TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

Tutkintotyö

Juha Kotiranta

Konsepti kodintekniikan etäohjaukseen

Työn ohjaaja: Diplomi-insinööri Tony Torp
Työn teettäjä: Sasken Finland Oy, valvojana Diplomi-insinööri Otto Chrons
Tampere 2008

TAMPEREEN AMMATTIKORKEAKOULU

Tietotekniikka
Kotiranta, Juha Konsepti kodintekniikan etäohjaukseen
Tutkintotyö 45 sivua
Työn ohjaaja Diplomi-insinööri Tony Torp
Työn teettäjä Sasken Finland Oy, valvojana Diplomi-insinööri Otto Chrons
Toukokuu 2008
Avainsanat DLNA, UPnP, NFC, Widget, WRT, AJAX

TIIVISTELMÄ

Tämän insinöörityön tavoitteena on tutkia uusia tekniikoita, joilla voisi toteuttaa kodintekniikan etäohjauksen. Kodintekniikan ja digitaalisen tiedon määrän valtava kasvu kuluttajien kotona on aiheuttanut tarpeen saada laitteet yhdistettyä toisiinsa, ja hoitaa niiden ohjaus helposti yhdestä paikasta. Monilla näistä laitteista on oma näyttö, käyttöliittymä ja kaukosäädin. Tämä vaatii käyttäjältä paljon opiskelua ja säätimien määrä saattaa muodostua ongelmaksi. Tiedon siirtäminen laitteesta toiseen vaatii usein erillisten johtojen asentamista ja laitteiden asetusten määrittämistä. Kuluttajien kannalta vaivaton tapa saada laitteet ja tietoverkko käyttöön, on tavoite johon uudet tekniikat pyrkivät. Erityisesti laitteiden keskitetty ohjaus ja mahdollisuus tehdä se etäohjauksena.

Teknologiateollisuus on viime vuosina kehittänyt erilaisia tiedonsiirtomenetelmiä, joiden käyttöä kodintekniikan ohjaukseen on mahdollista käyttää. Käsittelen työssä automaattisesti konfiguroituvaa kotiverkkoa (DLNA), kosketusetäisyyden tiedonsiirtoa (NFC) sekä Web Run-Time -widgetiä mobiililaitteessa olevan etäkäyttöliittymän moottorina. Selvitän myös onko näillä tekniikoilla mahdollista tehdä käyttäjän kannalta loogisempaa ohjausta laitteiden kesken ilman monimutkaista asetusten määrittämistä.

Käsittelemäni tekniikat ovat varsin uusia, ja tästä johtuen tietoa ja sovelluksia on saatavilla rajoitetusti. Suurin osa käyttämästäni lähdemateriaalista on

sähköisiä dokumentteja, koska painettua kirjallisuutta ei ole juurikaan saatavilla. Työssä ei paneuduta tekniikoiden tarkkoihin yksityiskohtiin, vaan käydään läpi tärkeimmät tiedot, jotta tämän konseptin kokonaisuuden pystyy ymmärtämään.

TAMK University of Applied Sciences

Information technology
Kotiranta, Juha Concept for remote controlling home devices
Final thesis 45 pages
Supervising teacher Tony Torp (MSc)
Commissioned by Sasken Finland Oy, supervisor Otto Chrons (MSc)
May 2008
Key words DLNA, UPnP, NFC, Widget, WRT, AJAX

ABSTRACT

The goal of this thesis is to study new technologies that could provide possibility to remote control home devices over internet. Amount of devices at customers homes have increased rapidly during last years. The need for easing the task of setting up home networks and possibility to access devices remotely is needed by users. The technology industry has developed many new technologies that can be used for automatically set up network between devices like Digital Living Network Alliance. In this thesis I will introduce briefly DLNA as a future home technology and Near Field Communication for short range wireless communication method. In this concept NFC is used for transferring remote control UI to mobile device that is used for controlling. The remote control UI is developed with Nokia's Web Run-Time, which is new runtime environment for Nokia's S60 platform. Since many of these technologies are quite new, most of the source material documents are eBooks. All technologies mentioned are only covered briefly so that the base concept of this thesis is easier to understand.

SISÄLLYSLUETTELO

Käytetyt lyhenteet ja termit

S60	Series 60. Nokia Oy:n kehittämä alusta älypuhelimille.
Web Run-Time	Itsenäisten web-widgettien ajamiseen tarkoitettu ympäristö Nokian S60-laitteisiin.
WRT	Katso Web Run-Time
NFC	Near field communication. Kosketusetäisyyden tiedonsiirtotekniikka.
DLNA	Digital Living Network Alliance. Teknologiayritysten liittouma uuden standardin kehittämiseksi.
UPnP	Universal Plug and Play, verkkoyhteyden automaattiseen luomiseen tarkoitettu tekniikka.
WWW	World Wide Web
AJAX	Asyncronous Javascript and XML. Yhdistelmä Javascriptiä ja XML-kieltä asynkronisen tiedonsiirron mahdollistamiseksi WWW-selaimessa.
HTML	Hypertext Markup Language, kieli jolla tuotetaan WWW-sivuja.
HTTP	Hypertext Tranfer Protocol, protokolla jota käytetään HTML-sivujen siirtämiseen internetissä.
PHP	Hypertext Preprocessor, scriptikieli jota ajetaan usein HTTP-palvelimessa.
XML	Extensible Markup Language, kieli jolla voidaan esittää tietorakenteita.
SOAP	Simple Object Access Protocol, XML-kuvauskieltä käyttävä HTTP:n päällä kulkeva protokolla.
ITU-T-näppäimistö	Monissa mobiililaitteissa oleva numeronäppäimistö, jolla voi myös kirjoittaa tekstiä.
CSS	Cascading Style Sheet, kieli, jolla tehdään HTML-sivun tyylimäärityksiä.

1 JOHDANTO

Nykyajan hyvinvointivaltioissa asuvien ihmisten kodit ovat pullollaan huipputekniikkaa. Teknisten laitteiden tarjonnan kasvaessa ja hintojen alentuessa kuluttajat ovat valmiita hankimaan kotiinsa uusia elektronisia laitteita. Internet, matkapuhelin, digitaalinen televisio, kotiteatterijärjestelmä, kannettava tietokone ja kannettava musiikkilaite löytyvät jo monista kodeista. Kun eri valmistajien laitteita on kotona paljon ja jokaisella niistä on omanlaisensa käyttöliittymä, aiheuttaa niiden opetteleminen käyttäjälle ylimääräistä työtä ja informaatiotaakkaa. Verkkoasetusten automatisoinnin, helposti ymmärrettävien käyttöliittymien ja keskitetyn ohjauksen tarve on selkeä puute markkinoilla olevissa laitteissa.

Lähivuosien aikana yhä useammat uusista kodinlaitteista sisältävät valmiudet käyttää kiinteää tai langatonta kotiverkkoa sekä internetiä. Tämä antaa mahdollisuuden kodissa olevan laitteen ohjaamiseen ajasta tai paikasta riippumatta. Tässä insinöörityössä esitellään konsepti, jonka avulla kodintekniikan yksinkertainen etäohjaus älypuhelimella voitaisiin toteuttaa. Siinä on tarkoitus yhdistää uusia tekniikoita, jotka jo itsessään ovat helppokäyttöisiä, mutta yhdistettynä tuovat lisäarvoa toisilleen. Näiden tekniikoiden yhdistelmällä voisi älypuhelimesta saada kodin keskitetyn ja monikäyttöisen ohjaimen.

Luvussa 2 käydään läpi DLNA-tekniikkaa, jonka tavoitteena on yksinkertainen mediasisällön jakaminen kodin laitteiden kesken. Sen avulla pystyy myös kytkeytymään automaattisesti IP-verkkoon. Luvussa 3 kerrotaan NFC-tekniikasta, joka on kosketusetäisyydellä tapahtuvaa tiedonsiirtoa. NFC-tekniikan avulla etäkäyttöliittymän siirtäminen mobiililaitteeseen olisi käyttäjälle looginen ja vaivaton toimenpide. Luku 4 käy läpi Web Run-Time -

teknologiaa, joka on Nokian juuri julkaisema ohjelmointialusta. Web Run-Timea käytän ohjaussovelluksen toteuttamiseen matkapuhelimessa. Luku 5 käsittelee esimerkkiohjelman suunnittelua ja toteutusta. Esimerkkiohjelmana toteutetaan WRT-web-widget, jonka avulla voidaan esimerkiksi ajastaa TV-nauhoituksia etänä kotona olevalle digiboxille.

2 DIGITAL LIVING NETWORK ALLIANCE

Digital Living Network Alliance (lyhyemmin DLNA) on huomattavien teknologiayritysten liittouma, jonka tarkoituksena on mahdollistaa digitaalisen tiedon jakaminen eri laitteiden välillä. Tähän liittoumaan kuuluu yli 250 kulutuselektroniikan, tietotekniikan ja mobiililaitteiden valmistajaa. Vuonna 2004 julkaistiin ensimmäiset suuntaviivat joiden avulla eri valmistajat voisivat sovittaa laitteensa toimimaan yhteistyössä muiden valmistajien laitteiden kanssa. DLNA ei sinänsä ole mikään uusi tekniikka vaan laitevalmistajien standardointisopimus, joka mahdollistaa yhteensopivien laitteiden tuottamisen kuluttajien käyttöön. /7/

2.1 DLNA:n perusteet

Teknisten laitteiden käyttäjille DLNA näkyisi parempina ja helppoina palveluina sekä vastaisi haasteeseen hallinnoida koko ajan kasvavaa digitaalisen median määrää. DLNA:ta tukevat laitteet muodostaisivat automaattisesti yhteyden toisiin DLNA-laitteisiin langattoman tai langallisen verkon välityksellä.

DLNA:n on tarkoitus mahdollistaa käyttäjälle

- helppo pääsy digitaaliseen musiikkiin kotona
- vaivaton digitaalisten kuvien hallinta ja katselu
- digitaalisen sisällön kuljettaminen mukana ulkona ollessa
- jaettu monen käyttäjän ympäristö, jossa voi nauhoittaa ja näyttää digitaalista sisältöä.

Tavoitteen saavuttamiseen suunnitellussa strategiassa painotetaan avainasioina

- teollisuuden yhteistyötä
- standardeihin perustuvaa yhteensopivuutta

- hyviä tuotteita.

Teollisuuden yhteistyö

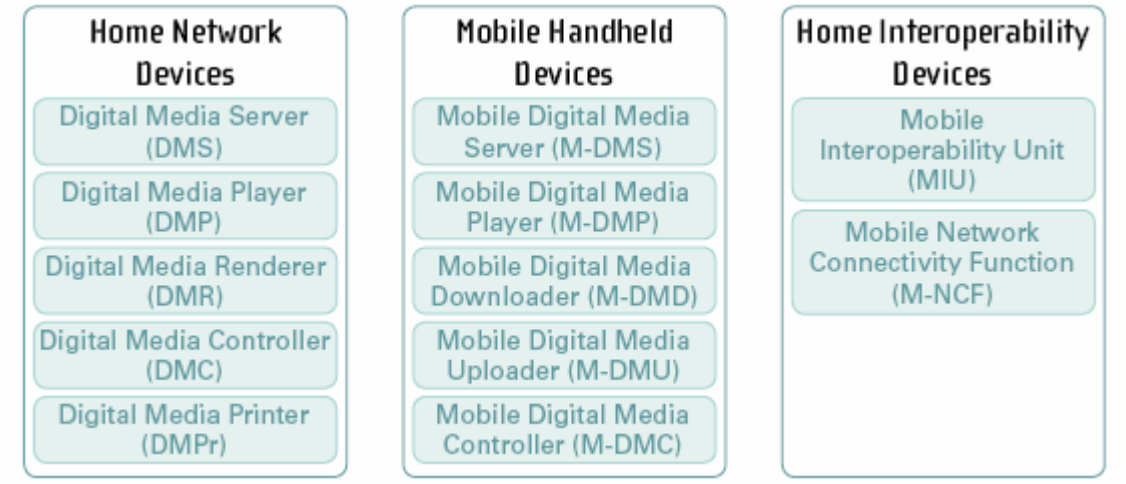
DLNA on julkaissut yleiset suunnitteluohjeet, joita DLNA:han liittyneet yritykset noudattavat tuotteissaan. PC- ja mobiililaitteiden johtavat valmistajat ovat ensimmäisinä olleet tukemassa tätä ideaa. Tavoitteena on saada mahdollisimman laaja edustus kaikilta teollisuuden aloilta. /7/

Standardeihin perustuva yhteensopivuus

Suunnitteluohjeet sisältävät ohjeet siitä mitkä tekniikat tulee DLNA-yhteensopivasta laitteesta löytyä, ja kuinka niitä tulee käyttää. Näitä tekniikoita käytetään rakennuspalikoina, jotta yhteensopivuus eri valmistajien laitteiden välillä löytyy. Suunnitteluohje kattaa fyysisen tallennuksen, tiedonsiirron, tiedon formaatin, tiedonsiirron protokollat ja myös digitaalisen tiedon suojaukseen käytetyn DRM (Digital Rights Management) -mekanismin. Kaikilla tekniikoilla on omat foorumit, joissa niistä saa ajantasaista tietoa. /7/

Hyvät tuotteet

Jotta DLNA:n käyttö yleistyy tarvitaan tietenkin tuotteita, jotka kiinnostavat kuluttajia. DLNA-tuotteet jaotellaan kolmeen ryhmään, joissa on yhteensä kaksitoista laiteluokkaa, jotka näkyvät kuvassa 3.1.



Kuva 3.1 DLNA laiteryhmät ja -luokat /7/

Laiteluokka kertoo, minkälaiseen käyttöön laite on suunniteltu. Jokaiseen laiteluokkaan liittyy käyttötapauksia, jotka laitteen tulisi täyttää, jotta se voi kuulua ryhmään. Myös muut DLNA-laitteet voivat päätellä laiteluokan perusteella, mitä palveluita on tarjolla. Esimerkiksi mobiililaite voi toimia median soittajana tai tarvittaessa kauko-ohjaimena kahden muun DLNA-laitteen välillä. /7/

Sama laite voi kuulua useampaan laiteluokkaan. Alla on esimerkkinä listattu muutamia yleisimpiä laiteluokkia ja niihin sopivia laitteita. /8/

Digital Media Server (DMS) -laitteet voivat tarjota, nauhoittaa ja tallentaa mediaa. Laitteet voivat sisältää käyttäjä- ja laitehallinnan sekä tarjota rikkaan käyttöliittymän, kuten

- digitaalinen video-nauhoitin
- pöytä- tai kannettava tietokone
- multimediamatkapuhelin
- kovalevyllinen kotiteatterijärjestelmä.

Digital media player (DMP) -laitteet mahdollistavat verkkoon tallennetun digitaalisen median toistamisen. Tällaisia laitteita ovat esimerkiksi

- TV-monitori
- kotiteatterijärjestelmä
- kämmentietokone
- multimediamatkapuhelin
- pelikonsoli.

Käyttötapaukset

DLNA on ottanut käyttäjälähtöisen lähestymistavan toiminnallisuuden kehittämiseen. Käyttötapaukset kuvaavat eri tilanteita, joissa käyttäjät haluavat käyttää tallentamaansa mediaa. Näin eri tekniikoita ei käytetä tekniikan takia vaan käyttäjän tarpeen vuoksi. Koska käyttäjän tarpeet muuttuvat hitaammin, ovat käyttötapaukset käyttökelpoisia vaikka tehtävän toteuttamiseen käytettävät tekniikat vaihtuisivat.

2.2 DLNA:n tekniikka

Tietoliikenne

Digitaalisen kodin tietoliikenne perustuu IPv4 (Internet Protocol version 4) -protokollaperheeseen. Intenet protokolla mahdollistaa liikenteen myös muihin verkkoihin sekä sitä on on yleisesti käytetty päätelaitteissa jo pitkään. Internet protokolla on edullisin tekniikka toteuttaa kotiverkossa ja antaa lisäksi seuraavat edut:

- Tietoliikennepaketit voivat kulkea monien laitteiden läpi yhteyttä otettaessa. Esimerkiksi yhteys voidaan avata tietokoneelta langallisella yhteydellä ja tietoliikennepaketit kulkevat langattoman lähiverkon kautta toiseen laitteeseen, kuten televisioon.
- Pääsy Internetiin kaikilta kodin laitteelta, jotka tukevat Internet protokollaa.

Laitteiden ja palveluiden havainnointi ja hallinta

DLNA-laitteet havaitsevat sopivan verkon ja laitteet saapuessaan verkon piiriin. Ne tekevät verkkoasetukset automaattisesti (kuten IP-osoite) ja myös tarkkailevat mitä muita laitteita ja palveluita verkossa on tarjolla. Tähän automatisointiin käytetään UPnP:n (Universal Plug and Play) DCP-kehystä (Device Control Protocol), joka on DLNA:n toiminnan runko. UPnP on pakollinen vaatimus laitteelle, joka aikoo tukea DLNA:ta. Seuraavassa kohdassa käydään tätä tekniikkaa tarkemmin läpi. /7/

2.3 Universal Plug and Play DLNA:n runkona

Tullessaan markkinoille Plug and Play helpotti aikoinaan PC:n käyttäjien elämää tehden laitekohtaiset asetukset automaattisesti. Sitä ennen oli käyttäjän itse esiteltävä lisätty laite käyttöjärjestelmälle ja tehtävä kaikki monimutkaisetkin asetukset. Nyt Universal Plug and Play pyrkii laajentamaan tämän automaattisen toiminnan tietokoneverkkoihin. UPnP on suunniteltu näkymättömäksi käyttäjälle, jolloin ei tarvitse tehdä mitään asetuksia verkkoon liittymiseksi. Liittyessään verkkoon UPnP-laite löytää verkosta muut UPnP-laitteet ja saa tietoonsa niiden palvelut, sekä jakaa tarvittaessa tietoa omista palveluistaan. /10/

UPnP:n standardia ylläpitää UPnP Forum, joka on DLNA:n tavoin teollisuusyritysten yhteistyön tulos. UPnP Forum pyrkii kehittämään standardeja, jotka kuvaavat laiteprotokollia ja ylläpitää myös XML-pohjaisia laitekuvausviestejä. UPnP Forum on myös käynnistänyt eri osa-alueille omia spesifisiä ryhmiä joiden vastuulla on tehdä jatkokehitystä ja testialustojen suunnittelu sekä esimerkkisovellusten tekeminen.

UPnP on avoin arkkitehtuuri ja perustuu olemassa oleviin TCP/IP ja Internet-protokolliin mahdollistaen vaivattoman liittymisen olemassa oleviin verkoihin. UPnP ei myöskään ole laite-, alusta- tai

käyttöjärjestelmäriippuvainen, vaan se toimii kaikissa laitteissa, joihin se halutaan lisätä. UPnP ei myöskään toimita valmista ohjelmointirajapintaa vaan jokaisen laitteen ja käyttöjärjestelmän toimittajan on suunniteltava omiin tarpeisiin sopiva API. /10/

UPnP-verkon laiteroolit

Kun laite kytkeytyy UPnP-verkkoon, alkaa se toimimaan annettujen asetusten mukaisesti. UPnP-verkossa voi laite toimia asiakkaana (control point), ohjattuna laitteena tai palvelimena (device) tai tarjota molemmat palvelut.

UPnP-laitteen IP-osoite

Kun UPnP-laite ensimmäisen kerran kytkeytyy verkkoon, etsii se DHCP-palvelinta IP-osoitteen saamiseksi. Jos DHCP-palvelin löytyy, on laitteen käytettävä sille määrättyä osoitetta. Tilanteessa, jossa verkko ei sisällä DHCP-palvelinta, täytyy laitteen määritellä itselleen IP-osoite käyttämällä Auto IP -toimintoa. Auto IP määrittelee laitteelle IP-osoitteen 169.254/16-alueella. Suosituksen mukaan laitteen olisi hyvä valita satunnainen osoite väliltä 169.254.1.0–169.254.254.255, jotta mahdollisuus osoitekonfliktiin vähenee. Laitteen täytyy myös testata onko valittu osoite vapaana. Jos osoite on jo käytössä, tulee laitteen määritellä uusi osoite. Tämä toimenpide toistetaan, kunnes laite saa itselleen toimivan osoitteen. Kun laitteen osoite on määritelty Auto IP:llä, tulee sen tietyin väliajoin etsiä DHCP-palvelinta. Näin siirtyminen ylläpitämättömistä verkoista ylläpidettyihin onnistuu vaivattomasti. /11/

UPnP-laitteen liittyminen verkkoon, vaihe 1: Discovery

Kun laitteen IP-osoite on saatu, alkaa seuraava vaihe – havainnointi (discovery). Laite lähettää multicast-osoitteeseen (239.255.255.250:1900) havainnointiviestejä, jotka sisältävät tietoja laitteen omista palveluista ja sulautetuista laitteista. Näin laite mainostaa itseään verkossa ja muut verkon laitteet ovat tietoisia uudesta palvelusta. Havainnoitiprotokollan tiedot kertovat laitteen perustiedot, kuten osoiteen, tunnisteen ja osoittimen

tarkempiin tietoihin. Jos jokin laitteista haluaa tarkempia tietoja laitteen palveluista, tulee sen pyytää niitä erikseen. /11/

UPnP-laitteen liittyminen verkkoon, vaihe 2: Description

Kun tieto uudesta laitteesta verkossa leviää, voivat siitä kiinnostuneet laitteet pyytää lisää informaatiota kuvauspyynnöllä (description). Kuvauspyyntö sisältää kaksi osaa, joista ensimmäinen kuvaa laitetta (device descriptor) ja toinen kuvaa sen palveluita (service descriptor). Laitekuvaus sisältää toimittajakohtaisia tietoja kuten mallin nimen, sarjanumeron ja valmistajan WWW-osoitteen jne. Palvelukuvaus kertoo palvelusta sen nimen, tyypin, osoitteen palvelukuvaukseen, osoitteen palvelun hallinnointiin ja osoitteen jonne voidaan lähettää tapahtumaviesti (event). Kuvausten tekeminen on laitteen valmistajan vastuulla ja UPnP Forum on määritellyt valmiit pohjat kuvausten tekemiselle. /11/

UPnP-laitteen liittyminen verkkoon, vaihe 3: Control

Kuvauspyynnön saatuaan voi laite tehdä hakuja toisen laitteen palveluihin eli alkaa kontrolloimaan sitä (control). Kuvauspyynnön perusteella saadaan selville toisen verkossa olevan laitteen palvelun osoite, johon lähetetään ohjauspaketti. Vastauksena saadaan takaisin palveluun liittyviä tuloksia tai virheraportti. Vastauksessa voi tulla myös tietoja palvelun senhetkisestä tilasta. /11/

UPnP-laitteen liittyminen verkkoon, vaihe 4: Eventing

Kun laite on käynyt kaikki aikaisemmat verkkoon liittymiseen tarvittavat vaiheet läpi, voi se tilata (subscribe) itselleen tietoja yhden tai useamman laitteen tapahtumista. Käytännössä laitteet ilmoittavat oman tilansa muutoksista niille laitteille, jotka ovat tilanneet sen itselleen. Näin laitteet pystyvät päivittämään verkossa tarjottavia palvelutietoja kätevästi ilman toistuvaa laitetietojen pyytämistä. Tapahtumankäsittely ei ole laitteille pakollista, mutta se on monissa tapauksissa käytännöllistä. /11/

UPnP-laitteen liittyminen verkkoon, vaihe 5: Presentation

Kun kaksi ensimmäistä vaihetta eli Discovery ja Description on suoritettu, voi laite jo ottaa vastaan käyttöliittymäesityksen, jonka kautta käyttäjä voi tutkia laitteen tilaa ja
antaa ohjauskäskyjä. Käyttöliittymän tulee olla UPnP-määritysten mukaisesti toteutettu HTML-sivuna. Jotta esittelysivu tulee näkyviin, on laitteen lähetettävä HTTP GET -käsky kuvauskäskyn yhteydessä saatuun presentation-osoitteeseen. Vastauksena tulee HTML-sivu joka näytetään laitteen selaimessa.

UPnP-viestien rakenne

Kaikki viestit joilla UPnP-laitteet kommunikoivat keskenään käyttävät SOAP (Simple Object Access Protocol) -protokollaa. Viestit kirjoitetaan XML-kuvauskielellä ja siirretään laitteiden välillä HTTP-protokollaa käyttäen. SOAP on standardiprotokolla jolla on tarkasti määritelty rakenne, mutta se on riippumaton käytettävistä ohjelmointikielistä tai alustoista. Se sopiikin hyvin UPnP:n tiedonsiirtoprotokollaksi.

SOAP-viestiä kutsutaan kirjekuoreksi (envelope). Alla on esimerkki UPnP-ohjauspyyntöpaketista, joka on tehty seuraten SOAP:n määrittelyjä:

```
POST path of control URL HTTP/1.1
HOST: host of control URL:port of control URL
CONTENT-LENGTH: bytes in body
CONTENT-TYPE: text/xml; charset="utf-8"
SOAPACTION: "urn:schemas-upnp-org:service:serviceType:v#actionName"
<?xml version="1.0"?>
<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <s:Body>
    <u:actionName xmlns:u="urn:schemas-upnp-org:service:serviceType:v">
      <argumentName>in arg value</argumentName>
    </u:actionName>
  </s:Body>
</s:Envelope>
```

Aluksi määritellään HTTP:n otsikkotiedot. Ensimmäinen on kutsuttavan käskyn polku, joka annetaan POST-komennolle. HOST on laitteen IP-osoite. Sen jälkeen tulee viestin pituus merkkeinä ja viestin MIME-tyyppi, jonka tulee olla ”text/xml”. SOAPACTION kertoo kutsuttavan SOAP-palvelun nimen, sen jälkeen viestin runkona tulee varsinainen SOAP-kirjekuori. Punaisella merkityt osat kirjekuoressa ovat tiettyyn pyyntöön liittyviä merkintöjä, joiden nimet vaihtuvat kutsuttavan käskyn mukaan. /11/

2.4 DLNA:n yhteistyökumppanit

- UPnP Implementers Corporation
- UPnP Forum
- Wi-Fi Alliance
- Consumers Electronics Association (CEA)
- Digital Video Broadcasting Project (DVB)
- Home Gateway Initiative Project (HGI)
- Alliance for Telecommunications Industry Solutions (ATIS)

3 NEAR FIELD COMMUNICATION

Near Field Communication (NFC) on uusi protokolla ja rajapinta, joka mahdollistaa lyhyellä kantamalla tapahtuvan tiedonsiirron. NFC:tä voisi myös kuvata kosketukseen perustuvaksi tiedonsiirroksi, koska sen toimintaetäisyys on vain muutamia senttimetrejä. Tämän tiedonsiirron käynnistäminen on helppoa ja intuitiivista. Kun käyttäjä haluaa käynnistää tiedonsiirron toisen laitteen kanssa, hänen tarvitsee vain koskettaa sitä, jolloin tiedonsiirto käynnistyy. NFC on standardoitu teknologia ja perustuu RFID:n teknologiaan. NFC:n markkinointi ja suunnittelu on kuitenkin suunnattu enemmän kuluttajille kuin RFID:n. /9/

3.1 NFC-protokolla

Protokolla (NFCIP-1) perustuu langattomaan rajapintaan. Siihen tarvitaan aina vähintään kaksi osapuolta, jotka pyrkivät kommunikoimaan keskenään. Tästä syystä protokollaa kutsutaan myös vertaisprotokollaksi (peer-to-peer). /9/

Rajapinta toimii vapaalla 13.56MHz radiotaajuudella. Tämä merkitsee sitä, että tällä taajuudella voi NFC-laitetta käyttää rajoituksitta ilman lisenssejä. Ainoastaan maakohtaiset säädökset rajoittavat RF-laitteiden lähetystehoa, eli käytönnössä ne vaikuttavat kantomatkaan. Yleisesti ottaen NFC:n teoreettinen kantomatka on 0:n ja 20cm:n välillä, mutta määrittellyssä se on rajattu 4cm:n. /9/

NFC-laite voi toimia RFID:stä poiketen sekä lukijana että tunnisteena. Tämä mahdollistaa myös kaksisuuntaisen kommunikoinnin half-duplex-tilassa. Koska vastaanottoon ja lähetykseen käytetään samaa radiotaajuutta, tulee laitteen ensin tarkistaa onko kaista vapaana. NFC:tä voi käyttää joko passiivi- tai aktiivitilassa. Aktiivitilassa molemmat laitteet muodostavat oman

radiokentän tiedon välitystä varten. Passiivitilassa on lukijan muodostettava radioaaltokenttä ja tunnistelaite ottaa energiansa tästä radioaaltokentästä. NFC:n kautta voi tietoa siirtää 106, 212 tai 424 kbit/s. Jokaisella nopeusalueella käytetään eri modulaatiota. Tämän takia on laitteiden sovittava, mitä siirtonopeutta käytetään, jotta modulaatio ja bittikoodaus menevät oikein. Näitä asetuksia ei voi muuttaa kesken tiedonsiirron. /9/

3.2 NFC-tiedonsiirtoformaatti

NFC-Forum on kehittänyt NDEF-formaatin (NFC Forum Data Exchange Format) binäärisen tiedon siirtoa varten. Tämä formaatti ei ota kantaa fyysiseen laitteeseen tai tiedonsiirtoprotokollaan, vaan toimii tiedon kapseloijana. Tämä formaatti on tehty mahdollisimman kevyeksi, ja yksi NDEF-tietue sisältää kolme peruselementtiä kuvaamaan sen kuormaa (payload): /13/

- **Kuorman pituus** kertoo kuinka monta tavua on tietueessa on kuormaa. Maksimikuorma voi olla 2^{32}-1 tavua.
- **Kuorman tyyppi** kertoo minkä tyyppistä tietoa tietue sisältää. NDEF-tukee URI (Uniform Resource Identifier)-, MIME-tyyppiä (Multipart Internet Mail Extension) sekä NFC:n omia tietotyyppejä.
- **Kuorman tunniste** on URI-tyyppinen tunniste, jonka avulla voidaan linkittää kuormia.

NDEF-tietue voi sisältää minkä tyyppistä tietoa tahansa. Kehittäjien tulee kuitenkin huomioida se, että NDEF ei itsessään sisällä minkäänlaista virheen käsittelyä. Sisällön eheyden tarkistus jää sovelluskehittäjän tehtäväksi. /13/

NDEF-viestin luominen ja siirto

Ohjelman NDEF-viesti voi sisältää yhden tai useampia tiedostoja, jotka ovat standardeja tyypiltään. NDEF-generaattori kapseloi jokaisen tiedoston NDEF-tietueiden sisälle kuormaksi. NDEF-tietueet linkitetään sen jälkeen yhdeksi NDEF-viestiksi. Tässä vaiheessa viesti voidaan siirtää NFC-linkin yli toiseen

laitteeseen tai se voidaan kirjoittaa NFC Forum tagiin. Laite joka tuodaan lähelle tagia, lukee sen sisällön eli NDEF-viestin NDEF-parsijalle, joka käsittelee viestin sisällön ja lähettää sen edelleen sopivalle ohjelmistolle. /13/

Eräs tärkeä NFC:n sovelluskohde on älykkäät julisteet (smart poster). Kun niitä koskettaa NFC-lukijalaitteella esim. matkapuhelimella, luetaan NFC-tagi, josta voidaan saada lisäinformaatiota tai tehdä jokin toiminto laitteessa, esimerkiksi avata jokin www-osoite selaimessa. NFC Forum on kehittänyt valmiin NDEF-tietueen tällaisia julisteita varten. Taulukossa 3.1 näkyy tämän viestin rakenne. /14/

Offset	Content	Length	Explanation
0	0xD1	1	NDEF header. TNF = 0x01 (Well Known Type). SR=1, MB=1, ME=1
1	0x02	1	Record name length (2 bytes)
2	0x12	1	Length of the Smart Poster data (18 bytes)
3	“Sp”	2	The record name
5	0xD1	1	NDEF header. TNF = 0x01, SR=1, MB=1, ME=1
6	0x01	1	Record name length (1 byte)
7	0x0E	1	The length of the URI payload (14 bytes)
8	“U”	1	Record type: “U”
9	0x01	1	Abbreviation: “http://www.”
10	“nfc-forum.org”	13	The URI itself.

Taulukko 3.1 Yksinkertainen URI NFC-tagin tietokentässä /14/

NDEF-tietueen kautta voidaan käynnistää laitteessa jokin palvelu. Esimerkiksi matkapuhelimessa tämä palvelu voi olla tekstiviestin lähettäminen, puhelun aloittaminen tai internetselaimen aukaiseminen johonkin osoitteeseen jne. Tällainen toimenpide voidaan helposti lisätä NDEF-viestiin yhdellä NDEF-tietueella, kuten taulukossa 3.2 esitetään.

Offset	Content	Length	Explanation
26	0x11	1	NDEF record header (SR=1, TNF=0x01)
27	0x03	1	The length of the record name
28	0x01	1	The length of the “act” payload.
29	“act”	3	Record type: “act”
32	0x00	1	Action = Launch browser

Taulukko 3.2 Internetselaimen käynnistäminen laitteessa /14/

3.3 NFC:n ero muista langattomista verkoista

NFC eroaa muista langattomista tiedonsiirtoprotokollista selkeästi siinä, että sen kantama on todella lyhyt. Se ei siis kilpaile muiden langattomien verkkojen kanssa vaan toimii aivan omalla segmentillään. NFC:n avulla saadaan myös hyvä tietoturva, koska sen kautta kommunikoivat laitteet voivat olla kohtuullisen varmoja siitä, että tiedonsiirto on turvattu. Verkon kantama on niin lyhyt, että sen sisään ei voi huomaamatta tuoda muita laitteita. /9/

NFC-tiedonsiirto on käyttäjälle helppo ymmärtää, koska laitteet vain tuodaan riittävän lähelle toisiaan. Koko tiedonsiirtoprosessi käynnistyy ja etenee käyttäjän näkymättömissä. Käyttäjästä tilanne vaikuttaa siltä, että laitteet tunnistavat toisensa kosketuksesta. /9/

Eräs tärkeä ominaisuus on NFC:n kyky käyttää kommunikoinnissa passiivitilaa. Tämä on etu käytettäessä akulla toimivia laitteita. Passivitilassa vain toisen kommunikoivista laitteista täytyy muodostaa radiokenttä ja käyttää virtaa siihen. Toinen laitteista voi käyttää tätä kenttää ja indusoida siitä tarvittavan energian. Akkukäyttöisen laitteen ei tarvitse käyttää omasta energiavarastostaan silloin mitään NFC:llä kommunikoimiseen. /9/

NFC:tä voidaan käyttää yhteistyössä muiden protokollien kanssa. Sen avulla voidaan automatisoida yhteydenmuodostus laitteiden välille, kuten

esimerkiksi Bluetooth-yhteyden muodostus. Näitä protokollia voidaan sitten käyttää tiedonsiirtoon pidemmällä välimatkalla. Tällä tavalla käytettynä ei muilla pidemmän kantaman protokollilla tarvitse erikseen etsiä laitetta, jonka kanssa tiedonsiirto halutaan käynnistää.

4 WEB-WIDGET

Web-widgetit ovat pieniä itsenäisiä sovelluksia, jotka on alunperin suunniteltu tietokoneiden käyttöjärjestelmien työpöydille. Ensimmäiset työpöytä-widgetit esiteltiin Applen Mac OS X:n yhteydessä. Nykyään widgettejä saa lähes kaikkien käyttöjärjestelmien työpöydille. Käyttöjärjestelmissä on omia ympäristöjä widgetien ajamiseen, mutta on myös mahdollista asentaa kolmannen osapuolen toimittamia ympäristöjä, esimerkiksi Yahoo Widget (kuva 4.1).

Kuva 4.1 Yahoo työpöytä-widget

Widgetit noutavat yleensä verkosta jonkin tietyn yksittäisen palvelun esimerkiksi säätietoja tai uutisia ja näyttävät sen käyttäjälle. Ne vievät vähän tilaa työpöydältä ja ovat usein graafisesti näyttäviä ja toiminnoiltaan suoraviivaisia. Widgetien idea onkin olla yksinkertaisia tietolähteitä eikä täyden palvelun sovelluksia. Widgetit ajetaan asiakkaan koneella erillään internetselaimesta, mutta ne ovat toteutettavissa Web 2.0 tekniikoilla, joilla useimmat uudet internetsivut toteutetaan. Näille tekniikoille on paljon osaajia, joten on myös paljon erilaisia widgettejä.

4.1 Mobiili-web-widget

Viime vuosien aikana on mobiililaitteiden internetyhteydet yleistyneet ja nopeutuneet. Useimmat mobiililaitteiden valmistajat haluavatkin, että ihmiset siirtäisivät työpöydiltä saamansa internetin käyttöön liittyvät tottumukset mobiiliympäristöön. Haasteena on kuitenkin saada laitteiden käyttö ja tiedon etsiminen yhtä helpoksi kuin pöytäkoneilla. Jos tämä onnistuu, ihmiset varmasti käyttäisivät mobiililaitteita samassa määrin tiedon hakuun internetistä kuin pöytäkoneita. Widgetit ovat eräs osa-alue, jolla tiedonhaku on saatu helpoksi ja nopeaksi tietokoneen työpöydällä. Widgetin saamiseksi matkapuhelimiin on Nokia Oy kehittänyt uuden ajoympäristön Web Run-Timen, jonka toivotaan muuttavan ihmisten käyttökokemuksia pienten mobiililaitteiden osalta parempaan suuntaan, ja nopeuttavan sovelluskehitystä. /1/

Mobiililaitteella internetin selailun ongelmana on pienet näytöt, ja ihmisten tottumukset käyttöliittymän suhteen. Myös syötteen antaminen laitteelle ITU-T-näppäimistöllä on hankalaa, joten URL:n syöttäminen ei ole niin joustavaa kuin PC:llä. Widget taas avautuu yhdellä napin painalluksella, ja tarjoaa palvelun internetistä sopivalla käyttöliittymällä. Koska widgetin kehittäminen ei vaadi niin suuria työresursseja kuin ohjelman tekeminen muilla kielillä, on pienten ohjelmien tekeminen kannattavaa. Widgetit tulevat luultavasti olemaan mobiililaitteille huomattavasti arvokkaampia kuin pöytäkoneille. /3/

4.2 Nokia S60 Web Run-Time

S60 Browser

Vuonna 2006 esiteltiin S60 3rd Edition -ympäristöön uusi verkkosivujen selain Webkit. Se teki mahdolliseksi lukea sivustoja, jotka oli alunperin suunniteltu työpöytäselaimille. Uusi S60-selain perustui avoimen lähdekoodin projekteihin WebCore ja JavaScriptCore, joita käytetään Applen Safari-

selaimessa. Suurin hyöty uudessa selaimessa on todella hyvä tuki eri standardeille ja tekniikoille, joita käytetään kehitettäessä verkkosivustoja. Tärkeimpinä mainittakoon W3C:n standardit HTML 4.01, XHTML 1.0, CSS 1–2, DOM 1–2. Selain pystyy näyttämään hyvin myös graafisen sisällön tukiessaan käytetyimpiä grafiikkaformaatteja. /1/

Web Run-Time

Vuoden 2007 syksyllä julkaistiin S60 FP2, jossa oli lisätty Web Run-Time -ympäristö osaksi S60-järjestelmää. Tämä ympäristö on tarkoitettu mobiiliwidgetin ajamiseen kaikissa S60 3.2 -laitteissa. Tällä hetkellä (2008) Nokian N95 on ainoa markkinoilla oleva puhelinmalli, johon saa WRT:n päivityksenä. Tämä nykyinen julkaisu on vielä beta-asteella, joten lopullisen julkaisu voi tukea myös muita S60-versioita.

Sovelluskehittäjä ei tarvitse syvällistä tietämystä S60-ympäristöstä, vaan WRT on erillinen ajoympäristö. WRT antaa silti sovelluskehittäjälle rajoitetun pääsyn laitteen resursseihin. Widget:n voidaan saada tieto vapaasta muistista, näyttää levytilan käyttö, käsitellä laitteen valoja sekä värinä-toimintoa. Nokian suunnitelmissa on myöhemmin lisätä pääsy esimerkiksi GPS- ja PIM (Personal Information Managment) -tietoihin, kameraan ja bluetoothiin.

Ohjelmointi

WRT widgetit tehdään samoilla standardoiduilla työkaluilla kuin useimmat verkkosivustot. Niitä ei kuitenkaan ajeta verkon yli, vaan widgetit asennetaan käyttöjärjestelmään, ja ne voidaan käynnistää samalla tavoin kuin natiivit S60-sovellukset omasta kuvakkeesta. Web Run-Time pohjautuu S60-selaimeen mahdollistaen näin kaikkien sen tukemien tekniikoiden käytön. S60-selaimen runsas ja standardien mukainen toteutus antaa hyvän pohjan lähteä toteuttamaan dynaamisia ja näyttäviä widgetiä. /1/

Web Run-Time -widgetit ovat yleensä AJAX-sovelluksia. AJAX mahdollistaa tiedon siirtämisen verkon yli widgetille asynkronisesti. Tämä tarkoittaa, että tiedon haku aloitetaan taustalla ja sivu päivitetään Javascriptin avulla vasta tiedon ollessa käytettävissä. Muita tapoja tiedon dynaamiseen lataamiseen WRT-widgetin kanssa ei ole. Widget käyttää oletuksena yhtä ja samaa HTML-tiedostoa sisällön näyttämiseen, joten Javascript ohjelmointia tarvitaan, jos haluaa käyttää eri näkymiä ohjelmassa. Web Run-Time -widgetin sisältö voidaan tehdä myös pelkästään käyttämällä useita staattisia HTML-sivuja ja CSS:ää, mutta WRT-ympäristö lataa jokaiselle uudelle HTML-sivulle oman selainkomponentin, joten laitteen muistia kuluu runsaasti. /2/

Koska WRT-widget ajetaan laitteessa lokaalisti, ei kehittäjän tarvitse huolehtia sivun raskaudesta siinä määrin kuin tavallisten internetsivujen kanssa. Kuvia ja grafiikkaa voi käyttää widgetin ulkoasua suunnitellessa, sillä ne siirretään laitteeseen vain kerran. Siirron jälkeen kaikki resurssit voidaan lukea laitteen levyltä. Tämä nopeuttaa widgetin latausaikaa ja käyttöä. Ainoastaan muuttuva data kannattaa hakea internetyhteyden yli.

WRT:n widgetit ovat koko ruudun sovelluksia poiketen näin työpöydille tehdyistä widgeteistä. Ne muistuttavat myös siinä mielessä natiivia S60-sovellusta. WRT-widgetiin voidaan tehdä myös S60:stä tuttu Options Menu ja käyttää Rigth Soft Keytä. Käyttöliittymässä tulee myös huomioida näytön orientaation ja koon muutokset, koska monissa päätelaitteissa näitä voidaan vaihtaa. Javasript tarjoaa näytön koon muuttuessa tapahtuman (onchange) jossa voidaan käyttöliittymä ladata tarvittaessa uudestaan.

WRT-widgetin asentaminen laitteeseen

Jotta widget saadaan asennetuksi laitteeseen, tulee siitä luoda asennuspaketti. Paketti on zip-tiedosto, jonka tarkennin vaihdetaan .wgz nimiseksi. Paketissa tulee olla myös Applen-widgeteistä tuttu Info.plist-tiedosto. Tämä XML-

kuvauskielellä kirjoitettu tiedosto sisältää informaation, jota Web Run-Time käyttää ajaessaan widgetiä. Info.plist:n tietoja kaytetään myös kertomaan käyttöjärjestelmälle esimerkiksi ohjelman ikoni-tiedoston nimi jne. Muu paketin sisältö ja rakenne on sovelluskehittäjän päätettävissä. Asennuspaketin voi asentaa laitteeseen monilla tavoin. Yleisin tulee todennäköisesti olemaan asennus S60-selaimen kautta. Tällöin www-palvelimen, jolla widget sijaitsee tulee tukea ”x-nokia-widget” MIME-tyyppiä. Myöskin bluetooth- ja infrapunasiirto onnistuu ja Messaging-ohjelma käynnistää widgetin asennuksen, kun viestiä avataan.

5 ESIMERKKIOHJELMA: S60 WEB-WIDGET DLNA-DIGIBOXIN OHJAAMISEEN

Työnantaja ehdotti esimerkkiohjelmaksi digiboxin hallinnoimiseen tarkoitetun web-widgetin tekemistä. Ehdotus oli mielenkiintoinen ja erilainen lähestymistapa widgetin käyttöön. Web-widget toimisi digiboxin etäkäyttöliittymänä. Tarkoitus oli mahdollistaa widgetin kautta TV-ohjelmien ajastettu tallennus, tiedostonhallinta ja joidenkin digiboxin tietojen sekä asetusten hallinta.

Konseptiin liittyi tärkeänä osana NFC:n avulla tapahtuva käyttöliittymän siirto kohdelaitteeseen, joka on älypuhelin. Tämä olisi käyttäjälle intuitiivinen tapa valita, minkä laitteen käyttöliittymän haluaa puhelimeen siirtää. Se tapahtuisi siis koskettamalla kohdelaitetta omalla älypuhelimella.

DLNA-tuki digiboxissa ja älypuhelimessa helpottaisi verkkoasetusten määrittämistä laitteissa. Se toisi mukanaan myös muita etuja. DLNA-standardin mukaan laitteen tulee tarjota rajapinta SOAP-käskyille. Jos tarvittavat toiminnot voisi tehdä tätä rajapintaa käyttäen, olisi tiedonsiirto web-widgetin ja digiboxin välillä standardeja tukeva. Myös widgetin siirto kohdelaitteiden välillä voitaisiin tehdä IP-verkkoa käyttäen. DLNA ei ole kuitenkaan pakollinen vaatimus tämän konseptin toimimiseksi.

5.1 Havaitut toteutuksen ongelmat

Heti alusta pitäen havaittiin, että konseptia jossa NFC, widget ja digiboxi toimisivat yhteen saumattomasti ei vielä ollut mahdollista tehdä käytännössä. Ensimmäinen ongelma oli digiboxi. Markkinoilla ei ollut sellaista digiboxia joka sisältäisi HTTP-palvelimen, DLNA:n, NFC:n ja API:n jolla digiboxin toimintoja voitaisiin ohjata. Toinen ongelma oli sellaisen puhelimen

puuttuminen markkinoilta, joka tukisi samalla NFC:tä ja Web Run-Timea. Päädyin ratkaisuun, jossa widget tehdään tämän hetken kohdelaitteeseen (N95) ja digiboxia edustaa Apache-palvelimelle tehty sovellus.

5.2 Käyttötapaukset

Tässä luvussa kuvataan ohjelman käyttötapaukset. Osa käyttötapauksista on sellaisia, joita ei vielä pysty toteuttamaan, mutta kuuluvat olennaisena osana suunnitelmaan. Käyttötapauksista käsitellään vain tärkeimmät, eli ne jotka liittyvät läheisesti työn konseptin ymmärtämiseen.

5.2.1 Widgetin siirto puhelimeen

Jossakin kohtaa digiboxia on hotspot-alue. Kun tätä aluetta kosketetaan NFC:tä tukevalla puhelimella, lähettää siinä oleva NFC-tagi viestin, joka aloittaa käyttöliittymäpaketin siirron. Käytännössä tämä siirto voi tapahtua monella tavalla. NFC:tä voi käyttää itsessään koko käyttöliittymäpaketin siirtoon. Se voi myös palauttaa URL:n jossa widget-paketti on, ja lisäksi on mahdollista siirtää paketti käynnistämällä Bluetooth-yhteys. Tässä tapauksessa kätevin tapa on todennäköisesti siirtää paketti bluetoothilla.

5.3 Digiboxia edustavan palvelinsovelluksen toteutus

Työssä ei ollut tarkoitus toteuttaa aidon digiboxin ohjausta heti, vaan päätin tehdä LAMP (Linux, Apache, MysSQL, PHP) –ympäristöön sovelluksen, joka toimii digiboxin korvaajana. Tiedonsiirtoon päätin käyttää HTTP-protokollan päällä toimivaa Web services arkkitehtuuria tarjoten julkisen rajapinnan ja saaden sen myös standardin mukaiseksi. Web services koostuu pääosin XML-dokumenteista ja XML-sanomista. Web servicen rajapinta kuvataan WSDL (Web Services Description Language)-dokumentissa ja tiedonsiirtoon

käytetään SOAP-kuorta (envelope). Molemmat sekä WSDL ja SOAP käyttävät kuvauskielenä XML:ää.

Digiboxien valmistajan taholta voisi Web services -tekniikan käyttö olla viisas ratkaisu, koska se mahdollistaisi kommunikoinnin erilaisista järjestelmistä riippumatta alustoista tai sovelluslogiikasta. Jos digipoksi on DLNA-sertifioitu pitäisi siinä löytyä SOAP-rajapinta. Se minkälainen se on riippuu tietenkin valmistajasta. Oikean digiboxin kanssa voi kuitenkin joutua käyttämään HTTP:n GET ja POST kutsuja perinteisellä tavalla tiedonsiirtoon.

5.3.1 Web service

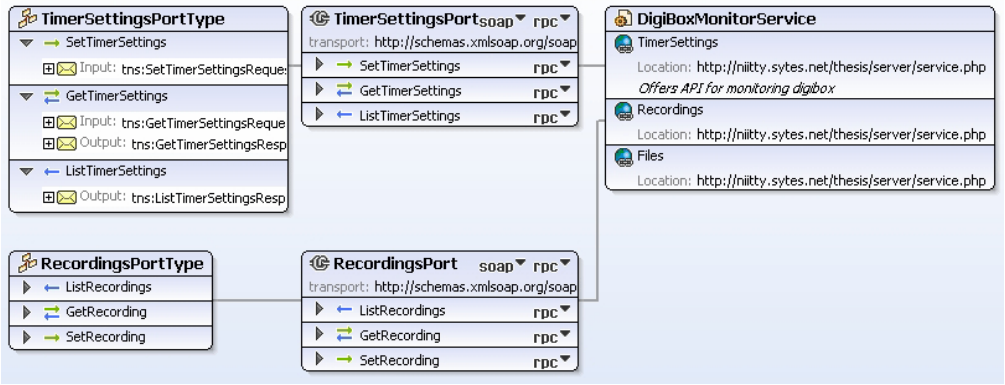
Digiboxi-sovelluksen ytimenä on PHP:llä toteutettu palvelin tietokannan hallintaan. PHP tarjoaa SoapServer API:n, jonka avulla SOAP kutsujen tekeminen ja käsittely on varsin suoraviivaista. SoapServer tarvitsee WSDL-rajapintamäärittelyn oikeiden SOAP-kutsujen tekemiseksi. WSDL on XML-dokumentti jossa palvelun rajapinta kuvataan tarkasti. Aluksi WSDL määrittelee käytettävät tietotyypit. Tietotyyppi voi olla yksinkertainen kokonaisluku tai luokkakokoelma. Alla on esimerkki eräästä työssä käyttämästäni tietotyypin määrittelystä.

```
<xs:element name="TimerSettings">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="Id" type="xs:unsignedInt"/>
     <xs:element name="Type" type="xs:string"/>
     <xs:element name="StartTime" type="xs:int"/>
     <xs:element name="Length" type="xs:int"/>
     <xs:element name="Channel" type="xs:string"/>
     <xs:element name="Filename" type="xs:string"/>
     <xs:element name="Repeat" type="xs:string"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
```

Jokainen SOAP kutsu määritellään porttityyppissä (PortType) joka kertoo minkälainen kutsu on kyseessä ja mitä tietoa pyynnön ja vastauksen yhteydessä siirretään. Kutsu voi siirtää tietoa neljällä eri tavalla:

- Ensin pyyntö sitten vastaus
- Vain vastaus
- Vain pyyntö
- Ensinvastaus sitten pyyntö

Porttityyppi sisällytetään porttiin (Port) joka julkaisee rajapinnan. Kun tarvittaville SOAP kutsulle on tehty portit, linkitetään WSDL-dokumentti Web service-palveluun ja tiedoston kautta saadaan tieto palvelun rajapinnasta (kuva 1).



Kuva 5.1 Osa digiboxisovelluksen WSDL-rakennetta

SOAP-kutsuille tulee tehdä vastaavat funktiot PHP-luokassa. Näissä funktioissa tehdään varsinainen työ siirretyn tiedon jatkokäsittelyssä (kuva 5.3). Toteutettu PHP-luokka annetaan parametrina SoapServer:lle WSDL-tiedoston kanssa (kuva 5.2).

```
329  // Start soap service and handle client requests
330  $server = new SoapServer(
331      "../wsdl/digiboxmonitorservice.wsdl",
332      array("encoding" => "iso-8859-1") );
333  $server->setClass("DigiboxMonitorService");
334  $server->handle();
```

Kuva 5.2 SOAP palvelun käynnistys

SoapServer:n handle()-käskyn jälkeen palvelu on valmis käsittelemään SOAP-kutsuja.

```
101     // SetRecording
102     // WSDL operation
103     //--------------------------------------------------
104     public function SetRecording( $values )
105     {
106         $this->AddToLog( 'SetRecording: Called' );
107
108         $query = "REPLACE INTO `Recordings`
109             (`Id`,`Name`,`Channel`,`Size`,`Date`)
110             VALUES (
111             '".$values->Id."',
112             '".$values->Name."',
113             '".$values->Channel."',
114             '".$values->Size."',
115             '".$values->Date."' )";
116
117         $this->AddToLog( 'SetRecording: query: '.$query );
118
119         mysql_query("START TRANSACTION");
120         if ( mysql_query( $query ) )
121         {
122             mysql_query("COMMIT");
123             $this->AddToLog( 'SetRecording: database updated' );
124         }
125     }
```

Kuva 5.3 PHP:llä toteutettu SOAP kutsu

5.3.2 Digiboxisovelluksen käyttöliittymä

Käyttöliittymä on hyvin yksinkertainen johtuen siitä, että palvelu vain näyttää sisältämiään tietoja eikä niitä muokata tästä käyttöliittymästä. Käyttöliittymässä on yksinkertaisia listoja jotka näyttävät mitä palvelun tietokanta sisältää. Se on toteutettu käyttämällä XHTML-, CSS2- ja Javascript-kieltä (kuva 5.4).

Kuva 5.4 Digiboxisovelluksen käyttöliittymä

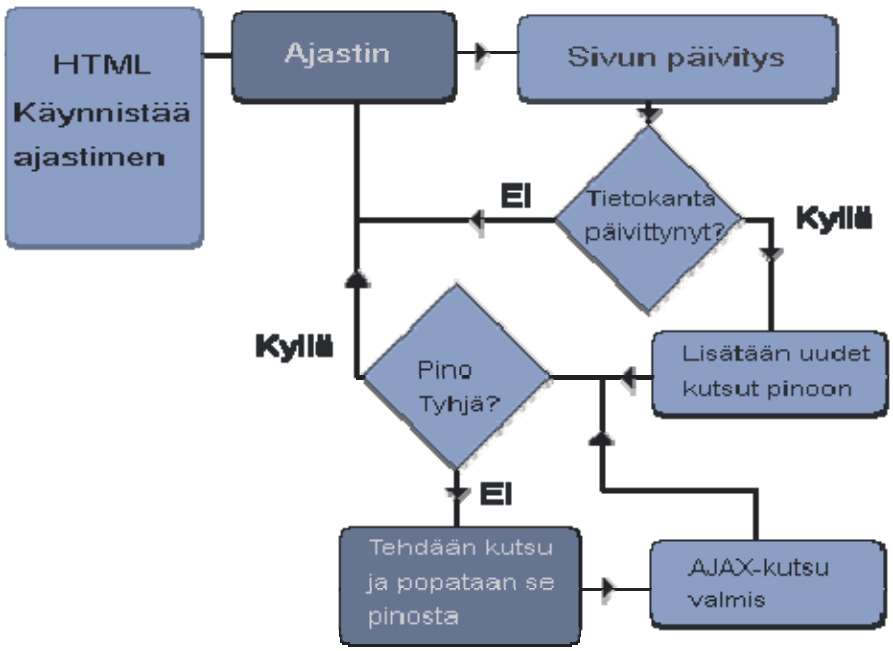
Käyttöliittymä on täysin erillään PHP-palvelusta ja se hakee tiedot listoihin SOAP-kutsuilla. Koska sivun on tarkoitus monitoroida tietokannan tilaa, rakensin siihen ajastimella toimivan komponentin, joka tarkkailee tietokannan tilan muutoksia. Ajastinkomponentti käyttää AJAX:a (Asynchronous Javascript And XML) tiedon välitykseen tekemällä kyselyn tietokantapalvelimelle taustalla. Kun jokin listoista on muuttunut, tehdään uusi AJAX haku kyseisen listan tiedoille. Jos haku onnistuu päivitetään lista Javascriptillä, muussa tapauksessa sivu pysyy ennallaan. Näin saadaan miellyttävämpi päivitys sivulle vain tiedon muuttuessa sen sijaan, että sivu ladattaisiin kokonaan uudestaan tietyin välein.

Listojen päivittämisessä ilmeni pieni ongelma AJAX-kutsun kanssa. Jokaiselle listalle tehdään oma SOAP-kutsu AJAX:a käyttäen. Näin tulee useita perättäisiä AJAX-kutsuja. XMLHttpRequest-kutsu tehdään asynkronisesti, eikä tulosta jäädä odottamaan koodissa, vaan tulokset käsitellään callback-funktiossa kun ne saapuvat. Koska kerrallaan voi käynnissä olla vain yksi XMLHttpRequest-instanssi, aiheutti tämä perättäisten kutsujen aikana javascript-tulkille virhetilanteen. Selkein ratkaisu mielestäni oli tehdä kutsuille oma pinoluokka. Aina AJAX-kutsua tarvittaessa laitetaan

se pinoon ja ajetaan pinon käsittelykäsky. Kun vastaus on saatu, suoritetaan pinossa oleva seuraava kutsu. Näin kutsuja voi lisätä tarvittavan määrän ja ne suoritetaan peräkkäin kunnes pino on tyhjentynyt. Kuva 5.5 näyttää koodiesimerkin kutsupinon luonnista ja käsittelystä. HandleCalls()-funktio käsittelee kutsupinoa ja sitä kutsutaan uudestaan XMLHttpRequest:lle annetussa callback-funktiossa aina kun edellinen AJAX-kutsu on tehty. Tämä tapahtumaketju kuvataan kaaviossa 5.1.

```
46  // ---------------------------------------------------------------------
47  // Initialize page settings and content
48  // ---------------------------------------------------------------------
49  function LoadPage()
50  {
51      // stack handles multiple ajax calls in serial way
52      // needed AJAX request is pushed on to stack and called in turns
53      ajaxCallStack = new AJAXCallStack();
54      ajaxCallStack.Push( new CallStackItem('ListFiles') );
55      ajaxCallStack.Push( new CallStackItem('ListTimerSettings') );
56      ajaxCallStack.Push( new CallStackItem('ListRecordings') );
57      HandleCalls();
58  }
59
60  // ---------------------------------------------------------------------
61  // HandleCalls
62  // Make AJAX request if call is in stack
63  // ---------------------------------------------------------------------
64  function HandleCalls()
65  {
66      if ( !ajaxCallStack.Empty() )
67      {
68          request = ajaxCallStack.Pop();
69          DoSoapRequest( request.action, request.parameters );
70      }
71  }
```

Kuva 5.5 AJAX kutsupinon hallinta



Kaavio 5.1 Digiboxipalvelimen käyttöliittymäsivun päivityksen vuokaavio

5.4 Digiboxin ohjaus-widget

Digiboxin ohjaukseen suunnitellun widgetin on tarkoitus olla mahdollisimman suoraviivainen ja selkeä. Koska kyseessä on sovelluksen ensimmäinen prototyyppi, tehtiin suunnittelu ja toteutus iteraatioiden kautta. Widgetille oli muutamia vaatimuksia, jotka oli täytyttävä, mutta muuten sen ominaisuudet ja toiminnallisuus muuttuivat kehitystyön edetessä. WRT on uusi ympäristö ja käytössä oleva versio oli vasta beta-asteella. Kaikki asiat eivät siis toimineet halutulla tavalla ja joidenkin ongelmien kiertäminen aiheutti paljon lisätyötä. Satunnaisia kaatumisia tuli kehitettäessä widgetiä S60 3rd Edition FP2 -versiolla. Varsinaisella laitteella (N95) asiat toimivat kuitenkin paremmin kuin SDK:lla.

5.4.1 Määrittely

Widgetillä oli vaatimuksena mahdollisuus ajastaa TV-lähetyksen nauhoitus, listata digiboxissa olevat tiedostot ja selata aikaisempia nauhoituksia.

Tiedoston listauksen lisäksi on myös mahdollisuus poistaa tiedostoja. Näin käyttäjä voi tarvittaessa saada lisää tilaa ohjelmien nauhoitukselle.

Widgetissä on neljä näkymää, joiden kautta käyttäjä hallinnoi sovellusta ja digiboxia. Näkymät ovat

- **nauhoitukset**, jossa on listattuna digiboxilla nauhoitetut lähetykset
- **tiedostonhallinta**, jossa käyttäjä voi listata kaikki tiedostot, jotka on siirretty digiboxiin
- **ajastukset**, jossa käyttäjä näkee nauhoitukseen ajastetut ohjelmat
- **uusi ajastus**, jonka kautta voidaan tehdä uusia ajoitettuja nauhoituksia.

Näkymien vaihdon voi tehdä S60 Options Menun kautta kuten kuvassa 5.6 tai pikanäppäimellä, joka on selection keyn painallus vasemmalle. Tällöin näytön alareunaan ilmestyy lista, jossa näkyy eri näkymien ikonit, ja mahdollisuus valita niistä jokin, jolloin valittu näkymä aktivoituu.

Kuva 5.6 Options Menu

Käyttäjä voi selata näkymien listoja painamalla selection keytä ylös tai alas. Jokaisen listan rivin kohdalla käyttäjä voi painaa selection key:tä oikealle, jolloin ruudun oikeasta reunasta liukuu esiin toimintavalikko. Tästä valikosta voi kyseisen rivin sisällölle valita siihen sisällytettyjä toimintoja.

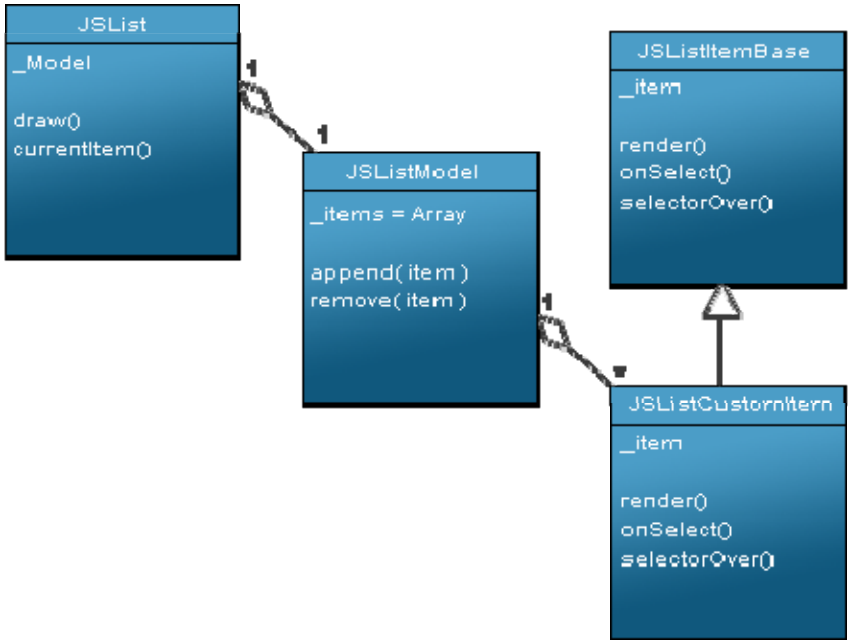
Näkymiä tehdessä tulee huomioida, että lähes jokaisen näkymän sisältö joudutaan siirtämään tetoverkon yli. Tiedonsiirto voi olla maksullista, ja näkymien sisältö tulisi hakea uudestaan vain silloin, kun käyttäjä haluaa sen tehdä. Widgetissä tulisi olla asetus, jossa listojen päivityksen voi laittaa automaattiseksi tai manuaaliseksi.

5.4.2 Suunnittelu ja toteutus

Widgetin suunnitteluvaiheessa oli aika miettiä miten toteuttaa tarvittavat käyttöliittymäkomponentit. WRT ei sisällä valmiita komponentteja käyttöliittymän tekemiseen lukuunottamatta S60 options menua, joten suurin osa käyttöliittymästä on suunniteltava ja tehtävä itse.

Lista

Koko sovelluksen peruselementiksi suunnittelin dynaamisen listan, jonka jokaisen rivin sisällön ja ulkoasun voi muokata omaan tarpeeseen sopivaksi. Listan suunnittelussa on otettu tavoitteeksi uudelleen käytettävyys. Kaikki widgetin näkymät käyttävätkin tätä listaa hieman eri tavoilla. Lista luodaan HTML:n DIV-elementin sisäänja se sovittaa kokonsa tämän elementin mukaan. Lista ei myöskään saa venyttää elementtiä, koska silloin alas ja ylös navigoitaessa koko sivu rullaa mukana. Tämä ilmiö ei ole haluttava, koska se sekoittaa navigoinnin. Jos listassa on enemmän rivejä kuin sen korkeus sallii näyttää, tulee listan rullata sisältöään sen elementin sisällä, johon se on laitettu. Lista pitää kirjaa sille syötetyistä elementeistä Model-luokassa, joka on koosteena listassa. Model-luokka ottaa vastaan JSListModelItem-rajapintaluokasta periytyviä olioita. Tämä hierarkia näkyy kaaviossa 5.2.



Kaavio 5.2 Javascript-listan rakenne

Kun objekti periytetään rajapintaluokasta, siitä löytyvät listan tarvitsemat metodit. Riville syötettävän elementin muun toteutuksen voi tehdä haluamallaan tavalla. Tällä menetelmällä saa ohjelman suunnittelija suuremman vapauden ulkoasun ja toiminnallisuuden suhteen.

Näkymien vaihto

Widgetin näkymien vaihtoon halusin suunnitella geneerisen menetelmän, jota olisi mahdollisuus käyttää myös tulevissa widgeteissä. Widgeteissä näkymät ovat HTML:n DIV-elementtejä, joiden näkyvyyttä hallinnoidaan DIV-elementin display-ominaisuudella. Kun DIV-elementti halutaan pois ruudulta, asetetaan display-ominaisuus ”none”-tilaan. Takaisin ruudulle DIV-elementin saa kun display-ominaisuus asetetaan tilaan ”block”.

Näkymien hallinnointiin suunnittelin näkymäpinon, johon jokainen ohjelmassa käytettävä näkymä tulee lisätä. Tämä pino tukee indeksointia, eli näkymän voi aktivoida tietyllä indeksillä. Jokaiselle näkymälle tulee määrittää näkymätunniste. Näkymät rekisteröidään pinoon antamalla viittaus näkymän

elementtiin ja näkymän tunniste. Näkymiä voi rekisteröinnin jälkeen aktivoida antamalla näkymän tunnisteen argumenttina activateView()-funktiolle kuten kuvassa 5.7 näkyy. Pinoluokka pitää kirjaa siitä mikä näkymä oli viimeisin aktiivinen ja kun uusi näkymä aktivoidaan, edellinen näkymä deaktivoidaan. Pinoluokka tukee myös takaisinkutsufunktiota, joka suoritetaan aina kun uusi näkymä aktivoidaan. Tätä voidaan käyttää esimerkiksi menun muuttamiseen eri näkymille.

Tiedonsiirto

Widget käyttää tiedonsiirtoon AJAX-menetelmää ja kommunikointiin SOAP:a. Digiboxisovellukseen olin jo tehnyt funktiot joiden avulla SOAP-kutsujen tekeminen AJAX:n kautta on osittain automatisoitu. Päätin käyttää samaa koodia myös widgetissä, jolloin SOAP-kutsut muodostetaan olioista niiden jäsenmuutujien perusteella.

```
/**
 * View id constants
 */
var MAIN_VIEW = 11;
var FILELIST_VIEW = 12;
var RECORDINGSLIST_VIEW = 13;
var TIMERSETTINGSLIST_VIEW = 14;
var TIMERSETTINGSFORM_VIEW = 15;

// register views
mainView = document.getElementById("MainView");
timerSettingsView = document.getElementById("TimerSettingsView");
fileListView = document.getElementById("FileListView");
recordingsView = document.getElementById("RecordingsView");
setTimerSettingsView = document.getElementById("SetTimerSettingsVi

views = new WidgetViewStack();
views.registerView( MAIN_VIEW, mainView );
views.registerView( TIMERSETTINGSLIST_VIEW, timerSettingsView );
views.registerView( FILELIST_VIEW, fileListView );
views.registerView( RECORDINGSLIST_VIEW, recordingsView );
views.registerView( TIMERSETTINGSFORM_VIEW, setTimerSettingsView )
// set callback function to run when view is changed
views.setViewChangedCallback( viewActivatedCB );
```

Kuva 5.7 Näkymien hallinnan alustus

Ongelma HTML input -elementtien kanssa

Uutta ajastusta varten tein lomakkeen, jonka kautta käyttäjä voi syöttää ajastukseen tarvittavia tietoja. Helpoin tapa ottaa vastaan käyttäjän syötteitä, on käyttää HTML input -elementtiä. Web-selain luo input tagista kentän, johon syöte voidaan kirjoittaa. Tämä toimii myös WRT:ssä hyvin, kun käytössä on kursoritila, eli näytöllä näkyy kursori-ikoni, jota voi liikutella puhelimen joystickilla. Kursori tulee tällöin siirtää halutun elementin päälle ja painaa valintapainiketta, jotta elementti aktivoituu. Halusin kuitenkin käyttöliittymän toimivan niin, että käyttäjä voi joystickilla siirtyä suoraan elementistä toiseen. Input-elementtiin tulee silloin lisätä tabindex-atribuutti ja antaa elementille focus()-käsky aina kun se aktivoidaan, ja blur()-käsky kun siirrytään pois elementistä. Tämän tekniikan käyttäminen aiheutti kuitenkin joka kerta S60-emulaattorin kaatumisen. Kohdelaitteessa ympäristö ei kaatunut, mutta minkäänlaista näppäinsyötettä ei input-elementille saanut annettua.

Tämän bugin vuoksi jouduin tekemään hieman ylimääräistä työtä, eli kehittämään omatekoisen input-kentän. Tämä input-kenttä pohjautuu DIV-elementtiin ja ottaa vastaan QWERTY-näppäinsyötteet ja tukee myös ITU-T-näppäimistöä. Jotta käyttäjä huomaa, että kyseessä on input-kenttä, on kentässä myös vilkkuva kursori. Input-kenttä on widgetissä toteutettu listaelementtinä ja se aktivoituu automaattisesti, kun valinta siirtyy input-elementin sisältävälle riville. Uudesta input-kentästä tuli muunneltavuudeltaan varsin onnistunut ja kehitystyö tulee todennäköisesti jatkumaan.

5.5 Jatkokehitys

Koko konseptin jatkokehitys riippuu markkinoille tulevien laitteiden ominaisuuksista. Jos sopivia laitteita tulee markkinoille, aion toteuttaa jonkin laitteen ohjauksen S60 WRT-widgetillä. Myös NFC:n käyttäminen widgetin

siirtämiseen kodinlaitteesta matkapuhelimeen on tarkoitus toteuttaa käytännössä myöhemmin. Kun WRT:tä tukevia matkapuhelimia tulee markkinoille enemmän, tulee widgetin kehityksessä huomioida muutamia lisäseikkoja. Todennäköisesti jokaisessa uudessa WRT:tä tukevassa laitteessa on erilaiset näytöt ja resoluutiot. Widgetin kehityksessä tulee sen jälkeen huomioida grafiikan ja näkymien skaalaus eri resoluutioille entistä tarkemmin.

Widgetin ulkoasuun ja käytettävyyteen tulee myös varmasti parannuksia. Tällä hetkellä grafiikka ei skaalaudu eri resoluutioille. Käytetävyyden osalta jää nähtäväksi miten paljon WRT:n virallisessa julkaisussa pystyy vaikuttamaan S60 Options Menun ja Soft Keyn toimintaan. Jos esimerkiksi oikeaan Soft Keyhin liittää jonkin muokatun toiminnan, on siitä seurauksena, että näkyviin tulee S60 Control Pane. Tämä ei kaikkien sovellusen kohdalla ole paras vaihtoehto.

6 PÄÄTELMÄT

Työssä käytiin läpi tiedonsiirtoon liittyen DLNA, UPnP ja NFC tekniikat, joiden tavoitteena on yksinkertaistaa eri laitteiden välisen tiedonsiirron käynnistäminen käyttäjän näkökulmasta katsottuna. Tämä on hieno tavoite ja tarpeellinen voimakkaasti kasvavalla kodintekniikan alalla.

DLNA oli tätä työtä tehtäessä jo saatavilla joidenkin valmistajien laitteissa, kuten Sonyn Playstationissa ja Nokian N95 8GB:ssä. Standardit ovat jo käyttökelpoisia ja laitteita varmati tulee enemmän saataville tulevaisuudessa. DLNA:n kautta tapahtuva verkon luonti on täysin automaattinen, ja tiedon jakamiseen on selkeät ja yksinkertaiset menetelmät. DLNA on yhdistänyt olemassaolevia standardeja ja tekniikoita toimivaksi kokonaisuudeksi. Tämän tyyppinen tapa yhdistellä olemassa olevia tekniikoita, laajentuu koko ajan tietotekniikassa. Menetelmällä saavutetaan suuri etu tuottavuudessa ja ylläpidettävyydessä, sekä saadaan riippumaton standardi kaikille laitevalmistajille.

NFC tekee vielä tuloaan Euroopan markkinoilla, mutta on esimerkiksi Aasiassa suosittu menetelmä maksamisessa. NFC:n eniten käytetty ja mainostettu ominaisuus tulee varmasti olemaan mobiilimaksaminen. Sen ominaisuudet antavat kuitenkin aihetta odottaa innovatiivisempiakin sovelluksia. Markkinoilla on jo NFC:n kautta toimivia digitaalisia valokuvakehyksiä ja sitä voi käyttää esimerksi etäkäyttöliittymän käynistämiseen matkapuhelimessa. NFC:n etuna on sen helposti ymmärrettävä vuorovaikutus ympäristön kanssa, eli kosketus. Yhdistettynä esimerkiksi DLNA:n kanssa, saadaan aikaan varsin monipuolinen langaton tiedonsiirtoympäristö.

Nokia Oyj:n S60-ympäristöön kehittämä Web Run-Time on vielä kehitysvaiheessa, mutta osoittautunut hienoksi lisäksi mobiiliohjelmointiin.

Näyttävien ja interaktiivisten verkkosovellusten tekeminen on WRT:n avulla tehokasta, koska kehitys tehdään web-ohjelmoinnista tutuilla työkaluilla. Tällä tekniikalla on paljon sovelluskohteita, vaikka useimmat sillä tehdyt widgetit ovat jonkin suositun verkkopalvelun mobiiliversioita. WRT:n virallisessa julkaisussa saadaan saada käyttöön enemmän tietoja puhelimesta, kuten GPS, yhteys- ja kalenteritiedot. Tämä mahdollistaa jo varsin monipuolisten sovellusten kehittämisen. Mikään ei estä tekemästä laajaa tietokantasovellusta tai GPS-karttasovellusta. Todennäköisesti tulevaisuudessa useat mobiililaitteiden käyttöliittymien osat saattavat olla tehty Web Run-Timella, varsinkin kun sen saa koohtuullisella vaivalla portattua eri käyttöjärjestelmiin.

WRT:n suurin ongelma on keskeneräinen ja epävakaa kehitysympäristö. Myös sovellusten ohjelmoinnissa käytetyn Javascriptin debuggaus on haasteellista. Virhetilanteiden selvittämiseen ei ole vielä selkeää keinoa, ja suurin osa kehitystyöstä tehdään käytännössä työpöytäselaimilla. Valmiissa tuotteessa on toivottavasti saatu parannuksia näihin ongelmiin.

LÄHTEET

1. Widgets Introducing Web Run-Time. [Sähköinen dokumentti.] Forum Nokia. [Viitattu 20.2.2008] Saatavissa: http://sw.nokia.com/id/f8e738e4-cbeb-4b40-b92f-5e8c7cf8dd48/Introducing_Web_Run-Time_v1_1_en.pdf
2. Web Run-Time API Reference. [Sähköinen dokumentti.] Forum Nokia. [Viitattu 20.2.2008] Saatavissa: http://sw.nokia.com/id/b9ad2b23-07ea-46cd-badf-f0ba3df97da3/Web_Run_Time_API_Reference_v1_1_en.pdf
3. Web Run-Time Going Beyond Browsing [Sähköinen dokumentti.] Forum Nokia [Viitattu 3.4.2008] Saatavissa: https://mktools.forum.nokia.com/pics/Widgets_Nielsen.pdf
4. Käyttöliittymä älykotiin [Sähköinen dokumentti.] [Viitattu: 15.4.2008] Saatavissa: http://lauri.sokkelo.net/files/kayttoliittyma_alykotiin.pdf
5. Wide Area Media Sharing with UPnP/DLNA [Sähköinen dokumentti.] Naraynan Venkitaraman, Motorola Labs [Viitattu 15.4.2008] Saatavissa: http://ieeexplore.ieee.org/iel5/4446297/4446298/04446370.pdf?tp=&arnumber=4446370&isnumber=4446298
6. DLNA homepage [www-sivu] [Viitattu: 15.4.2008] Saatavissa: http://www.dlna.org/about_us/about/
7. DLNA White paper [Sähköinen dokumentti.] DLNA [Viitattu 15.4.2008] Saatavissa: http://www.dlna.org/industry/join/organization/DLNA_white_paper.pdf
8. DLNA Use Cases [Sähköinen dokumentti.] DLNA [Viitattu 16.4.2008] Saatavissa: http://www.dlna.org/industry/join/organization/DLNA_Use_Cases.pdf
9. Near Field Communication white paper [Sähköinen dokumentti.] ECMA International [Viitattu:] Saatavissa: http://www.ecma-international.org/activities/Communications/2004tg19-001.pdf
10. Universal Plug and Play in Windows XP [WWW-sivu] Microsoft [Viitattu: 17.04.2008] Saatavilla: http://technet.microsoft.com/fi-fi/library/bb457049(en-us).aspx

11. UPnP Device Architecture 1.0 [Sähköinen dokumentti.] UPnP Forum [Viitattu: 17.04.2008] Saatavilla: http://www.upnp.org/specs/arch/UPnP-DeviceArchitecture-v1.0.pdf
12. Near Field Communication Interface and Protocol [Sähköinen dokumentti.] ECMA International [Viitattu: 20.4.2008] Saatavissa: http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-340.pdf
13. NFC Data Exchange Format (NDEF) Technical Specification [Sähköinen dokumentti.] NFC Forum [Viitattu: 20.4.2008] Saatavissa: http://www.nfc-forum.org/specs/spec_license/download_spec/d11789e06e9893a16de5f23584a75b31b20a94ff/NFCForum-TS-NDEF_1.0.pdf
14. Smart Poster Record Type Definition [Sähköinen dokumentti.] NFC Forum [Viitattu: 20.4.2008] Saatavissa: http://www.nfc-forum.org/specs/spec_license/download_spec/d11789e06e9893a16de5f23584a75b31b20a94ff/NFCForum-SmartPoster_RTD_1.0.pdf