa myös erilaisten käyttöliittymien rakentamisen helposti ja nopeasti paneeleita vaihtamalla.

**Päävalikko-näyttö**

Päävalikko sisältää lyhyen kuvauksen sovelluksen toiminnoista. Käyttäjä ohjataan pääsivulle ensimmäisellä kerralla käynnistettyään sovelluksen ja kirjauduttuaan sisään. Pääsivulla sijaitsee linkki sovelluksen käyttöohjeeseen, jota klikkaamalla MHG Mobile PC -sovellus yrittää avata Internetissä sijaitsevan käyttöohjeen käyttäen tietokoneen oletukseksi asetettua Internet-selainta. Käyttöohje sijaitsee yrityksen palvelimella, jotta se voidaan korvata helposti uudella versiolla päivityksen yhteydessä. Sovelluksen Päävalikko-näyttö on kuvassa 5 seuraavalla sivulla.



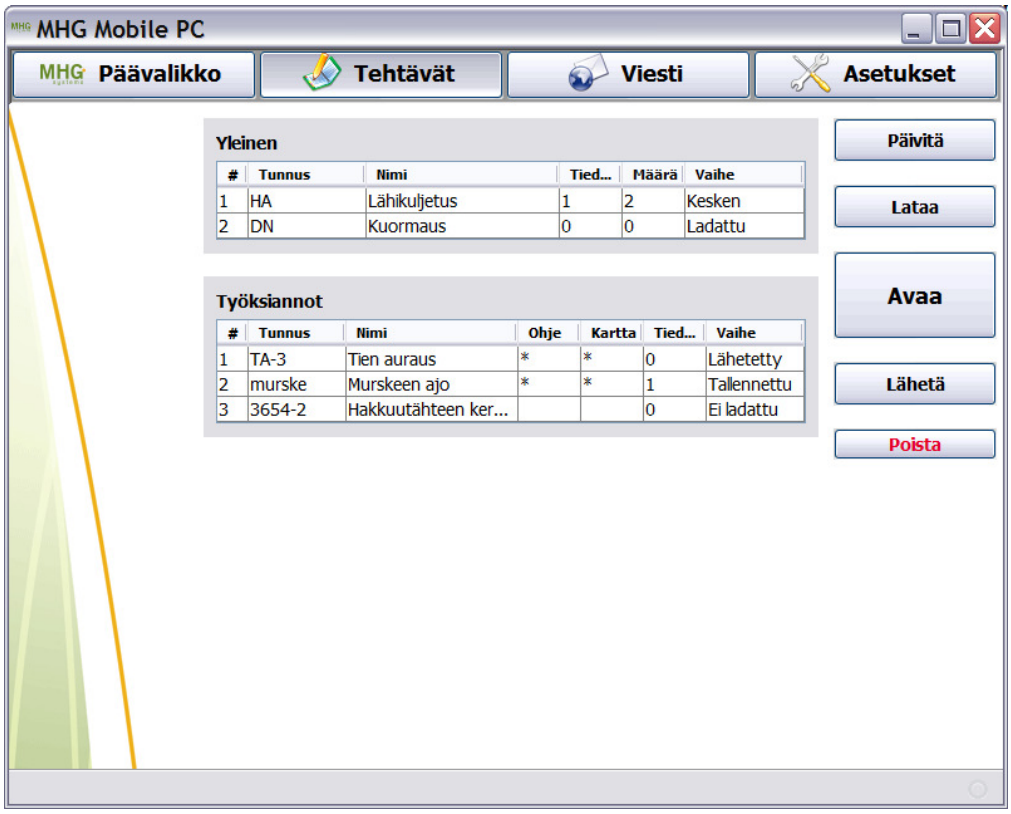

**KUVA 5 Sovelluksen päävalikko**

### Tehtävät-näyttö

Tehtävät-näytöllä sijaitsee kaksi taulukkoa tehtävälistoille. Yleiset tehtäväpohjat ovat listattuna ylemmässä taulukossa ja työksiannot ovat listattuna alemmassa taulukossa. Taulukoista saa selville helposti tehtävien tilan sekä tehtävien tärkeimpiä tietoja. Tiedoissa näkyvät mm. onko tehtäviin liitetty karttaa ja ohjetta. Tehtäviin lisättyjen tiedostojen sekä tietueiden määrä yleisissä tehtävissä näkyy taulukossa.

Taulukoiden rakenne on toteutettu dynaamisella rakenteella. Taulukoiden rivien määrä muuttuu automaattisesti, kun taulukkoon lisätään rivejä tai siitä poistetaan rivejä. Visuaalisen ilmeen säilyttämiseksi taulukkoon jää yksi tyhjä rivi viimeisen rivin poistamisen jälkeen. Taulukko muuttaa myös samalla paneelin kokoa joka sisältää kyseisen taulukon ja näytön ryhmittely pysyy ehjänä. Taulukon sarakkeet on myös toteutettu dynaamisesti. Yleiset tehtävät sekä työksiannot taulukoissa on täysin samat sarakkeet, mutta vain osa sarakkeista näytetään. Yleisissä tehtävissä ei tällä hetkellä voi käyttää karttaa tai ohjetta, joten kyseiset sarakkeet on piilotettuina yleiset tehtävät taulukossa. Työksiannot ovat aina kertakuitattavia tehtäviä, joten ne eivät voi sisältää montaa tie-

tuetta ja näin ollen tietueiden määrää ilmoittava sarake on piilotettu työksiannot taulukosta. Tehtävien hallinta Tehtävät-näytön avulla kuvattuna kuvassa 6.




**KUVA 6 Tehtävät-näyttö**

Tehtäviä hallitaan oikeassa laidassa olevilla toimintonapeilla, jotka on järjestetty toimintajärjestykseen ylhäältä alaspäin. Ylimpänä sijaitseva päivitä-nappi suorittaa tehtävälistan haun palvelimelta. Tehtävälistan päivitys lisää listalle uusien tehtävien tunnukset ja nimet sekä merkkaa tehtävän tilaksi ”Ei ladattu”.

Listalta valitut lataamattomat tehtävät ladataan painamalla lataa-nappia. Mikäli ladattu tehtävä sisältää kartan ja ohjeen, tulee siitä merkintä taulukkoon. Palvelin lähettää aina sekä kartan että ohjeen, mutta molemmille on oma merkintä tehtävälistassa. MHG Mobile -matkapuhelinasiakasohjelma pystyy käsittelemään karttaa ja ohjetta vain parina, ja samoin palvelin lähettää aina kartan ja ohjeen parina. MHG Mobile PC -sovellus haluttiin toteuttaa joustavammin kuin olemassa oleva nykyinen järjestelmä, mahdollistaen kartan ja ohjeen käyttämisen erillisinä, mikäli palvelin muutetaan myöhemmin lähettämään ne erikseen. MHG Mobile PC:ssä kartta ja ohje käsitellään eril-

lisinä tehtävän lisätietoina, ja sovellus pystyy vastaanottamaan kumman tahansa erikseen tai molemmat yhdessä.

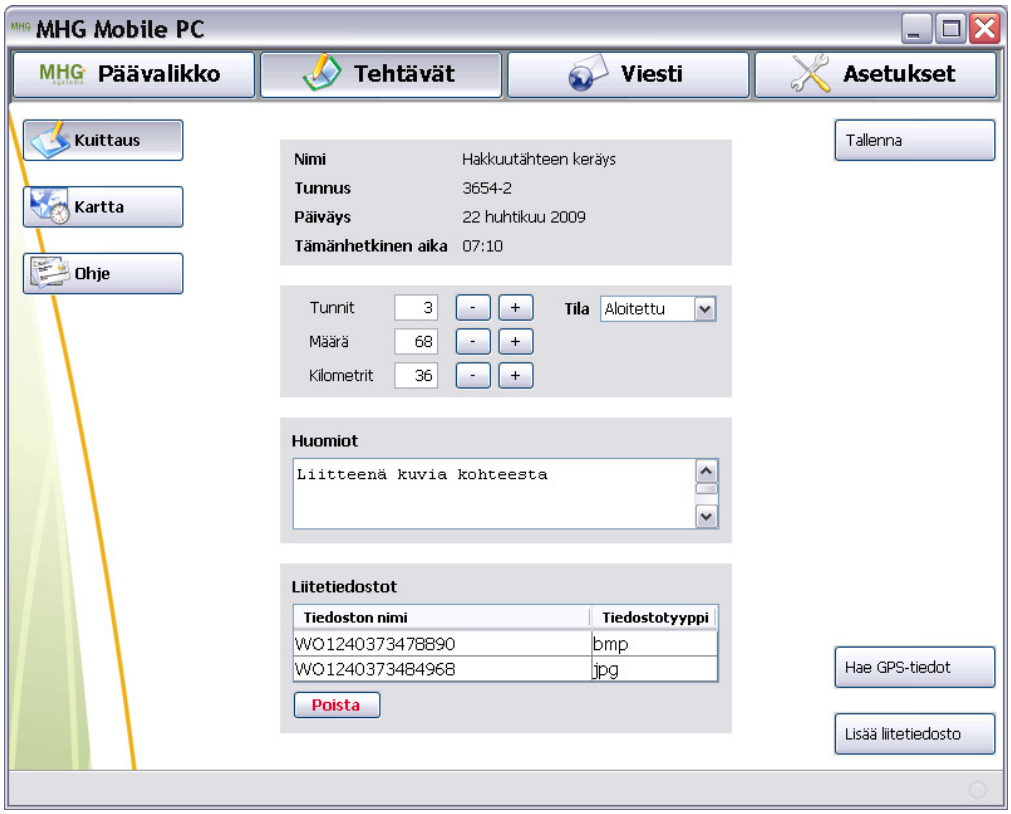
Valittuna oleva tehtävä avataan tarkasteltavaksi ja kuitattavaksi painamalla avaa -nappia. Mikäli useampia tehtäviä on valittuna, avataan listalla ylimmäisenä oleva tehtävä. Sovellusta käytettäessä useimmiten käytetty tehtävälistan toiminto on tehtävän avaaminen, joten avaa-nappi on selvyyden vuoksi kooltaan kaksi kertaa suurempi kuin muut toimintonapit.

Tallennetut tehtävien kuittaukset lähetetään toiminnanohjausjärjestelmään painamalla lähetä-nappia. Tehtäviä lähetettäessä sovelluksen alapalkissa on näkyvillä lähetyksen etenemispalkki ilmoittamassa lähetyksen etenemisestä. Mikäli useita tehtäviä on valittu, lähetetään tehtävät yksi kerrallaan listan järjestyksessä.

Tehtävien poistaminen listalta tapahtuu valitsemalla poistettavat tehtävät ja painamalla poista-nappia. Poistaminen vahvistetaan erillisellä ponnahdusikkunalla, jossa ilmoitetaan poistettavien tehtävien nimet ja kysytään käyttäjältä varmistusta tehtävien poistamiseen. Tehtävien poisto poistaa kaikki valitut tehtävät, välittämättä tehtävien tilasta tai tehtäviin tallennetuista tiedoista. Poistamisen yhteydessä poistetaan myös kaikki tehtäviin tallennetut kuittaukset liitetiedostoineen. Suljettaessa sovellusta tehtävät joiden tila on ”Ei ladattu” poistetaan listalta. Kaikki ladatut, tallennetut tai lähetetyt tehtävät säilyvät listalla myös sovelluksen seuraavalla käyttökerralla. Käytettävyyden helpottamiseksi poista-nappi on tarkoituksella muita nappeja pienempi ja napin teksti on punaisella värillä.

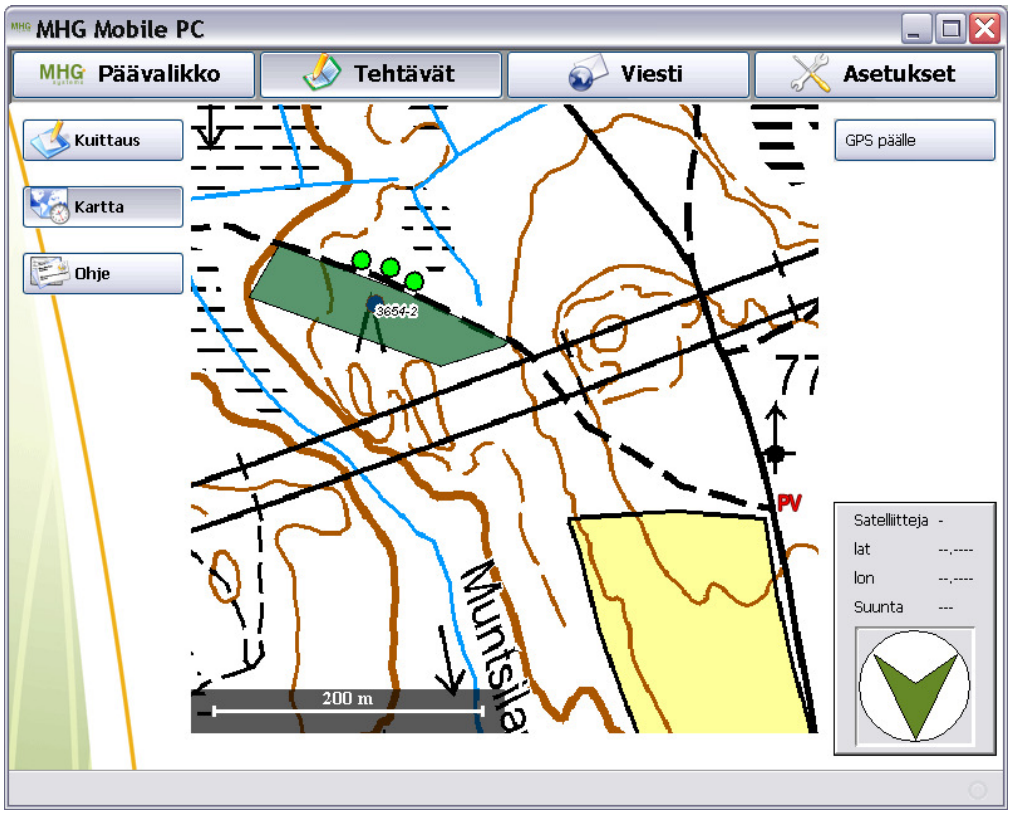
**Tehtävä-näyttö**

Tehtävä avautuu Tehtävät-näytön tehtävälistalta Tehtävä-näytölle, joka sisältää kolme alasivua. Tehtävä aukeaa aina kuittaus-alasivulle. Alasivua voidaan vaihtaa vasemmassa reunassa sijaitsevista napeista. Kartan tai ohjeen napit voivat olla himmennettyjä ja käytöstä poistettuja, mikäli kyseisellä tehtävällä ei ole karttaa tai ohjetta. Kartta ja ohje sisältävät tietoja tehtävästä sovelluksen käyttäjälle, ja kuittaus-alasivulta voidaan syöttää kuittaustietoja.



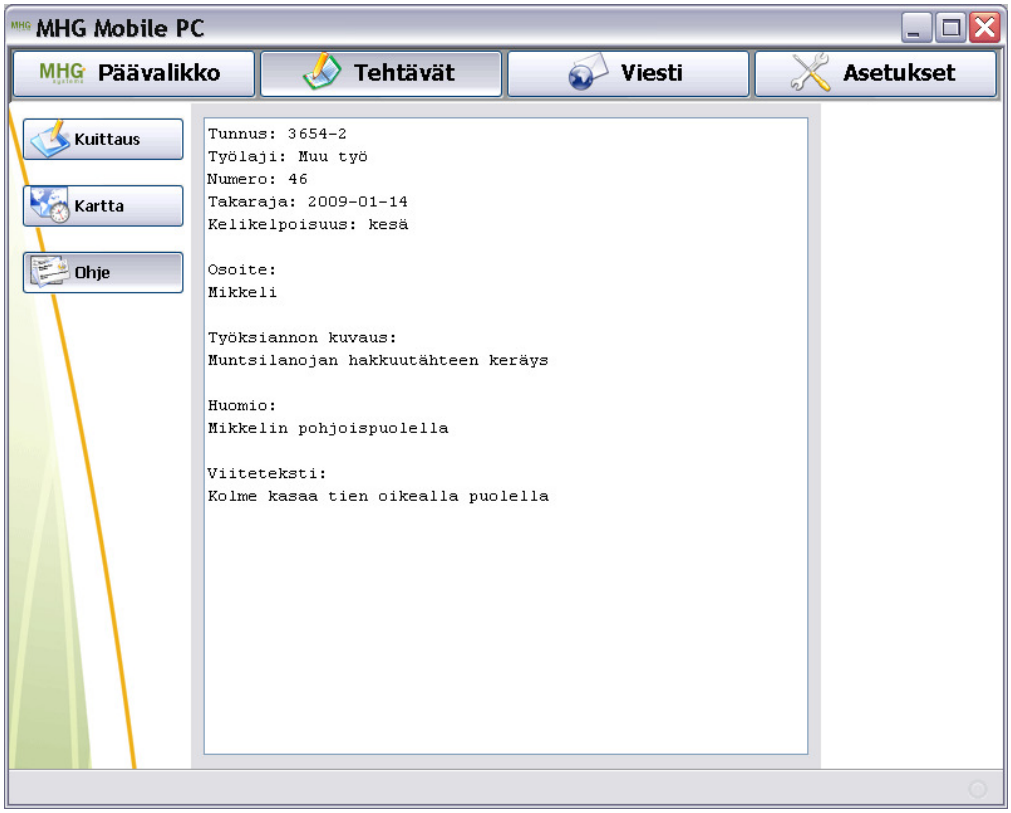

**KUVA 7 Tehtävä-näytön kuittaus-alasivu**

Kuvassa 7 näkyvällä Tehtävä-näytön kuittaus-alasivulla näytetään käyttäjälle tehtävän tunnistetiedot. Kuittaus alasivulla käyttäjä täyttää tehtävästä vaaditut kuittaukset. Osa kuittauksista on pakollisia ja osa valinnaisia, tehtävän kuitattavat tiedot voidaan määrätä Internet-palvelusta tehtävää lisättäessä. Kuitattavia arvoja on kolmea erilaista. Kuitattava arvo voi olla tekstiä tai numeroarvo tai kuitattava tieto valitaan palvelimelta saatavasta arvolistasta, joka esitetään näytöllä pudotusvalikkona. Kuittaukseen on mahdollista lisätä vapaamuotoista tekstiä, GPS-sijainti sekä liitetiedostoja. Näyttö on sisällöltään ja toiminnallisuudeltaan laajin koko sovelluksessa. Kuitattavat tiedot jaetaan erillisiin visuaalisiin paneeleihin, jotta käyttö on helppoa ja selkeää. Tehtävän tyypistä riippuu mitä paneeleita käyttäjälle näytetään.




**KUVA 8 Tehtävä-näytön kartta-alasivu**

Internet-käyttöliittymässä tehtävälle tallennettu kartta näkyy käyttäjälle Tehtävä-näytön kartta-alasivulla. Kartta-alasivu on esitetty kuvassa 8. Karttasivulla voidaan käyttää myös GPS-paikannusta kohteen löytämiseen, jolloin kartan päällä on näkyvissä nuoli osoittamassa käyttäjän sijaintia, mikäli käyttäjän on kartan alueella. Suurimmalla osalla sovelluksen käyttäjistä on käytössään navigaattori, joten käyttäjällä ei ole tarvetta navigoida tehtävän kartan avulla. GPS-paikannus on sovelluksessa toteutettu kevyesti, vain lisäarvoa antavana palveluna.

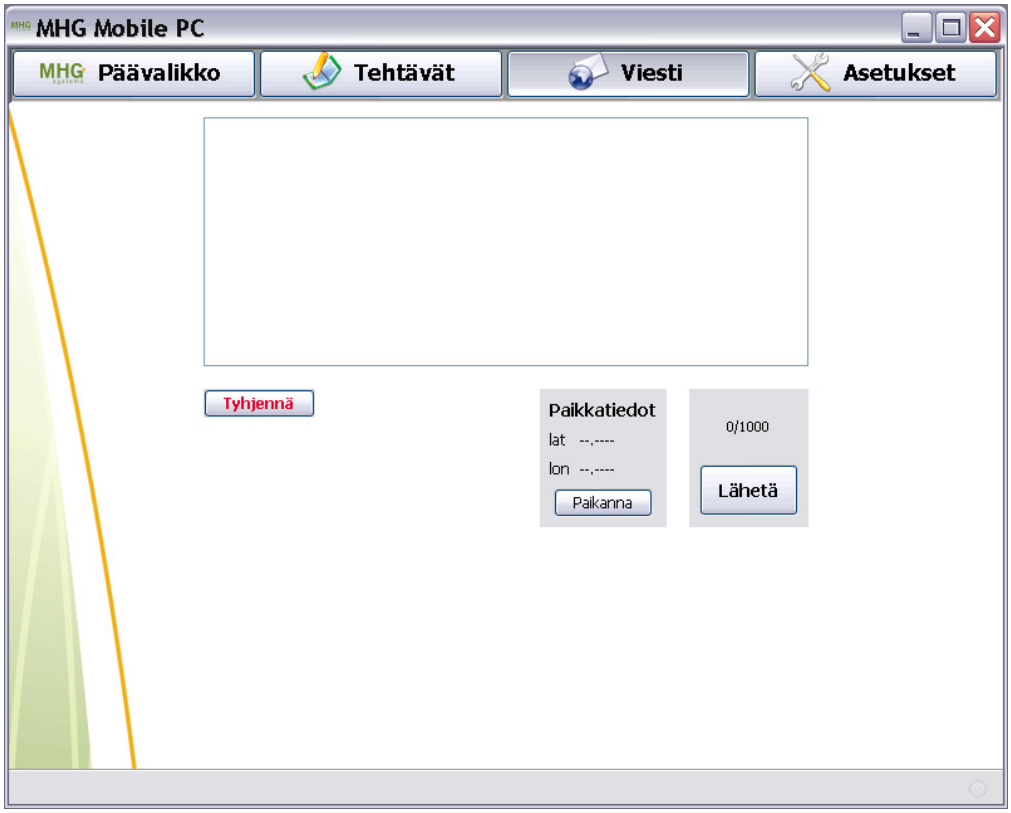



**KUVA 9 Tehtävä-näytön ohje-alasivu**

Tehtävä-näytön ohje-alasivulla on kartan mukana lähetettävät ohjeet, jotka palvelin kokoaa automaattisesti Internet-käyttöliittymässä tehtävälle syötetyistä tiedoista. Malli tiedoista joita ohje on esitelty kuvassa 9. Ohje voi sisältää muitakin tietoja kuin mitä on esitetty kuvassa. Ohjeet on suunniteltu avuksi kartan käyttöön ja palvelin ei lähetä ohjeita mikäli tehtävällä ei ole karttaa.

### Viesti-näyttö

Viesti-näytöllä voidaan kirjoittaa viesti ja lähettää se järjestelmään pääkäyttäjän luettavaksi. Viestiin voidaan liittää GPS-sijainti. Viesti lähetetään Internetyhteyttä käyttäen. GPS-sijaintia ei tarvitse kirjoittaa viestin tekstiosaan, sillä sijainti lisätään lähetettyyn viestiin erillisenä tietona. Kirjoitettu viesti säilyy näytöllä, kunnes viesti lähetetään tai tyhjennetään tyhjennä-napista. Sovellus tallentaa viestin automaattisesti rekisteriin, joten keskeneräinen viesti säilyy sovelluksessa Viesti-näytöllä, vaikka sovellus sammutettaisiin välillä. Viesti-näyttö kuvattuna seuraavalla sivulla kuvassa 10.




**KUVA 10 Viesti-näyttö**

### Paikannus-näyttö

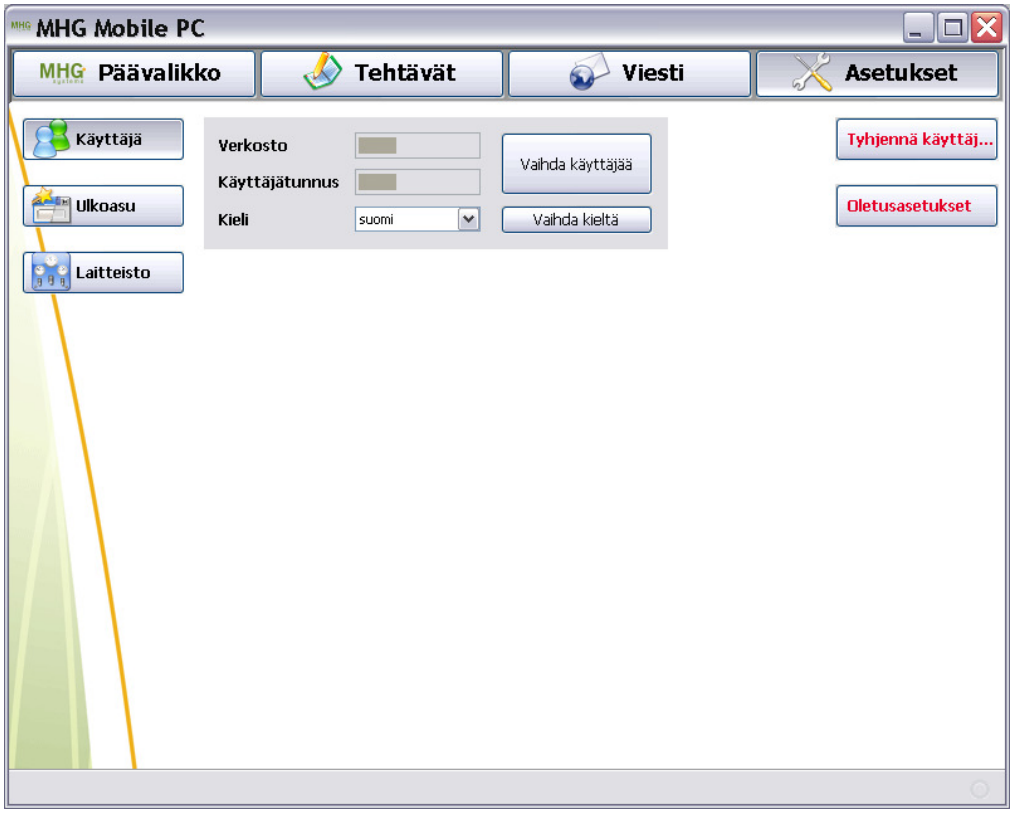
Sovellusta toteutettaessa kävi ilmi erillisen paikannus tarpeettomuus, koska käyttäjä voi tarkistaa oman sijaintinsa helposti esimerkiksi Viesti-näytöltä. Paikannus-näyttö poistettiin ensimmäiseen julkaisuversioon mennessä ja tästä syystä kaikissa kuvankaappauksissa sovelluksesta päänäyttöjen navigointipainikkeita on näkyvillä vain neljä kappaletta, eikä viittä, kuten alun perin oli tarkoitus. Paikannus-näyttö toteutettiin, mutta julkaisuversiossa näyttö on piilotettu, poistamalla painike päänäyttöjen navigointipainikkeista sekä näyttöön viittaava kuvaus päävalikosta. Paikannus-näytölle on tiedossa uusiokäyttöä.

### Asetukset -näyttö

Päänäyttöjen navigoinnin asetukset-painikkeesta avautuu käyttäjäasetukset-alasivu, ja muille alasivuille voidaan vaihtaa vasemman laidan alasivujen navigointipainikkeista. Kaikissa asetuksien alasivuissa on oikeassa laidassa näkyvillä kaksi toimintopainiketta. Ylimmäinen toimintopainike ”Tyhjennä käyttäjäasetukset” poistaa rekisteristä

kaikki sovelluksen tallentamat tiedot sekä poistaa tietokoneelta kaikki käyttäjän tallentamat kuittaukset. Tietojen poiston jälkeen sovellus sammuttaa itsensä ilmoittaen tilanteesta käyttäjälle ponnahdusikkunalla. Seuraavalla käynnistyskerralla sovellus käynnistyy täysin samoin kuin ensimmäisellä käynnistyskerralla asennuksen jälkeen. Toinen toimintopainike ”Oletusasetukset” muuttaa ainoastaan sovelluksen ulkoasu- ja laiteasetukset oletusasetuksiin, mutta käyttäjä pysyy toimenpiteen jälkeen kirjautuneena sovellukseen.

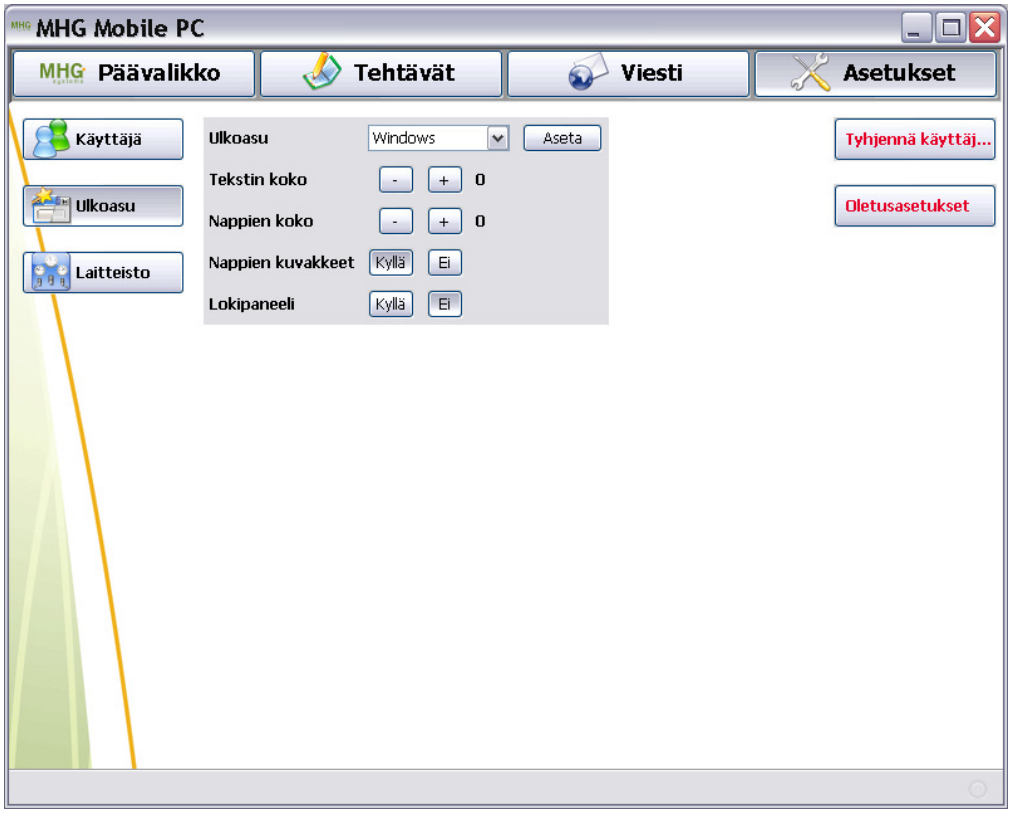
Käyttäjäasetuksissa on näkyvillä sisään kirjautuneen käyttäjän verkostotunnus ja käyttäjätunnus, kenttinä joita ei voida muokata. Kenttien vieressä sijaitsee painike, joka avaa käyttäjänvaihdon erilliseen ponnahdusikkunaan. Käyttäjätietojen alapuolella sijaitsee pudotusvalikko josta voidaan valita sovelluksen kieli. Kielivaihtoehdot ovat verkostokohtaisia ja sisään kirjautuessa sovellus tallettaa palvelimen tarjoamat kielivaihtoehdot, jotka tämän jälkeen näkyvät ponnahdusikkunassa. Kuvassa 11 on sovelluksen käyttäjäasetukset. Verkosto- ja käyttäjätunnukset ovat tarkoituksella peitettyinä kuvassa.




**KUVA 11 Asetukset-näytön käyttäjäasetukset-alasivu**

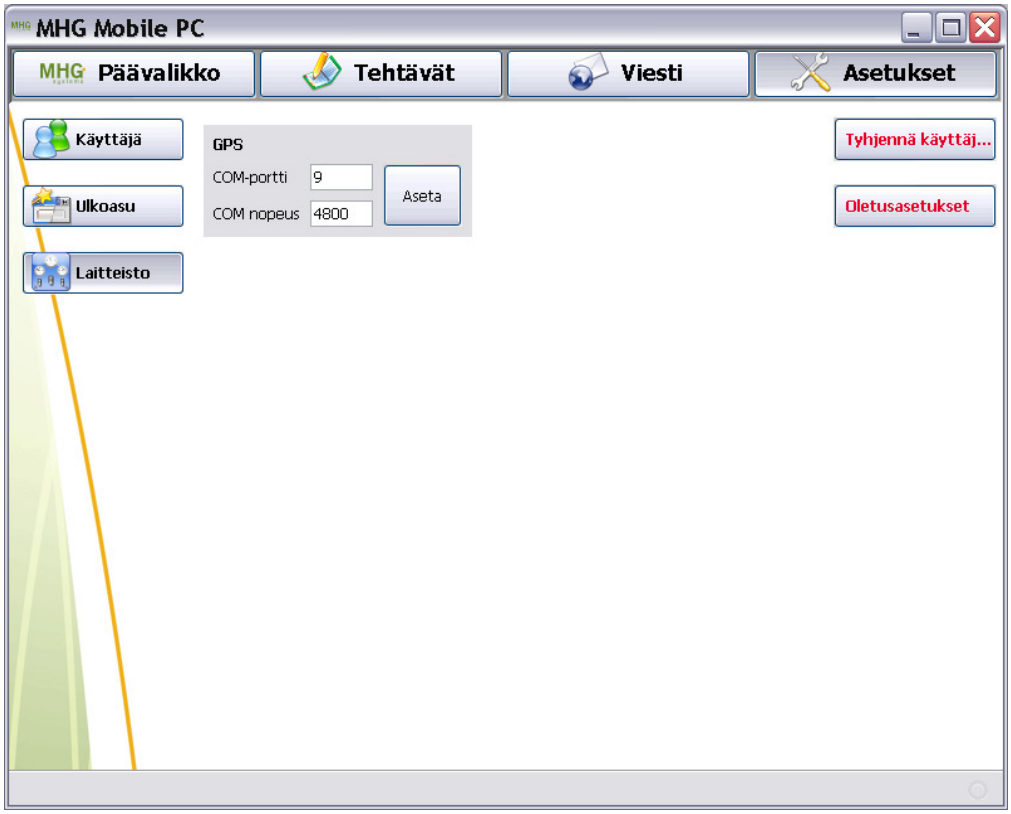
Ulkoasuasetuksissa voidaan vaihtaa sovelluksen ulkoasua (Look and Feel) valitsemalla uusi ulkoasu pudotusvalikosta. Ulkoasu-alasivusta on malli kuvassa 12. Pudotusvalikon sisältö riippuu käytettävästä tietokoneesta ja käyttöjärjestelmästä. Sovelluksen käynnistyessä tutkitaan mahdolliset ulkoasuvaihtoehdot, jotka listataan pudotusvalikkoon.

Tekstin koon muuttaminen tapahtuu plus- ja miinusnapeilla, jotka muuttavat kokoa suuremmaksi tai pienemmäksi. Positiivinen luku ilmoittaa oletuskokoa suurempaa tekstin kokoa ja negatiivinen luku pienempää. Tekstin oletuskoko voidaan määritellä *Config*-luokan avulla. Nappien koon muuttaminen tapahtuu samalla tavoin kuin tekstin koon muuttaminen. Napin koon muuttaminen muuttaa napin koon lisäksi myös napin sisältämän tekstin kokoa. Navigointinapit sekä toimintonapit ovat kiinteän kokoisia ja niiden koko ei muutu, ainoastaan tekstin koko napin sisällä muuttuu. Nappien kuvakkeiden piilottamiseen on asetuksissa ”Kyllä” ja ”Ei” painikkeet. Lokipaneelin näkyvyyttä voi muuttaa käyttäen samanlaisista painikkeista, mutta piilotettunakin paneeli ilmestyy näkyviin uuden viestin saapuessa.




**KUVA 12 Asetukset-näytön ulkoaseasetukset-alasivu**

Laitteistoasetuksissa ei ole vielä ensimmäisessä julkaisuversiossa mitään muita muutettavia asetuksia kuin GPS-laitteen COM -portti ja sarjaliikennenopeus, kuten kuvasta 13 voi hyvin havaita. Syöttövirheiden välttämiseksi COM-portti syötetään puhtaana numeroarvona, esim. COM9 valitaan antamalla COM-portin arvoksi ”9”. Syöttökentissä on tarkistukset virheellisille arvoille, eikä sovellus hyväksy niitä. Mikäli syöttää virheelliset arvot, sovellus palauttaa oletusarvot.




**KUVA 13 Asetukset-näytön laitteistoasetukset-alasivu**

### 5.2.2 Palvelimen ja asiakasohjelman välinen rajapinta

MHG Mobile PC -asiakasohjelma kommunikoi palvelimen kanssa täysin samoin kuin MHG Mobile -matkapuhelinasiakasohjelma. Käyttämällä jo olemassa olevaa rajapintaa on mahdollista pitää palvelimen lähdekoodi mahdollisimman vähillä muutoksilla, jolloin MHG Mobile PC:n käyttöönotto on mahdollisimman joustavaa. Sama palvelin pystyy kommunikoimaan vähillä muutoksilla sekä MHG Mobilen että MHG Mobile PC:n kanssa.

Palvelimen ja MHG Mobile asiakasohjelman välinen tiedonsiirto on toteutettu käyttämällä tilanteeseen kulloinkin sopivinta tiedonsiirtomuotoa. Palvelin lähettää tehtäviä asiakasohjelmalle standardimuodossa. Tehtävien lisäksi palvelin lähettää asiakasohjelmalle tehtäviin liitetyt ohjeet sekä karttakuvan. Tiedonsiirto MHG Mobile PC:ltä palvelimelle täytyi toteuttaa muuttamatta MHG Mobilen käyttämää tiedonsiirtomuotoa. Viestit asiakasohjelmalta palvelimelle välitetään http-viesteinä.

Kommunikointi MHG Mobile PC:n ja palvelimen välillä toteutettiin ohjelmoimalla uudelleen MHG Mobilen Java ME:llä toteutetut luokat Java SE:llä MHG Mobile PC:n. Viestien välitystä testattaessa käytettiin kumpaakin asiakasohjelmaa rinnakkain yhtenevän toimivuuden aikaansaamiseksi. Sovelluksen tiedonsiirtoa testattiin myös oikeita käyttöolosuhteita vastaavalla tiedonsiirtonopeudella, käyttäen testilaitteessa ollutta GPRS-modeemia. Käyttäjä pidetään ajan tasalla tiedonsiirron etenemisestä sovelluksen tilariville ilmaantuvan etenemispalkin avulla.

### 5.2.3 Kartta ja GPS-toiminto

Palvelin muodostaa karttakuvan, joka toimitetaan asiakasohjelmalle tehtävän mukana. Kartta on kiinteän kokoinen ja mikäli kartta ei mahdu sille varattuun tilaan, ilmestyvät vierityspalkit, joilla karttaa voi liikuttaa. MHG Mobile PC -asiakasohjelmassa piirretään karttakuvan päälle nuolen muotoinen polygoni, joka kuvaa käyttäjän sijaintia. Sijainti saadaan GPS-laitteelta, mikäli sellainen on käytettävissä ja polygonin sijainti karttaruudulla lasketaan kartan mukana toimitettavien bounding box (bbox) rajaarvojen perusteella.

GPS-ominaisuus toteutettiin käyttäen hyväksi avoimen lähdekoodin GPSylon-kirjastoa, joka käyttää sisäisesti avoimen lähdekoodin RXTX-sarjaliityntäkirjastoa. Molemmat kirjastoista ovat *Lesser Gnu Public License* (LGPL) -lisensoituja, joka sallii kirjastojen käytön kaupallisessa sovelluksen ilman sovelluksen lähdekoodin julkistamista. GPSylon-kirjasto on täysin toimiva, joskin sen kehitys vaikuttaa pysähtyneen. Kirjasto kuitenkin täyttää tämän projektin vaatimukset ja LGPL-lisensoituna kirjastoa on mahdollisuus tarvittaessa laajentaa. GPSylon-kirjaston ominaisuuksia on hyvin vähän dokumentoitu ja suuri osa toiminnoista täytyi selvittää kokeilemalla, sekä hyödyntäen vähäisiä esimerkkejä. Suurin hyöty GPS-toiminnosta on käyttäjän sijain-

nin raportointi järjestelmään. Käyttäjän oman sijainnin näyttäminen kartalla ja GPS:n avulla navigointi koettiin olevan toissijainen ominaisuus ja se toteutettiin varsin kevyesti.

Merkittävä puute GPSylon-kirjastossa oli, ettei sen avulla voinut laskea eri sijaintien välisiä etäisyyksiä. MHG Mobile puolestaan käyttää Nokian GPS API:a, joka tarjoaa myös etäisyyden laskemisen oman sijainnin ja toisen sijainnin väliltä. Etäisyyttä omasta sijainnista varastolle ei voitu toteuttaa helposti, eikä sitä julkaisuversioon ehditty ollenkaan toteuttaa. Etäisyyden laskeminen varastolle on julkaisuversion merkittävin ero MHG Mobilen ominaisuuksiin, kaikilta muilta osin MHG Mobile PC:n ominaisuudet ovat laajemmat kuin MHG Mobilen. Samalla etäisyyden laskeminen jäi yhdeksi ensimmäisistä kehityttävistä asioista sovelluksessa.

# 6 YHTEENVETO

Teorian selvittäminen MHG Mobile PC:n suunnittelua ja toteutusta varten tapahtui samaan aikaan, kun sovellusta ohjelmoitiin. Sovelluksen suunnittelu sujui ilman suurempia ongelmia. Käyttöliittymän rakenne muodostui nopeasti, mutta yksityiskohtien säätämisessä kesti kauan aikaa. Aikaa toteutuksessa vei varsinkin komponenttien sijaintien ja kokojen saaminen skaalautumaan sulavasti erikokoisille näytöille.

Muutamia pieniä ongelmia tuli vastaan MHG Mobile PC:n testauksessa, jotka täytyi korjata ennen julkistamista. Joitakin ongelmia esiintyi myös palvelinpäässä, joka oli ohjelmoitu vastaamaan MHG Mobilelle, koska MHG Mobile PC pystyi saamaan aikaan virhetilanteita, jotka olivat MHG Mobilelle mahdottomia. MHG Mobile PC:n testaaminen oli GPS-toimintoa lukuun ottamatta nopea suorittaa. GPS:n testaus ei onnistunut sisätiloissa, joten jokaista testausta varten oli sovellus käännettävä, asennettava kannettavaan tietokoneeseen ja käytävä ulkona testaamassa. Mikäli sovelluksesta löytyi korjattavaa, oli palattava työpisteeseen tekemään muutos ohjelmakoodiin ja toistettava sama testaustoimenpide. GPSylon GPS-kirjaston yhdistämisessä olikin alkuun hieman ongelmia heikon dokumentaation takia.

Ohjelmakoodin määrä paisui valtavaksi, omia luokkia tuli yli 20 kpl ja tuhansia rivejä ohjelmakoodia. Yksistään Tehtävä-näytön kuittaus-alasivun keskimmäisessä paneelissa on omaa toiminnallista ohjelmakoodia noin 1500 riviä ja lisäksi noin 3000 riviä GUI Builderin lisäämää käyttöliittymän koodia näytölle piirretyistä komponenteista. Pääohjelmaluokka on toinen suurempi luokka, siinä on omaa ohjelmakoodia yli 2000 riviä. Muut luokat olivat huomattavasti pienempiä. Käyttöliittymän ohjelmakoodi oli kokonaan itse tehtyä, kun toiminnallisesta koodista osassa oli mahdollista käyttää aiemmin tehtyjä luokkia. Ohjelmakoodiin jäi toki paljon siistimistä, koska suuri osa suunnittelusta tapahtui toteutuksen yhteydessä. Kuvien tullessa muualta, oli mahdollista keskittyä pelkästään ohjelmointiin ja samalla sovellukselle tuli ammattimainen ilme käyttämällä ammattilaisten piirtämiä ikoneita. Yhteisellä työpanoksella on mahdollista saada näyttävämpi ja toimivampi sovellus kuin mitä yksi ihminen voisi yksinään saada aikaiseksi.

MHG Mobile PC oli ensimmäinen Java SE:llä ohjelmoimani Windows-sovellus. Sovellus täytti sille asetetut vaatimukset ja julkaistiin MHG Mobilen rinnalle toiseksi asiakasohjelmaksi. Insinöörityön tutkimusongelmaan saatiin ratkaisu, matkapuhelinsovellus voidaan kohtuullisella vaivalla muuntaa helposti käytettäväksi Windows-sovellukseksi. Muunnos onnistuu varsin vaivattomasti käyttämällä Swing application frameworkia, jonka hyödyt tulivat esille MHG Mobile PC -sovellusta toteutettaessa. Parannettavaa ja korjattavaa sovellukseen toki vielä jäi, kaikkea ei ollut mahdollista hioa aivan loppuun asti aikarajan puitteissa. Seuraava kehitettävä ominaisuus tulee olemaan seuranta-ominaisuus (tracking), jossa tullaan hyödyntämään käyttämättä jäänyttä Paikannus-näyttöä. Seurantapalvelua käytettäessä asiakasohjelma lähettää sijaintitietoja palvelimelle tasaisin väliajoin ja Internet-palvelusta voidaan seurata kartan avulla sovellusta käyttävien käyttäjien liikkumista ja viimeisimpiä sijainteja. Insinöörityö tuotti hyvän alustan sovelluksen jatkokehittämistä silmälläpitäen.

Muutama asia tuli esille insinöörityötä tehdessä, jotka mahdollistivat matkapuhelinsovelluksen muuntamisen Windows-sovellukseksi lyhyen aikarajan sisällä ja vähäisellä kokemuksella Windows-sovelluksen ohjelmoinnista. Merkittävin hyöty oli Swing application frameworkin käyttö, joka tarjosi valmiin rungon sovellukselle. Tärkeä tekijä käyttöliittymän suunnittelussa ja toteutuksessa oli vapaus toteuttaa käyttöliittymä parhaalla mahdollisella tavalla ilman suurempia rajoitteita. Olemassa olevan ohjelmakoo-

din käyttö joko suoraan tai hieman muokkaamalla helpotti ohjelmointiurakkaa. Tarkka tieto muunnettavan sovelluksen toiminnoista oli eduksi ohjelmoitaessa toimintoja toiselle ohjelmointikielelle. Valmiiden kuvien käyttö edesauttoi myös osaltaan sovelluksen nopeaa toteutusta ja lisäsi sovelluksen näyttävyyttä. Ilman näitä mainittuja hyötyjä, olisi ohjelmointiurakka ollut huomattavasti hankalampi.

MHG Mobile PC -sovelluksen suunnittelu onnistui ja suunnittelun tuloksena syntyi toimiva asiakasohjelma. MHG Mobile PC:n julkistus tapahtui Saksan Leipzigissä järjestetyillä Enertec-messuilla 27. - 29.1.2009. MHG Mobile PC -sovellus on tästä lähtien ollut saatavilla vaihtoehtona MHG Mobile -matkapuhelinasiakasohjelmalle. Insinöörityön tavoite täyttyi ja toimeksiantaja MHG Systems Oy pystyi laajentamaan palveluaan uudella asiakasohjelmalla. Parannettavaa jäi toki monellakin osa-alueella sovelluksessa, mutta ohjelmistojen kehityksellä on tapana jatkua vielä pitkään julkaisemisen jälkeenkin.

**LÄHTEET**

Cooper, Alan Reimann, Robert 2002. About Face 2.0 : The essentials of interaction design. Indianapolis, IN: Wiley Publishing.

Horstmann, Cay S. Cornell, Gary 2003. Inside Java 2. Käännös Veli-Pekka Ketola. Helsinki: IT Press.

Kalimo, Anna (toim.) 1996. Graafisen käyttöliittymän suunnittelu : opas ohjelmistojen käytettävyyteen. Helsinki: Tietotekniikan kehittämiskeskus TIEKE ry

O'Connor, John 2007. Using the Swing application framework (JSR 296). WWW-dokumentti . http://java.sun.com/developer/technicalArticles/javase/swingappfr/. Päivitetty 26.1.2009. Luettu 18.4.2009

Shneiderman, Ben. 1998. Designing the user interface : strategies for effective human-computer interaction. Reading: Addison-Wesley.

Travis, Greg 2003. Store objects using the Preferences API. WWW-dokumentti . http://www.ibm.com/developerworks/java/library/j-prefapi.html. Ei päivitystietoa. Luettu 12.4.2009

Zukowski, John 2002. Mastering Java 2, J2SE 1.4. San Francisco: Sybex