LAPPEENRANNAN TEKNILLINEN YLIOPISTO
TEKNISTALOUDELLINEN TIEDEKUNTA
TIETOTEKNIIKAN OSASTO

Mobiiliteknologia liikkuvan työntekijän apuna

Diplomityön aihe on hyväksytty 27.11.2007

Työn tarkastajina toimivat dosentti Jouni Ikonen ja professori Jari Porras sekä ohjaajana DI Tommi Kallonen

Lappeenrannassa 5.12.2007

Teemu Vilkko
Tervahaudankatu 1 B 7
53850 Lappeenranta
050 - 351 9776

TIIVISTELMÄ

Lappeenrannan Teknillinen Yliopisto
Teknistaloudellinen tiedekunta
Tietotekniikan osasto
Teemu Vilkko

Mobiiliteknologia liikkuvan työntekijän apuna

Diplomityö
2007
59 sivua, 16 kuvaa, 4 taulukkoa, 1 liite
Tarkastajat: Dosentti Jouni Ikonen
Professori Jari Porras
Ohjaaja: DI Tommi Kallonen
Hakusanat: liikkuva työntekijä, mobiiliteknologia, rakennusteollisuus, RFID
Keywords: nomadic worker, mobile technology, construction industry, RFID

Liikkuvat työntekijät työskentelevät pääasiassa muualla kuin vakituisessa työpisteessään ja siirtyvät usein työpisteestä toiseen. Myös kenttäolosuhteissa työntekijöiden on usein tarpeen päästä käsiksi tietoihin, jotka on tallennettu sähköisesti yrityksen tietojärjestelmään. Kentällä työskennellessä voidaan tietojärjestelmän kanssa kommunikointiin käyttää mobiiliteknologiaa, eli kannettavia päätelaitteita ja langattomia tiedonsiirtotekniikoita.

Tässä diplomityössä tutkitaan mobiiliteknologian soveltuvuutta liikkuvan työntekijän apuvälineeksi. Ensin vertaillaan erilaisia mobiiliteknologioita sekä selvitetään mobiiliympäristön ominaispiirteitä ja rajoitteita. Työn käytännön osuudessa toteutetaan mobiililaitteelle asiakasohjelma, jolla rakennusalan työntekijä voi tunnistaa betonisia rakennuselementtejä, hakea niihin liittyviä tietoja tietokantapalvelimelta sekä päivittää tietoja. Toteutuksessa otetaan huomioon mobiiliympäristön ominaispiirteiden vaikutus käytettävyyteen, ja sovelluksesta pyritään tekemään mahdollisimman helppokäyttöinen.

ABSTRACT

Lappeenranta University of Technology
Faculty of Technology Management
Department of Information Technology
Teemu Vilkko

Mobile technology as a tool for nomadic workers

Master's thesis
2007
59 pages, 16 figures, 4 tables, 1 appendix
Examiners: Adjunct Professor Jouni Ikonen
Professor Jari Porras
Instructor: M.Sc. (Tech.) Tommi Kallonen
Keywords: nomadic worker, mobile technology, construction industry, RFID

Nomadic workers travel often between different work sites and work mainly outside their regular work place, but often require access to information stored electronically in corporate information systems. While working in field conditions, communication with an information system can be achieved by using mobile technology, i.e. mobile devices and wireless communication.

This master's thesis researches the use of mobile technology to assist nomadic field workers in their tasks. First different mobile technologies are compared and constraints that characterize mobile computing are explained. In the practical part of the thesis client software is developed for a mobile device. The software allows a nomadic construction worker to identify concrete elements and to acquire and update information concerning them. The characteristics of mobile computing and their effect on usability are taken into account when implementing the client software and the software is designed to be as easy to use as possible.

SISÄLLYSLUETTELO

1 Johdanto ... 4
1.1 Liikkuvan työntekijän kommunikointi tietojärjestelmän kanssa ... 5
1.2 Aikaisempia ratkaisuja ... 6
1.3 Rakenne ... 7

2 Mobiiliympäristö ... 8
2.1 Mobiililaitteet ... 9
2.2 Käyttöjärjestelmät ... 11
2.3 Langaton tiedonsiirto ... 13
2.3.1 GPRS, EDGE ja 3G ... 13
2.3.2 WLAN ... 15
2.3.3 Bluetooth ... 16
2.4 Käyttöliittymäsuunnittelu ... 16

3 RFID-tunnistus ... 20
3.1 Tunnisteet ... 20
3.2 Taajuusalueet ... 22
3.3 Tiedon tallennus ... 23

4 Tietojärjestelmän liityntävaihtoehtoja ... 25
4.1 Web Service Description Language (WSDL) ... 26
4.2 SOAP ... 27
4.3 Universal Description, Discovery and Integration (UDDI) ... 28
4.4 XML-RPC ... 29

5 Mobiiliteknologia rakennusteollisuudessa ... 31
5.1 Tietojärjestelmä ... 33
5.2 Käyttötapaukset ... 34

6 Toteutus 38
6.1 Tietokantarajapinta 38
6.2 Päätelaitteen valinta 39
6.3 Asiakassovelluksen rakenne 40
6.4 Käyttöliittymä ja toiminta 42
6.5 Testaus ja analyysi 48
6.5.1 Havaitut ongelmat 50
6.5.2 Jatkokehitys 52

7 Johtopäätökset 54

Lähteet 56

Liitteet

LYHENNELUETTELO

3G	3rd Generation
AGPS	Assisted GPS
EDGE	Enhanced Data Rates for GSM Evolution
EDR	Enhanced Data Rate
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HF	High Frequency
HSDPA	High Speed Downlink Packet Access
HSPA	High Speed Packet Access
HSUPA	High Speed Uplink Packet Access
HTTP	Hypertext Transfer Protocol
ISM	Industrial, Scientific and Medical
LF	Low Frequency
MVC	Model-View-Controller
OS	Operating System
RFID	Radio Frequency Identification
RPC	Remote Procedure Call
TDMA	Time Division Multiple Access
UDDI	Universal Description, Discovery and Integration
UHF	Ultra High Frequency
UMTS	Universal Mobile Telecommunications System
W3C	World Wide Web Consortium
W-CDMA	Wideband Code Division Multiple Access
WLAN	Wireless Local Area Network
WSDL	Web Service Description Language
XML	Extensible Markup Language

1 JOHDANTO

Liikkuvan työn määrä lisääntyy, ja yhä useampi työntekijä tekee työtään vakituisen työpaikkansa tai kotinsa ulkopuolella. Paljon liikkuvia työntekijöitä ovat esimerkiksi myyjät, konsultit sekä huolto- ja kuljetushenkilöstö. Työkohteesta toiseen siirtyvät työntekijät voivat tarvita itse kohteeseen tai siellä suoritettavaan työtehtävään liittyvää tietoa, mutta tällainen tieto on yhä kasvavassa määrin tallennettuna yrityksen tietojärjestelmään, jolloin sen käsittely vaatii tietokoneen ja tietoliikenneyhteyden. Työntekijä voi käydä tehtävien välillä yrityksen toimistolla noutamassa aina seuraavan työtehtävänsä vaatimat tiedot paperilla, mutta ylimääräinen edestakainen matkustus paitsi hidastaa tehtävien suoritusta, myös aiheuttaa kuluja. Jos työntekijä pystyy itse noutamaan tarvitsemansa tiedot suoraan työkohteeseensa, voidaan tehostaa tehtävien suoritusta sekä vähentää ylimääräisiä kuluja. Kun työtehtäviin liittyvät tiedot noudetaan reaaliaikaisesti työtä tehdessä, voidaan varmistua siitä että käytettävissä on uusimmat mahdolliset tiedot ja lisäksi työtehtävien suorituksen reaaliaikainen raportointi voi tuoda yritykselle logistiikkaetuja. Apuna voidaan käyttää mobiiliteknologiaa, eli kannettavia päätelaitteita sekä langattomia tietoliikennetekniikoita, joita käyttäen työntekijä voi itsenäisesti ja paikasta riippumatta noutaa työtehtävissään tarvitsemansa tiedot yrityksen tietojärjestelmästä tai kuitata työn suoritetuksi ja hakea uusia työtehtäviä.

Tämä diplomityö on osa Mobilding-hanketta, jossa tutkitaan mobiiliteknologian hyödyntämistä rakennusteollisuuden elinkaaritiedon hallinnassa läpi rakennuksen elinkaaren, alkaen sen suunnittelusta ja jatkuen koko käyttöiän. Rakennusprojekteihin osallistuu monia eri osapuolia rakennuttajista suunnittelijoihin ja elementtivalmistajiin, joiden välinen kommunikaatio on tärkeä osa onnistunutta projektia. Jokainen osapuoli tuottaa projektin aikana suuren määrän siihen liittyvää tietoa. Suuri osa rakennusprojekteihin liittyvästä tiedosta on perinteisesti tallennettu eri osapuolten omiin järjestelmiin tai paperimuotoon, ja tällöin tieto on hankalasti saatavissa ja jaettavissa osapuolten kesken. Jos tiedot rakennuksen koko elinkaaren ajalta tallennetaan keskitetysti osapuolten yhteiseen tietojärjestelmään, josta niihin pääsevät käsiksi kaikki projektiin osallistuvat osapuolet, voidaan parantaa tiedon saatavuutta sekä kommunikaatiota. Parantunut kommunikaatio tehostaa logistiikkaa ja voi myös parantaa

laatua kun tieto muutoksista tai ongelmista on reaaliajassa kaikkien osapuolten nähtävissä. Langattomalla tietoliikenneyhteydellä varustetulla mobiililaitteella rakennusprojektiin osallistuvat liikkuvat työntekijät, kuten elementtien asentajat, pääsevät käsiksi tarvitsemaansa tietoon sujuvasti ja voivat ilmoittaa työtehtävien suorittamisesta tai ongelmista paikasta riippumatta.

1.1 Liikkuvan työntekijän kommunikointi tietojärjestelmän kanssa

Kannettavat päätelaitteet ja langattomat tietoliikennetekniikat tarjoavat uusia mahdollisuuksia liikkuville kenttätyöntekijöille. Mobiiliteknologian avulla työntekijät voivat työkohteessa työskennellessään muodostaa yhteyden yrityksensä tietojärjestelmään ja noutaa tarvitsemansa tiedot suoraan omaan päätelaitteeseensa. Työntekijän tehtäviin sopivimman mobiililaitteen valintaa voi rajoittaa työtehtävien luonne. Mitä enemmän työntekijä liikkuu työpisteestä toiseen, sitä pienempi ja kevyempi laitteen kannattaa olla. Jotkin työympäristöt voivat myös asettaa vaatimuksia laitteen kestävyydelle, kuten erityisen pölyiset, kosteat tai kylmät tilat.

Tässä diplomityössä kehitetään rakennusteollisuuden liikkuvan työntekijän tarpeisiin järjestelmä, jolla voidaan kommunikoida yrityksen tietojärjestelmän kanssa. Työssä tutkitaan mobiiliympäristön erikoispiirteitä sekä niiden vaikutusta sovelluksen käytettävyyteen ja käyttöliittymäsuunnitteluun. Lisäksi selvitetään avoimia kysymyksiä mobiililaitteen ja tietojärjestelmän välisestä kommunikoinnista; mitä tietoa siirretään, miten tieto siirretään, milloin tieto siirretään sekä missä muodossa tieto siirretään. Tietokantapalvelimelle suunnitellaan rajapinta, joka mahdollistaa työtehtäviin liittyvien tietojen haun tietokannasta sekä niiden lisäyksen ja muokkauksen. Lisäksi valitaan käyttökohteeseen sopivin mobiililaite, jolle toteutetaan rajapintaa hyödyntävä asiakassovellus tietojen siirtoon ja esitykseen.

Toteutettava mobiilisovellus avustaa rakennusteollisuuden liikkuvia työntekijöitä useissa erilaisissa betonielementteihin liittyvissä työtehtävissä betonitehtaalta rakennuspaikalle. Eri työtehtävissä sovelluksella siirretään mobiililaitteen ja

tietojärjestelmän välillä erilaista elementteihin liittyvää tietoa, kuten elementin mittatietoja, vikaraportteja ja kuittauksia. Sovelluksella voidaan tunnistaa betonisia rakennuselementtejä niihin valuvaiheessa upotettujen RFID (Radio Frequency Identification) -tunnisteiden avulla, ja niiden paikantamisessa voidaan hyödyntää GPS (Global Positioning System) -satelliittipaikannusta.

Tavoitteena on toteuttaa helppokäyttöinen ja monipuolinen sovellus rakennusteollisuuden liikkuvalle työntekijälle. Toteutuksen onnistumisessa on tärkeää hyvä käytettävyys vaihtelevissa työolosuhteissa betonitehtaalla, varastoissa ja rakennuspaikalla. Valittavan laitteen ja tietoliikenneyhteyden täytyy soveltua monenlaisiin tilanteisiin ja olosuhteisiin. Olosuhteet voivat vaihdella sisätiloista ulkotiloihin, ja ympäristö voi olla pölyinen tai likainen. Lämpötila voi olla alhainen, jolloin esimerkiksi mobiililaitteiden pienten näppäimien painelu on vaikeaa hansikkaat kädessä. Tietoliikenneyhteyden valintaa määrittelevät toisaalta käyttökohteen sijainti ja toisaalta yhteyden nopeuden riittävyys. Betonitehtaalla tai varastorakennuksessa voi olla olemassa kiinteä langaton verkko, mutta valmistumassa olevalle rakennukselle sellainen on pystytettävä erityisesti tätä tarkoitusta varten tai käytettävä jotain toisenlaista verkkoratkaisua.

1.2 Aikaisempia ratkaisuja

Mobiiliteknologian käyttöä työntekijöiden apuvälineenä on tutkittu ennenkin, mutta harvemmin rakennusteollisuuden näkökulmasta. Eräs kehitetty järjestelmä mahdollistaa yksinkertaistettujen rakennuspiirustusten siirtämisen työmaalle kämmentietokoneella (Fogli et al. 2005). Työmaalla tehdään joskus muutoksia piirustuksiin tai lisätään niihin kommentteja, mutta nämä muutokset eivät aina kantaudu takaisin alkuperäisiin piirustuksiin. Kehitetyllä järjestelmällä työmaan esimies voi tehdä kämmentietokoneellaan piirustuksiin muutoksia ja lähettää päivitetyt piirustukset tietokantaan, josta muutokset voidaan siirtää alkuperäisiin piirustuksiin.

Meluisaan ja turvallisuuskriittiseen voimalaitosympäristöön on suunniteltu ja kehitetty mobiilisovellus, jolla voidaan koordinoida hajautetusti työskenteleviä työntekijöitä

(Kjedskov et al. 2006). Järjestelmän suunnitteluprosessi koostui kahdesta päätekijästä, etnografisesta kenttätutkimuksesta sekä oliopohjaisesta suunnittelusta. Etnografisella tutkimuksella kerättiin runsaasti tietoa voimalaitostyöntekijöistä sekä heidän työtehtävistään haastattelemalla ja tarkkailemalla. Koska meluisassa ympäristössä on vaikea puhua puhelimella, sovellusta testanneet työntekijät pitivät sen tarjoamasta mahdollisuudesta kommunikoida tekstimuodossa. Tekstiä ei tarvitse kirjoittaa itse, vaan käyttäjä valitse listoista sopivat lauseenjäsenet, joista viesti muodostuu.

Nyt kehitettävän mobiililaitteisiin perustuvan järjestelmän kaltaisia on muutamia jo käytössä. Finnair-konserniin kuuluvan Northport Oy:n työntekijät käyttävät työaikojen kirjaamiseen sekä työtehtävien jakoon ja kuittaukseen matkapuhelimia ja RFID-tunnisteita (IBM 2006). Järjestelmän on kehittänyt IBM ja teknologian toimittaa Nokia, jonka Field Force Solution -ratkaisuun se perustuu. Field Force Solution perustuu Nokian RFID-ominaisuudella varustettuihin puhelimiin, jotka kommunikoivat keskuspalvelimen kanssa tekstiviesteillä tai salatun HTTP-yhteyden yli GPRS:llä (Vuorinen 2005).

1.3 Rakenne

Luvussa kaksi esitellään mobiiliympäristön erityispiirteitä sekä mobiililaitteita ja langattoman tiedonsiirron tekniikoita. Lisäksi käsitellään käyttöliittymäsuunnittelua sekä yleisesti että erityisesti mobiililaitteita ajatellen. Luvussa kolme tutustutaan RFID-tunnistustekniikkaan ja luvussa neljä käsitellään erilaisia tietojärjestelmän liityntävaihtoehtoja. Luvussa viisi tutustutaan rakennusteollisuuteen sekä esitellään sovelluksen käyttötapaukset. Luku kuusi käsittelee työn käytännön toteutusta. Siihen kuuluu tietojärjestelmän rajapinnan määrittely, sopivan päätelaitteen valinta sekä sovelluksen rakenne ja toiminta. Lisäksi analysoidaan sovelluksen toimivuutta. Lopuksi luvussa seitsemän tehdään yhteenveto työn tuloksista ja esitellään johtopäätöksiä.

2 MOBIILIYMPÄRISTÖ

Mobiililaitteiden liikuteltavuus tekee niiden käytöstä vapaampaa kuin perinteisten, kiinteästi asennettavien langallisten laitteiden, mutta toisaalta ne ovat myös monin tavoin rajoittuneempia. Mobiiliympäristön rajoitteet voidaan jaotella neljään eri tyyppiin (Satyanarayanan 1996):

- Laitteet ovat resursseiltaan rajoittuneita kiinteisiin verrattuna
- Liikkuvuus tuo mukanaan riskejä
- Tietoliikenneyhteydet ovat suorituskyvyltään ja luotettavuudeltaan vaihtelevia
- Laitteiden virtalähde on rajallinen

Mobiililaitteiden koko ja paino rajoittavat käytettävissä olevaa suoritintehoa, muistin määrää sekä tallennuskapasiteettia. Vaikka laitteet parantuvat resursseiltaan koko ajan, ne ovat aina vähätehoisia kiinteisiin laitteisiin verrattuna. Nykyisin laitteiden prosessointiteho riittää kuitenkin esimerkiksi hyvälaatuisen videokuvan esittämiseen ja myös tallennuskapasiteettia voidaan laajentaa muistikorteilla useisiin gigatavuihin.

Liikuteltavat laitteet ovat alttiimpia vahingoittumiselle ja katoamiselle kuin kiinteästi sijoitetut laitteet, ja niiden varastaminen on huomattavasti helpompaa. Tähän liittyy myös tietoturvariski, jos luottamuksellista tietoa sisältävä laite päätyy vääriin käsiin.

Langaton tietoliikenneyhteys voi katkeilla useasti ja odottamattomasti, joten ei voida olettaa että tiedonsiirto toimii aina. Toisaalta verkkoyhteyden hetkellinen katkeaminen ei saa vaikeuttaa tai estää työtehtävien suoritusta. Langattomien verkkojen tiedonsiirtonopeus on useimmiten melko vaatimaton langallisiin verrattuna. Lisäksi nopeus voi vaihdella, koska kaista on jaettu muiden tukiaseman käyttäjien kanssa, tai käyttäjän liikkuessa heikomman signaalin alueelle. Myös tiedonsiirron viiveet ovat suuria. Nopeus on kuitenkin riittävä yksinkertaisiin tekstipohjaisiin viesteihin. Viestien pakkaamisella voidaan niiden kokoa pienentää, jolloin niiden lähetys ja vastaanotto on nopeampaa (ja halvempaa). Tällä on myös haittansa, sillä viestin pakkaaminen ja vastaavasti purkaminen vaativat laskentatehoa (ja siten lisäävät virrankulutusta)

laitteelta ja siihen kuluu jonkin verran aikaa. Hitaalla yhteydellä ja vähän kuormitetulla palvelimella suurikokoisten viestien pakkauksen on kuitenkin havaittu nopeuttavan toimintaa (Tian et al. 2004). Epäluotettavan tietoliikenneyhteyden haittoja voidaan vähentää tallentamalla kriittisimmät tiedot mobiililaitteeseen joko etukäteen tai välimuistiin aina kun uutta tietoa on haettu verkosta. Tällöin yhteyden katkeaminen ei vaikeuta työntekoa.

Vaikka akkuteknologia kehittyykin, ovat mobiililaitteiden virtalähteet aina rajallisia, ja säästeliäs virrankulutus on huomioitava sekä laitteisto- että ohjelmistosuunnittelussa. Suurin osa mobiililaitteiden virrankulutuksesta aiheutuu prosessorin laskentatoimista sekä tiedonsiirrosta. Mobiililaitteiden suoritusteho kasvaa koko ajan ja niillä käytetään entistä raskaampia sovelluksia, joten myös niiden virrankulutus kasvaa. Jos laitteen valmiusaika halutaan pitää samana, täytyy akun olla suurempi, mutta laitevalmistajat eivät halua kasvattaa akkujen kokoa, koska kuluttajat vieroksuvat suuria ja painavia laitteita. Prosessorin aiheuttamaa virrankulutusta voidaan useasti vähentää siirtämällä raskaiden laskentatoimien suoritus mobiililaitteelta palvelimelle (Rudenko et al. 1998). Laskentatoimissa säästetty virrankulutus ei välttämättä riitä kompensoimaan tiedonsiirrossa kulutettua virtaa, etenkään sellaisessa ympäristössä, jossa on paljon häiriöitä. Tiedonsiirron virheiden takia tietoa joudutaan siirtämään uudelleen ja siten virrankulutus nousee.

2.1 Mobiililaitteet

Mobiililaitteiksi kutsutaan kannettavia laitteita, jotka soveltuvat tiedon käsittelyyn ja esittämiseen sekä langattomaan tiedonsiirtoon. Tällaisia laitteita ovat esimerkiksi matkapuhelimet, kämmentietokoneet ja kannettavat tietokoneet. Kuvassa 1 on erilaisia mobiililaitteita, ylimpänä Nokia 770 Internet Tablet ja alarivissä vasemmalta lukien Nokia E60 ja E61 -puhelimet sekä Toshiba Pocket PC e740 ja HP iPAQ Pocket PC h5450 -kämmentietokoneet.

Kuva 1. Erilaisia mobiililaitteita

Mobiililaitteiden kirjo on suuri. Laitteen ominaisuuksien määrän kasvaessa myös sen koko kasvaa. Asteikon toisessa päässä ovat kannettavat tietokoneet, joissa on suuret ja tarkat näytöt, täysi näppäimistö ja hyvät ohjausvälineet (kosketuslevy sekä mahdollisuus liittää hiiri). Toisessa päässä asteikkoa ovat matkapuhelimet. Niiden näytöt ovat pieniä, näppäimet ovat pieniä ja niitä on vähän. Puhelimien ohjaus toimii useimmiten näppäimillä, mutta joissain malleissa voi olla ohjauskynä tai kosketusnäyttö. Näiden ääripäiden väliin sijoittuvat erilaiset kämmentietokoneet.

Koska mobiililaitteiden näytöt ja tekstinsyöttökeinot ovat tavallisiin pöytätietokoneisiin verrattuna rajoittuneita, on sovellusten tiedonesitystapaan ja käyttäjältä vaadittavien toimenpiteiden suunnitteluun kiinnitettävä huomiota. Pienelle ruudulle ei mahdu kerralla paljon tietoa, joten kaikki tarpeeton tulee karsia pois. Koska monissa laitteissa ei ole kunnollista näppäimistöä ja napit ovat pieniä, tekstin syöttö on vaikeaa ja hidasta, ja käyttäjältä vaadittavien toimenpiteiden tulee olla suoritettavissa mahdollisimman vähillä näppäimenpainalluksilla. Hiiren kaltaisen osoitinlaitteen puute hidastaa useilla laitteilla niiden käyttöä, mutta toisaalta kosketusnäyttö tai osoitinkynä voi olla hiirtä nopeampi ja helpompi käyttää.

Päätelaitteet voidaan jakaa kahteen luokkaan perustuen tehtävien suorituksen jakoon palvelimen ja päätelaitteen välillä. Niin sanottu ”thin client” on pitkälti riippuvainen keskuspalvelimesta, kun taas ”thick client” suorittaa suuren osan prosessoinnista ja tiedon varastoinnista itse. Koska thin client ei tallenna mitään tietoa itse, kaikki tarvittava tieto on noudettava palvelimelta. Mobiiliympäristössä laitteiden resurssien vähäisyys puoltaa thin client -ajattelutapaa, kun taas usein epäluotettava verkkoyhteys puoltaa thick client -mallia. Näitä malleja voidaan myös yhdistellä ja hämärtää niiden välistä rajaa, esimerkiksi varastoimalla tieto keskuspalvelimella, mutta suorittamalla prosessointi asiakaslaitteessa. Smart client model -mallissa (Chu et al. 2004) on yhdistämällä thin- ja thick clientien ominaisuuksia minimoitu toisaalta asiakassovelluksen virrankulutus (sekä latausaika) ja toisaalta käyttäjän havainnoima vasteaika.

Mobiililaitteista erityisesti matkapuhelimien toiminnan odotetaan olevan huomattavasti luotettavampaa kuin esimerkiksi kotitietokoneiden, vaikka niillä voidaan käyttää monia hyvinkin monimutkaisia sovelluksia. Useimmiten matkapuhelin on päällä kuukausia yhteen menoon, eikä jumiutumisia ja uudelleenkäynnistyksiä pidetä normaalina käyttäytymisenä. Puhelimen tulee toipua virhetilanteista eikä se saa hukata tallennettuja tietoja. Käyttäjät odottavat myös puhelimella käytettävien ohjelmien käynnistyvän nopeasti. Työpöytäkoneissa raskaiden sovellusten käynnistäminen voi kestää useita minuutteja, mutta matkapuhelimella kymmenen sekuntia on käyttäjän mielestä jo pitkä aika.

2.2 Käyttöjärjestelmät

Matkapuhelimille ja kämmentietokoneille on olemassa useita erilaisia käyttöjärjestelmiä. Osa on laitevalmistajien omaan käyttöönsä kehittämiä suljettuja järjestelmiä, osa taas avoimia.

Symbian OS on Symbian Ltd:n kehittämä kohtuullisen pienitehoisille ja vähämuistisille kannettaville laitteille, kuten älypuhelimille, tarkoitettu avoin käyttöjärjestelmä (Digia Inc. 2003). Symbian OS on tämän hetken käytetyin mobiilikäyttöjärjestelmä 67 %

markkinaosuudellaan (Canalys 2007). Älypuhelimien näyttöjen muoto ja ohjaustavat poikkeavat toisistaan paljon, joten Symbian-puhelimissa käyttöjärjestelmä ja käyttöliittymä on erotettu toisistaan. Symbian OS tarjoaa puhelimen ydintoiminnot, ja matkapuhelinvalmistaja voi kehittää itse tai lisensoida ja muokata tuotteeseensa sopivan käyttöliittymän. Tärkeimmät käyttöliittymät ovat Nokian kehittämät S60 (aiemmalta nimeltään Series 60) ja Series 80 sekä UIQ Technologiesin kehittämä UIQ. S60-käyttöliittymä on suunniteltu yhdellä kädellä käytettäviin älypuhelimiin, Series 80 kommunikaattoreihin ja UIQ kosketusnäytöllisiin puhelimiin. Sovelluskehitys Symbian OS -käyttöjärjestelmälle on mahdollista Java- tai C++-kielellä sekä S60-alustalle myös Python-kielellä. S60 3rd Edition -laitteille on saatavilla myös .NET Compact Framework -toteutus, jolla voidaan ajaa .NET-sovelluksia (Red Five Labs 2007).

Windows Mobile on Microsoftin kehittämä käyttöjärjestelmä kämmentietokoneille ja älypuhelimille. Se on suunniteltu muistuttamaan ulkonäöltään ja ominaisuuksiltaan työpöytä-Windowsia. Microsoftin käyttöjärjestelmät ovat 14 % osuudellaan mobiililaitemarkkinoiden toisella sijalla (Canalys 2007). Windows Mobile -kämmentietokoneista käytetään markkinanimeä PocketPC. Uusin versio, Windows Mobile 6, perustuu Windows CE 5.0 -ytimelle ja jakautuu kolmeen eri versioon: Classic, Professional ja Standard. Classic on tarkoitettu perinteisille kämmentietokoneille joissa ei ole matkapuhelinominaisuuksia ja Professional taas sellaisille kämmentietokoneille joita voidaan käyttää myös puhelimena. Standard on tarkoitettu älypuhelimille. Windows Mobile -alustalle voidaan kehittää natiivisovelluksia C++-kielellä tai .NET Compact Framework -sovelluksia C#- tai VB.NET-kielellä.

Tavallisesti työasemissa tai palvelimissa käytettävästä Linuxista on olemassa mobiililaitteille optimoituja jakelupaketteja, joissa on pienennetty muistintarvetta sekä lisätty tuki uusille prosessoreille. Mobiili-Linuxin etuihin kuuluu mm. perustuminen avoimeen lähdekoodiin sekä lisenssimaksuttomuus. Avoin lähdekoodi avaa sovelluskehittäjille uusia mahdollisuuksia, ja monet työpöytä-Linuxille kehitetyt komponentit voidaan helposti siirtää mobiiliversioon. Linuxin osuus vuonna 2006 myytyjen mobiililaitteiden käyttöjärjestelmistä oli 6 % (Canalys 2007). Linux-

käyttöjärjestelmää käyttävät monet Motorolan älypuhelimet sekä Nokian Internet Tablet -laitteet.

Palm OS on Palm Inc:n alun perin kosketusnäytöllisille kämmentietokoneille kehittämä käyttöjärjestelmä, mutta sitä on käytetty myös mm. matkapuhelimissa, pelikonsoleissa ja viivakoodilukijoissa. Palm valmistaa käyttöjärjestelmän lisäksi myös laitteita. Palm OS -sovelluskehityksessä käytetään pääosin C/C++-kieliä, mutta se on mahdollista myös Javalla, Pythonilla ja monilla muilla kielillä.

Nokia OS on Nokian kehittämä matkapuhelinkäyttöjärjestelmä. Vain Nokian omissa puhelinmalleissa käytettävät Series 30 ja Series 40 -käyttöliittymät perustuvat Nokia OS -ytimelle. Series 40 muistuttaa ulkoisilta ominaisuuksiltaan paljon Nokian Symbian-käyttöliittymää S60:tä. Nokia OS -puhelimille voidaan kehittää sovelluksia vain Javalla.

2.3 Langaton tiedonsiirto

Kenttäolosuhteissa liikkuva työntekijä tarvitsee yrityksen tietojärjestelmän kanssa kommunikointiin jonkin langattoman tiedonsiirtomenetelmän. Eri teknologioilla on omat etunsa ja haittansa. Eroavaisuuksia on mm. tiedonsiirron nopeudessa ja kuuluvuusalueessa, joten käytettävä verkkotekniikka täytyy valita käyttökohteen vaatimusten mukaisesti. Jos työntekijä liikkuu laajalla alueella, ei tehtävään sovellu mikään pienen kuuluvuusalueen tekniikka. Tarvittavan tiedonsiirtonopeuden suuruus riippuu siirrettävän tiedon määrästä.

2.3.1 GPRS, EDGE ja 3G

Matkapuhelinverkon tiedonsiirtotekniikoiden etuna on kuuluvuusalue. Tällä hetkellä Suomessa GPRS on saatavilla kaikkialla GSM-verkon alueella eli lähes koko maassa pieniä katvealueita lukuun ottamatta. Uudempien, nopeampien tiedonsiirtotekniikoiden kuuluvuus on vielä melko puutteellinen. Myös datayhteyksien hinnoittelu on nykyisin melko kuluttajaystävällistä. Hinnoittelun perusteena on siirretyn tiedon määrä tai kiinteä kuukausihinta eikä yhteyden kesto. Koska radiotien häiriöt kasvavat, kun siirrytään

kauemmas GSM-tukiasemasta, voidaan tietoa joutua lähettämään uudelleen. Uudelleenlähetyksistä johtuen tiedonsiirrossa saavutettu nopeus pienenee.

GPRS on pakettipohjainen tiedonsiirtotekniikka, joka toimii GSM-verkossa. Toisin kuin vanhempi piirikytketty GSM-datayhteys, GPRS-yhteys käyttää verkon resursseja vain kun tietoa siirretään, joten sitä voidaan pitää auki koko ajan resursseja tuhlaamatta. Tämä on erityisen hyödyllinen ominaisuus www-sivujen selailun kaltaisessa käytössä, jossa suurimman osan aikaa yhteys on käyttämättä, kun käyttäjä lukee hakemaansa sivua. GSM-verkossa radiotien jakoon useamman käyttäjän kesken käytetään aikajakokanavointia (Time Division Multiple Access eli TDMA). TDMA:ssa kaista jaetaan tietyn pituisiin aikaväleihin joissa tieto lähetetään. Käyttämällä yhtä aikaväliä GPRS:llä saavutetaan 14,4 kbit/s tiedonsiirtonopeus, joten maksimimäärällä eli kahdeksalla aikavälillä ylletään 115,2 kbit/s nopeuteen (Dornan 2001). Se, kuinka montaa aikaväliä voidaan käyttää, riippuu päätelaitteesta.

EDGE (Enhanced Data Rates for GSM Evolution) on laajennus GSM-verkkoon, joka vaatii päivityksen tukiasemaan. EDGE toimii Suomessa lähinnä kaupungeissa ja niiden lähiympäristöissä. Ainoa ero GPRS:ään verrattuna on erilainen signaalin modulointi, jolla saavutetaan moninkertainen tiedonsiirtonopeus aikaväliä kohden eli 59,2 kbit/s (Dornan 2001). Käyttämällä kahdeksaa aikaväliä voidaan siis saavuttaa 473,6 kbit/s nopeus.

3G (3rd Generation) on yhteisnimitys erilaisille kolmannen sukupolven matkapuhelintekniikoille. Suomessa ja muualla Euroopassa käytössä oleva 3G-verkko on nimeltään UMTS (Universal Mobile Telecommunications System). UMTS-verkossa käytettävä verkkotekniikka on nimeltään Wideband Code Division Multiple Access (W-CDMA). UMTS-verkossa saavutetaan teoriassa n. 2 Mbit/s nopeus (Kaaranen et al. 2001). UMTS-verkon operaattoreiden sitoutuvat mitoittamaan verkkonsa niin että eri alueilla ja olosuhteissa saavutetaan vähintään tietty tiedonsiirron nopeus (Kaaranen et al. 2001). Syrjäseuduilla ja nopeasti liikkuvissa ajoneuvoissa tämä nopeus on 144 kbit/s, kaupungeissa ja hitaasti liikkuvissa ajoneuvoissa 384 kbit/s sekä kaupunkikeskustoissa ja kävelynopeudella liikuttaessa 2 Mbit/s.

HSPA (High Speed Packet Access) on ryhmä protokollia, joilla voidaan parantaa UMTS-verkon tiedonsiirtonopeutta. HSPA-perheeseen kuuluu kaksi protokollaa, HSDPA (High Speed Downlink Packet Access) nopeuttaa verkosta päätelaitteelle päin suuntautuvaa tiedonsiirtoa ja HSUPA (High Speed Uplink Packet Access) päätelaitteelta verkkoon päin. Teoriassa saavutettavat tiedonsiirtonopeudet ovat HSDPA:lle 14,4 Mbit/s ja HSUPA:lle 5,76 Mbit/s (Mulvey 2007).

Suomalaisten matkapuhelinoperaattoreiden GSM- ja 3G-verkkojen kuuluvuusaluekartat löytyvät yritysten verkkosivuilta. Elisa Oyj:n 3G-verkko kattaa vuoden 2007 loppuun mennessä 120 paikkakuntaa ja 70 % väestöstä (Elisa 2007). Elisan 3G/HSDPA -verkossa voidaan saavuttaa 2 Mbit/s tiedonsiirtonopeus verkosta asiakkaalle päin. Vastaavasti EDGE-verkon alueelle luvataan 60-80 kbit/s ja GPRS-alueelle 30-40 kbit/s nopeus. Muut operaattorit ilmoittavat verkkosivuillaan vain tiedonsiirron teoreettiset maksiminopeudet.

2.3.2 WLAN

Langattomassa lähiverkossa (WLAN, Wireless Local Area Network) saavutetaan suuri tiedonsiirtonopeus, mutta yhdellä tukiasemalla saavutettava kuuluvuusalue on melko pieni. Tavallisella laitteistolla kantama on muutamista kymmenistä metreistä hiukan yli sataan metriin ulkotiloissa. Tiheällä tukiasemaverkostolla voidaan kuitenkin kattaa esimerkiksi suurikin toimistorakennus. IEEE 802.11g -standardin mukaisessa WLAN-verkossa saavutetaan teoriassa 54 Mbit/s nopeus (IEEE 2003), käytännössä noin puolet tästä. Useimmiten yritykset rakentavat itse WLAN-verkon omaan käyttöönsä, jolloin niiden käytöstä ei koidu yritykselle muita kuin verkon rakennus- ja ylläpitokustannukset. WLAN-verkon rakentamiseen ei vaadita lisenssiä, koska ne toimivat lisenssivapaalla radiotaajuusalueella. Aikaisemmin WLAN on ollut käytettävissä lähinnä kannettavilla tietokoneilla ja kämmentietokoneilla, mutta tekniikan kehityttyä myös monet uuden sukupolven matkapuhelimet sisältävät WLAN-yhteyden.

2.3.3 Bluetooth

Bluetooth on lyhyen kantaman tiedonsiirtotekniikka, ja tyypillisesti se kantaa noin 10 metrin etäisyydelle (Miller & Bisdikian 2001). Maksimilähetysteholla (100 mW) voidaan kuitenkin teoriassa yltää noin 100 metrin kantamaan. Käytännössä Bluetoothilla voidaan kattaa pienehkö parin huoneen toimistorakennus.

Lyhyellä kantamalla ei vaadita suurta lähetystehoa joten Bluetooth-radion virrankulutus on alhainen, samoin laitteiston valmistuskustannukset. Bluetoothia ei ole suoranaisesti suunniteltu lähiverkkotekniikaksi vaan oheislaitteiden välijohtojen korvaajaksi. Bluetoothia käytetäänkin useimmiten vain kahden laitteen toisiinsa yhdistämiseen eikä suurien verkkojen muodostamiseen. Verkot ovat useimmiten yksityiskäytössä ja niiden käytöstä ei verkon rakentamisen lisäksi koidu muita kuluja kuin sen ylläpito. Bluetooth 2.0 -version mukanaan tuoma EDR-ominaisuus (Enhanced Data Rate) mahdollistaa teoriassa noin 3 Mbit/s tiedonsiirtonopeuden (Bluetooth SIG 2004). Käytännön testeissä (Lim et al. 2001) on selvitetty Bluetooth-internetyhteyspisteen tiedonsiirtonopeuksia usealla yhtäaikaisella käyttäjällä. Mitatut nopeudet vaihtelivat kymmenistä satoihin kilobitteihin sekunnissa.

2.4 Käyttöliittymäsuunnittelu

Käytettävyys, tai käyttäjäystävällisyys, kuvaa sitä kuinka hyvin jokin tuote soveltuu sille tarkoitettuun tehtävään ja tietylle kohderyhmälle. Tietotekniikassa käytettävyyteen liittyy laitteiden muotoilun lisäksi käyttöliittymäsuunnittelu. Hyvin toteutettu käytettävyys- ja käyttöliittymäsuunnittelu vähentää tuotanto- ja tuotetukikustannuksia sekä parantaa asiakastyytyväisyyttä (Butow 2007).

Jakob Nielsen on määritellyt kymmenen perussääntöä (Nielsen 2005), joita voidaan pitää käyttöliittymäsuunnittelun ohjeina ja joita noudattamalla voidaan parantaa sovelluksen käytettävyyttä. Säännöt ovat:

1. **Järjestelmän tilan näkyvyys**

   Järjestelmän tulee ajoissa ilmoittaa käyttäjälle mitä on tapahtumassa.

2. **Vastaavuus reaalimaailman kanssa**

   Järjestelmän tulee käyttää reaalimaailman tapoja ja käytäntöjä sekä käyttäjälle tuttua kieltä.

3. **Käyttäjän kontrolli ja vapaus**

   Käyttäjän tulee voida palata edelliseen tilaan tehdessään virheen.

4. **Yhtenäisyys ja standardit**

   Seuraamalla yleisiä käytäntöjä ehkäistään sekaannuksia.

5. **Virheiden estäminen**

   Poista virhealttiit tilanteet tai pyydä käyttäjältä varmistus toiminnon suorittamiseksi.

6. **Tunnistaminen muistamisen sijaan**

   Vähennä asioiden muistamisen tarvetta näkyvillä toiminnoilla.

7. **Joustavuus ja helppokäyttöisyys**

   Tarjoamalla toiminnoille oikoteitä nopeutetaan käyttöä. Anna käyttäjän muokata järjestelmää mieleisekseen.

8. **Esteettisyys ja minimalistisuus**

   Kaikki epäolennainen tieto heikentää olennaisen tiedon näkyvyyttä.

9. **Virheiden tunnistaminen, diagnosointi ja korjaaminen**

   Virheviestien tulee olla selkeitä ja ehdottaa toimenpiteitä virheen korjaamiseksi.

10. **Ohjeet ja dokumentaatio**

    Ohjeiden tulee olla helposti löydettävissä, tehtäväkohtaisia, eivätkä ne saa olla liian laajoja.

Mobiililaitteita pidetään yleisesti vain pienempinä versioina työpöytätietokoneista. Niiden käyttökohteet ja -ympäristö kuitenkin eroavat toisistaan suuresti, joten mobiililaitteiden käytettävyydellä on aivan erilaiset lähtökohdat (Holmquist 1999). Koska mobiililaitteiden luonteeseen kuuluu kannettavuus, ovat laitteet pieniä. Pienen koon mukanaan tuomia ominaisuuksia ovat pieni näyttö ja hankala tekstinsyöttö (Ballard 2007).

Näkyvimpänä ongelmana mobiililaitteiden käyttöliittymäsuunnittelussa on pieni näyttö. Kaikki esitettävä tieto ei välttämättä mahdu kerralla ruudulle, joten ruutua joudutaan vierittämään enemmän. Se vaikeuttaa pitkien tekstien lukemista johtuen ihmisten tavasta hahmottaa tekstikokonaisuuksia. Pienellä ruudulla ei voida myöskään käyttää ikkunointia, vaan yleensä yksi ohjelma peittää koko näytön, joten vaikkapa kalenterin tarkistelu kesken tekstiviestin kirjoituksen on vaikeaa tai jopa mahdotonta. Myös perinteisistä tietokoneiden näytöistä poikkeava ruudun muoto voi aiheuttaa ongelmia. Esimerkiksi kommunikaattori-tyyppisten puhelimien ruutu on hyvin leveä ja matala, jolloin vaikkapa www-sivua mahtuu kerralla näkyviin ruudulle hyvin vähän. Jos tästä vielä lohkaistaan kallisarvoisia kuvapisteitä www-selaimen otsikkoriville ja painikeriville kuten perinteisissä selaimissa, ei itse sivulle jää tilaa juuri yhtään.

Mobiililaitteiden pieni koko rajoittaa siinä käytettävien näppäimien määrä sekä kokoa. Tekstinsyöttö on suhteellisen hankalaa jopa kommunikaattorien täydellä QWERTY-näppäimistöllä johtuen näppäimien pienestä koosta, eivätkä käyttäjät siksi mielellään kirjoita pitkiä tekstejä. Käyttöliittymässä tulee minimoida käyttäjältä vaadittava tekstin kirjoitus, ja käyttää esimerkiksi valintalistoja joita käyttäjä voi selata ja valita oikean vaihtoehdon (Ballard 2007). Sovellukset voivat myös vähentää kirjoittamisen tarvetta poimimalla tietoa automaattisesti muista lähteistä, kuten GPS-paikantimelta, kamerasta, käyttäjän puhekomennoista tai kalenterista.

Mobiiliympäristön riippuvuus rajallisesta virtalähteestä voi myös vaikuttaa käyttöliittymäsuunnitteluun. Monimutkaiset graafiset muunnokset tarvitsevat paljon prosessointitehoa ja sitä kautta kuluttavat laitteen akkua. Myös epäluotettava tietoliikenneyhteys voi vaikuttaa käyttöliittymäsuunnitteluun. Jos sovelluksen käyttö on

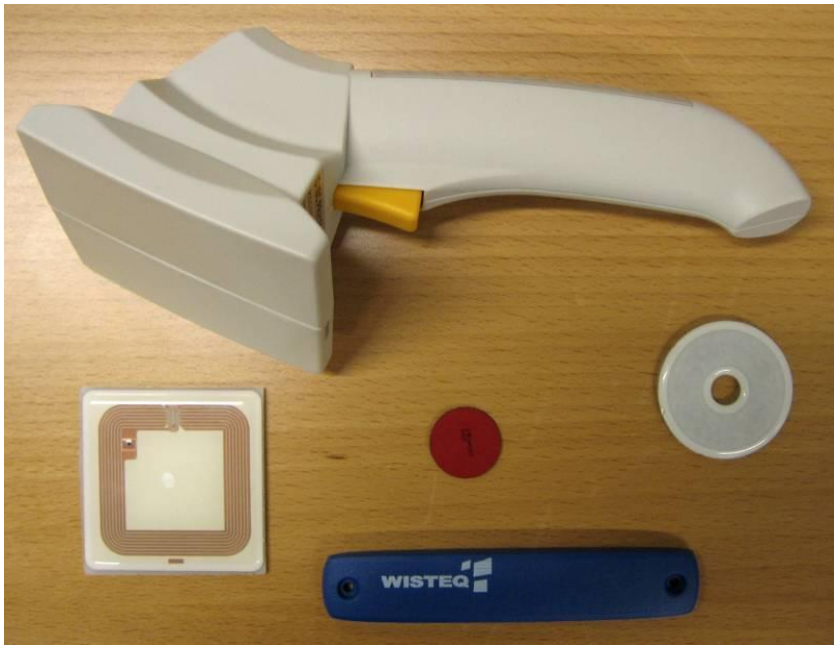
riippuvaista verkosta noudettavasta tiedosta, tulee mahdollistaa tietojen noutaminen ja laitteeseen tallentaminen etukäteen (Ballard 2007).

Itse laitteen käyttöjärjestelmä määrittelee sille tietynlaisen toimintalogiikan ja ulkonäön. On suositeltavaa että laitteelle kehitettävä sovellus seuraa tätä tyyliä, jolloin käyttäjä tuntee olonsa kotoisaksi sovellusta käyttäessään (Ballard 2007). Esimerkiksi OK- ja Peruuta-nappien paikkaa vaihtamalla saa helposti aikaan sekaannusta. Koska laitteet vaihtelevat toisistaan paljon, ei kaikille yleisesti päteviä suunnitteluohjeita ole, vaan kannattaa tutkia käyttöjärjestelmä- tai laitevalmistajan antamia ohjeita.

Eräs esimerkki mobiililaitteen käyttöliittymäsuunnittelusta on sovellus kenttätyöntekijöiden tiedonkeruun apuvälineeksi (Moe et al. 2004). Tutkimuksen pääpaino oli käytettävyysvaatimusten kartoittamisessa ja ymmärtämisessä sekä näiden vaatimusten täyttäminen käyttöliittymäsuunnittelussa. Sovelluksen käytettävyydelle asetettiin seuraavat vaatimukset: intuitiivisuus, tehokkuus, virheiden estäminen, vähäinen huomion tarve, tiedon keräys ja tarkistus pienellä vaivalla, selkeä tiedon esitys sekä joustavuus. Kämmentietokoneelle toteutettiin prototyyppi tiedonkeruusovelluksesta, jota käyttävät liikkuvat työntekijät maanalaisissa huoltotunneleissa työskennellessään. Jo lyhyen, muutaman tunnin koulutuksen jälkeen vain vähän tietokoneita käyttävät työntekijät oppivat keräämään tietoa uudella sovelluksella yhtä tehokkaasti kuin paljon kauemmin käyttämällään perinteisellä menetelmällä eli kynällä ja paperilla.

3 RFID-TUNNISTUS

RFID (Radio Frequency Identification) on radiotaajuuksia hyödyntävä etätunnistus-menetelmä. RFID-järjestelmän kaksi perusosaa ovat tunniste ja lukijalaite. RFID-lukija on laite, joka havaitsee lähietäisyydellä olevat tunnisteet ja pystyy lukemaan niiden sisältämät tiedot. Useimmiten lukijalaite ei tee lukemillaan tiedoilla itse mitään vaan se kommunikoi jonkin ulkoisen järjestelmän kanssa johon se ilmoittaa tunnisteista lukemansa tiedot (Finkenzeller 2003). Kuvassa 2 on esitelty muutamia ulkomuodoltaan erilaisia RFID-tunnisteita, jotka käyttävät eri taajuusalueita kommunikointiin. Luvussa 3.2 kerrotaan lisää RFID-järjestelmien käyttämistä taajuusalueista. Kuvassa 2 on myös käsikäyttöinen Feig Electronic ID ISC.PRH101-B -mallinen RFID-lukijalaite, joka on varustettu langattomalla Bluetooth-yhteydellä.



Kuva 2. Erilaisia RFID-tunnisteita sekä lukijalaite

3.1 Tunnisteet

RFID-tunnisteen tarkoitus on auttaa tunnistamaan tuote tai esine johon se on kiinnitetty. Tunniste on pieni laite, esimerkiksi nappi tai tarra, joka voidaan sisällyttää tuotteeseen valmistusvaiheessa tai kiinnittää jälkikäteen. Tunnisteet sisältävät mikropiirin sekä

antennin, jolla ne voivat vastaanottaa kyselyitä RFID-lukulaitteelta ja lähettää takaisin tietoa, yleensä sarjanumeronsa. RFID-tunnisteita voidaan käyttää viivakoodin korvaajina, ja niissä on useita etuja viivakoodeihin verrattuna. Niiden lukeminen ei vaadi näköyhteyttä, joten tunniste voidaan tuotteesta riippuen sijoittaa sen sisälle näkymättömiin, jossa se ei ole visuaalinen haitta eikä se irtoa vahingossa. RFID-tunnisteet voidaan lukea myös kauempaa kuin viivakoodit, ja niihin voidaan tallentaa enemmän tietoa. Tunnisteita käytetään lukuisissa erilaisissa käyttökohteissa. Niillä voidaan jäljittää ja tunnistaa karjaa sekä lemmikkieläimiä, rahtilavoja tai yksittäisiä tuotteita. Niitä käytetään myös kulunvalvontaan ja julkisen liikenteen maksujärjestelmissä sekä kaikissa Suomessa nykyisin myönnettävissä passeissa, eli ns. ”biopasseissa”.

RFID-tunnisteet voidaan luokitella monen eri perusteen mukaan. Yhteistä kaikille tunnisteille on kuitenkin tiedon säilytys sekä antenni jolla voidaan kommunikoida lukijalaitteen kanssa. Koska RFID-tunnisteita käytetään hyvin erilaisiin tarkoituksiin, niiden ulkomuoto ja materiaali vaihtelevat suuresti. Tunniste voi olla muodoltaan pyöreä tai neliskulmainen nappi, luottokortti, avaimenperä, kapseli, tai se voidaan jopa sijoittaa paperin kerrosten väliin.

Yleisin lajitteluperuste RFID-tunnisteille on niiden toimintaan tarvittavan sähkövirran lähde. Virtalähteen mukaan tunnisteet voidaan jakaa kahteen päätyyppiin, passiivisiin ja aktiivisiin (Bhatt & Glover 2006). Omaksi tyypikseen voidaan myös erottaa puoli-passiiviset tunnisteet.

Passiiviset tunnisteet eivät sisällä omaa virtalähdettä, vaan kommunikointiin ja prosessointiin tarvittava sähkövirta saadaan induktiolla lukijalaitteen muodostamasta sähkömagneettisesta kentästä. Passiiviset tunnisteet ovat pienikokoisia ja halpoja valmistaa, mutta niiden lukuetäisyys on pieni, parista sentistä muutamaan metriin.

Aktiiviset tunnisteet sisältävät oman virtalähteen joten lähetysteho voi olla suurempi. Tämän ansiosta signaalin läpäisykyky (esimerkiksi metallin tai veden) on suurempi ja niiden lukeminen onnistuu kauempaa, jopa sadan metrin päästä.

Puoli-passiiviset tunnisteet käyttävät omaa virtalähdettään prosessointiin, mutta saavat kommunikointiin tarvittavan virran lukijalaitteelta. Oman virtalähteen ansiosta puoli-passiiviset tunnisteet voivat esimerkiksi tarkkailla ympäristön lämpötilaa ja niiden lukuetäisyys voi olla hieman suurempi kuin täysin passiivisilla tunnisteilla.

3.2 Taajuusalueet

Koska RFID-järjestelmät kommunikoivat radioaalloilla, ne luokitellaan radiolaitteiksi, eikä niiden käyttö saa häiritä muiden radiolaitteiden toimintaa. RFID-tunnisteet käyttävät neljää eri radiotaajuusaluetta, LF (Low Frequency), HF (High Frequency), UHF (Ultra-High Frequency) ja mikroaaltoalue. Näillä taajuusalueilla on useampia ISM (Industrial, Scientific and Medical) -taajuuskaistoja, joita RFID-laitteet voivat käyttää häiritsemättä radio-, televisio-, matkapuhelin- tai viranomaissignaaleja. ISM-taajuuskaistat ovat teollisuuden, tieteen ja lääketieteen tarpeisiin varattuja radiotaajuuksia, joita voidaan käyttää ilman lisenssiä. Eri puolilla maailmaa ISM-taajuuskaistat poikkeavat hieman toisistaan, esimerkiksi joissakin maissa kaikki taajuusalueet eivät ole käytettävissä tai suurin sallittu lähetysteho on rajoitettu. Taulukossa 1 on esitetty tärkeimmät RFID-järjestelmien käyttämät taajuusalueet (Bhatt & Glover 2006).

Taulukko 1. RFID-tunnisteiden käyttämät radiotaajuudet

Taajuusalue	Taajuuskaistat
LF	alle 135 kHz
HF	6,78 MHz, 13,56 MHz, 27,125 MHz, 40,680 MHz
UHF	433,920 MHz, 869 MHz, 915 MHz
Mikroaalto	2,45 GHz, 5,8 GHz, 24,125 GHz

Alle 135 kHz:n taajuudet eivät kuulu ISM-taajuuksiin, mutta niillä operoivat RFID-laitteet toimivat lyhyillä etäisyyksillä joten häiriö ei ole suuri ongelma. Myöskään 869 MHz:n kaista ei ole ISM-kaista, vaan pelkästään Euroopassa käytettävä taajuusalue

lyhyen kantaman laitteille. 915 MHz:n ISM-kaista on käytettävissä ainoastaan Pohjois-Amerikassa.

Eri taajuusalueilla on erilaiset fysiikan laeista johtuvat ominaisuudet. Matalataajuiset signaalit läpäisevät paremmin vettä, mutta korkeataajuiset signaalit voivat sisältää enemmän informaatiota. Taajuusalue tulee siis valita tunnisteen käyttökohteen mukaan. Taulukossa 2 on esitetty tyypillisiä lukuetäisyyksiä eri taajuusalueiden tunnisteille (Lahiri 2006). Nämä lukuetäisyydet ovat teoreettisia maksimietäisyyksiä, ja niihin vaikuttavat monet tekijät, mm. antennin koko, lähetysteho ja ympäristön aiheuttamat häiriöt.

Taulukko 2. RFID-tunnisteiden tyypillisiä maksimilukuetäisyyksiä

Taajuusalue	Tyypillisiä lukuetäisyyksiä
LF	5-10 cm (passiivinen)
HF	1 metri (passiivinen)
UHF	9 metriä (passiivinen), 90 metriä (aktiivinen)
Mikroaalto	30 metriä (aktiivinen)

LF- ja HF-taajuusalueiden RFID-tunnisteet käyttävät muuttuvan magneettikentän indusoimaa jännitettä energian siirtoon lukijasta tunnisteeseen (Finkenzeller 2003). Tällöin lukijan ja tunnisteen sisältämät käämit toimivat samaan tapaan kuin muuntajassa. UHF- ja mikroaaltoalueen tunnisteet taas toimivat lukijan lähettämän sähkömagneettisen säteilyn avulla, joka mahdollistaa pidemmän lukuetäisyyden. Aiemmin kuvassa 2 esitetyistä tunnisteista alin, pitkulaisen muotoinen, on UHF-alueen tunniste, muut HF-tunnisteita.

3.3 Tiedon tallennus

RFID-tunnisteiden tiedontallennuskapasiteetti vaihtelee, suurimmillaan se voi olla muutamia kilotavuja. Yksinkertaisimmat, ns. yhden bitin tunnisteet eivät voi sisältää

tietoa, vaan ainoastaan niiden olemassaolo voidaan havaita lukijalaitteella. Pelkkä olemassaolon havaitseminen kuitenkin riittää joihinkin käyttökohteisiin, kuten kauppojen varkaudenestojärjestelmiin. Tietoa sisältävistä tunnisteista yksinkertaisimpiin on tallennettuna vain tunnisteen yksilöllinen sarjanumero. Sarjanumero on annettu tehtaalla eikä sitä voi muuttaa, joten näihin tunnisteisiin ei voida kirjoittaa tietoa. Suuremman kapasiteetin tunnisteisiin voidaan myös kirjoittaa. Suuret tallennuskapasiteetit vaativat enemmän virtaa tiedon prosessointiin, joten tällaisen tunnisteen täytyy mahdollisesti olla aktiivinen eli sisältää oma virtalähde.

4 TIETOJÄRJESTELMÄN LIITYNTÄVAIHTOEHTOJA

Liittymä tietojärjestelmään, eli sen rajapinta, on mahdollista toteuttaa usealla eri tavalla, mutta mobiililaitteiden suuresti vaihtelevat ominaisuudet asettavat rajoitteita ja rajapinnan tulee olla käytettävissä erilaisilla laitteilla. Sopivinta ratkaisua valitessa täytyy ottaa huomioon yhteensopivuus, tehokkuus ja laajennettavuus. Käyttämällä standardisoitua rajapintatekniikkaa saavutetaan parempi yhteensopivuus, ja uusien samaa rajapintaa käyttävien sovellusten kehitys on helpompaa ja nopeampaa.

Määrittelemällä oma tietoliikenneprotokolla tiedonvaihtoon socket-yhteyden yli voidaan viestien muoto optimoida tarkoitukseen sopivaksi, jolloin saavutetaan suurin tehokkuus ja tiedonsiirron nopeus. Tämä ratkaisu kuitenkin vaikeuttaa sovellusten toteutusta uusille alustoille, koska valmiita sovelluskehityksen apuvälineitä ei ole saatavilla, ellei myös niitä toteuteta itse.

Jos käytettävissä on kannettava tietokone suurine näyttöineen sekä verkkoyhteys, voidaan käyttöliittymä helposti toteuttaa dynaamisesti luotavilla www-sivuilla ja web-lomakkeilla. Tällöin ongelmaksi muodostuu toimenpiteissä tarvittavien oheislaitteiden, kuten GPS-paikantimen, tuottaman tiedon yhdistäminen automaattisesti www-selaimen tiedonsyöttökenttiin, eikä tämä keino sovellu esimerkiksi matkapuhelimella käytettäväksi. Tässä ratkaisussa myös tietoliikenneyhteyden täytyy olla luotettava, koska lomakkeiden tietoja ei voida tallentaa myöhempää lähetystä varten.

Yksi keino rajapinnan toteuttamiseen on Web Service. W3C (World Wide Web Consortium) on kansainvälinen järjestö, joka kehittää web-standardeja, tai suosituksia, kuten se itse kutsuu niitä. W3C on määritellyt Web Servicen seuraavasti (W3C 2004):

> *”Web Service on ohjelmistojärjestelmä, joka tukee tietokoneiden välistä vuorovaikutusta tietoverkon yli. Siinä on rajapinta, joka on määritelty kone-luettavassa muodossa (eritoten WSDL:llä). Muut järjestelmät keskustelevat Web Servicen kanssa kuvauksen mukaisesti käyttäen SOAP-viestejä, tyypillisesti*

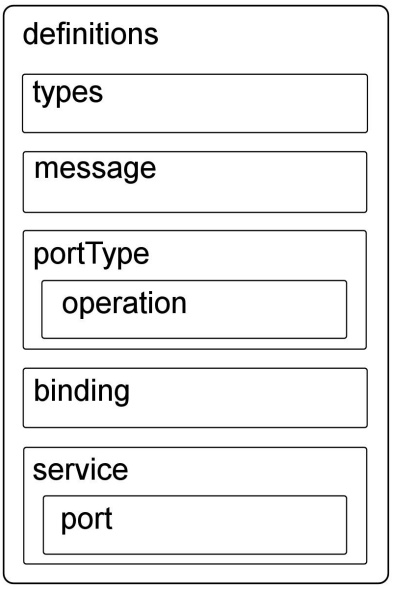
HTTP:n kuljettamina ja XML:llä muotoiltuina yhdessä muiden web-standardien kanssa.”

Web Serviceksi kuitenkin yleisesti nimitetään mitä tahansa sovellusta, jolla voidaan suorittaa funktiokutsuja tietoverkon kautta, ja joka käyttää standardeja Internet-protokollia sekä XML-muotoisia viestejä. Keskeisiä Web Service -määritelmiä ja -protokollia ovat WSDL (Web Service Description Language), SOAP ja UDDI (Universal Description, Discovery and Integration). Lisäksi tässä luvussa esitellään XML-RPC (XML Remote Procedure Call), joka on hieman yksinkertaisempi vaihtoehto SOAP:lle.

4.1 Web Service Description Language (WSDL)

WSDL on XML-pohjainen kuvauskieli jota käytetään Web Service -rajapintojen kuvaamiseen. Sillä määritellään kuinka kuvattavaa Web Serviceä käytetään, mitä funktioita voidaan suorittaa sekä SOAP-viestien muotoilu. Web Serviceä käyttävä sovellus voi WSDL-dokumentin perusteella muodostaa oikeanlaisen viestin ja lähettää sen oikeaan osoitteeseen. WSDL-dokumentin rakenne on esitetty kuvassa 3. Se muodostuu seuraavista osista:

- types - Määrittelee message-osan käyttämät tietotyypit
- message - Kuvaa lähetettävät ja vastaanotettavat sanomat
- portType - Yhdistää sanomat operaatioiksi
- binding - Määrittää käytettävän tiedonsiirtoprotokollan
- service - Määrittelee palvelun ”portit”, eli verkko-osoitteet



Kuva 3. WSDL-dokumentin rakenne

4.2 SOAP

W3C:n määritelmän mukaan SOAP on ”kevyt protokolla tiedonvaihtoon hajautetussa ympäristössä” eli se määrittää Web Servicen ja asiakassovelluksen välisten viestien muodon. Sovellus lähettää Web Servicelle SOAP-pyynnön, johon Web Service vastaa SOAP-vastauksella. SOAP-pyynnöt ja -vastaukset ovat XML-pohjaisia viestejä, jotka käyttävät siirtoprotokollanaan yleensä HTTP:tä. SOAP version 1.2 myötä sen nimi ei enää ole lyhenne mistään, mutta aiemmissa versioissa se tuli sanoista Simple Object Access Protocol.

Esimerkki SOAP-pyynnöstä joka kutsuu Web Servicen funktiota `GetLastTradePrice` parametrinaan `DIS`:

```
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <m:GetLastTradePrice xmlns:m="Some-URI">
      <symbol>DIS</symbol>
    </m:GetLastTradePrice>
```

```
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP-viestit lähetetään HTTP POST-viestinä. Kaikissa POST-viesteissä täytyy olla tieto lähetettävän tiedon tyypistä sekä sisällön koko. Nämä tiedot sisällytetään viestin otsikkokenttiin `Content-Type` ja `Content-Length`. Lisäksi SOAP-viestin otsikossa täytyy olla `SOAPAction` -kenttä, joka määrittää kutsuttavan funktion nimen. Esimerkki SOAP-viestin otsikkotiedoista:

```
POST /soapservice.php HTTP/1.0
User-Agent: SoapClient/1.0
Host: www.example.com
Content-Type: text/xml
Content-length: 296
SOAPAction: "urn:GetLastTradePrice"
```

Web Service vastaa `GetLastTradePrice` -kutsuun seuraavanlaisella SOAP-vastauksella:

```
<SOAP-ENV:Envelope"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <m:GetLastTradePriceResponse xmlns:m="Some-URI">
      <Price>34.5</Price>
    </m:GetLastTradePriceResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.3 Universal Description, Discovery and Integration (UDDI)

UDDI on eräänlainen Web Service -puhelinluettelo. Web Service -palveluntarjoaja voi rekisteröidä palvelunsa UDDI:n avulla, jolloin palveluita etsivät sovellukset voivat löytää sen. UDDI-rekisteri koostuu kolmesta osasta, valkoisista, keltaisista ja vihreistä sivuista. Valkoiset sivut sisältävät yleistä tietoa palvelun tarjoajasta, yhteystietoja,

sähköpostiosoitteita tai verkkosivujen osoitteen. Keltaiset sivut luokittelevat palvelun tarjoajan maantieteellisen sijainnin, liiketoiminta-alan tai jonkin muun ominaisuuden perusteella. Vihreät sivut sisältävät Web Servicen tekniset yksityiskohdat (WSDL-kuvauksen).

4.4 XML-RPC

XML-RPC on etäkutsuprotokolla, jolla voidaan suorittaa funktiokutsuja Internetin kautta. Se käyttää kutsujen muotoiluun XML:ää ja siirtoprotokollana HTTP:tä. XML-RPC eriytettiin SOAP:sta sen kehityksen alkuvaiheessa ja pidettiin hyvin yksinkertaisena. Sen viesteissä ei käytetä nimiavaruusmäärittelyjä eikä edes elementtien attribuutteja.

XML-RPC-kutsu sisältää yhden `methodCall` -elementin, joka taas sisältää `methodName` -elementin, jossa määritellään kutsuttavan funktion nimi. Tämän jälkeen listataan kutsun parametrit, jos kutsu niitä tarvitsee. Esimerkki XML-RPC-kutsusta, joka vastaa aiemmin luvussa 4.2 esitettyä SOAP-pyyntöä:

```
<?xml version="1.0"?>
<methodCall>
  <methodName>GetLastTradePrice</methodName>
  <params>
    <param>
      <value>
        <string>DIS</string>
      </value>
    </param>
  </params>
</methodCall>
```

XML-RPC-vastaus sisältää yhden `methodResponse` -elementin, joka sisältää funktiokutsun paluuarvot. XML-RPC-vastaus edellä esitettyyn pyyntöön näyttää seuraavalta:

```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
      <value>
        <double>34.5</double>
      </value>
    </param>
  </params>
</methodResponse>
```

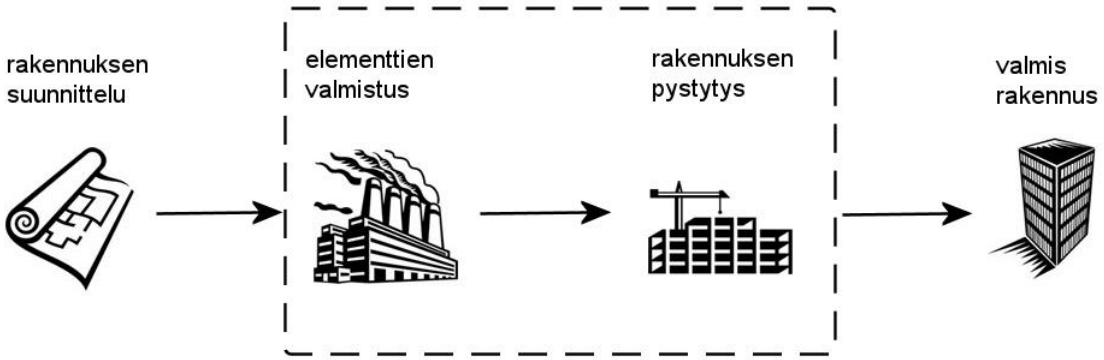
XML-RPC-kutsut lähetetään HTTP POST -viestinä. Minimissään viestissä täytyy olla seuraavat otsikkokentät:

```
POST /rpcserver.php HTTP/1.0
User-Agent: PHPRPC/1.0
Host: rpc.example.com
Content-Type: text/xml
Content-length: 154
```

`Host`-kenttä sekä POST-rivin osoite määrittävät yhdessä XML-RPC-palvelimen sijainnin. `User-Agent` -kentässä on jokin tunnistettava nimi kutsuvalle sovellukselle. `Content-Type` -kentän täytyy olla `text/xml`, ja `Content-length` -kentän täytyy sisältää viestin oikea koko.

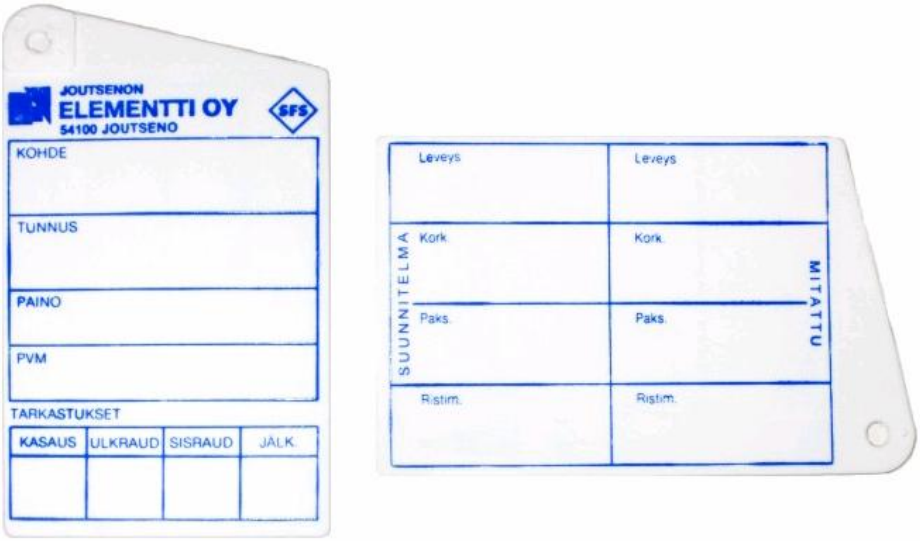
5 MOBIILITEKNOLOGIA RAKENNUSTEOLLISUUDESSA

Tämä diplomityö liittyy Mobilding-hankkeeseen, jossa tutkitaan mobiiliteknologian käyttöä rakennusteollisuuden elinkaaritiedon hallintaan. Rakennuksen elinkaaren vaiheita ovat suunnittelu, elementtien valmistus, rakennuksen pystytys sekä valmis rakennus (kuva 4). Tämän työn aihepiirissä keskitytään vaiheisiin rakennuselementtien valmistuksesta rakennuksen pystytykseen.



Kuva 4. Rakennuksen elinkaaren vaiheet

Rakennuselementtien tunnistaminen on perinteisesti suoritettu niihin kiinnitetyn paperisen tai muovisen elementtikortin perusteella. Kuvassa 5 on muovinen elementin tunnistekortti, johon kirjoitetaan elementin tunniste- ja mittatiedot. Tämän menetelmän etuina on nopea tunnistaminen ja riippumattomuus ulkoisista järjestelmistä. Menetelmän haittoja ovat kortin mahdollinen irtoaminen, jolloin tunnistetiedot kadotetaan, ja tietojen sijainti vain tässä yhdessä paikassa. Kortit irrotetaan viimeistään elementtiä asennettaessa, eikä elementtiä voida enää varmuudella tunnistaa asennuksen jälkeen. Valamalla elementin sisään RFID-tunniste saadaan se pysymään paikallaan niin kauan kuin rakennus on pystyssä. Tällöin elementti voidaan tunnistaa myös valmiista rakennuksesta, mikäli tunnisteen sijainti on tiedossa ja sellaisessa paikassa että se päästään lukemaan. Tunnistekorttiin kirjataan elementin tunnistekoodi, eikä elementtiä sen perusteella voida erottaa muista identtisen muotoisista elementeistä, mutta RFID-tunnisteiden avulla elementit voidaan tunnistaa yksilötasolla. Elementtikorttia ei voida täysin korvata, koska elementin paino täytyy olla näkyvissä sen nostoa varten.



Kuva 5. Muovinen elementtikortti

Mobilding-hankkeessa kehitetyssä järjestelmässä betonisiin rakennuselementteihin sisällytetään valuvaiheessa RFID-tunniste, joka lukemalla elementti voidaan jatkossa tunnistaa. Tunnisteet itsessään eivät sisällä mitään tietoa elementeistä vaan pelkän sarjanumeron, jonka perusteella elementti tunnistetaan yksilöllisesti. Tunnisteen lukemisesta ei siis ole mitään hyötyä ulkopuolisille. Tunnisteet sijoitetaan jokaiseen elementtiin tiettyyn ennalta sovittuun kohtaan, jotta ne voidaan helpommin löytää ja lukea. Kun tunniste on luettu lukulaitteella, voidaan sen sarjanumeron perusteella noutaa elementin tiedot tietokannasta. Tässä käytetään apuna mobiiliteknologiaa, kannettavia päätelaitteita sekä langattomia tiedonsiirtomenetelmiä.

Elementin suunnittelupiirustuksissa sille on merkitty tietyt ulkomitat. Elementin valua varten sille tehdään muotti, mutta muotin valmistuksessa voi tapahtua virheitä joiden takia valmistunut elementti ei täysin vastaa suunniteltua. Elementtien ulkomittojen tarkistaminen on tehty aikaisemmin perinteisillä menetelmillä, käyttäen mittanauhaa, kynää ja paperia. Kun mittatietojen tarkistus suoritetaan mobiililaitteella ja niitä kerätään tietokantaan, voidaan tietoja analysoida helposti. Niistä voidaan tehdä tilastoja, joista nähdään esimerkiksi jos elementit ovat liian pitkiä tai lyhyitä, ja tehdä tarvittavat muutokset valumuottiin. Elementin valmistaja voi myös todistaa valmistaneensa suunnitelmien mukaisen elementin, jos sen asennuksessa tulee ongelmia ja etsitään vastuullista osapuolta.

Aikaisemmissa testeissä on käyttökohteeseen sopivimmaksi tekniikaksi valittu HF-taajuusalueen RFID-tunnisteet (Kallonen 2006). Esitesteissä vertailtavina oli HF- ja UHF-alueiden tunnisteita, joista HF oli vähemmän altis väliaineen, eli tässä tapauksessa betonin, aiheuttamille häiriöille. Myös HF-tunnisteiden lukuetäisyys pieneni jonkin verran kun ne upotettiin betoniin. Jatkotesteissä vertailtiin kahta tekniikaltaan vastaavaa, mutta muodoltaan erilaista HF-tunnistetta, joista toinen oli pyöreä (halkaisija noin 45 mm) ja toinen neliömäinen (noin 60x60 mm). Tunnisteiden erot olivat siis ainoastaan antennin koossa ja muodossa. Testatuista tunnisteista neliömäinen osoittautui paremmin toimivaksi. Kun tunnisteet upotettiin 40 mm betonin pinnan alle, neliömäinen onnistuttiin lukemaan 90 mm etäisyydeltä betonin pinnasta, mutta pyöreä täytyi lukea 35 mm etäisyydeltä. Pelkässä ilmassa neliömäisen tunnisteen lukuetäisyys oli 160 mm ja pyöreän 80 mm. Myös veden, jään sekä tunnisteen asennon vaikutus lukuetäisyyteen tutkittiin.

Kallonen (Kallonen 2006) on suorittanut myös käytännön testejä, joissa RFID-tunnisteita sijoitettiin kolmeen erityyppiseen betonielementtiin, ulkoseinäelementtiin ja parvekkeen pieli- sekä lattiaelementtiin. Yhteensä 29 elementtiin valettiin tunniste, ja nämä tunnisteet yritettiin löytää valmiista rakennuksesta. Tunnisteista löydettiin ainoastaan 17, mutta kymmentä elementtiä ei päästy edes lukemaan, koska ne olivat mittauspäivänä rakennustarvikkeiden alla tai niiden eteen oli asennettu muita rakenteita. Tämä jättää jäljelle kaksi elementtiä joiden lukeminen ei onnistunut. Nämä elementit ovat saattaneet jäädä löytymättä siksi, että ne oli asennettu viereiseen taloon jossa käytetään samanlaisia elementtejä. Käytännön testeissä tuli ilmi tärkeä asia tunnisteiden sijoittamisesta. Koska osa tunnisteista jäi valmiissa rakennuksessa muiden rakenteiden taakse, tulee tunnisteiden sijoituspaikka elementissä siis valita huolellisesti, tutkimalla koko rakennuksen piirustuksia.

5.1 Tietojärjestelmä

Tietojärjestelmän ytimenä on palvelinkone, jolla toimii tietokanta- sekä www-palvelinohjelmisto. Palvelimen tietokanta sisältää kaikista rakennusprojekteissa käytettävistä elementeistä niiden tunnistetiedot, mitat, tilatiedot, sijaintitiedot sekä

virheraportit. Tietokannassa on myös tiedot projektien eri osapuolista ja käyttöoikeuksista. Tietokantaan saadaan tuotua tiedot rakennuksessa käytettävistä elementeistä Tekla Structures -suunnitteluohjelmasta automatisoidusti tai syöttämällä tiedot käsin www-käyttöliittymään elementin piirustusten perusteella. Tietokantaan lisätään elementteihin upotettavan RFID-tunnisteen sarjanumero, jonka perusteella voidaan yhdistää luettu tunniste oikeisiin tietoihin. Sarjanumerot syötetään tietokantaan tehtaalla käyttäen tavallista työasemaa ja lomakkeisiin perustuvaa www-käyttöliittymää. Ennen kuin RFID-tunniste sijoitetaan valettavaan elementtiin, se luetaan tietokoneeseen liitetyllä USB-liitäntäisellä RFID-lukijalla, josta sen sarjanumero saadaan lomakkeeseen. Samalla syötetään elementtitunnus. Näin tunnisteen ja elementin tietojen välille luodaan sidos, ja tästä lähtien lukemalla tunnus voidaan sitä vastaavan elementin tiedot saada tietokannasta. Tämän jälkeen täytyy vielä varmistaa että tietokantaan kirjattu RFID-tunniste upotetaan oikeaan elementtiin. Tämä voidaan tehdä vaikkapa merkitsemällä tussilla tunnisteen pintaan sen elementin tunnus, johon se kuuluu.

Tässä työssä palvelimelle suunnitellaan uusi rajapinta. Tätä rajapintaa voidaan käyttää mobiililaitteelle toteutettavalla sovelluksella palvelimen kanssa kommunikointiin. Rajapintaan määritellään funktiot eri käyttötapauksissa tarvittavien elementtitietojen hakuun ja päivitykseen.

5.2 Käyttötapaukset

Sovellukselle kartoitettiin useita erilaisia käyttötapauksia, jotka sijoittuvat elementin elinkaaren eri vaiheisiin elementtitehtaalta rakennukselle. Eri käyttötapauksissa siirretään erilaista elementteihin liittyvää tietoa mobiililaitteen ja tietokannan välillä. Viestit vaihtelevat erilaisista elementin tilan ilmoittavista kuittauksista varastopaikkahakuihin ja elementin mittatietoihin. Työntekijällä on työtehtävistään riippuen mukanaan mobiililaite sekä erilaisia oheislaitteita, RFID-lukijan lisäksi voidaan tarvita GPS-paikanninta tai laseretäisyysmittaria. Oheislaitteet yhdistetään puhelimeen langattomasti Bluetoothilla, ja niiden ulkonäkö sekä koko järjestelmän rakenne on esitetty kuvassa 6. Elementin tunnistaminen tapahtuu lukemalla sen sisältämä RFID-tunniste. Tunnisteen sisältämän sarjanumeron perusteella elementti

voidaan tunnistaa yksilöllisesti. Tunnisteen lukeminen ja elementin yksilöiminen on edellytys useimmille seuraavissa kappaleissa esitellyille käyttötapauksille. Taulukossa 3 luetellaan toteutetulle sovellukselle kartoitetut käyttötapaukset sekä niissä tarvittavat oheislaitteet.

Kuva 6. Järjestelmän rakenne

Taulukko 3. Sovelluksen eri käyttötapauksissa tarvittavat oheislaitteet

Käyttötapaus	**Tarvittavat oheislaitteet**
Tarkastus ja mittaus	RFID-lukija, laseretäisyysmittari
Varastosijainnin ilmoitus	RFID-lukija, GPS-paikannin (ei pakollinen)
Varastokartan haku	GPS-paikannin (ei pakollinen)
Tietojen haku ja asennuskuittaus	RFID-lukija
Vikailmoituksen tekeminen	RFID-lukija
Vikailmoitusten haku ja päivitys	RFID-lukija

Elementin *tarkastus ja mittaus* suoritetaan elementtitehtaalla kun valettu elementti on kuivunut. Mittaamalla valmistunut elementti saadaan selville, vastaako se suunnitelmien mukaisia mittoja. Ensin elementti tunnistetaan lukemalla RFID-tunniste ja sen jälkeen mitataan laseretäisyysmittarilla, joka yhdistetään puhelimeen Bluetoothilla. Mittalaite

välittää mittaamansa etäisyydet puhelimeen, ja elementin todelliset mittatiedot lähetetään tietokantaan jossa se samalla kuitataan tarkastetuksi. Palvelin lähettää paluuviestissä varoituksen, jos elementin toteutuneet ulkomitat ylittävät niille asetetut toleranssirajat.

Varastosijainnin ilmoitus suoritetaan varastoitaessa elementti joko betonitehtaan tai rakennuspaikan varastoon. Elementti tunnistetaan ja kuitataan varastoiduksi, jolloin sen sijainti ilmoitetaan tietokantaan. Ulkovaraston tapauksessa varastopaikan sijainti voidaan ilmoittaa puhelimeen liitetyn GPS-paikantimen antamien koordinaattien perusteella. Jos paikanninta ei ole käytettävissä tai elementti varastoidaan sisävarastoon, voidaan sijainti ilmoittaa sanallisessa muodossa kirjoittamalla varastopaikan tunnus.

Varastokartan haku voidaan suorittaa kun etsitään tietyn tyyppistä elementtiä varastoalueelta. Varastoalueesta sekä elementtien sijainnista voidaan hakea karttakuva tietokannasta. Koska etsittävän elementin RFID-tunnistetta ei tietenkään voida lukea, hakuperusteena käytetään elementtitunnusta ja halutun varastoalueen tunnusta. Palvelimen palauttamaan karttakuvaan on merkitty haetun tyyppisten elementtien sijainnit kyseisellä varastoalueella, ja siihen voidaan myös merkitä käyttäjän oma sijainti, mikäli GPS-paikannin on käytettävissä.

Elementin *tietojen haku* voi olla tarpeen sitä asennettaessa. Ennen elementin asennusta se voidaan tunnistaa ja noutaa tietokannasta sen mittatiedot (suunnitellut sekä toteutuneet), paino, suunniteltu asennuspäivämäärä, tilatieto sekä lista mahdollisista vioista. Tässä vaiheessa asentaja voi tarkastaa, vastaavatko elementin toteutuneet mitat suunniteltuja, ja ottaa mahdolliset poikkeamat huomioon asennuksessa. Kun elementti on asennettu paikalleen, asentaja lähettää *asennuskuittauksen* palvelimelle.

Jos elementissä havaitaan jokin valmistus- tai muu vika, voidaan tietojärjestelmään lähettää *vikailmoitus*. Ensin elementti tunnistetaan, ja vikailmoitusta varten valitaan listasta jokin esimääritelty vikatyyppi. Sen jälkeen voidaan tarvittaessa kirjoittaa tarkempi vian kuvaus tai ottaa puhelimen kameralla vikakohdasta kuva tietokantaan tallennettavaksi.

Tunnistetulle elementille voidaan tehdä *vikailmoitusten haku ja päivitys*. Vikailmoitusten perusteella voidaan ryhtyä korjaustoimenpiteisiin sekä välttää saman vian kirjaaminen useaan kertaan. Vikailmoituksista saadaan vikatyypin lisäksi yksityiskohtaisemmat vikakuvaukset. Kun vika on korjattu, sen tila voidaan päivittää tietokantaan.

6 TOTEUTUS

Työn käytännön osuudessa toteutettiin valittuja tekniikoita käyttäen mobiililaitteelle sovellus rakennusteollisuuden liikkuvan työntekijän tarpeisiin. Sovellusta toteutettaessa pyrittiin ottamaan huomioon sekä mobiiliympäristön että liikkuvan rakennustyöntekijän työympäristön tekniikalle asettamat rajoitteet ja vaatimukset. Sovelluksesta pyrittiin myös tekemään mahdollisimman helppokäyttöinen, jotta se ei avustamisen sijaan vaikeuttaisi työtehtävien suoritusta.

6.1 Tietokantarajapinta

Tietokantapalvelimelle suunniteltiin uusi rajapinta, jonka avulla internet-yhteydellä varustetulla mobiililaitteella voidaan käsitellä tietokannan tietoja (katso kuva 6 aiemmin). Mobiililaitteen ja tietojärjestelmän välillä vaihdettavien viestien sisältö määriteltiin, mutta rajapinnan toteutusta ei tehty itse. Rajapinnan toteutuksessa päädyttiin SOAP-viesteihin, johon suurimpana syynä oli yhteensopivuus. SOAP on yleisesti käytetty, joten sille on olemassa laaja valikoima sovelluskehitystyökaluja eri alustoille. Työkalujen avulla voidaan luoda automaattisesti rajapinnan WSDL-kuvauksen perusteella koodia SOAP-viestien käsittelyfunktioihin, joten uusien asiakasohjelmien toteutus erilaisille päätelaitteille helpottuu. SOAP-viestit ovat XML-pohjaisuudesta johtuen kookkaita suhteessa niiden sisältämään tietomäärään. Määrittelemällä omat viestiformaatit voitaisiin viestien koko optimoida mahdollisimman pieneksi, mikäli tiedonsiirron määrä halutaan minimoida. Tällöin kuitenkin funktiot viestien käsittelyyn joudutaan ohjelmoimaan uudestaan aina kun asiakassovellus halutaan siirtää uudelle alustalle. Toteutetun SOAP-rajapinnan WSDL-kuvaus on esitetty liitteessä 1. Rajapintaan määriteltiin joukko funktioita, joita kutsumalla tietokannasta voidaan hakea tai päivittää eri käyttötapauksissa tarvittavaa tietoa. Funktiot kuvauksineen on esitetty taulukossa 4.

Taulukko 4. Rajapinnan funktiot

Funktio	Kuvaus
getInfo	Hakee elementin tunniste-, mitta-, sekä tilatiedot
sendMeasurements	Lähettää elementin mittatiedot
sendLocation	Lähettää elementin sijainnin
sendState	Päivittää elementin tilan
getMapLocation	Hakee varastoalueen kartan
getStorages	Hakee tietyn tyyppisten elementtien GPS-koordinaatit
reportError	Lähettää vikaraportin
sendErrorFile	Lähettää vikaraporttiin liittyvän tiedoston
getErrors	Hakee listan elementille raportoiduista vioista
updateErrorState	Päivittää vikaraportin tilan

6.2 Päätelaitteen valinta

Mobiilisovellus päätettiin toteuttaa Symbian S60 -matkapuhelimelle C++-kielellä. Matkapuhelin on pienen kokonsa vuoksi tähän käyttökohteeseen hyvin soveltuva laite, ja rakennustyöntekijöillä on useimmiten joka tapauksessa puhelin mukanaan. Puhelimet ovat myös fyysiseltä rakenteeltaan suhteellisen kestäviä esimerkiksi kämmentietokoneisiin verrattuna, ja rakennuksilla tai tehtailla työskennellessä laitteet saavat usein kolhuja. Matkapuhelin on työntekijöille entuudestaan tuttu laite, joten sen käyttöönoton opettelussa ei mene niin kauan aikaa, vaikka älypuhelimet ovatkin monimutkaisempia kuin perusmallit. Matkapuhelimen etuna moniin muihin päätelaitteisiin verrattuna on myös edullinen hinta.

Tiedonsiirtotekniikoista GPRS:llä saavutetaan suurin peittoalue, eikä se vaadi oman verkon rakentamista, kuten WLAN-verkkoa käytettäessä, joten se on hitaahkosta nopeudestaan huolimatta sovelias valinta. Myös GPRS-verkossa on pieniä katvealueita,

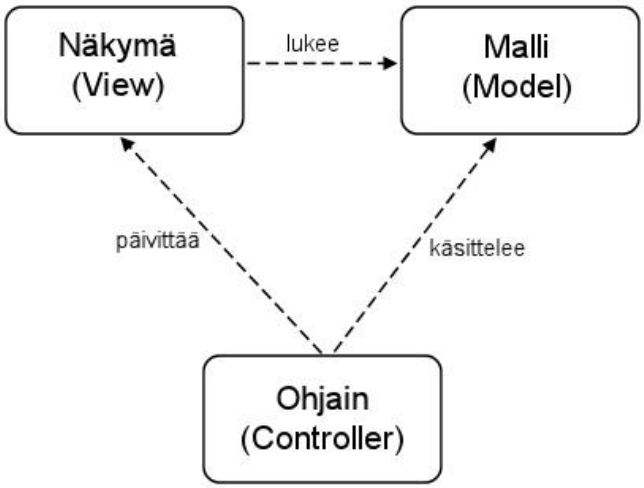
joissa tietoliikenneyhteys ei toimi. Lisäksi rakennuksilla voi olla esimerkiksi kellaritiloja, joissa heikon kuuluvuuden vuoksi tietoliikenneyhteys ei ole saatavilla, ja tämä voi haitata sovelluksen käyttöä. Ongelmaan voidaan varautua tallentamalla sekä tarvittavat tiedot laitteeseen etukäteen että palvelimelle lähetettävät tiedot myöhempää lähetystä varten, mutta tämän työn puitteissa ei sellaista ominaisuutta toteutettu.

Työssä käytetylle Symbian S60 2nd Edition -alustalle löytyi vain yksi C++-kielinen SOAP-implementaatio, gSOAP (Van Engelen 2007). Jos matkapuhelimelle halutaan toteuttaa uusi asiakasohjelma, joka käyttää samaa tietokantarajapintaa, S60 3rd Editioniin sisältyvä Nokia Mobile Web Services Framework tarjoaa ohjelmointirajapinnat Web Servicen hyödyntämiseen.

6.3 Asiakassovelluksen rakenne

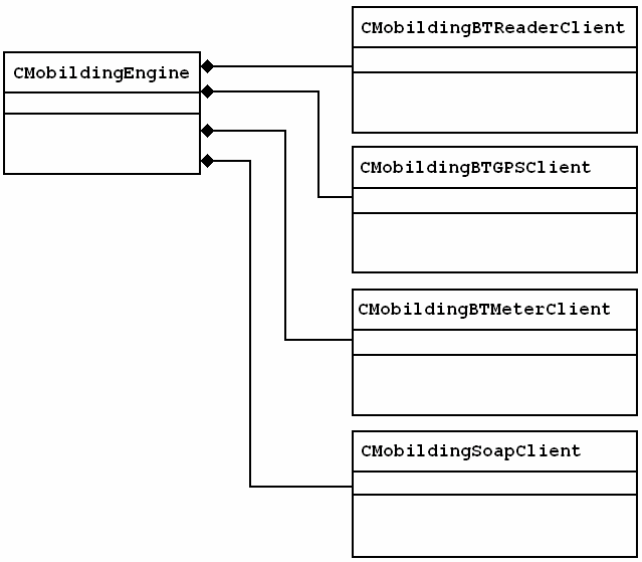
Mobiililaitteelle toteutettu asiakassovellus noudattaa Symbian-ohjelmoinnissa yleisesti käytettyä MVC-suunnittelumallia (Model-View-Controller), joka on esitetty kuvassa 7. MVC-mallissa sovellus jaetaan kolmeen osaan (Digia Inc. 2003):

- *Malli* (Model) sisältää sovelluksen tilan ja toiminnallisuuden
- *Näkymä* (View) esittää mallin sisältämän tiedon käyttäjälle
- *Ohjain* (Controller) vastaanottaa käyttäjän komennot



Kuva 7. MVC-suunnittelumalli

MVC-mallissa eri osat ovat irrallisia, ja muutokset yhdessä osassa eivät yleensä vaadi mitään muutoksia muissa osissa. Sovellus on myös hyvin uudelleenkäytettävissä ja helposti siirrettävissä toiselle käyttöliittymälle vain *Näkymä*-osaa muuttamalla. Toteutetun mobiilisovelluksen *Malli*-osa sisältää sen ydintoiminnallisuuden, eli Bluetooth-laitteisiin sekä SOAP-viestien käsittelyyn liittyvät funktiot. Kuvassa 8 on esitetty pääpiirteissään *Malli*-osan luokkakaavio.



Kuva 8. Sovelluksen *Malli*-osan luokkakaavio

`CMobildingEngine` -luokka tarjoaa yhteisen rajapinnan kaikille Bluetooth- sekä SOAP-funktioille. Luokka `CMobildingBTReaderClient` vastaa RFID-lukijan yhteydenmuodostuksesta sekä sen lähettämän tiedon vastaanotosta. `CMobildingBTGPSClient` -luokka vastaa GPS-paikantimen yhteydenhallinnan ja tiedon vastaanoton lisäksi koordinaattitietojen tulkitsemisesta. Lasermittalaitteen yhteydenhallinnasta vastaa luokka `CMobildingBTMeterClient`. Koska mittalaite lähettää myös paljon muuta kuin pelkän mittatiedon, `CMobildingBTMeterClient` -luokka tulkitsee vastaanottamastaan binääridatasta tarpeellisen tiedon, eli luetun etäisyyden. `CMobildingBTSoapClient` -luokka sisältää funktiot SOAP-viestien käsittelyyn. Luokan toteutuksessa käytettiin hyväksi GPL-lisenssin alaista gSOAP-työkalua (Van Engelen 2007) C++-koodin luomiseen Web Servicen WSDL-kuvauksesta.

6.4 Käyttöliittymä ja toiminta

Sovelluksen käyttöliittymästä pyrittiin tekemään mahdollisimman helppokäyttöinen. Tekstin kirjoituksen tarve on minimoitu, ja suurin osa valinnoista tehdään listasta valitsemalla. Ohjelman perustila ennen laitteiden yhdistämistä sekä käyttöliittymän eri osien nimet on esitetty kuvassa 9. Luvussa 2.4 esiteltyjen Nielsenin ohjeiden ensimmäisessä kohdassa neuvotaan näyttämään käyttäjälle järjestelmän tila (Nielsen 2005). Ohjelman Navipane-palkissa on koko ajan näkyvissä Bluetooth-laitteiden yhteyden tila, niin että kun laite on yhdistetty, laitteen nimen ympärille ilmestyy kaarisulkeet. Tietokantahakujen aikana käyttäjälle näytetään animaatio, jolla ilmaistaan operaation olevan käynnissä eikä käyttäjä luule ohjelman jumittuneen. Otsikkopalkissa on ohjelman perustilassa näkyvissä ”Mobilding” -teksti, joka vaihtuu ilmaisemaan ohjelman eri tiloja.

Kuva 9. Käyttöliittymän perustila ja eri osien nimet

Käyttöliittymän ulkonäössä noudatetaan mahdollisimman pitkälle S60-käyttöliittymän yleistä ulkoasua Nielsenin ohjeiden neljännen kohdan mukaisesti (Nielsen 2005). Oikeanpuoleinen toimintonäppäin peruuttaa toimintoja ja poistuu eri näkymistä. Vasemmalla näppäimellä suoritetaan valinnat ja hyväksymiset. Käyttöliittymä käyttää

hyväkseen valmiita komponentteja, kuten listoja ja tiedonsyöttöruutuja, joten ulkonäkö on hyvin pitkälle yhteneväinen muiden S60-sovellusten kanssa. Sovelluksen käyttöliittymä myös noudattaa puhelimen asetuksista valittua väriteemaa.

Nielsenin ohjeiden seitsemännessä kohdassa neuvotaan antamaan käyttäjän säätää käyttöliittymä mieleisekseen (Nielsen 2005). Koska ohjelmassa on niin monta toimintoa joista käyttäjä pystyy valitsemaan RFID-tunnisteen luettuaan, voidaan ohjelman asetusvalikossa valita käyttäjän tarvitsemat toiminnot ja siten piilottaa tarpeettomat toiminnot. Tällöin valittavissa olevien toimintojen lista saadaan pidettyä mahdollisimman pienenä, eikä käyttäjän tarvitse selata pitkää listaa. Jos asetuksissa on valittuna vain yksi ainoa toiminto, ei toimintolistaa näytetä lainkaan kun tunniste luetaan, vaan toiminto suoritetaan heti. Käyttäjän valitsemien toimintojen perusteella ohjelman perusnäkymässä näytetään tai piilotetaan tietyt ohjeistustekstit. Esimerkiksi jos käyttäjä ei ole asetuksista valinnut elementtien mittausta käytettäväksi toiminnoksi, ei sovelluksen päänäkymässä näytetä ohjetekstiä ”Yhdistä laser-mittari”.

Sovelluksella voidaan muodostaa langaton Bluetooth-yhteys tarvittaviin oheislaitteisiin eli RFID-lukijaan, GPS-paikantimeen sekä laseretäisyysmittariin. Käyttäjä voi aloittaa lähistöllä olevien Bluetooth-laitteiden etsimisen ohjelman valikosta ja valita löytyneistä laitteista oikean. Bluetooth-laitteiden etsiminen on aikaa vievä prosessi, mutta laitteiden yhdistämistä voidaan jatkossa nopeuttaa syöttämällä ohjelman asetuksiin Bluetooth-laitteiden laiteosoitteet, jolloin yhteys muodostetaan automaattisesti ohjelman käynnistyksen yhteydessä. Laiteosoitteet voidaan myös lukea yhdistetystä laitteesta, jolloin vältytään osoitteen kirjoittamiselta. Automaattinen laitteiden yhdistäminen nopeuttaa huomattavasti sovelluksen käyttöönottoa, koska laitteita ei tarvitse enää etsiä ennen yhdistämistä. Tämä ominaisuus vastaa Nielsenin ohjeissa kohdassa seitsemän mainittuja oikoteitä (Nielsen 2005).

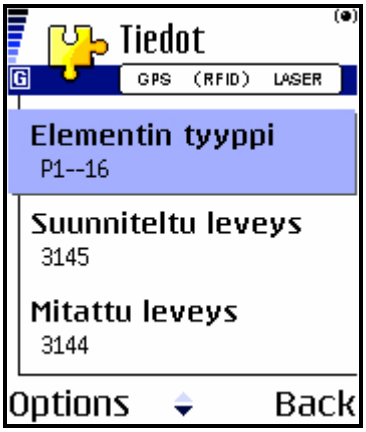
Elementin mittauksessa käyttäjän tulee pystyä asettamaan laseretäisyysmittari tarkasti oikeaan kohtaan elementin reunaan sekä suuntaamaan lasersäde oikein. Tästä syystä sovelluksen käyttöliittymä suunniteltiin siten, että se ei vaadi käyttäjältä puhelimen näppäimien painelua kesken mittauksen. Tällöin puhelin voidaan laskea pois kädestä ja

siten vapautetaan molemmat kädet etäisyysmittarin käsittelyyn. Jos jotakin tiettyä mittaa ei haluta mitata, sovellus sallii mittauksen ohituksen, jolloin mitta jätetään nollaksi. Ennen mittatietojen lähetystä voidaan myös lukea yksittäisiä mittoja uudelleen.

Tärkein oheislaite on RFID-lukija, jolla rakennuselementti voidaan tunnistaa yksilöllisesti lukemalla siihen sijoitettu RFID-tunniste. Kun lukijaan on onnistuneesti muodostettu Bluetooth-yhteys, sillä voidaan lukea tunniste painamalla liipaisinta tunnisteen lähistöllä. Feig Electronicin lukijalaite voidaan tietokoneen avulla säätää myös niin, että liipaisinta ei tarvitse erikseen painaa, vaan lukija lukee tunnisteen heti kun sellainen osuu lähistölle. Kun tunniste on luettu, lukija lähettää sen sarjanumeron Bluetooth-yhteyden kautta matkapuhelimelle, joka puolestaan näyttää ruudullaan kuvassa 10 näkyvän listan elementtiin liittyvistä toiminnoista, joista käyttäjä voi valita haluamansa. Näiden toimintojen lisäksi käyttäjä voi ohjelman päävalikosta valitsemalla hakea elementtitunnuksen ja varastoalueen nimen perusteella varastoalueesta karttakuvan, jossa näkyy haluttujen elementtien sijainti.

Kuva 10. Lista toiminnoista, joita tunnistetulle elementille voidaan suorittaa

Tietojen haku -toiminnossa sovellus hakee palvelimelta elementtiin liittyvät tiedot. Paluuviestin saatuaan sovellus esittää elementin tiedot näytöllä listana, joka on esitetty kuvassa 11. Tietojen haku suoritetaan rajapinnan SOAP-funktiolla `getInfo`. Funktiokutsun parametrina on elementin RFID-tunnisteesta luettu sarjanumero, ja palvelimen lähettämä paluuviesti sisältää elementin tunniste-, mitta- ja tilatiedot sekä mahdollisesti listan raportoiduista vikatyypeistä.



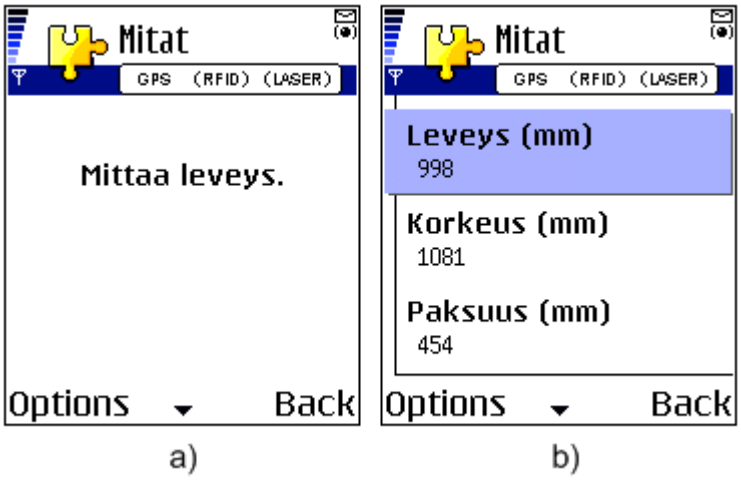
Kuva 11. Elementin tiedot

Varastokuittauksessa sovellus tarkistaa ensin, ovatko käyttäjän GPS-koordinaatit käytettävissä varastosijainnin ilmoittamiseen. Jos koordinaatit puuttuvat, käyttäjää pyydetään syöttämään varastotunnus tekstimuodossa. Varastokuittauksen SOAP-funktio on `sendLocation`, jonka parametrina on sarjanumeron lisäksi GPS-koordinaatit (pituus- ja leveysaste) sekä varastotunnus. Paluuviesti sisältää tiedon kuittauksen onnistumisesta.

Asennuskuittauksessa sovellus lähettää tilanmuutosviestin palvelimelle. Asennuskuittaus käyttää funktiota `sendState`, jonka parametreina ovat sarjanumero sekä ”asennettu” -tilaa vastaava tunnuskoodi. Paluuviestissä ilmoitetaan kuittauksen onnistuminen.

Ennen elementin mittauksen aloittamista sovellus kehottaa yhdistämään mittalaitteen, ellei sitä ole jo yhdistetty. Mittauksessa sovellus ohjeistaa ruudulla käyttäjää lukemaan yksi kerrallaan elementin mitat, aina samassa järjestyksessä. Kuvassa 12 a näkyy ohjeistus seuraavaksi luettavasta mitasta. Yksittäisten mittojen mittaus voidaan ohittaa kesken mittauksen valitsemalla ohjelman valikosta ”Ohita mitta” -valinta, jolloin kyseinen mitta jätetään nollaksi. Kun käyttäjä on mitannut kaikki mitat, ruudulla näytetään yhteenvetona kuvassa 12 b näkyvä lista luetuista mitoista, joita käyttäjä pääsee vielä selailemaan. Listasta valitun mitan voi halutessaan mitata uudelleen valitsemalla päävalikosta ”Lue mitta uudelleen” -toiminto. Mittatietojen lähetys käyttää rajapinnan funktiota `sendMeasurements`, jonka parametreina ovat sarjanumero sekä

mittatiedot. Paluuviesti voi sisältää tiedon mittojen lähetyksen onnistumisesta tai varoituksen, jos mitat ylittävät toleranssirajat.

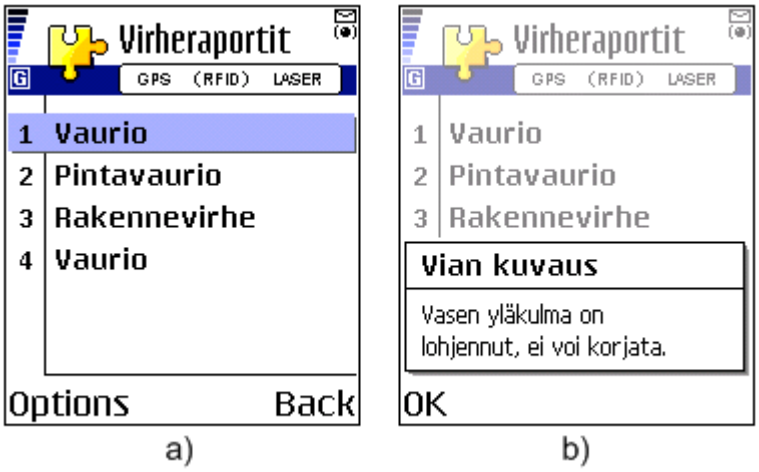


Kuva 12. Mittausnäkymän ohjeteksti (a) sekä lista mitatuista ulkomitoista (b)

Vikaraporteille on esimääritelty useita eri vikatyyppejä, joista käyttäjän on ensin valittava yksi. Vikaraporttia tehdessään käyttäjä voi päävalikosta toiminnon valitsemalla syöttää tarkemman vikakuvauksen tekstinä sekä ottaa vikakohdasta kuvan puhelimen kameralla (kuva 13). Vikaraportti lähetetään `reportError` -funktiolla, jonka parametreina ovat sarjanumero ja vikakuvaus. Funktiokutsun paluuviesti sisältää vikaraportin tunnusnumeron. Tämä tunnusnumero lisätään parametriksi `sendErrorFile` -funktioon, jolla raporttiin liittyvä kuva (tai mikä tahansa tiedosto) lähetetään palvelimelle.

Kuva 13. Kameranäkymä

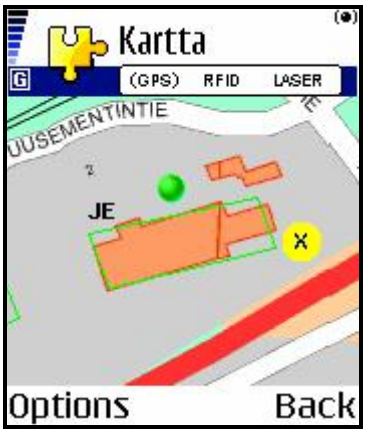
Vikaraporttien haussa sovellus noutaa `getErrors` -funktiolla elementin sarjanumeron perusteella sille tehdyt vikaraportit. Kuvassa 14 a näkyy ruudulla lista elementille raportoiduista vioista, ja valitsemalla yhden vikatyypin käyttäjä saa nähtäväkseen kyseisestä viasta tarkemman kuvauksen (kuva 14 b). Jos vika on korjattu, ilmoituksen tila voidaan päivittää, jolloin kyseistä vikaa ei enää palauteta vikaraporttien haussa. Vikailmoituksen tilan päivitys käyttää funktiota `updateErrorState`, jonka parametreina ovat vikailmoituksen tunnusnumero, vian uusi tila sekä tarvittaessa tekstimuotoinen kuvaus.



Kuva 14. Lista elementille raportoiduista vioista (a) sekä tarkempi vikakuvaus (b)

Elementin varastosijaintien karttahaku suoritetaan valitsemalla toiminto ”Hae varastokartta” ohjelman päävalikosta. Sovellus pyytää käyttäjää syöttämään elementin tunnuksen sekä varastoalueen nimen, joiden perusteella muodostetaan haku palvelimelle käyttäen `getMapLocation` -funktiota. Funktiokutsun parametreina lähetetään myös halutun kartan koko pikseleinä sekä mittakaava. Paluuviesti sisältää itse kuvan, tiedon kuvan formaatista sekä kartan vasemman ylänurkan ja oikean alanurkan GPS-koordinaatit. Näiden koordinaattitietojen perusteella voidaan käyttäjän oma sijainti piirtää kartalle, mikäli puhelimeen yhdistetty GPS-paikannin vastaanottaa koordinaattitietoja satelliiteilta. Kuvassa 15 on puhelimen ruudulle piirretty palvelimelta noudettu kartta. Karttaan on palvelimella lisätty halutun elementin sijainti (ympyrä jossa risti) sekä varastopaikan nimi ja alueen rajaus. Puhelimella kartan päälle on lisätty GPS-

paikantimen koordinaattien perusteella henkilön oma sijainti (pallo) joka päivittyy reaaliajassa käyttäjän liikkuessa.



Kuva 15. Karttakuva varastoalueesta

6.5 Testaus ja analyysi

Sovelluksen testaus kenttäolosuhteissa suoritettiin vain suppeasti. Todellisessa käyttöympäristössä, eli betonitehtaalla, testattiin elementtien mittausta, niiden varastointia sekä karttahakua. Laajamittaisempi kenttätestaus suoritetaan myöhemmin, ja testeissä saatujen kokemuksien perusteella sovelluksen kehitystä jatketaan.

Tehtaalla testattiin betonielementtien mittausta useilla erimuotoisilla ja -kokoisilla elementeillä. Valun jälkeen kuivuneet elementit oli varastoitu niin sanottuihin ”kampatelineisiin”, joissa elementit ovat pystyasennossa. Koska elementit sijaitsevat kammassa lähellä toisiaan, mittaus voidaan suorittaa vain reunimmaiselle elementille. Tämän takia elementit täytyy mitata heti kun ne nostetaan telineeseen. Lisäksi korkeiden ulkoseinäelementtien korkeuden mittauksessa tarvitaan tikapuita. Elementtien mittauksessa tarvitaan jonkinlainen mittausvaste, joka sijoitetaan mitattavan elementin reunaa vasten, jolloin etäisyysmittarin lasersäde suunnataan siihen kun mittaus suoritetaan. Mittausvasteen tulee olla sellainen, että säde on helppo suunnata siihen, ja sen täytyy pysyä paikallaan ilman apua, koska työntekijän on suoritettava mittaus yksin. Testeissä käytetty vaste valmistettiin paksusta vanerista, kiinnittämällä kaksi palaa toisiinsa suorassa kulmassa. Elementin ristimitan mittaus osoittautui haastavaksi, koska

lasersäde piti suunnata hyvin tarkasti vastekappaleen nurkkaan. Ristimitan mittauksessa täytyi käyttää mittalaitteen alapäästä esiin käännettävää piikkiä, jonka pää sijoitettiin siihen elementin nurkkaan, josta mittaus suoritettiin. Mittalaitteen sekä esiin käännetyn piikin ulkonäkö näkyy kuvassa 16. Kun piikki on käännetty esiin, laite mittaa etäisyyden sen päästä mittalaitteen etu- tai takareunan sijaan. Piikin tarkka sijoitus elementin nurkkaan vaatii käyttäjältä huolellisuutta, ja tämä yhdistettynä lasersäteen kohdistamiseen teki elementin ristimitan mittauksesta vaikeaa. Itse laseretäisyysmittari (Hilti PD 38) on hyvin tukevarakenteinen laite, mutta kohdistusapuna ja suorakulmana toimiva piikki on hieman heikko ja taipuisa.

Kuva 16. Laseretäisyysmittari piikki esiin käännettynä

Mittaustesteissä koehenkilönä toimi elementtitehtaan työntekijä. Ensin koehenkilölle opastettiin järjestelmän käyttöä, jonka jälkeen tämä suoritti itsenäisesti elementin mittauksen. Testissä tuli ilmi muutamia toiveita sovelluksen käyttöliittymän suhteen. Puhelimen toivottiin ilmoittavan äänimerkillä kun mittalaitteen lähettämä etäisyystieto on vastaanotettu onnistuneesti. Käyttöliittymän toivottiin myös näyttävän mitatut etäisyydet kesken mittausprosessin, eikä pelkästään sen jälkeen kun kaikki on mitattu. Myös mahdollisuutta palata edelliseen mittaan kesken mittausprosessin toivottiin.

Elementtien varastointia sekä karttahakua testattiin elementtitehtaan piha-alueella. Kyseinen tehdas ei tavallisesti varastoi elementtejä ulkotiloihin, eikä piha ollut kovin suuri, joten testitilanne ei täysin vastannut todellista tilannetta. Tietokantaan oli etukäteen syötetty piha-alueen kartta sekä sen koordinaattitiedot. Testissä ”varastoitiin” pelkkä RFID-tunniste, ei oikeata elementtiä.

Varastointitestissä käytettiin GPS-paikanninta, jonka ilmoittamien koordinaattien perusteella elementin sijainti ilmoitettiin tietojärjestelmään ja myöhemmin seurattiin omaa sijaintia varastoalueen kartalta. Paikantimen käynnistyksestä noin minuutin kuluttua laite alkoi lähettää koordinaattitietoja puhelimelle. Ensin suoritettiin varastokuittaus, jossa varastointisijainnin koordinaatit välitettiin palvelimen tietokantaan. Tämän jälkeen palvelimelta haettiin varastoidun elementin sekä elementtitehtaan tunnuksien perusteella kartta, jossa näkyi aikaisemmin varastoidun elementin sijainti. Käyttäjän oma sijainti piirtyi kartalle, ja kuljettaessa ympäri piha-aluetta sijainti päivittyi puhelimen ruudulla vastaamaan oikeaa sijaintia. Lähistöllä sijaitsevat suuret rakennukset heikentävät GPS-paikannuksen tarkkuutta, mutta jopa elementtitehtaan suuren hallin seinän vieressä kartalle piirretty sijainti vastasi hyvin tarkasti oikeaa sijaintia. Elementtien paikantamisessa riittää, että päästään näköetäisyydelle elementistä, joten testien mukaisessa tehdasympäristössä saavutetulla paikannustarkkuudella voidaan löytää useita metrejä pitkä elementti varastoalueelta.

6.5.1 Havaitut ongelmat

Elementtien mittauksessa havaittuja ongelmia olivat sopivan mittauksessa käytettävän vastekappaleen kehittäminen sekä poikkeavan muotoisten elementtien mittaus. Yksinkertainen vanerinen vastekappale toimi hyvin kaikissa muissa paitsi ristimitan mittauksessa. Ristimittauksessa lasersäde täytyi kohdistaa L-kirjaimen muotoisen vastekappaleen saumakohtaan, joten mittalaitteen suuntaus oli vaikeaa. Vastekappaleeseen voidaan lisätä esimerkiksi vinottain 45 asteen kulmassa oleva lisäkappale, johon lasersäteen kohdistus on helpompaa. Joskus elementtien reunoissa on viisteitä tai syvennyksiä, jolloin voi olla epäselvää, mistä kohtaa mittaus tulee suorittaa. Myös mittausvasteen asettaminen paikalleen voi olla tällöin vaikeaa.

Puhelimen ja oheislaitteiden yhdistäminen Bluetoothilla aiheutti ongelmia. Sovelluskehityksen jo alettua huomattiin, että testilaitteeksi valittu Nokia 6600 -puhelin ei tue useamman kuin yhden Bluetooth-laitteen yhtäaikaista yhdistämistä, joten joidenkin käyttötapauksien testaamista varten jouduttiin lainaamaan toinen puhelin. Useaa Bluetooth-yhteyttä tukevien S60 2nd Edition -puhelimien saatavuus on melko

huono. Siirtämällä sovellus S60 3rd Edition -alustalle voidaan käytettävä puhelin valita paljon suuremmasta valikoimasta. Toinen Bluetooth-ongelma liittyi käytettyyn Feig Electronic ID ISC.PRH101-B -malliseen RFID-lukijaan. Yhteyden muodostaminen lukijalaitteeseen osoittautui ajoittain ongelmalliseksi, koska lukijalaite pyrki aloittamaan itse Bluetooth-yhteyden muodostamisen, eikä vain odottanut yhteyspyyntöjä. Näin ollen kun puhelimelta yritettiin muodostaa yhteyttä, lukija ei hyväksynyt sitä koska se yritti jo itse muodostaa sitä.

Vaikka alustaksi valittu matkapuhelin on pieni laite, joutuu sovelluksen käyttäjä kantamaan mukanaan työtehtävistään riippuen yhdestä kolmeen oheislaitetta. Laitteiden määrää voidaan pienentää ainakin yhdellä käyttämällä sellaista puhelinmallia, jossa on sisäinen GPS-paikannin. Toinen vaihtoehto oheislaitteiden karsimiseksi on RFID-toiminnallisuuden sisältävän puhelimen käyttö, mutta kyseisten puhelinmallien tukemien RFID-tunnisteiden toimivuutta betonin sisällä ei ole testattu ja niiden lukuetäisyys on lyhyt ilman betoniakin.

GPS-laitteisiin liittyy myös toinen ongelma. Laitteen käynnistyksen jälkeen voi kestää ympäristöstä riippuen useita minuutteja ennen kuin paikannin saa satelliiteilta tarpeeksi tietoa, jotta se voi laskea sijaintikoordinaatit. Tästä viiveestä käytetään nimitystä Time To First Fix (TTFF). Koordinaattitietojen odottaminen voi viivästyttää aivan turhaan työtehtävien suoritusta. Yksi ratkaisu on käynnistää GPS-paikannin hyvissä ajoin ennen oikeaa tarvetta ja jättää se päälle. Tällöin kulutetaan turhaan akkua paitsi paikantimessa, myös puhelimessa, joka tarvitsee virtaa Bluetooth-radiolle. Assisted GPS (AGPS) on tekniikka, joka lyhentää koordinaattitietojen saamiseen tarvittavaa aikaa jopa sekunteihin (Djuknic & Richton 2001). AGPS-palvelin tietää, minkä langattoman verkon solun alueella päätelaite sijaitsee. Sen lisäksi palvelin vastaanottaa itse jatkuvasti GPS-signaalia. Näiden tietojen sekä päätelaitteen vastaanottaman puutteellisen GPS-tiedon perusteella palvelin voi arvioida päätelaitteen sijainnin. Koska AGPS-palvelin suorittaa signaalien prosessoinnin, vähenee myös päätelaitteelle kohdistuva kuormitus.

6.5.2 Jatkokehitys

Sovelluskehityksen aikana sekä testauksessa havaittuja ongelmia ja puutteita voidaan korjata jatkokehityksessä. Suurimmaksi ongelmaksi sovelluksen käyttöliittymässä todettiin matkapuhelimen hidas tekstinsyöttö. Vikaraportteihin voi joutua liittämään pitkiä vikakuvauksia, joten raportointia kannattaa helpottaa korvaamalla tekstin kirjoittaminen jollain vaihtoehtoisella menetelmällä. Vikakuvauksen kirjoittamisen sijaan sanelemalla se puhelimelle ja lähettämällä äänitiedosto palvelimelle voidaan säästää paljon aikaa ja etenkin vaivaa. Vaikka ääni pakattaisiin, ongelmana on ääntä sisältävien SOAP-viestien suuri koko, josta johtuen sen lähetys palvelimelle on hidasta.

Jatkokehityksessä sovellukseen voidaan lisätä uusia ominaisuuksia, esimerkiksi jonkinlainen apuväline elementtien kuormaukseen sekä mahdollisuus käyttää sovellusta ilman verkkoyhteyttä. Elementtien kuormausta voidaan helpottaa, jos lista kuormaan kuuluvista elementeistä sekä niiden sijaintitiedot voidaan noutaa palvelimelta. Nykyisellään sovellus on täysin riippuvainen verkkoyhteydestä, koska mitään tietoja ei tallenneta päätelaitteelle. Vaikka GPRS-verkko on kuuluvuusalueeltaan kattava, myös siinä on katvealueita, ja rakennuksilla voidaan työskennellä esimerkiksi kellaritiloissa, joissa kuuluvuus on heikko tai olematon. Tästä johtuen työntekijän tehtävien suoritus voi vaikeutua tai estyä kokonaan. Tällaisten tilanteiden ehkäisemiseksi on tarpeen ”offline” -toiminnallisuus, jolloin viestit tallennettaisiin väliaikaisesti mobiililaitteeseen, mikäli verkkoyhteys ei ole saatavilla. Elementtien asentaja voi työpäivän alussa ladata puhelimeensa päivän aikana asennettavien elementtien tiedot, ja työtehtävien suorituskuittaukset tallennettaisiin puhelimelle myöhempää lähetystä varten. Mobiililaitteiden ja palvelimen tiedot voidaan synkronoida aina työvuoron alussa sekä lopussa. Tästä aiheutuu toisaalta tietokannan puolella ongelmia jotka täytyy ratkaista. Jos työpäivän aikana ei saada kuittauksia elementtien asennuksesta, tietojärjestelmä voi tulkita tilanteen niin että työt seisovat ja lähettää automaattisen hälytyksen työmaan esimiehelle, vaikka todellisuudessa työt ovat edenneet normaalisti.

GPS-paikantimella voidaan saada koordinaatteja vain ulkotiloissa, mutta elementtien varastot voivat olla myös sisätiloissa. Koordinaattitietojen tarve onkin suurin

ulkovarastojen kohdalla, jotka voivat olla hyvinkin laajoja piha-alueita joista oikeiden elementtien löytäminen voi olla haastavaa. Sisätilavarastojen ilmoittaminen tietokantaan ei siis ole yhtä kriittistä, mutta myös sitä voidaan helpottaa esimerkiksi sijoittamalla varastopaikkaan RFID-tunniste, jonka sarjanumero on kytketty tietokannassa kyseiseen varastoon. Lukemalla varastoinnin yhteydessä elementin tunnisteen lisäksi tämä varastopaikan tunniste, voidaan siis automaattisesti tunnistaa myös sijainti sisätiloissa. Varastopaikan tunniste tai tunnisteet tulisi sijoittaa helposti löydettäviin paikkoihin. Tässä on myös virhetilanteen mahdollisuus, jos käyttäjä lukee elementin tunnisteen, vaikka sovellus pyytää lukemaan varastopaikan tunnisteen.

Nyt toteutetussa sovelluksessa on hyvin monta erilaista toimintoa, joita käyttävät eri työntekijät erilaisissa ympäristöissä, eikä yksi työntekijä tarvitse kaikkia toimintoja. Pilkkomalla sovellus pienempiin osiin voitaisiin se kohdistaa paremmin erilaisten tehtävien vaatimuksia vastaamaan. Esimerkiksi karttakuvien katselu on mukavampaa suuremmalta näytöltä, ja kämmentietokone voisi olla soveliaampi laite siihen tarkoitukseen. Tässä on haittapuolena kämmentietokoneiden matkapuhelinta suurempi alttius rikkoutumiselle, joten sitä pidetään huonosti rakennus- ja tehdasympäristöön sopivana laitteena.

7 JOHTOPÄÄTÖKSET

Yhä useampi työntekijä tekee työtään jossain muualla kuin vakituisessa työpisteessään ja liikkuu paikasta toiseen. Nämä liikkuvat työntekijät voivat tarvita työtehtävissään tietoa joka on tallennettuna yrityksen tietojärjestelmään, mutta kenttäolosuhteissa pääsy tietojärjestelmään voi olla vaikeaa tai mahdotonta. Tässä diplomityössä kehitettiin mobiililaitteelle sovellus vastaamaan rakennusteollisuuden liikkuvien työntekijöiden useisiin eri käyttötarpeisiin. Suunnitteluvaiheessa vertailtiin erilaisia teknologioita sekä päätelaitteen että tietoliikenneyhteyden osalta, joista sopivimmat valittiin käytettäväksi sovelluksen toteutuksessa. Toteutusvaiheessa suunniteltiin tietojärjestelmään rajapinta, jota voitiin hyödyntää tietojen siirrossa mobiililaitteeseen, sekä ohjelmoitiin asiakassovellus matkapuhelimelle.

Matkapuhelin soveltuu työntekijän apuvälineeksi hyvin, kunhan sovelluksen helppoon käytettävyyteen sekä selkeään tiedonesitystapaan kiinnitetään tarpeeksi huomiota. Matkapuhelimet ja muut mobiililaitteet suunnitellaan mahdollisimman pieniksi, jotta niiden mukana kantaminen olisi helppoa, mutta pienten laitteiden käytettävyys ei aina ole hyvä. Näkyvimpiä ongelmia ovat pienet näytöt sekä hidas tekstinsyöttö. Yksityiskohtaisten kuvien esitys on pienellä näytöllä vaikeaa, joten kuvia voi olla tarpeen yksinkertaistaa mobiililaitteilla esittämistä varten. Tekstinsyöttöä ei voida nopeuttaa, mutta sen määrää voidaan vähentää.

Toteutetusta sovelluksesta pyrittiin tekemään mahdollisimman selkeä ja helppokäyttöinen, jotta se avustaisi työntekijää eikä sen sijaan hidastaisi tehtävien suoritusta. Valitsemalla matkapuhelimen ruudulla esitettävien karttakuvien mittakaava sopivaksi voitiin kompensoida pienen ja epätarkan näytön rajoitteita. Suurimmaksi ongelmaksi koettu tekstinsyöttö minimoitiin käyttämällä ensinnäkin mahdollisimman paljon valintalistoja ja toisekseen tiedonsyötön apuvälineitä. Tiedonsyöttöä automatisoitiin käyttämällä RFID-tekniikkaa betonielementtien tunnistamiseen, GPS-paikannusta sijainnin määrittämiseen sekä laseretäisyysmittaria, josta mittatiedot voitiin siirtää suoraan puhelimeen. Työntekijän apuvälineinä toimivat oheislaitteet helpottavat sovelluksen käyttöä, mutta haittapuolena on mukana kannettavien laitteiden määrän

kasvaminen. Monissa uusissa puhelimissa on sisäänrakennettu GPS-paikannin ja joissakin RFID-lukija, joten niiden hyödyntämistä kannattaa tutkia.

Osa sovelluksen ominaisuuksista jäi tämän työn puitteissa testaamatta kenttäolosuhteissa, joten testausta jatketaan. Havaittujen ongelmien ja puutteiden perusteella sovellusta kehitetään edelleen, ja myös uusien ominaisuuksien lisäämistä on suunniteltu. Tärkeimpiä sovelluksen jatkokehityksen kohteita ovat mahdollisuus vikaraporttien saneluun sekä käyttö ilman verkkoyhteyttä.

LÄHTEET

Ballard, B. 2007. Designing the Mobile User Experience. John Wiley & Sons. 260 s. ISBN 0-470-03361-4

Bhatt, H. & Glover, B. 2006. RFID Essentials. O'Reilly. 276 s. ISBN 0-596-00944-5

Bluetooth SIG 2004. Bluetooth Specification Documents. [verkkodokumentti][viitattu 22.11.2007]
Saatavissa: http://www.bluetooth.com/Bluetooth/Learn/Technology/Specifications/

Butow, E. 2007. User Interface Design for Mere Mortals. Addison-Wesley Professional. 312 s. ISBN 0-321-44773-5

Canalys 2007. Canalys research release: 64 million smart phones shipped worldwide in 2006. [verkkodokumentti] 12.2.2007 [viitattu 29.10.2007]
Saatavissa: http://www.canalys.com/pr/2007/r2007024.htm

Chu, H., You, C. & Teng, C. 2004. Challenges: Wireless Web Services. Tenth International Conference on Parallel and Distributed Systems, 2004. ICPADS 2004. Proceedings. 7.-9.7.2004. s. 657-664.

Digia Inc. 2003. Programming for the Series 60 Platform and Symbian OS. John Wiley & Sons. 521 s. ISBN 0-470-84948-7

Djuknic, G.M., Richton, R.E. 2001. Geolocation and assisted GPS. Computer, Volume 34, Issue 2. Helmikuu 2001. s. 123-125.

Dornan, A. 2001. The Essential Guide to Wireless Communications Applications. Prentice-Hall PTR. 315 s. ISBN 0-130-31716-0

Elisa Oyj 2007. Elisan kotisivut, Mobiilidatan tuotetiedot. [verkkodokumentti][viitattu 8.11.2007]
Saatavissa: http://matkaviestinta.elisa.fi/public/elisa.do?id=hen_liit_peruspalvelut,liittyman_peruspalvelu_h_0032.htm

Finkenzeller, K. 2003. RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification 2nd Edition. John Wiley & Sons. 427 s. ISBN 0-470-84402-7

Fogli, D., Marcante, A., Mussio, P., Oliveri, E., Padula, M. & Scaioli, R. 2005. Building yard on line: a distributed and mobile system for supporting building workers. 14th IEEE international Workshops on Enabling Technologies: infrastructure For Collaborative Enterprise. Proceedings. 13.-15.6.2005. s. 195-200.

Holmquist, L.E. 1999. Will Baby Faces Ever Grow up?. 8th International Conference on Human-Computer Interaction, 1999. Proceedings, Volume 1. s. 706-709.

IBM 2006. Finnair, IBM ja Nokia ratkaisivat liikkuvien työntekijöiden haasteita. IBM Uutiset. [verkkodokumentti] 12.6.2006 [viitattu 31.10.2007]
Saatavissa: http://www.ibm.com/news/fi/fi/2006/06/fi060612RFID.html

IEEE 2003. IEEE Std 802.11g-2003, IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 4: Further Higher Data Rate Extension in the 2.4 GHz Band. The Institute of Electrical and Electronics Engineers. 78 s. ISBN 0-7381-3701-4

Kaaranen, H., Ahtiainen, A., Laitinen, L., Naghian, S. & Niemi, V. 2001. UMTS Networks, Architecture, Mobility and Services. John Wiley & Sons. 302 s. ISBN 0-471-48654-X

Kallonen, T. 2006. RFID-tekniikan käyttö betonielementtien tunnistamiseen. Diplomityö. Lappeenrannan teknillinen yliopisto, tietotekniikan osasto. 64 s.

Kjedskov, J., Nielsen, C.M., Overgaard, M., Pedersen, M.B., Stage, J. & Stenild, S. 2006. Designing a Mobile Communicator: Combining Ethnography and Object-Oriented Design. Proceedings of the 20th Conference of the Computer-Human interaction Special interest Group (Chisig) of Australia on Computer-Human interaction: Design: Activities, Artefacts and Environments. 20.-24.11.2006. s. 95-102

Lahiri, S. 2006. RFID Sourcebook. IBM Press. 276 s. ISBN 0-13-185137-3

Lim, Y., Kim, S., Min, L., & Ma, J.S. 2001. Performance evaluation of the Bluetooth-based public internet access point. 15th International Conference on Information Networking, 2001. Proceedings. 31.1.-2.2.2001. s. 643-648.

Miller, B. A. & Bisdikian, C. 2001. Bluetooth Revealed. Prentice-Hall. 303 s. ISBN 0-13-090294-2

Moe, K.H., Dwolatzky, B. & Olst, R.V. 2004.Designing a usable mobile application for field data collection. 7th AFRICON Conference in Africa, Volume 2. 15.-17.9.2004. s. 1187-1192.

Mulvey, D. 2007. HSPA. Communications Engineer, Volume 5, Helmikuu/Maaliskuu 2007. s. 38-41.

Nielsen, J. 2005. Heuristics for User Interface Design - Ten Usability Heuristics. [verkkodokumentti] [viitattu 12.11.2007]
Saatavissa: http://www.useit.com/papers/heuristic/heuristic_list.html

Red Five Labs 2007. Red Five Labs kotisivut. [verkkodokumentti] [viitattu 27.11.2007]
Saatavissa: http://www.redfivelabs.com/

Rudenko, A., Reiher, P., Popek, G. & Kuenning, G. 1998. Saving portable computer battery power through remote process execution. ACM SIGMOBILE Mobile Computing and Communications Review, Volume 2, Issue 1. Tammikuu 1998. s. 19-26.

Satyanarayanan, M. 1996. Fundamental challenges in mobile computing. Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing. 23.-26.5.1996. s. 1-7.

Tian, M., Voigt, T., Naumowicz, T., Ritter, H. & Schiller, J. 2004. Performance Considerations for Mobile Web Services. Elsevier Computer Communications Journal, Volume 27, Issue 11. Heinäkuu 2004. s. 1097-1105.

Van Engelen, R. 2007. gSOAP kotisivut. [verkkodokumentti] [viitattu 29.10.2007] Saatavissa: http://www.cs.fsu.edu/~engelen/soap.html

Vuorinen, P. 2005. Applying the RFID technology for Field Force Solution. Proceedings of the 2005 Joint Conference on Smart Objects and Ambient intelligence: innovative Context-Aware Services: Usages and Technologies. 12.-14.10.2005. s. 39-41.

W3C 2004. Web Services Architecture. [verkkodokumentti] 11.2.2004 [viitattu 30.10.2007] Saatavissa: http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/

WEB SERVICE -RAJAPINNAN WSDL-KUVAUS


```
<?xml version="1.0"?>
<definitions name="MobildingServer"
targetNamespace="urn:MobildingServer"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="urn:MobildingServer"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns="http://schemas.xmlsoap.org/wsdl/">
 <types xmlns="http://schemas.xmlsoap.org/wsdl/">
  <schema xmlns="http://www.w3.org/2001/XMLSchema"
  targetNamespace="urn:MobildingServer">
   <complexType name="errorlist">
    <complexContent>
     <restriction base="SOAP-ENC:Array">
      <attribute ref="SOAP-ENC:arrayType" wsdl:arrayType="xsd:int[]"
      />
     </restriction>
    </complexContent>
   </complexType>
   <complexType name="errors">
    <all>
     <element name="error_type" type="xsd:int" />
     <element name="error_desc" type="xsd:string" />
     <element name="error_id" type="xsd:int" />
    </all>
   </complexType>
   <complexType name="errorsArray">
    <complexContent>
     <restriction base="SOAP-ENC:Array">
      <attribute ref="SOAP-ENC:arrayType"
      wsdl:arrayType="tns:errors[]" />
     </restriction>
    </complexContent>
   </complexType>
   <complexType name="storage">
    <all>
     <element name="gpslat" type="xsd:double" />
     <element name="gpslong" type="xsd:double" />
     <element name="storageid" type="xsd:string" />
     <element name="rfid" type="xsd:string" />
    </all>
   </complexType>
   <complexType name="storageArray">
    <complexContent>
     <restriction base="SOAP-ENC:Array">
      <attribute ref="SOAP-ENC:arrayType"
      wsdl:arrayType="tns:storage[]" />
     </restriction>
    </complexContent>
   </complexType>
   <complexType name="elementinfo">
    <all>
     <element name="elementid" type="xsd:string" />
```

```
        <element name="shape" type="xsd:string" />
        <element name="width" type="xsd:int" />
        <element name="height" type="xsd:int" />
        <element name="thickness" type="xsd:int" />
        <element name="weight" type="xsd:int" />
        <element name="width_measured" type="xsd:int" />
        <element name="height_measured" type="xsd:int" />
        <element name="thickness_measured" type="xsd:int" />
        <element name="weight_measured" type="xsd:int" />
        <element name="cross_measured" type="xsd:int" />
        <element name="planned_installdate" type="xsd:int" />
        <element name="currentstate" type="xsd:int" />
        <element name="reported_errors" type="tns:errorlist" />
      </all>
    </complexType>
  </schema>
</types>
  <message name="sendLocationRequest">
    <part name="rfid" type="xsd:string" />
    <part name="gpslat" type="xsd:double" />
    <part name="gpslong" type="xsd:double" />
    <part name="storageid" type="xsd:string" />
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
  </message>
  <message name="sendLocationResponse">
    <part name="message" type="xsd:string" />
    <part name="success" type="xsd:boolean" />
  </message>
  <message name="getInfoRequest">
    <part name="rfid" type="xsd:string" />
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
  </message>
  <message name="getInfoResponse">
    <part name="return" type="tns:elementinfo" />
  </message>
  <message name="sendStateRequest">
    <part name="rfid" type="xsd:string" />
    <part name="newstate" type="xsd:int" />
    <part name="state_time" type="xsd:int" />
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
  </message>
  <message name="sendStateResponse">
    <part name="message" type="xsd:string" />
    <part name="success" type="xsd:boolean" />
  </message>
  <message name="reportErrorRequest">
    <part name="rfid" type="xsd:string" />
    <part name="error_type" type="xsd:int" />
    <part name="error_desc" type="xsd:string" />
    <part name="login" type="xsd:string" />
    <part name="password" type="xsd:string" />
  </message>
  <message name="reportErrorResponse">
    <part name="success" type="xsd:boolean" />
    <part name="message" type="xsd:string" />
```

```
  <part name="error_id" type="xsd:int" />
</message>
<message name="sendErrorFileRequest">
  <part name="error_id" type="xsd:int" />
  <part name="data" type="xsd:string" />
  <part name="file_extension" type="xsd:string" />
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
</message>
<message name="sendErrorFileResponse">
  <part name="success" type="xsd:boolean" />
  <part name="error" type="xsd:string" />
</message>
<message name="updateErrorStateRequest">
  <part name="error_id" type="xsd:int" />
  <part name="error_state" type="xsd:int" />
  <part name="state_message" type="xsd:string" />
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
</message>
<message name="updateErrorStateResponse">
  <part name="success" type="xsd:boolean" />
  <part name="error" type="xsd:string" />
</message>
<message name="getStoragesRequest">
  <part name="shape" type="xsd:string" />
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
</message>
<message name="getStoragesResponse">
  <part name="success" type="xsd:boolean" />
  <part name="storage" type="tns:storageArray" />
</message>
<message name="sendMeasurementsRequest">
  <part name="rfid" type="xsd:string" />
  <part name="width" type="xsd:int" />
  <part name="height" type="xsd:int" />
  <part name="thickness" type="xsd:int" />
  <part name="cross" type="xsd:int" />
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
</message>
<message name="sendMeasurementsResponse">
  <part name="success" type="xsd:int" />
  <part name="error" type="xsd:string" />
</message>
<message name="getErrorsRequest">
  <part name="rfid" type="xsd:string" />
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
</message>
<message name="getErrorsResponse">
  <part name="success" type="xsd:boolean" />
  <part name="errors" type="tns:errorsArray" />
</message>
<message name="getMapLocationRequest">
  <part name="dim_x" type="xsd:int" />
  <part name="dim_y" type="xsd:int" />
```

```
  <part name="storagearea" type="xsd:string" />
  <part name="shape" type="xsd:string" />
  <part name="scale" type="xsd:int" />
  <part name="draw_storage_area" type="xsd:boolean" />
  <part name="login" type="xsd:string" />
  <part name="password" type="xsd:string" />
</message>
<message name="getMapLocationResponse">
  <part name="success" type="xsd:boolean" />
  <part name="message" type="xsd:string" />
  <part name="filetype" type="xsd:string" />
  <part name="file" type="xsd:string" />
  <part name="coord_ltop_x" type="xsd:double" />
  <part name="coord_ltop_y" type="xsd:double" />
  <part name="coord_rbottom_x" type="xsd:double" />
  <part name="coord_rbottom_y" type="xsd:double" />
</message>
<portType name="MobildingServerPort">
  <operation name="sendLocation">
    <input message="tns:sendLocationRequest" />
    <output message="tns:sendLocationResponse" />
  </operation>
  <operation name="getInfo">
    <input message="tns:getInfoRequest" />
    <output message="tns:getInfoResponse" />
  </operation>
  <operation name="sendState">
    <input message="tns:sendStateRequest" />
    <output message="tns:sendStateResponse" />
  </operation>
  <operation name="reportError">
    <input message="tns:reportErrorRequest" />
    <output message="tns:reportErrorResponse" />
  </operation>
  <operation name="sendErrorFile">
    <input message="tns:sendErrorFileRequest" />
    <output message="tns:sendErrorFileResponse" />
  </operation>
  <operation name="updateErrorState">
    <input message="tns:updateErrorStateRequest" />
    <output message="tns:updateErrorStateResponse" />
  </operation>
  <operation name="getStorages">
    <input message="tns:getStoragesRequest" />
    <output message="tns:getStoragesResponse" />
  </operation>
  <operation name="sendMeasurements">
    <input message="tns:sendMeasurementsRequest" />
    <output message="tns:sendMeasurementsResponse" />
  </operation>
  <operation name="getErrors">
    <input message="tns:getErrorsRequest" />
    <output message="tns:getErrorsResponse" />
  </operation>
  <operation name="getMapLocation">
    <input message="tns:getMapLocationRequest" />
    <output message="tns:getMapLocationResponse" />
  </operation>
```

```
</portType>
<binding name="MobildingServerBinding"
type="tns:MobildingServerPort">
  <soap:binding style="rpc"
  transport="http://schemas.xmlsoap.org/soap/http" />
  <operation name="sendLocation">
    <soap:operation
    soapAction="urn:MobildingServer#MobildingServer#sendLocation" />
    <input>
      <soap:body use="encoded" namespace="urn:MobildingServer"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:MobildingServer"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>
  <operation name="getInfo">
    <soap:operation
    soapAction="urn:MobildingServer#MobildingServer#getInfo" />
    <input>
      <soap:body use="encoded" namespace="urn:MobildingServer"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:MobildingServer"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>
  <operation name="sendState">
    <soap:operation
    soapAction="urn:MobildingServer#MobildingServer#sendState" />
    <input>
      <soap:body use="encoded" namespace="urn:MobildingServer"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:MobildingServer"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>
  <operation name="reportError">
    <soap:operation
    soapAction="urn:MobildingServer#MobildingServer#reportError" />
    <input>
      <soap:body use="encoded" namespace="urn:MobildingServer"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
    <output>
      <soap:body use="encoded" namespace="urn:MobildingServer"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </output>
  </operation>
  <operation name="sendErrorFile">
    <soap:operation
    soapAction="urn:MobildingServer#MobildingServer#sendErrorFile" />
    <input>
```

```
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </input>
  <output>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </output>
</operation>
<operation name="updateErrorState">
  <soap:operation
  soapAction="urn:MobildingServer#MobildingServer#updateErrorState"
  />
  <input>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </input>
  <output>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </output>
</operation>
<operation name="getStorages">
  <soap:operation
  soapAction="urn:MobildingServer#MobildingServer#getStorages" />
  <input>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </input>
  <output>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </output>
</operation>
<operation name="sendMeasurements">
  <soap:operation
  soapAction="urn:MobildingServer#MobildingServer#sendMeasurements"
  />
  <input>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </input>
  <output>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </output>
</operation>
<operation name="getErrors">
  <soap:operation
  soapAction="urn:MobildingServer#MobildingServer#getErrors" />
  <input>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </input>
  <output>
    <soap:body use="encoded" namespace="urn:MobildingServer"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  </output>
</operation>
```

```
    <operation name="getMapLocation">
    <soap:operation
    soapAction="urn:MobildingServer#MobildingServer#getMapLocation" />
      <input>
        <soap:body use="encoded" namespace="urn:MobildingServer"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
      </input>
      <output>
        <soap:body use="encoded" namespace="urn:MobildingServer"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
      </output>
    </operation>
  </binding>
<service name="MobildingServerService">
  <documentation />
    <port name="MobildingServerPort"
    binding="tns:MobildingServerBinding">
      <soap:address
      location="http://XXX.XXX.XXX.XXX/webservice/mobile_service.php" />
    </port>
  </service>
</definitions>
```