Joonas Hämäläinen ja Timo Kantanen

Saunan etäkäyttö mobiililaitteella

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikan koulutusohjelma

Insinöörityö

19.11.2015

Metropolia

Tekijät Otsikko Sivumäärä Aika	Joonas Hämäläinen ja Timo Kantanen Saunan etäkäyttö mobiililaitteella 58 sivua 19.11.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Sulautetut järjestelmät
Ohjaaja	Lehtori Kimmo Saurén

Insinöörityön tarkoituksena oli toteuttaa saunan etäkäytön mahdollistava ohjausjärjestelmä. Järjestelmään kuuluu palvelin, joka vastaa saunan tilanteen seurannasta ja mekaanisten osien käskyttämisestä. Lisäksi siihen kuuluu mekaaninen, kiukaan säätönuppeihin asennettava aktuaattori. Tämä mekaaninen osa säätää kahdella servomoottorilla kiukaan nuppeja oikeisiin asentoihin. Kaikkea tätä ohjataan sovelluksella, jolla on mahdollista ajastaa saunan päällemenoaika ja säätää haluttua lämpötilaa.

Ohjelmistopuoli toteutettiin käyttämällä useita erilaisia vapaasti saatavilla olevia työkaluja. Mekaaninen kiukaaseen kiinnitettävä osa jäi tarkoituksella prototyyppiasteeseen tuotantosyistä. Varsinaista mekaanista testausta tehtiin rajatusti.

Insinöörityön lopputuloksena oli perustaltaan toimiva kokonaisuus, jota pystytään myös jatkossa kehittämään.

Avainsanat	etäkäyttö, sauna, palvelin, ohjausjärjestelmä

Authors Title Number of Pages Date	Joonas Hämäläinen and Timo Kantanen Remote access to sauna on mobile devices 58 pages 19 November 2015
Degree	Bachelor of Engineering
Degree Programme	Information and Communications Technology
Specialisation option	Embedded systems
Instructor	Kimmo Saurén, Senior Lecturer

The goal of this final year project was to design and implement a system which allows the user to heat a sauna remotely. The system includes a server which is responsible for monitoring the sauna and for giving commands to the mechanical parts. The system can be controlled with an Android application. The application can be used to schedule a specific time for heating the sauna and also to adjust the desired temperature.

The programming in this project was executed using a variety of free tools such as Eclipse. A mechanical part which attaches to the stove was built. It became a prototype, and actual mechanical testing was limited because of production reasons.

At the end of the final year project a fully operational, remotely controlled system had been implemented. The system is designed in a way which makes its future development possible.

Keywords	Android, sauna, remote access, server

Sisällys

Lyhenteet ja termit

1 Johdanto 1

2 Etäkäyttö ja -hallinta yleisesti 1

3 Raspberry Pi -tietokone 3

4 Android-käyttöjärjestelmä 7

4.1 Android-versiot 7
4.2 Android-ohjelmointi 10
4.3 Android-laitteet 14

5 Ohjausjärjestelmän mekaniikka 15

5.1 Suunnittelu 15
5.2 Komponentit 16

6 Etäohjattava saunajärjestelmä 20

6.1 Palvelinpuolen toiminta 20
6.2 Palvelinpuolen ohjelmisto 27
6.3 Palvelinpuolen ongelmat 30
6.4 Käytönvalvonta palvelimella 33
6.5 Palvelimen komponentit 34

7 Android-saunasovellus 35

7.1 Yleistä 35
7.2 Lämpötila 37
7.3 Asetukset 40
7.4 Ilmoitukset 42

8 Testaus 48

9 Jatkokehitys 51

9.1 Palvelimen jatkokehitys 51
9.2 Android-sovelluksen jatkokehitys 52
9.3 Mekaniikan jatkokehitys 54

10 Yhteenveto 55

Lyhenteet ja termit

SSH	Secure Shell on salattuun tietokoneliikenteeseen tarkoitettu tekstipohjainen protokolla, joka mahdollistaa etäkäytön kahden tietokoneen välillä.
Skripti	Ohjelma tai ohjelman osa, joka suorittaa tiettyä asiaa.
Telnet	Tekstipohjainen yhteysprotokolla pääteyhteyksiin internetin välityksellä. Se ei ole salattu yhteys.
RDP	Remote Display Control on Windows-käyttöjärjestelmän etäkäyttöprotokolla.
VNC	Virtual Network Computing on etäkäyttöön tarkoitettu protokolla, jolla pystytään käyttämään toisen tietokoneen graafista käyttöliittymää.
Unix	Laitteistoriippumaton käyttöjärjestelmä.
Linux	Avoimen lähdekoodin käyttöjärjestelmä, joka perustuu Unixiin.
FTP	File Transfer Protocol on tiedonsiirtomenetelmä kahden tietokoneen välille.
FIFO	First In First Out on Linux-ympäristössä prosessien välistä kommunikointia jono-periaatteella.
LED	Light-emitting-diode on puolijohdekomponentti, joka säteilee valoa virran kulkiessa sen lävitse.
UPS	Uninterruptible Power Supply on järjestelmä tai laite, joka takaa tasaisen virransyötön katkostilanteissa.
Android	Avoimeen lähdekoodiin perustuva Linux-pohjainen käyttöjärjestelmä mobiililaitteille. Sen on kehittänyt Google.

Eclipse	Ohjelmointiympäristö, joka tukee Java-, C-, C++- ja PHP-ohjelmointikieliä. Perustuu avoimeen lähdekoodiin.
IDE	Integrated Development Environment on ohjelmointiympäristö, joka tarjoaa monipuoliset kehitysominaisuudet.
ADT	Android Development Tools on Eclipseen ladattavissa oleva paketti, joka sisältää kaikki Android-sovelluskehitykseen tarvittavat osat.
Debug	Virheenkorjaus. Debuggaus tarkoittaa virheiden paikallistamista ja korjaamista ohjelmistosta.
Activity	Androidin luokkamuoto. Activityn avulla luodaan näytölle ikkuna, jossa sovelluksen määritetyt asiat näkyvät ja navigointi tapahtuu.
Fragment	Activityn ”ala-luokka”. Activity voi sisältää useita fragmentteja ja niin kauan kuin fragmentin sisältämä activity on auki, on niillä oma elinkaari.
API	Application Programming Interface on ohjelmointirajapinta, joka määrittää ohjelmien kyvyn ”keskustella” keskenään.
Log	Tiedostoon tai konsoli-ikkunaan tallennettu ja aikaleimattu tapahtuma, jota käytetään pääasiassa debuggaukseen.
Service	Android-sovelluksen ominaisuus, joka mahdollistaa ohjelman suorittamisen tausta-ajona ilman käyttäjän siihen puuttumista.
BroadcastReceiver	Androidin osa-alue, joka mahdollistaa tapahtumien rekisteröimisen Androidin omiin ominaisuuksiin.
TCP	Transmission Control Protocol on tietoliikenneprotokolla kahden tai useamman tietokoneen välille internetissä.
Crontab	Linux-käyttöjärjestelmän automaattinen ajastinpalvelu.

1 Johdanto

Insinöörityön tarkoituksena on tehdä Android-laitteilla etäkäytettävä ja -hallittava sähkökiukaan ohjausjärjestelmä. Työn tavoitteena on saada saunan lämmitys ja sen lämpötilan tarkkailu mahdolliseksi etäyhteyttä käyttävällä mobiililaitteella. Työ sisältää Android-sovelluksen, palvelinpuolen sovelluksen ja servomoottoreilla toimivan mekaanisen osan, joka vastaa kiukaan fyysisestä säätämisestä.

Työn tarkoitus on helpottaa ihmisten saunassa käymistä, joka on iso osa suomalaista perinnettä ja kulttuuria. Järjestelmä sopii varsinkin kiireisten ihmisten käyttöön, kun saunan voi laittaa lämpenemään jo töissä tai kotimatkalla. Yleisesti järjestelmän tulee toimia niin, että käyttäjä käynnistää Android-laitteeltaan saunasovelluksen, valitsee, mihin aikaan saunaa aletaan lämmittää, kuinka kauan sitä lämmitetään ja mihin lämpötilaan se asetetaan lämpenemään. Tämän jälkeen tiedot siirtyvät Raspberry Pi-tietokoneella sijaitsevalle palvelimelle, joka puolestaan käskee kiukaaseen asennettua mekaanista toimintayksikköä siirtämään kiukaan säätövipuja haluttuun asentoon.

Saunan on ennenkin voinut laittaa päälle matkapuhelimen kautta, mutta nämä ominaisuudet ovat aina kuuluneet isompaan, koko asunnon kattavaan sähköjärjestelmän hallintaan tai kiukaaseen, johon on varta vasten sisäänrakennettuna tämä ominaisuus. Tämän työn keskipisteenä on mekaaninen ohjain, joka asennetaan sähkökiukaan säätimien yhteyteen.

Saunan etäkäyttömahdollisuus on myös lisännyt saunasta alkavien tulipalojen määrää. Päälle laitettaessa on helppo unohtaa, onko saunassa esimerkiksi pyykkejä kuivumassa, ja tätä varten järjestelmän tulisi muistuttaa käyttäjää mahdollisista vaaratilanteista.

2 Etäkäyttö ja -hallinta yleisesti

Etäkäytöllä tarkoitetaan yleisesti jonkin tietokoneen tai laitteiston käyttöä verkon kautta toisella tietokoneella olematta itse paikalla. Yleensä etäkäyttöön käytetään tunnettuja protokollia, kuten esimerkiksi SSH (Secure Shell) tai Telnet, jotka ovat tekstipohjaisia. Niiden lisäksi on olemassa graafisia etäkäyttöprotokollia, kuten esimerkiksi Windows-

käyttöjärjestelmien oma RDP (Remote Display Control) ja Olivetti & Oracle Research Labin kehittämä VNC (Virtual Network Computing). Etäkäyttöprotokollia apuna käyttäen voidaan ottaa yhteys kotoa käsin esimerkiksi työpaikan tietokoneisiin ja työskennellä kotona samalla tavalla, kuin olisi itse läsnä työpaikalla. Kotikoneen ruudulla näkyy työpaikan koneiden työpöytänäkymä, jos käytössä on esimerkiksi graafinen etäkäyttöprotokolla. Etäkäytön yleisimpiä rajoituksia ovat hitaat verkkoyhteydet etäkäytettävän tietokoneen ja etäkäytön muodostavan tietokoneen välillä. Tämän huomaa varsinkin, jos käytössä on graafinen käyttöliittymä.

Etäkäyttö on ollut jo pitkään mahdollista Unix- ja Linux-käyttöjärjestelmissä. Suurin osa Linux- ja Unix-käyttäjistä hyödyntää etäyhteyttä päivittäin. Windows-käyttäjien keskuudessa etäkäytön on yleistänyt vasta Windows XP -käyttöjärjestelmän mukana tullut etäkäyttötuki. Kuvassa 1 on havainnollistava esimerkki etäyhteydestä.

Kuva 1: Toisen koneen käyttöä etähallintaa hyväksikäyttäen.

Etäyhteys koneiden välille muodostetaan siten, että toinen koneista ottaa TCP-yhteyden (Transmission Control Protocol) kohdekoneeseen. Tätä varten yhteyden pitää olla sallittuna palomuureissa sekä pääsylistoissa, jos sellaisia on. Yhteyden avaamisen jälkeen kohdekone lähettää etäyhteyttä ottavalle koneelle oman julkisen avaimensa. Etäyhteyttä pyytävä kone vertaa kohdekoneen avainta omaan avainlistaansa, johon on listattu jo entuudestaan tunnetut julkiset avaimet. Avaimen

löydyttyä listalta etäyhteyttä pyytävä kone muodostaa satunnaisen oman avaimen, jonka se lähettää kohdekoneelle tämän julkisella avaimella salattuna. Mikäli avainta ei löydy listalta, kysytään etäyhteyttä ottavalta koneelta, tahtooko se hyväksyä avaimen ja lisätä sen listaan. Avaimen hylkääminen johtaa yhteyden katkaisuun. Avaimen lisäämisen jälkeen toimitaan tavalliseen tapaan. Kohdekone purkaa avaimen omalla julkisella avaimellaan. Jatkossa kaikki liikenne koneiden välillä on salattua. Salatun yhteyden muodostuttua tehdään varsinainen käyttäjän tunnistaminen. Salattu yhteys on tarpeen muodostaa ennen käyttäjän tunnistamista, jotta käyttäjätunnus tai salasana ei vuoda. (1.)

SSH-protokolla

SSH on suomalaisen tekniikan lisensiaatti Tatu Ylösen keksimä protokolla vuodelta 1995, kun hän työskenteli tutkijana teknillisessä korkeakoulussa. Se on erittäin käyttökelpoinen protokolla hyvän tietoturvansa vuoksi. Se mahdollistaa muun muassa monen eri algoritmin käytön. Näin ollen yhden algoritmin vuotaminen ei vielä vaaranna koko palvelimen tietoturvaa, jos käytössä on monia algoritmeja eri paikoissa. Tietojärjestelmiin murtautuvat krakkerit joutuvat murtamaan useita eri algoritmeja ja usein tämän vuoksi luovuttavatkin ja siirtyvät helpompiin kohteisiin, joissa ei ole yhtä korkeatasoista tietoturvaa käytössä.

Tässä työssä käytetään SSH-protokollaa, koska se mahdollistaa myös sen, että sillä pystytään salaamaan FTP-liikenne, jota käytetään lähetettäessä tiedostoja mobiililaitteesta palvelimelle. Käyttäjän tunnistamiseen käytetään ensin julkisen avaimen vaihtoa ja sen jälkeen käyttäjätunnus-salasanaparia. Käyttäjän tunnistaminen salasanalla on riittävän turvallinen vaihtoehto ja huomattavasti helpompi vaihtaa kuin salainen avain-menetelmä. (2.)

3 Raspberry Pi -tietokone

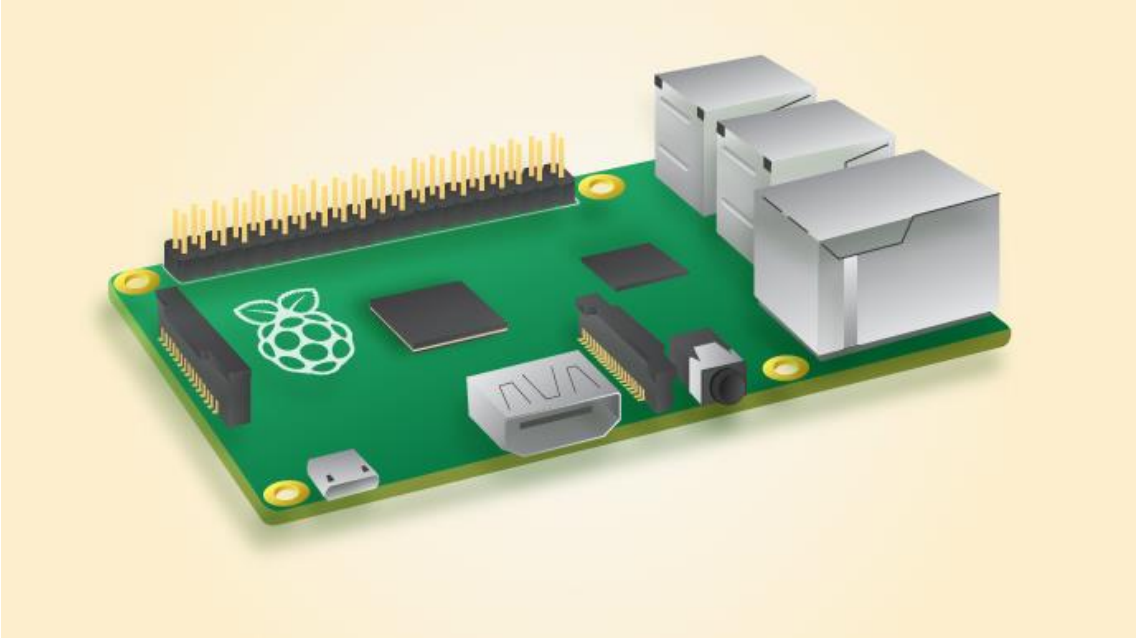
Raspberry Pi on luottokortin kokoinen yhden piirilevyn tietokone, jonka voi liittää tietokoneen näyttöön tai vaikka televisioon. Raspberry Pitä voi ohjata tavanomaiseen tapaan näppäimistöllä ja hiirellä tai vaikka etäyhteyden kautta. Kiintolevynä voi käyttää tavallista SD-korttia. Virran Raspberryyn voi tuoda esimerkiksi matkapuhelimen laturilla, jos se on microUSB. Videokuvan saa Raspberrystä joko HDMI-kaapelilla tai

RCA-komposiittipiuhalla. Äänet voi ottaa ulos samasta HDMI-liitännästä tai sitten erikseen 3,5 mm:n jakista.

Raspberry Piin kehitti brittiläinen Raspberry Pi Foundation helmikuussa 2012. Sen päätavoitteena oli luoda laite, jonka avulla kenen tahansa olisi helppo oppia tietokoneisiin liittyviä asioita, kuten esimerkiksi tietojenkäsittelyä tai ohjelmointia. Raspberry Pillä pystyy tekemään kaikkia niitä asioita, joita tavallisella tietokoneellakin pystyy tekemään. Voi selata internetiä, katsella videoita, pelata erilaisia pelejä tai vaikka itse tehdä pelejä Raspberrylle. Raspberrya onkin käytetty kekseliäästi erilaisiin projekteihin ympäri maailman, sillä siinä on myös ulkoisia pinnejä, joita voi ohjelmoida esimerkiksi vastaanottamaan lämpötila-anturista lukemia. A-mallissa on 26 pinniä, kun taas vastaavassa B-mallissa on 40 pinniä. (3.)

Raspbery Pi -piiri

Raspberry Pissä on niin sanottu SoC-piiri eli System-on-Chip-siru BCM2835 (Broadcom), joka toimii Raspberryn sydämenä. Se yhdistää kaikki laitteen osat yhteen vähän niin kuin emolevy. Havainnollistava esimerkki on kuvassa 2. Osiin kuuluu esimerkiksi prosessori, näytönohjain, piirisarja, digitaalisen signaalin käsittelyprosessori, USB ja SDRAM-muisti sekä sen ohjain. Käytännössä todella pieneen tilaan on saatu mahdutettua nykyisen älypuhelimen tehoinen tietokone. Tehoa prosessorissa on 700 MHz:n verran ja keskusmuistia version 1.0 B-mallissa 512 MB.

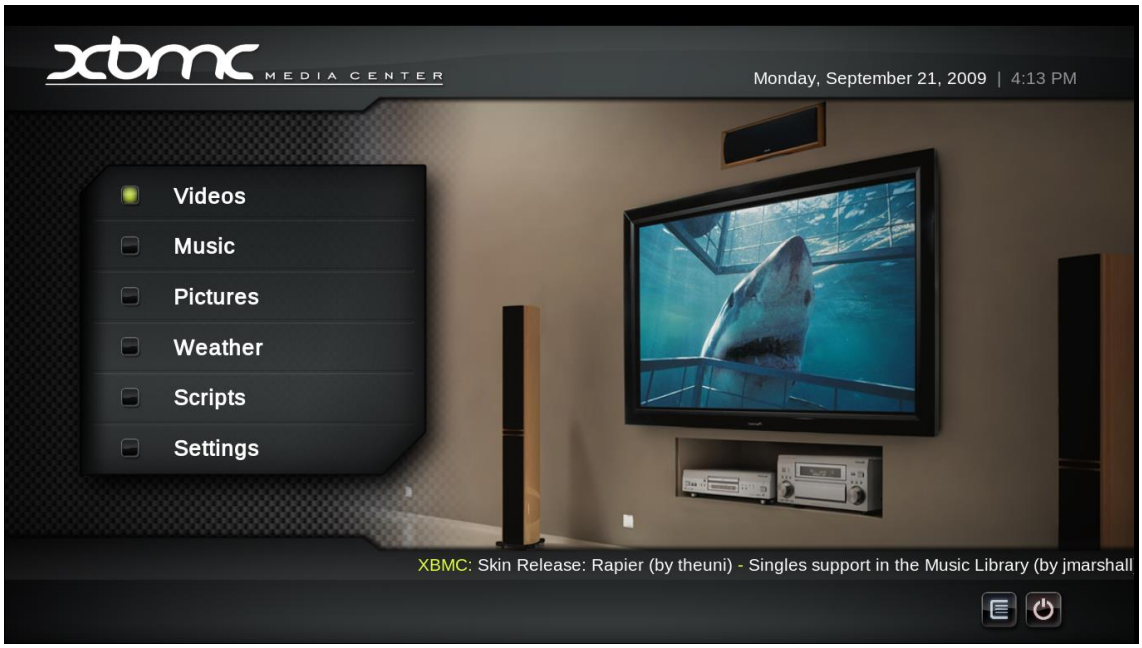


Kuva 2: Raspberry Pi-tietokone yläpuolelta (3).

Raspberry Piista on nykyään olemassa kaksi eri versiota, niin sanottu vanha eli Raspberry Pi 1.0 ja uudempi versio Raspberry Pi 2.0. Vanhassa 1.0-versiossa oli vielä neljä erilaista mallia, A ja A+ sekä B ja B+. A-mallit olivat selkeästi vielä kehitysversioita verrattuna B-malleihin. A-malleissa ei ollut edes valmiina verkkosovitinta, vaan se tuli vasta versiossa B. Hyvänä esimerkkinä voidaan pitää myös muistin määrän kaksinkertaistamista mallista A (256 MB) malliin B (512 MB). Muistin määrän kaksinkertaistaminen paransi huomattavasti myös käyttäjän kokemusta, sillä kaikki toimi paljon ripeämmin eikä tietokone koko ajan jumittanut. Rasperry Pi 2 on kaksi kertaa ensimmäistä versiota tehokkaampi. Versiossa 2.0 on uusi neliydinprosessori, jossa on tehoa 900 MHz. Keskusmuisti on niin ikään kaksinkertaistettu edellisistä versioista aina 1 GB:een asti. Myös verkkosovitinta on parannettu huomattavasti.

Raspberry Pi käyttää käyttöjärjestelmänä Linuxiin pohjautuvia käyttöjärjestelmiä, jotka on varta vasten räätälöity Raspberry Pin laitteistoa varten. Raspberry Pille on saatavissa lukematon määrä erilaisia käyttöjärjestelmiä. Yksi eniten käytetyistä käyttöjärjestelmistä on Rasbian, joka on Debian-Linuxiin pohjautuva. Rasbianin mukana tulee suuri määrä valmiiksi käännettyjä ohjelmia ja paketteja, joten tämä käyttöjärjestelmä sopii hyvin myös aloittelijoille. Rasbian on kehitetty yhdessä Raspberry Pin kehittäjien kanssa. Muita Raspberry Pille saatavia käyttöjärjestelmiä ovat esimerkiksi Arch, Fedora, Gentoo ja RISC OS. Listaa voisi jatkaa, mutta nämä ovat ehkä käytetyimmät käyttöjärjestelmät. (4.)

Raspberry Pita voi käyttää myös kotiteatterina. Kuvassa 3 näkyy kotiteatterina käytetyn Raspberryn Pin käyttöliittymä. Siihen saa ladattua käyttöjärjestelmiä, mikä tekee mahdolliseksi muuttaa pienen laitteen täydelliseksi mediakoneeksi olohuoneeseen. Tämä mahdollistaa esimerkiksi lähiverkon koneille tallennettujen videoiden katsomisen Raspberryn kautta.



Kuva 3: Raspberry Pi-kotiteatterin käyttöliittymä (5).

Raspberryllä voi myös toistaa lempikappaleitaan. Suosituin käyttöjärjestelmä tähän tarkoitukseen on ehkä XBMC eli nykyinen Kodi. Raspberry Pita on myyty vuoden 2015 kesäkuuhun mennessä noin viisi miljoonaa kappaletta, ja tämäkin määrä kertoo jotain Raspberryn suosiosta (6). Nykyään on myös mahdollista rakentaa Raspberry Pista vanhan ajan Arcade-pelikone, mikä mahdollistaa vanhojen kolikkopelien pelaamisen tällä laitteella. Moni onkin rakentanut itselleen tällaisen retron pelikoneen, kuten kuvassa 4.

Kuva 4: Raspberry Pi -tietokoneesta rakennettuja Arcade-pelikoneita (7).

4 Android-käyttöjärjestelmä

Android on alun perin Android Inc. -nimisen yrityksen kehittelemä Linuxiin perustuva käyttöjärjestelmä. Yritys perustettiin vuonna 2003 Kaliforniassa, kun sen jäsenet Andy Rubin, Rich Miner, Nick Sears ja Chris White halusivat kehittää käyttöjärjestelmää digitaalikameroille. Digikameroiden markkinat olivat kuitenkin niin pienet, että he päättivät muuttaa suunnitelmaansa ja siirsivät kehityksen kohteen puhelimiin tarkoituksenaan luoda älykkäämpiä, kosketusnäytöllä varustettuja mobiililaitteita, jotka tietäisivät enemmän käyttäjästään (8.)

4.1 Android-versiot

Vuonna 2005 Google osti Android Inc:n ja jatkoi sen kehitystä alkuperäisten kehittäjien kanssa. Google myös perusti ryhmän nimeltä Open Handset Alliance, jonka tarkoitus on osallistua Androidin kehitykseen. Ryhmä koostuu laite- ja sovelluskehittäjistä sekä telekommunikaatioyrityksistä. (8.) Androidin betaversio 1.1 julkaistiin vuonna 2007 ja ensimmäinen kaupallinen versio 1.0 vuonna 2008. Ensimmäinen kaupallinen Android-käyttöjärjestelmää käyttävä puhelin oli HTC Dream, joka julkaistiin lokakuussa 2008. Julkaisu sai sekalaisen vastaanoton, koska puhelimesta puuttui ominaisuuksia, joita oli tarjolla muilla mobiilikäyttöjärjestelmillä (10). Huhtikuussa 2009 Google julkaisi Android 1.5 -päivityksen koodinimellä "Cupcake". Siitä lähtien kaikki Android-käyttöjärjestelmän

uudet versiot ovat saaneet jälkiruokaan liittyvän koodinimen, jotka näkyvät taulukossa 1.

Taulukko 1: Android-versioiden koodinimet ja julkaisuajat.

Versio	Koodinimi	Julkaisu	API(Application programming interface)
1.0	—	23.9.2008	1
1.1	—	9.2.2009	2
1.5	Cupcake	27.4.2009	3
1.6	Donut	15.9.2009	4
2.0–2.1	Eclair	26.10.2009	5-7
2.2	Froyo	20.5.2010	8
2.3–2.3.7	Gingerbread	06.12.2010	9-10
3.0–3.2	Honeycomb	22.2.2011	11-13
4.0	Ice Cream Sandwich	18.10.2011	14-15
4.1	Jelly Bean	9.7.2012	16-18
4.4	KitKat	31.10.2013	19-20
5.0–5.1	Lollipop	12.11.2014	21-22

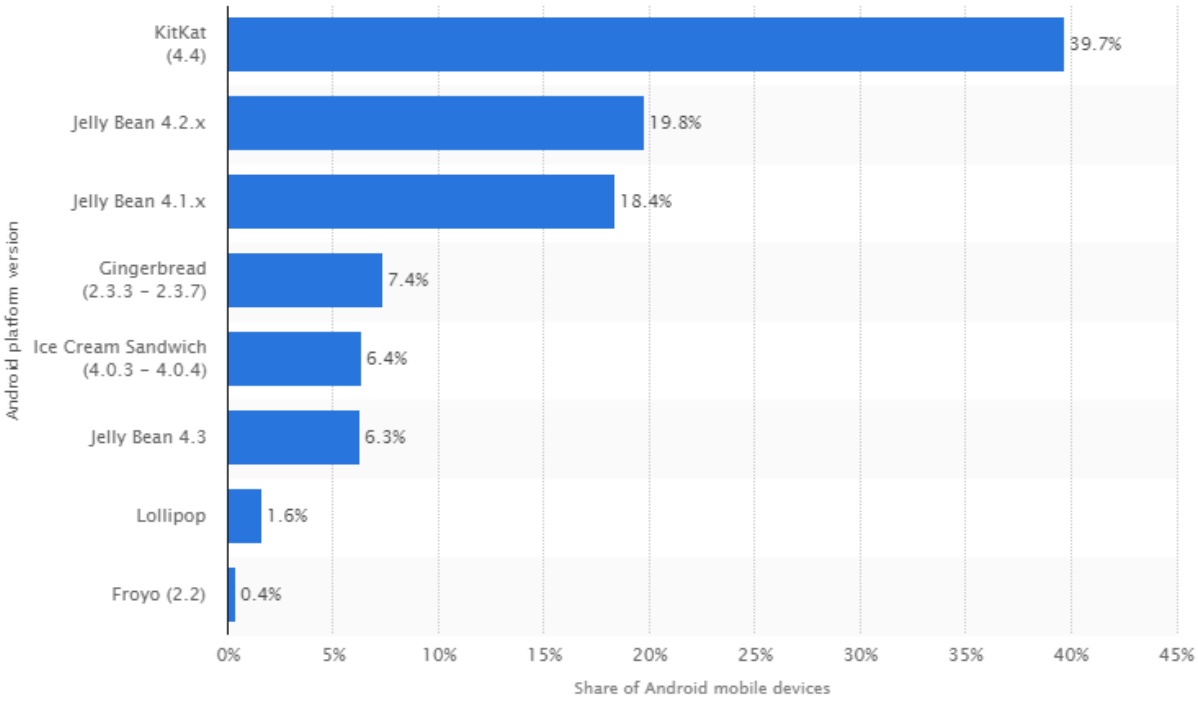
Android 2.2 "Froyo" toi mukanaan paljon uusia ja kaivattuja ominaisuuksia, kuten Googlen Chrome-selaimen V8-JavaScript-moottorin liittäminen Androidin selaimeen (11). Versio paransi myös laitteen nopeutta, muistia ja tehoa (9) sekä lisäsi Adobe Flash –tuen (12).

Versio 2.3 "Gingerbread" paransi akun elinikää optimoimalla laitteen hereilläoloaikaa. Se myös mahdollisti tuen isoille näyttöruuduille ja usealle kameralle. Ensimmäinen enimmäkseen tableteille suunnattu versio oli 3.0 "Honeycomb". Se oli myös ensimmäinen versio, joka tuki kaksiydinprosessoreita tai tehokkaampia prosessoreita. (9.)

Android 4.0 "Ice Cream Sandwich" julkaistiin vuonna 2011, ja uusina ominaisuuksina oli HD-videon tallennus ja mahdollisuus ottaa laitteen näytöstä kuvakaappaus painamalla virta- ja äänenpienennysnäppäintä samanaikaisesti. Versio 4.1 "Jelly Bean" on toiseksi eniten käytössä oleva versio, ja sen parannukset käyttöjärjestelmään olivat pääasiassa käyttöliittymän käytön mukavuuden lisäys. (9.)

Eniten käytössä oleva käyttöjärjestelmäversio on Android 4.4 "Kitkat". Tämä johtuu siitä, että versio kehitettiin sopimaan monelle eritehoiselle laitteelle. Muita parannuksia olivat muun muassa käyttöliittymän parannus, mahdollisuus nauhoittaa ruudun toimintaa ja askeltunnistin. (9.)

Viimeisin julkistettu versio on 5.0 "Lollipop" ja sen 5.1-päivitys. Muutoksia ovat Dalvik-prosessivirtuaalikoneen korvaaminen Android Runtime (ART) -ympäristöllä. Dalvik vastasi sovellusten suorittamisesta ja asennusvaiheessa Java-luokkien muuttamisesta Dalvik exe -formaattiin. ART kasvattaa Android-sovellusten suorituskykyä huomattavasti ja parantaa samanaikaisten tehtävien suorittamista (13). Muita uudistuksia ovat tuet 64-bittisille suorittimille ja usealle SIM-kortille sekä projekti Volta, joka parantaa akun elinikää (9). Tulossa olevina versioina on tiedossa vain Android 6.0 ”Marshmallow”, joka julkaistaan todennäköisesti vuoden 2015 loppupuolella tai vuoden 2016 alkupuolella. Saatavilla olevien versioiden yleisyys helmikuussa 2015 on havainnollistettu kuvassa 5.



Kuva 5: Android-versioiden jakauma Android-käyttöjärjestelmissä helmikuussa 2015 (14).

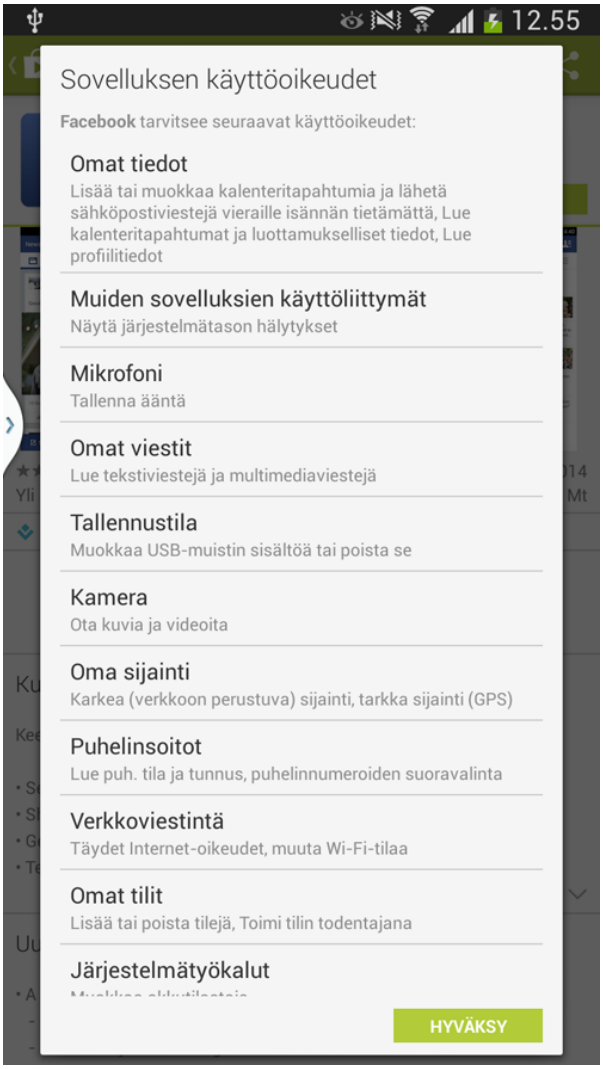
4.2 Android-ohjelmointi

Androidin lähdekoodi on avoin, se käyttää pääasiassa Java-ohjelmointikieltä Googlen omilla kirjastoilla ja kaikki halukkaat voivat kehittää Android-sovelluksia. Kehittämistä varten Google on luonut Android Studio -nimisen kehitysympäristön, joka sisältää kaiken tarvittavan sovellusten tekemiseen ja testaamiseen. Tämä ohjelma on ilmaiseksi saatavilla kehittäjien sivuilta. Sieltä voi myös ladata tarvittavat lisäosat muihin kehitysympäristöihin, kuten Eclipseen. Näin kehittäjät voivat valita mieleisensä kehitysympäristön. Android-sovelluksen käyttöliittymä on ensimmäinen asia, jonka käyttäjä sovellusta käyttäessään kohtaa, ja sen määrittely sovellusta kehitettäessä on tehty tehokkaasti ja mahdollisimman helpoksi. Käyttöliittymän osat määritellään Layout XML –tiedostossa, ja ohjelmaa ajettaessa Android asettaa määritellyt komponentit oikeille paikoilleen (15).

Android-kehittämistä varten on olemassa monia erilaisia tutoriaaleja ja oppimateriaaleja, joista osa on muotoiltu internetissä käytäviksi pienoiskursseiksi. Näistä osan ovat luoneet online-kursseihin erikoistuneet yritykset, ja ne saattavat maksaa satoja dollareita (16). Vapaasti käytettävissä olevan oppimateriaalin avulla

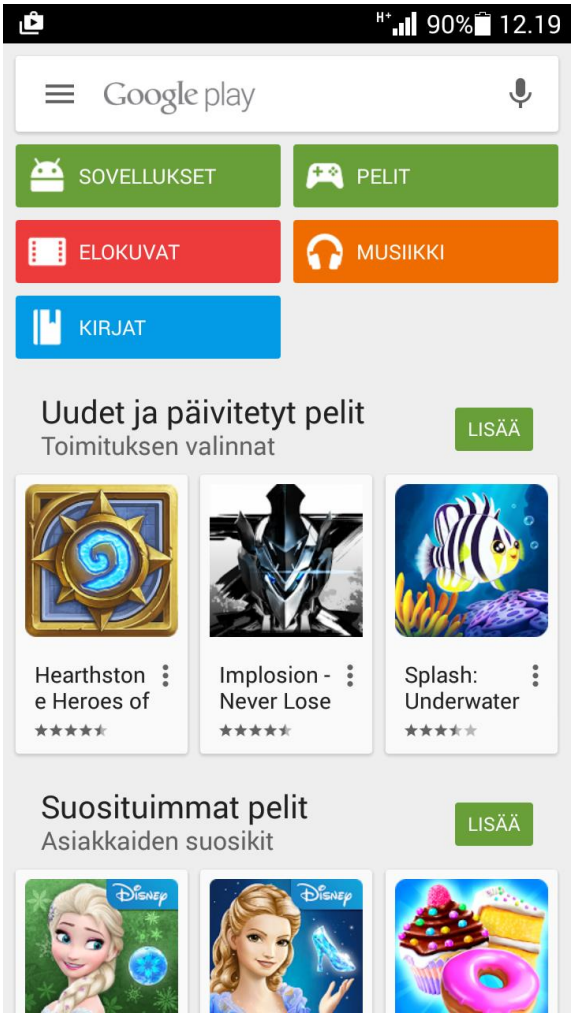
lähes kaikki, jotka ovat perehtyneet ohjelmointiin, pystyvät aloittamaan oman Android-sovelluksensa kehittämisen.

Valmiita sovelluksia ja pelejä voi julkaista Google Play -kaupassa, josta saa paljon myös ilmaiseksi ladattavissa olevia ohjelmia. Julkaisu onnistuu Play-palvelussa sen jälkeen, kun on avannut itselleen kehittäjätilin. Google ottaa 30 prosenttia kaikesta julkaistun sovelluksen tuotosta. Vastapainoksi Google Play -palvelua moderoidaan eli valvotaan tarkasti, jolloin turhia tai viallisia ja haittaohjelmia sisältäviä sovelluksia ei siellä julkaista. Jos kuitenkin sovellus, joka sisältää haittaohjelmia tai on muuten epäilyttävä käytökseltään, julkaistaan palvelussa, se otetaan nopeasti pois ihmisten saatavilta. Esimerkkinä tällaisesta sovelluksesta oli jonkin aikaa palvelussa ollut taskulamppusovellus, joka kuitenkin pääsi käsiksi sellaisiin puhelimen tietoihin, mihin sen ei toimiakseen tarvinnut päästä. Näitä tietoja saatettiin lähettää eteenpäin tuntemattomalle palvelimelle. (17.) Tämän takia kannattaakin tarkistaa ennen sovelluksen lataamista, ovatko kaikki sen vaatimat pääsyoikeudet tarpeellisia sovellukselle. Taskulamppusovelluksen tapauksessa käyttäjän tiedot lähetettiin eteenpäin, vaikka käyttäjä olisi estänyt sovelluksen pääsyn tietoihin. Hyvänä nyrkkisääntönä voidaan pitää sitä, että lataa vain luotettujen tai tuttujen kehittäjien sovelluksia. Jos epäilee kehittäjän luotettavuutta, kannattaa tehdä jonkinlainen taustaselvitys tekijästä esimerkiksi käyttämällä Googlea. Kuvassa 6 näkyvät Play-kaupasta ladattavan Facebook-sovelluksen vaatimat oikeudet.



Kuva 6: Google Play -kaupasta ladattavan sovelluksen oikeudet (18).

Sovellusten pääsyoikeudet tarkoittavat sovellukselle annettua oikeutta käyttää mobiililaitteen toimintoja tai tietoja omaan tarkoitukseensa. Oikeuksia voidaan antaa esimerkiksi laitteen GPS:n käytölle, jolloin sovellus voi tarkistaa mobiililaitteen sijainnin. Tämä on pakollinen lähes kaikissa kartta- ja paikannussovelluksissa. Myös pääsy arkaluontoiseen tietoon voidaan sallia, kuten laitteen historiatietoihin tai kontaktilistaan pääsy. Näihin tietoihin ei pitäisi päästää muita kuin luotettuja sovelluksia. Esimerkiksi suosittu WhatsApp-sovellus, joka on tehokas tekstiviestit korvaava pikaviestin, hyödyntää laitteen kontaktilistaa lisäämällä sen avulla automaattisesti yhteydet listasta löytyvien henkilöiden kanssa. Tämä vähentää huomattavsti käyttäjän vaivaa lisätä kaikki kontaktit manuaalisesti sovellukseen. (19.) Kuvassa 7 on Google Play -kaupan mobiiliversion etusivu.



Kuva 7: Google Play -kaupan etusivu. Kaupasta saatavat sovellukset ja pelit on jaoteltu omiin luokkiinsa. Kaupasta saa myös musiikkia, elokuvia ja e-kirjoja.

Android-käyttöjärjestelmä mahdollistaa sen helpon muokkaamisen omien mieltymysten mukaiseksi. Käyttäjä voi helposti muuttaa Androidin ulkonäköä monella eri tavalla esimerkiksi lataamalla saatavilla olevia teemoja tai taustakuvia. (20.) Käyttäjät voivat lisätä mobiililaitteensa päänäytölle pienoisohjelmia eli Widgetejä, joista voi nähdä nopeasti erilaista tietoa, kuten sää- tai pikaviestintietoa. Pienoisohjelma toimii samalla myös pikakuvakkeena varsinaiselle sovellukselle (21), jolloin sitä koskettamalla avautuu enemmän tietoa. Googlen kehittämä Android tarjoaa valmiiksi asennettuja, Googlen omia ohjelmia mobiililaitteissaan, kuten Google Maps, Gmail ja Google Chrome. Näitä sovelluksia ei kuitenkaan ole pakko käyttää, ja ne voi helposti korvata haluamallaan sovelluksella (22). Valmiiksi asennettuja sovelluksia ei kuitenkaan pysty poistamaan ilman puhelimen ”roottaamista” eli kaikkien toimintojen hallintaa.

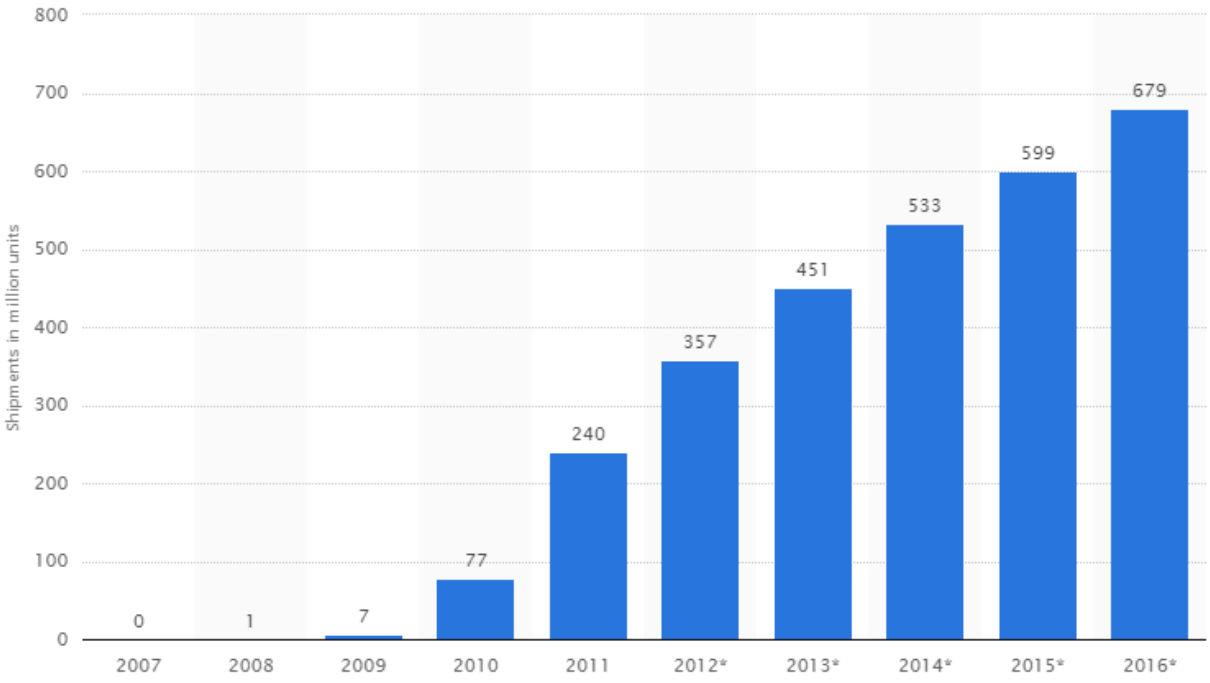
4.3 Android-laitteet

Android ei ole käytössä pelkästään tableteissa ja älypuhelimissa, vaan se on jo levinnyt käyttöön myös esimerkiksi televisioihin ja pelikonsoleihin sekä autoihin (23). Android ohitti Nokian Symbian-käyttöjärjestelmän markkinaosuuden vuonna 2010 (24), ja tällä hetkellä se on suosituin mobliilikäyttöjärjestelmä. Toisena on Applen iOS. Käytetyimmät käyttöjärjestelmät neljän vuoden ajalta on esitelty kuvassa 8.

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
Q4 2014	76.6%	19.7%	2.8%	0.4%	0.5%
Q4 2013	78.2%	17.5%	3.0%	0.6%	0.8%
Q4 2012	70.4%	20.9%	2.6%	3.2%	2.9%
Q4 2011	52.8%	23.0%	1.5%	8.1%	14.6%

Kuva 8: Käytetyimpien mobiilikäyttöjärjestelmien markkinaosuuksien viimeisten vuosineljännesten tiedot vuosilta 2011-2014 (25).

Android-laitteiden kysyntä on kasvanut räjähdysmäisesti vuodesta 2009, eikä se näyttäisi hidastuvan vielä vähään aikaan. Kuvassa 9 nähdään laitteiden viennin kasvu.



Kuva 9: Android-laitteiden viennin kasvu (27).

Android-käyttöjärjestelmien päivitys muiden valmistajien laitteille on tuottanut ongelmia käyttäjille ja laitevalmistajille, koska he joutuvat räätälöimään uuden version jokaisen Android-laitteen laitevaatimusten mukaan. Tämä tuo lisäkustannuksia laitevalmistajille, ja usein vain uudemmat laitteet saavat uusimman Android-version ja vanhemmat laitteet jätetään päivittämättä. Googlen omat Nexus-sarjan laitteet tukevat paremmin käyttöjärjestelmäpäivityksiä, koska Google kehittää Android-versiot juuri näiden laitteiden vaatimusten mukaisesti. (26.)

5 Ohjausjärjestelmän mekaniikka

Luku 5 käsittelee insinöörityöprojektissa käytettyjä komponentteja ja sitä, miten ne yhdistetään ohjausjärjestelmän prototyypin kanssa sähkökiukaaseen sekä erilaisia kytkentöjä.

Projektissa ei ollut käytössä oikeaa sähkökiuasta, vaan kiuasta simuloitiin vanerista rakennettuna sähkökiukaan jäljitelmänä. Testilaitteessa oli samanlaiset osat kuin tavanomaisessa sähkökiukaassa, joten se kävi yhtä hyvin testaukseen kuin oikeakin sähkökiuas. Prototyypin testaaminen on myös kätevämpää ja turvallisempaa tehdä sisätiloissa kuin saunassa, missä on märkää ja kuuma. Vanerista tehty sähkökiukaan jäljitelmä on huomattavasti helpompi siirtää kuin oikea sähkökiuas.

5.1 Suunnittelu

Komponenttien suunnittelua rajoitti merkittävästi vaihtoehtojen vähäisyys ja prototyyppiä varten tarvittavien komponenttien hinta-laatusuhde. Huomioon piti ottaa muun muassa saunatilojen kosteus ja kuumuus sekä ympäristö, johon on haastavaa suunnitella elektroniikkaa. Komponenttien piti olla veden- ja lämmönkestäviä. Tällaiset elektroniikkakomponentit maksavat yleensä paljon, ja tässä projektissa oli käytössä hyvin minimaalinen, omakustanteinen budjetti. Todennäköisesti saunakiukaassa ei tarvitsisi olla muuta vedenpitävää kiinni kuin servomoottorit. Servomoottoreihin yhdistetään riittävän pitkät vedenpitävät kaapelit, ja itse palvelin sijaitsee toisessa tilassa, jossa ei tarvitse huomioida vaikeita olosuhteita. Lämpömittarin tulisi olla

saunatiloissa, joten sen tulee olla myös vedenkestävä. Tämän vuoksi lämpötilasensoriksi valittiin vedenpitävä digitaalinen DS18B20-lämpötilasensori.

Projektin alussa tehtiin päätös toteuttaa pelkkä prototyyppi, jossa tätä haasteellista ympäristömuuttujaa ei tarvitse ottaa huomioon. Prototyypissä on vanerista tehty kiuasta simuloiva neliskanttinen laatikko. Vaneriin kiinnitettiin myös sähkökiukaassa olevat säätövivut, joihin servomoottorit asennetaan kehikolla. Servomoottoreiden täytyy olla kiinnitettyinä kehikkoon ja kalibroituina oikeaan asentoon, jotta ne liikkuisivat aina saman verran ja samaan asentoon tietyissä lämpötiloissa.

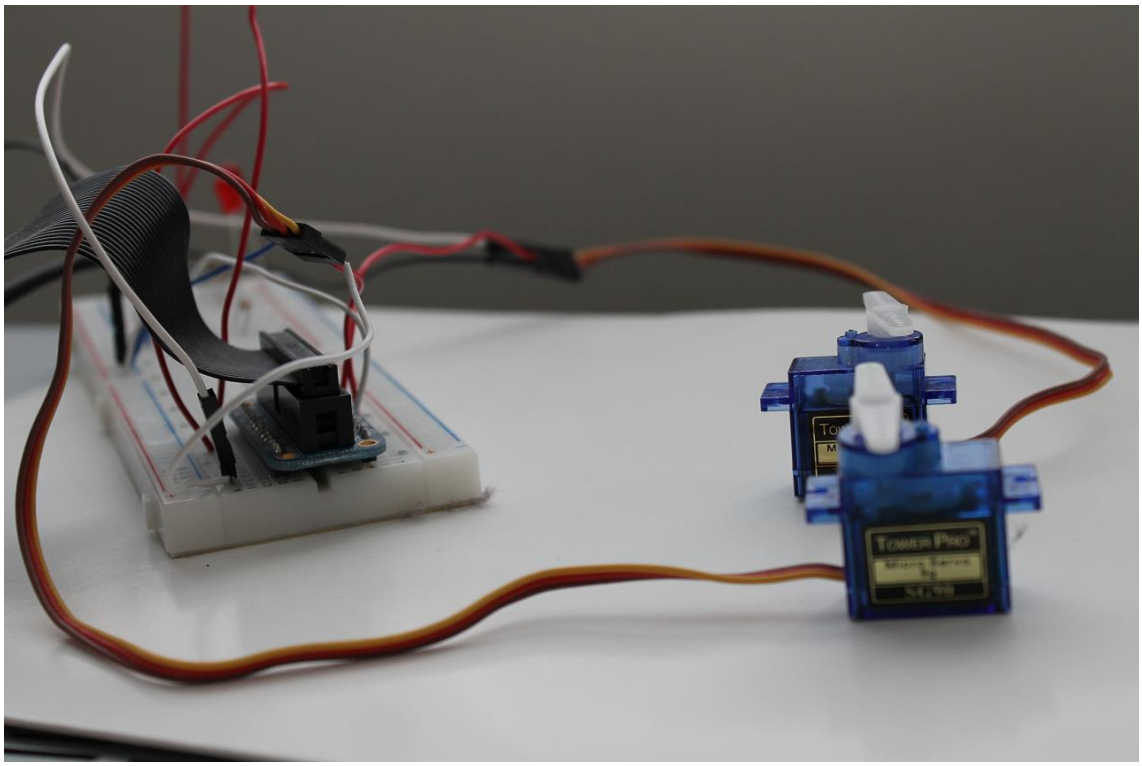
5.2 Komponentit

Ohjausjärjestelmän mekaniikka koostuu peruskomponenteista, joita saa elektroniikkakaupoista. Projektissa käytössä oli vedenkestävä digitaalinen DS18B20-lämpötilasensori, jolla pystyy seuraamaan saunan lämpötilaa reaaliaikaisesti. Saunasovellus ilmoittaa käyttäjälle, kun sauna on oikean lämpöinen ja sinne voi mennä. Lämpötilasensorilla on myös toinen erittäin tärkeä käyttötarkoitus. Se on tämän saunasovelluksen turvallisuuden kannalta välttämätön. Saunaohjelmaan toteutettiin lämpötilasensoria hyväksikäyttäen turvamekanismi. Lämpötilan noustessa määritellyn asteluvun verran saunan lämmityslämpötilan yli, palvelinpuoli sammuttaa saunan saman tien. Palvelin kääntää ensin molemmat säätövivut nolla-asentoon, ja sen jälkeen se myös sammuttaa laitteiston.

Servoina mukana oli kaksi kuvassa 10 näkyvää SG90-servomoottoria, joita käytettiin sähkökiukaan säätövipujen säätämiseen. Toinen servo säätää saunan ajastusta ja sitä, kuinka kauan sauna on päällä.

Kuva 10: SG90-servomoottori ja sen kiinnitysruuveja.

Kuvassa 11 ovat molemmat servot kytkettyinä kytkentälevylle. Toisesta säätövivusta säädellään saunan lämpötilaa.

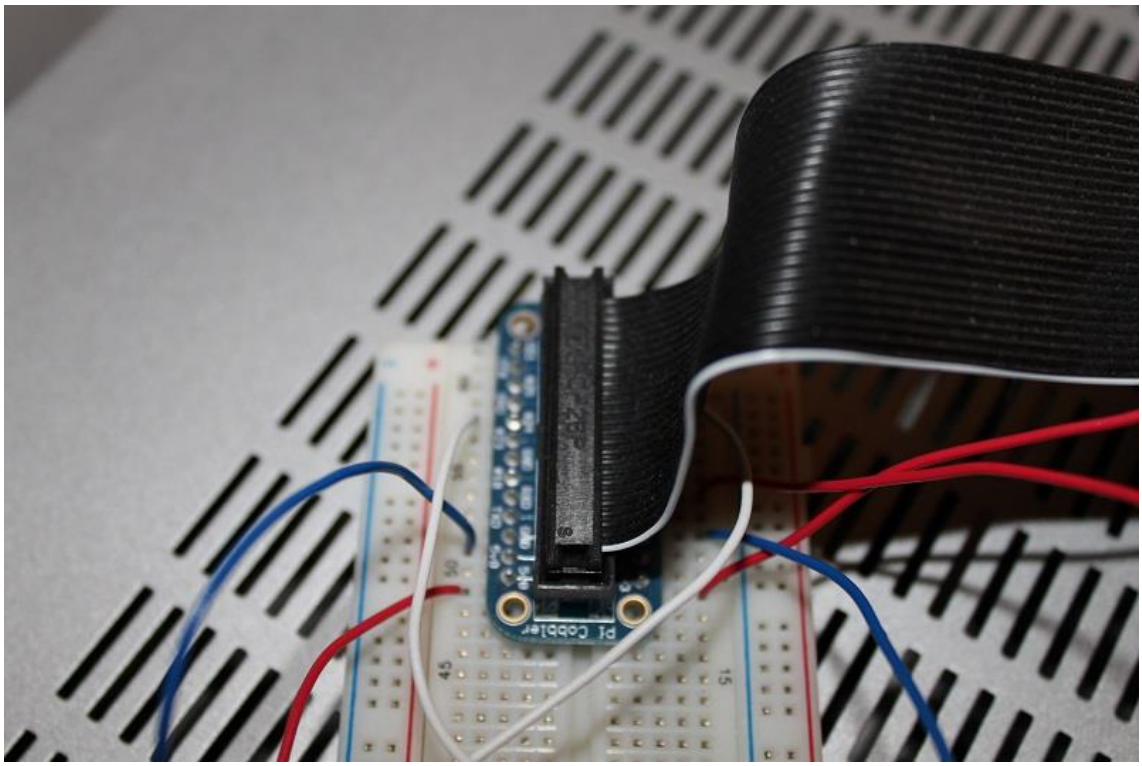


Kuva 11: Servot kytkettiin kytkentälevyyn.

SG90-servomoottorit valittiin niiden pienen ja kompaktin koon takia. Niissä on myös todella paljon tehoa pieneen kokoonsa nähden. Prototyypin kanssa testaukseen ne olivat juuri oikeanlaiset, eikä tehokkaampia servoja tarvittu. Servot kiinnitettiin telineeseen, joka pitää niitä paikallaan. Teline kiinnitettiin sähkökiukaaseen, niin että servot osuvat mekaanisiin säätövipuihin. Säätövivut liikkuvat servojen liikkeen mukana.

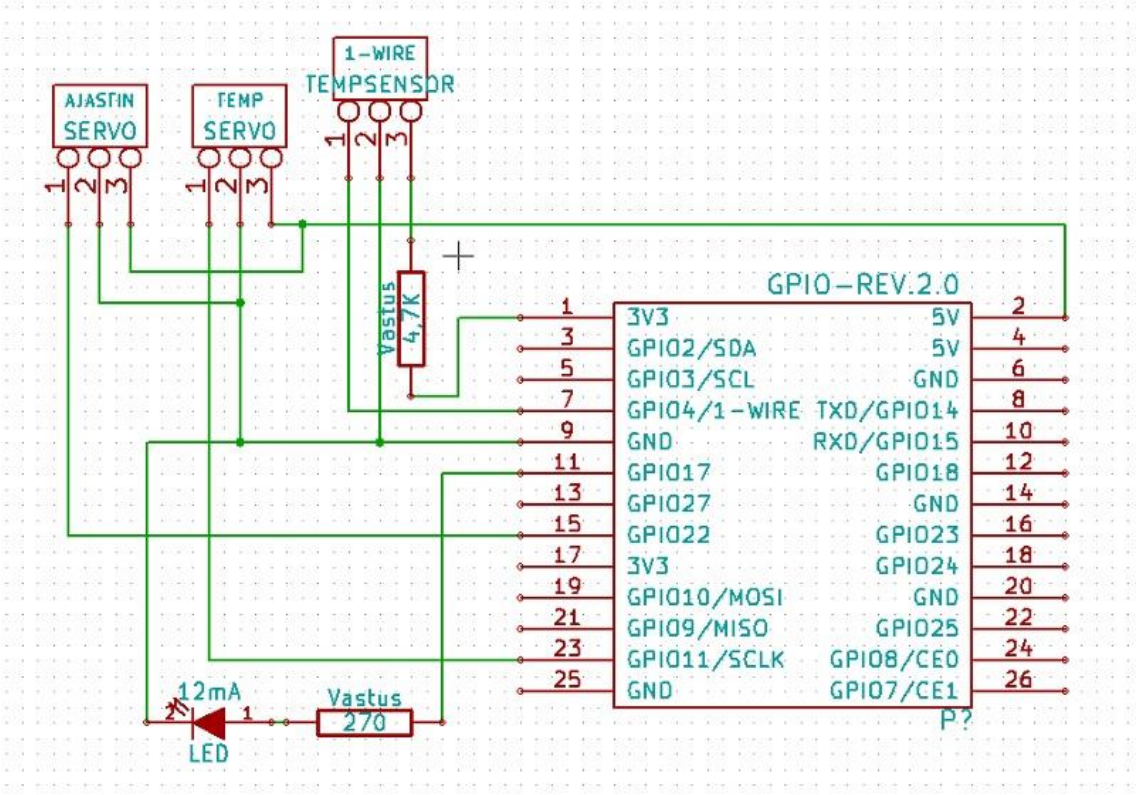
Merkkivalona käytössä oli tavallisia keltaisia LED:ejä. LED:ien käyttövirta on 12 mA, joten niille kulkevaa virtaa rajoitettiin 270 ohmin vastuksella. LED:t saavat virtansa 3,3 V:n GPIO-pinnistä. Yhteensä koko Raspberry saa virtaa noin 2 A:n verran. Se saa virtansa tavallisesta microusb-liittimestä, joka on kytketty 230 V:n verkkovirtaan.

Servot kytkettiin Rasperry Piin GPIO-pinneihin. Raspberryn GPIO-pinneihin kiinnitettiin myös 26-porttinen IDE-kaapeli, jolla jokainen pinni vedettiin isommalle kytkentälevylle. Tämä mahdollisti helpommat kytkennät ja testaukset komponenteille, joka näkyy kuvassa 12.



Kuva 12: IDE-liitäntä Raspberrystä kytkentälevylle.

Kytkennät tehtiin lämpötilaa säätelevästä servosta yhteiseen maapinniin (GND), servon +5 V:n jännite kytkettiin +5 V:n GPIO-pinniin ja servon data input kytkettiin GPIO-pinniin numero 11. Ajastusta säätelevä servo kytkettiin muilta osin samoihin pinneihin kuin lämpötilaa säätelevä paitsi data input, joka kytkettiin GPIO-pinniin numero 22. Lämpötilasensori kytkettiin +3,3 V:n GPIO-pinniin ja yhteiseen maa-pinniin (GND). Lämpötilasensorin datalinja kytkettiin sisään tulevaan GPIO-pinniin numero 4. GPIO-pinnistä numero 4 luetaan lämpötilasensorin lähettämää 9-bittistä dataa. Molemmat ohjelmistot käyttävät tätä samaa tietoa hyväkseen. Kytkennät on esitetty kuvassa 13.



Kuva 13: Mekaanisten osien kytkentäkaavio.

6 Etäohjattava saunajärjestelmä

Palvelinpuolen arkkitehtuurin suunnittelu selventää, millaista ohjelmakoodia palvelimen ohjelmiston täytyy sisältää toimiakseen suunnitellun käyttötarpeen mukaisesti. Säädöt ja ajastukset palvelimeen tehtiin Harvian sähkökiukaiden kuvausten mukaisesti (23). Palvelinpuolen toteutuksen ja käyttöönoton kanssa ilmeni monia erilaisia ongelmatilanteita.

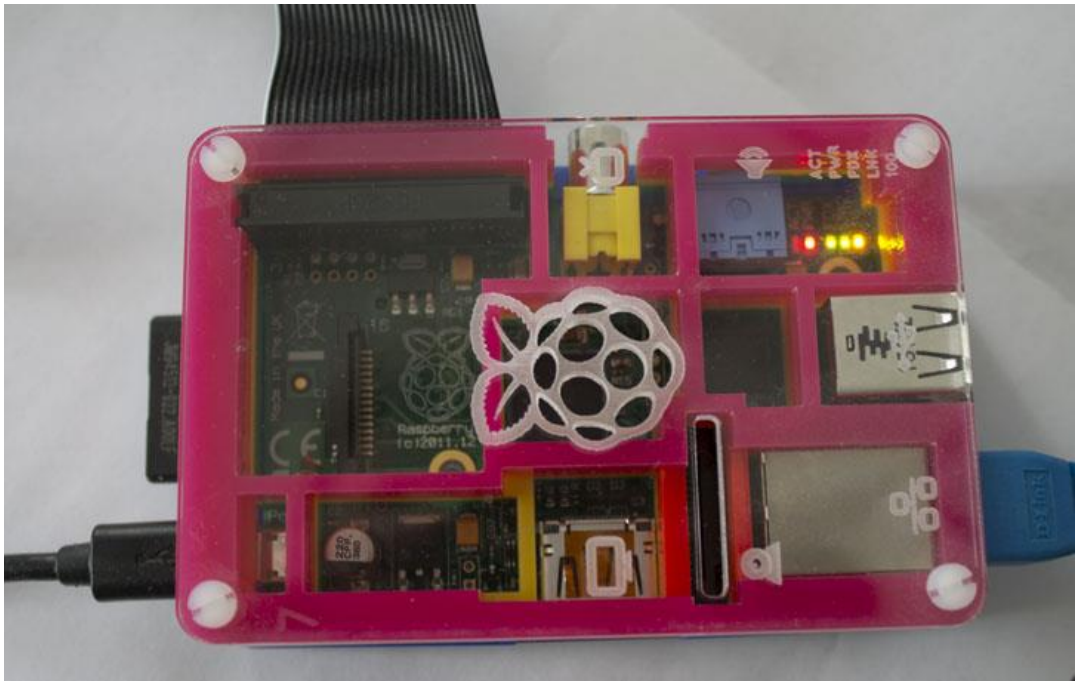
6.1 Palvelinpuolen toiminta

Etäohjattavan saunajärjestelmän palvelinpuolen ohjelmakoodin toiminta perustuu pitkälti siihen, että etäyhteydellä muodostetaan yhteys palvelimeen ja yhteyden muodostamisen jälkeen siellä voidaan suorittaa haluttuja toimintoja. Palvelimen käskytys hoidetaan suurimmaksi osaksi siirtämällä sinne suojattua SFTP-protokollaa käyttäen tiedostoja, joissa käskyt ovat. Palvelin lukee tiedostot, joissa käskyt ovat ja

toimii niiden mukaisesti. Virhetilanteen sattuessa palvelin ilmoittaa lokitiedostoon virheestä.

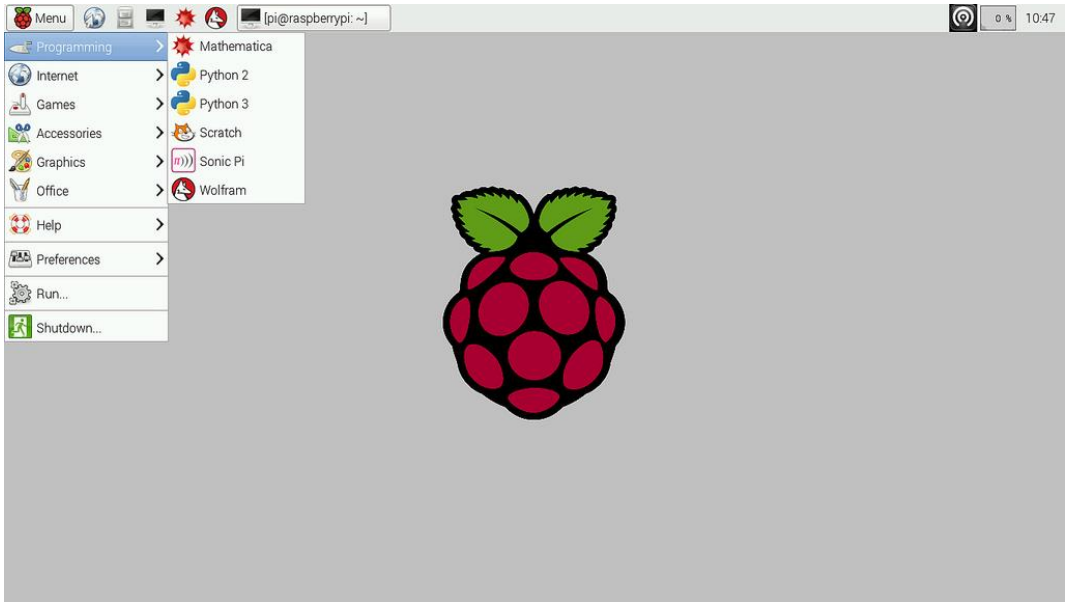
Kuva 14: Raspberry Pi ilman suojakoteloa.

Palvelin rakennettiin Raspberry Pi model B:n yhden piirilevyn tietokoneelle, joka on hieman luottokorttia isompi piirilevy. Siihen on lisäksi saatavilla erilaisia suojakoteloita, jotka suojaavat piirilevyn pintaa kolhuilta ja mahdollisilta vaurioitumisilta. Kuvassa 14 piiri ilman suojakoteloa ja kuvassa 15 suojakotelon kanssa.



Kuva 15: Raspberry Pi suojakotelon kanssa.

Raspberryyn asennettiin Raspberry Pin kotisivuilta saattavissa oleva käyttöjärjestelmä Raspbian, joka pohjautuu Linux Debian -käyttöjärjestelmään. Rasbian on räätälöity sopivaksi toimimaan erityisesti Raspberryssa. Raspbian, joka näkyy kuvassa 16, valittiin käyttöjärjestelmäksi, koska siinä on paljon sellaisia ohjelmistopaketteja, joita tässä projektissa tarvittiin.

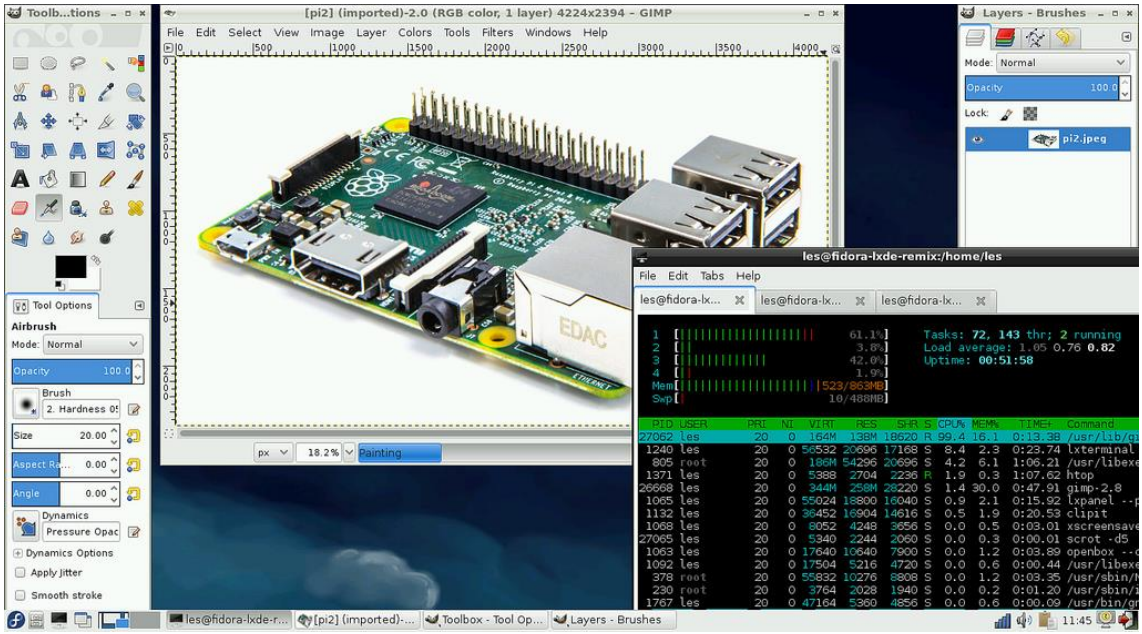


Kuva 16: Raspbian-työpöytä.

Muita vahvoja vaihtoehtoja Raspbianille olivat Ubuntu Mate, joka ei valitettavasti tässä vaiheessa tukenut vielä projektissa välttämättömiä GPIO-pinnejä. Toinen valittavissa ollut käyttöjärjestelmä oli Fedora, jossa ei ole suuria puutteita tai suorituskykyongelmia Raspbianiin verrattuna. Ubuntu MATE ja Fedora näkyvät kuvissa 17 ja 18.

Kuva 17: Ubuntu MATE -työpöytä.

Raspbian kuitenkin otettiin käyttöön myös siksi, että sitä suositellaan ensimmäiseksi käyttöjärjestelmäksi Raspberryyn. Lisäksi se oli tarpeeksi tehokas ja yksinkertainen suorittamaan sille annettuja käskyjä. Se on saanut myös monissa arvosteluissa parhaat pisteet luotettavuuden ja helppokäyttöisyyden ansiosta. (27.)



Kuva 18: Fedora-työpöytä.

Palvelinpuolen toimimiseen halutulla tavalla riittivät pääominaisuuksiksi SSH-tuki, peruspalvelut ja Rasbian-käyttöjärjestelmän tuki graafiselle käyttöliittymälle. Mahdollisesti tulevaisuudessa kun komponentit vaihtuvat ja uusia ominaisuuksia tulisi lisää ohjausjärjestelmään. Olisi mahdollista ja ehkä myös tarpeellista vaihtaa käyttöjärjestelmää.

Raspberryssä käytössä oleva palvelin on Linux-pohjainen, joten siinä pätevät samat perusteet, kuin jokaisessa Linux-käyttöjärjestelmässä. Istunnot muodostetaan käyttäjätunnus-salasanaparilla. Mobiililaitteelle luotiin oma käyttäjätunnus, joka käyttää sitä ottaessaan SSH-yhteyden palvelimeen. Näin ollen on helppo lisätä useita laitteita samaan palvelimeen vain luomalla uusi käyttäjätunnus niille.

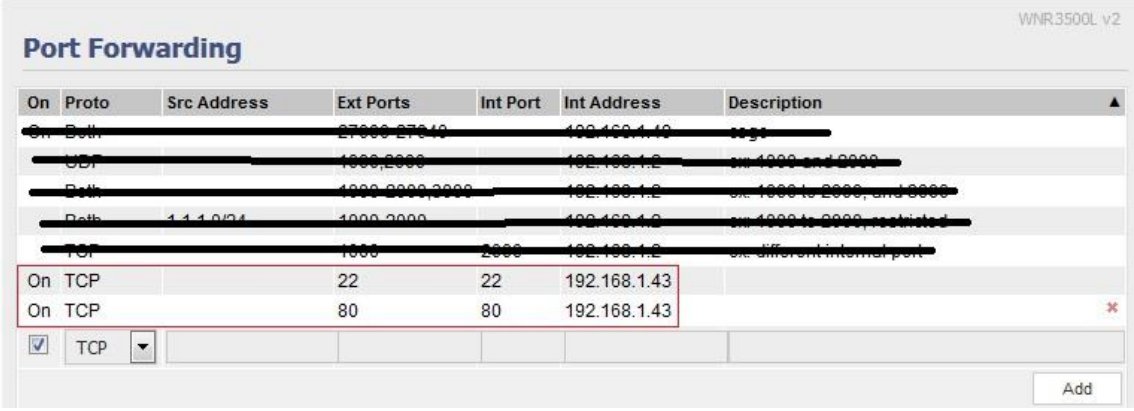
Etäyhteys palvelimeen

Etäyhteyden muodostaminen palvelimeen toteutettiin SSH-protokollaa käyttäen. Palvelin kytkettiin kotona omaan lähiverkkoon Ethernet-kaapelilla. Palvelimella on siis oma IP-osoite sisäverkossa, ja se on esitetty kuvassa 19.

```
~ $ ifconfig eth0 |grep "inet addr"
inet addr:192.168.1.43  Bcast:192.168.1.255  Mask:255.255.255.0
```

Kuva 19: Palvelimen sisäverkon IP-osoite.

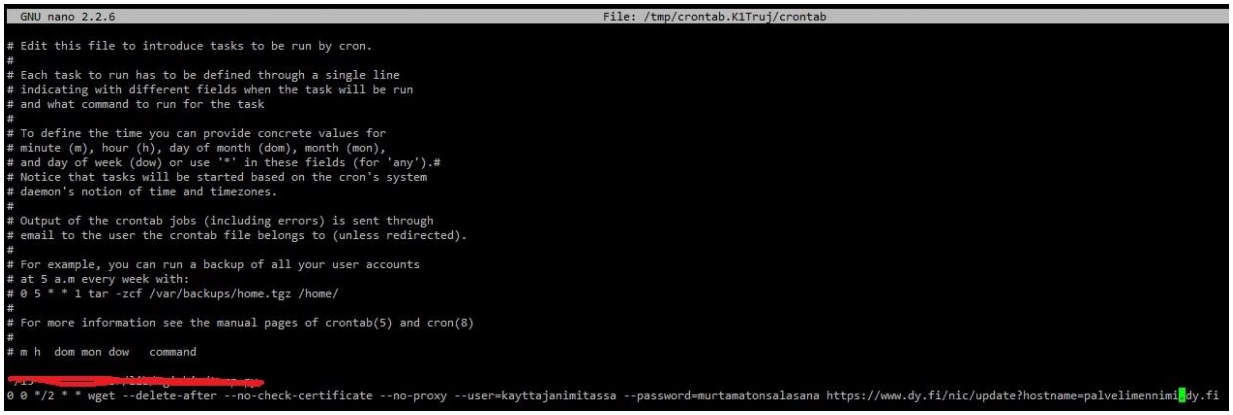
Tämä ei pelkästään riitä siihen, että palvelimeen pystyy muodostamaan etäyhteyden ulkomaailmasta, sillä välissä on myös reititin, joka hoitaa reitityksen ulkomaailmaan. Reitittimen asetuksiin jouduttiin tekemään muutoksia, jotta se osaa ohjata mahdollisen palvelimeen kohdistuneen liikenteen suoraan palvelimelle. Tämä ratkaistiin porttien uudelleenohjauksella, josta esimerkki kuvassa 20. Porttien uudelleenohjaamisen jälkeen ei ollut vielä mahdollista muodostaa yhteyttä palvelimen sisäverkon IP-osoitteella, vaan yhteys tulee muodostaa reitittimen ulkoiseen IP-osoitteeseen. Reititin osaa porttien uudelleenohjauksen ansiosta ohjata palvelimelle suunnatun liikenteen suoraan palvelimelle. Tällöin yhteyden muodostaminen onnistuu.



Kuva 20: Reitittimen porttien uudelleenohjaus.

Kuluttajalaajakaistoja tarjoavat operaattorit saattavat vaihtaa IP-osoitteita tietyin väliajoin, ja tämän vuoksi piti keksiä tapa muodostaa etäyhteys palvelimeen, niin ettei jokaisella kerralla tarvitsisi ottaa uutta IP-osoitetta selville. Tätä varten käyttöön otettiin DNS-nimipalvelujärjestelmä. Tunnukset luotiin ilmaiselle suomalaiselle sivustolle dy.fi, joka tarjoaa palvelua palvelimen IP-osoitetteen tallessapitoon ja mahdollistaa yhteydenoton siihen helpommin muistettavalla nimellä. Etäyhteys muodostetaan tämän jälkeen nimeen, joka palvelimelle on annettu, ja dy.fi-palvelu ohjaa palvelimen julkiseen IP-osoitteeseen. (28.)

Palvelimelle lisättiin automaattinen komento, joka suoritetaan kahden päivän välein päivittämään IP-osoite vastaamaan haluttua nimeä dy.fi-palveluun. Riskinä on, että IP-osoitteen vaihto osuu edellä mainitun kahden päivän väliin, jolloin komento täytyy käydä käsin suorittamassa. Kuvassa 21 näkyvät Crontabin asetukset palvelimelta.



Kuva 21: IP-osoitteen päivitys Crontabissa.

Skriptit palvelimella

Palvelimella on toiminnassa erilaisia apuskriptejä, jotka pitävät palvelimen toiminnon mahdollisimman tehokkaana ja sen ylläpitämisen helppona. Palvelimella on käytössä käytön valvontajärjestelmä, joka kerää erilliseen lokitiedostoon dataa. Tiedosto voi kasvaa isoksi ja vaikealukuiseksi ajan kuluessa. Sitä varten luotiin skripti, joka tarkistaa tiedoston koon ja tarpeen mukaan poistaa turhat rivit jättäen viimeiset kymmenen riviä talteen. Tämä skripti suoritetaan joka päivä kello 12, ja näin vain viimeisimmät käytön valvontajärjestelmän merkinnät ovat näkyvissä ja tiedoston koko pysyy pienenä. Varsinaista myyntiversiota tai edes pitkäaikaista testaamista varten olisi hyvä, ettei skripti poistaisi vanhoja lokitietoja vaan siirtäisi ne toiseen tiedostoon päivämäärällä merkittyinä. Näin olisi mahdollisuus vikatilanteiden tarkistamiseen pitkänkin ajan jälkeen sekä toiminnan seuraamiseen. Tiedostoja voisi säilyttää viikon ajan, minkä jälkeen ne poistettaisiin. Kuvassa 22 on lokitiedoston puhdistusskripti.

```
#!/bin/bash

FILE=/home/homectrl/scanner.log
TMPFILE=/home/homectrl/tmp.log

FILESIZE=$(stat -c%s "$FILE")
if [ $FILESIZE -gt 10000 ] ; then
    cat $FILE > $TMPFILE
    tail $TMPFILE > $FILE
    rm -rf $TMPFILE
fi
```

Kuva 22: Lokitiedoston puhdistusskripti

Käytössä on myös skripti, joka seuraa pääprosessin toimintaa ja tarkistaa tietyin väliajoin prosessin toiminnan. Kuvassa 23 on esitetty tämä skripti. Skriptin havaitessa, että prosessi ei ole käynnissä, se käynnistää sen välittömästi uudestaan. Prosessin ollessa käynnissä skripti tulostaa /dev/nulliin grepin tuloksen eli käytännössä ei tee mitään. Skripti on ajastettu tarkistamaan prosessin tilan joka viides minuutti.

```
#!/bin/bash

process=servoIOtesti
makerun="/home/homectrl/bin/servoIOtesti"

if ps ax | grep -v grep | grep $process > /dev/null
then
  exit
else
  $makerun &
fi
```

Kuva 23: Pääprosessin ”elossa pito” -skripti.

6.2 Palvelinpuolen ohjelmisto

Palvelinpuolen ohjelmisto toteutettiin C-kielellä ja osittain myös C++-kielellä. Palvelimella toimii yksi niin sanottu pääohjelma, joka kutsuu muita ohjelmia sitä mukaa, kuin niitä tarvitaan. Pääohjelman toimintaperiaate on hyvin yksinkertainen. Se odottaa ulkopuolelta tulevaa etäyhteyttä ja on niin sanotussa waiting-tilassa sitä ennen. Näin ollen ei turhaan käytetä tietokoneen resursseja, ennen kuin yhteys palvelimeen on muodostettu. Tämä toteutus on hyvin suosittu asiakas-palvelinmalli, joka käyttää TCP-yhteyksiä hyväkseen.

Palvelin ja asiakasohjelmisto juttelevat keskenään socketien kautta. Socketit ovat eräänlaisia kytköksiä tietokoneiden tai prosessien välillä. Tämä mahdollistaa sen, että palvelin ja asiakasohjelmisto voivat keskustella keskenään internetin välityksellä. Kun pääohjelma huomaa yhteydenoton asiakaspuolen ohjelmalta, se jää odottelemaan asiakkaan käskyjä. Asiakasohjelma pystyy ilmoittamaan palvelimelle lähettämästään tiedostosta, jossa käskyt saunan lämmitykseen ovat. Tämän jälkeen palvelin kirjoittaa ”putkeen” tiedon asiakkaan lähettämästä tiedostosta. Tätä putkea kutsutaan FIFO:ksi tai ”named pipeksi”. FIFO:n toista päätä lukee erillinen skannausohjelma, joka yrittää lukea FIFO:sta tietoa siitä, onko tiedostoja saapunut. Kun skannausohjelma lukee

FIFO:sta tiedoston saapuneen hakemistoon, se käy skannaamassa hakemiston. Myös tämä skannausohjelma on waiting-tilassa, joten tietokoneen resurssit eivät ole turhaan käytössä.

Asiakasohjelma voi halutessaan myös tiedustella palvelimelta lämpötilatietoja. Asiakas lähettää palvelimelle käskyn, että haluaa vastaanottaa palvelimelta lämpötilatietoja, jolloin palvelin lähettää ne asiakkaalle. Palvelin luo tässä tapauksessa lapsiprosessin, joka käy suorittamassa lämpötilan tuottavan ohjelman, minkä jälkeen lapsi palauttaa prosessin tarjoaman tiedon asiakasohjelmalle. Palvelin- ja lapsiprosessi keskustelevat keskenään myös tätä FIFO:a hyväksi käyttäen. Lämpötilaohjelman suorittamiseen luodaan aina erillinen lapsiprosessi, jotta palvelin on aina valmiina ottamaan uusia yhteyksiä vastaan asiakasohjelmilta.

Kun asiakasohjelma ilmoittaa palvelimelle tiedoston saapuneen, käy skannausohjelma vielä tarkistamassa tiedoston oikeellisuuden. Tämä tarkistus on tarpeellinen siltä varalta, että tiedosto ei sisällä turhanpäiväistä dataa tai haittaohjelmia. Kaikki muut tiedostot, joissa on väärät tiedostopäätteet tai väärä sisältö, poistetaan automaattisesti hakemistosta. Jatkokäsittely lukee tiedoston sisältämät käskyt ja poistaa tiedoston sen jälkeen hakemistosta. Tällä tavalla vältetään myös palvelimen tilan loppuminen kesken.

Skannausohjelma etsii tiedostoja joiden tiedostopääte on ilmoitettu taulukossa 2.

Taulukko 2: Palvelimen toiminta tiedostopäätteen perusteella.

Toiminta palvelimella	Tiedostopääte
Laite kytketään päälle	.on
Laite kytketään pois päältä	.off
Laite ja prosessi sammutetaan kokonaan	.quit

Kun skannausohjelma löytää hakemistosta tiedoston tietyllä tiedostopäätteellä, se toimii tiedostopäätteen mukaisella toiminnolla, jotka on esitetty taulukossa 1. Skannausohjelma käynnistää tiedostosta lukemisen vain, jos tiedostopääte on ”.on”-päätteinen. Palvelin sulkee laitteen, jos tiedostopääte on ”.off”, tai se sulkee koko laitteen ja prosessin, jos tiedostopääte on ”.quit”. Kaikissa muissa tapauksissa palvelin vain poistaa tiedoston.

Palvelimella suoritettavat eri toiminnot on toteutettu ohjelmakoodissa tiedostopäätteiden perusteella. Ohjelmakoodissa käytetään niin sanottuja flagejä, jotka ohjaavat ohjelman toteutusta. Ohjelman käynnistyessä ensi kertaa on OFF-flag

asetettu ”true”-tilaan, joka kertoo palvelimelle, että laite on pois päältä. Skannausohjelma odottaa, kunnes hakemistoon tulee ”.on”-päättellä oleva tiedosto, joka vastaavasti kytkee OFF-flagin ”false”-tilaan ja asettaa samalla ON-flagin ”true”-tilaan. Näin palvelin tietää, että laite on päällä, ja käynnistää tiedostosta lukemisen, minkä jälkeen ohjelma poistaa tiedoston.

Palvelin olettaa tiedostossa olevan tietynlaisessa formaatissa olevaa dataa, jossa käskyt tulevat. Tiedostossa käytetään arvopareja eli parametrejä, joilla asetataan halutuille tiedoille arvoja. Esimerkiksi jos sauna halutaan lämmittää 100 asteeseen, asetetaan tiedostoon parametri taulukon 3 mukaisesti. Tämän jälkeen tiedosto lähetetään palvelimelle, josta palvelin sitten lukee tiedoston sisällön. SAUNA_HEAT-parametrissa olisi tässä tapauksessa arvo ”100”, ja sen perusteella palvelin asettaisi saunan haluttuun lämpötilaan, joka on 100 astetta.

Taulukko 3: Palvelimen tunnistamat parametrit.

Palvelimen tunnistama parametri	Toiminto palvelimella
SAUNA_HEAT	Saunan tavoitelämpötila
SAUNA_ON_TIME	Saunan päälläoloaika
SAUNA_TIMER	Ajastettu saunan lämmitys

Valitettavasti Raspberry Pissa ei ole kunnollista tukea PWM-signaaleille eli pulssinleveysmodulaatiolle. Tämän takia vaihtoehdoiksi jäi joko itse kirjoittaa kirjasto, joka mahdollistaa pulssinleveysmodulaation, tai käyttää valmista kirjastoa. Valmis kirjasto pigpio.h lisättiin käyttöön, ja sitä käyttämällä on mahdollista tehdä mistä tahansa Raspberry Piin GPIO-pinnistä pulssinleveysmodulaatiota tuottava pinni. Kirjastossa on noin 10 000 riviä koodia, josta tässä työssä hyödynnettiin vain pulssinleveysmodulaatiota sisältävä osa. Kaikki muu koodi tuotettiin itse. (23.)

Palvelinpuolen ohjelmisto jaettiin standardikirjastojen lisäksi kolmeen erilliseen kirjastoon. Yhdessä kirjastossa on itse kirjoitettu C++-koodi, jolla pystyy valitsemaan, minkä tahansa Raspberry Pin GPIO-pinnin. Tämä kirjasto on nimeltään GPIOClass.h. Pinnille voi sen lisäksi antaa ominaisuudeksi olemaan joko input- tai output-pinni. Tämä määrittelee sen, tuottaako pinni signaalia ulospäin vai ottaako se signaalia sisäänpäin. Tätä kirjastoa käytetään muun muassa sytyttämään merkkivalo eli LED-valo saunan saavuttaessa halutun lämpötilan. Toisen kirjaston nimi on homecontroller.h. Se on C-kielellä kirjoitettu toteutus, jossa on sisällä kaikki hakemistojen skannaus, tiedostonpäätteiden selvitys, muistinvapautus ja saunasovelluksen toimivuuden kannalta tärkeimmät muut funktiot ja määritelmät.

Kolmas kirjasto on nimeltään sauna.h. Tämä kirjasto ohjaa servomoottoreita ja hakee lämpötilan saunasta. Kirjasto myös liittää pigpio.h-kirjaston ja käyttää sen tarjoamia pulssinleveysmodulaatioita. Kirjaston sauna.h tarkoitus on sisältää servojen asentojen määritelmät sekä funktiot servojen liikuttelemiseen. Servojen asentojen määritelmä tarkoittaa määrityksiä esimerkiksi servojen ääriasennoille, keskiasennolle ja minimiasennolle. Asennoille on laskettu tarkasti arvot sen takia, että ne vastaisivat mahdollisimman tarkasti sähkökiukaan säätönappuloiden asteikkoja.

6.3 Palvelinpuolen ongelmat

Eniten haasteita palvelinpuolella tuotti koodin saaminen sellaiseksi, että se ei aikaansaanut muistivuotoja. Muistivuodot kaatavat prosessin, kun se on varannut liikaa muistia käyttöjärjestelmästä. Näin ollen prosessin suorittaminen ei ole kovin stabiilia eikä muutenkaan tehokasta, jos se aiheuttaa muistivuotoja. Lisäksi sen joutuu käynnistämään useita kertoja uudestaan. Aluksi muistivuotojen löytäminen oli erittäin haastavaa, koska palvelinpuolen ohjelmisto on monimutkainen.

Ohjelmaa testattiin aluksi tulostamalla ruutuun sekä käytetyn että vapautetun muistin määrää, mutta tämä osoittautui ongelmalliseksi ilman virheenjäljitysohjelmaa. Muistivuotoja on vaikea jäljittää, ja useasti niiden korjaamiseen kuluu paljon aikaa. Rasperry Pille ei ollut saatavilla virheenjäljitysohjelmaa pääohjelmaa kirjoitettaessa ja testattaessa.

Nykyään Rasperry Pille on saatavilla virheenjäljitysohjelma nimeltä Valgrind, josta esimerkki kuvassa 24.

```
sewardj@phoenix:~/newmat10$ ~/Valgrind-6/valgrind -v ./bogon
==25832== Valgrind 0.10, a memory error detector for x86 RedHat 7.1.
==25832== Copyright (C) 2000-2001, and GNU GPL'd, by Julian Seward.
==25832== Startup, with flags:
==25832== --suppressions=/home/sewardj/Valgrind/redhat71.supp
==25832== reading syms from /lib/ld-linux.so.2
==25832== reading syms from /lib/libc.so.6
==25832== reading syms from /mnt/pima/jrs/Inst/lib/libgcc_s.so.0
==25832== reading syms from /lib/libm.so.6
==25832== reading syms from /mnt/pima/jrs/Inst/lib/libstdc++.so.3
==25832== reading syms from /home/sewardj/Valgrind/valgrind.so
==25832== reading syms from /proc/self/exe
==25832==
==25832== Invalid read of size 4
==25832==    at 0x8048724: BandMatrix::ReSize(int,int,int) (bogon.cpp:45)
==25832==    by 0x80487AF: main (bogon.cpp:66)
==25832==  Address 0xBFFFF74C is not stack'd, malloc'd or free'd
==25832==
==25832== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
==25832== malloc/free: in use at exit: 0 bytes in 0 blocks.
==25832== malloc/free: 0 allocs, 0 frees, 0 bytes allocated.
==25832== For a detailed leak analysis, rerun with: --leak-check=yes
```

Kuva 24: Valgrind-virheenjäljitysohjelman raportti.

Raspberry Pi käyttää ARM11-arkkitehtuurin prosessoria, eikä Valgrind tukenut ARM-arkkitehtuurin prosessoreita vielä toteutuksen aikaan. Ongelma kierrettiin kääntämällä ja suorittamalla ohjelmakoodi toisella Linux-koneella, jossa Valgrind toimi. Virheenjäljitysohjelman palautteesta pystyttiin seuraamaan ja paikallistamaan muistivuotoja, minkä jälkeen ongelmakohtiin lisättiin koodia, joissa vapautettiin kaikki varattu muisti. Muistinvapautuskoodista on esimerkki kuvassa 25. Lopuksi ohjelmakoodi saatiin korjattua täysin muistivuodottomaksi.

```
void free_array(char** array, size_t size){
    size_t c;
    for(c=0;c<size;c++){
        free(array[c]);
        array[c]=NULL;
    }
    free(array);
    array=NULL;
}
```

Kuva 25. Esimerkki palvelinpuolen pääohjelman yhdestä tärkeimmistä muistinvapautus-funktiosta.

Muistivuotojen lisäksi ongelmia aiheutti aluksi käytetty liian nopea hakemiston skannaus. Hakemiston skannaus oli asetettu suoritettavaksi ilman viivettä, jolloin prosessori oli koko ajan varattuna. Tämä aiheutti suuren prosessorikuorman turhaan, ja käyttöjärjestelmä tappoi prosessin nopeasti. Tämän selvittyä muutettiin koko pääohjelman toimintaperiaatetta. Hakemiston skannaus muutettiin suoritettavaksi ainoastaan silloin, kun tiedosto oli saapunut hakemistoon.

Servomoottoreiden ongelmana oli muun muassa saada pulssinleveysmodulaatiota toimimaan Raspberry Pissä. Haasteen toi se, ettei laitteistossa ollut valmista moduulia tuottamaan pulssinleveysmodulaatiota. Jos käyttäjä tahtoo käyttää pulssinleveysmodulaatiota Raspberry Pissä, se on tuotettava ohjelmistolla. Pulssinleveyssignaali yritettiin tuottaa aluksi itse, mutta se osoittautui erittäin vaikeaksi, eikä omatekoiseen signaaliin voinut luottaa täysin. PWM:n luontia varten löytyi kirjasto, jossa oli pulssinleveyssignaalin tuottaminen mistä tahansa Raspberry Pin GPIO-pinnistä. Kirjasto otettiin käyttöön, ja se toimi hyvin. Kirjasto vaati asennuksen käyttöjärjestelmään ja liittämisen ohjelmakoodiin.

6.4 Käytönvalvonta palvelimella

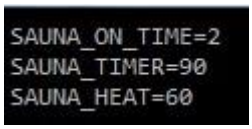
Kun käyttäjä antaa käskyjä palvelimelle, jokainen käsky tallentuu tiedostoihin, joista voidaan seurata tarkasti, mitä palvelimella on tapahtunut. Käytönvalvontaa suoritetaan Linux-käyttöjärjestelmään kuuluvalla ominaisuudella, joka tallentaa esimerkiksi yhteydenottoyritykset eri tiedostoon kuin esimerkiksi crontabissa suoritettavat komennot. Palvelimelle toteutettiin vielä oma käytönvalvontajärjestelmä, jolla pystytään seuraamaan tarkemmin palvelimen toimintaa. Kuvassa 26 on esimerkki tiedostosta, jossa näkyy käytönvalvonnan tulostuksia. Tiedostossa olevasta datasta pystyy päättelemään palvelimen toimivuuden ja ongelmatilanteet. Aikataulutuksen vuoksi haluttua hälytysjärjestelmää palvelimelle ei ehditty toteuttaa. Hälytysjärjestelmä tarkkailisi käytönvalvontatiedostoja ja palvelimen resurssinkäyttöä. Hälytysjärjestelmä lähettäisi hälytyksen sähköpostitse, jos se huomaisi jotain tavanomaisesta poikkeavaa. Tämä ominaisuus todennäköisesti toteutetaan palvelimelle myöhemmin.

```
Sat Feb 14 19:49:22 2015 Device turned ON=1
Sat Feb 14 19:49:24 2015 60  astetta asetettu saunaan! (SAUNA_HEAT)
Sat Feb 14 19:49:24 2015 60  minuuttia saunan lämpeämiseen. Servo ei vielä kytketty. (SAUNA_ON_TIME)
Sat Feb 14 19:49:24 2015 Tätä ominaisuutta ei ole vielä tehty.(SAUNA_TIMER)
Sat Feb 14 19:49:34 2015 Device turned OFF=0
Sat Feb 14 20:13:53 2015 Device turned ON=1
Sat Feb 14 20:13:55 2015 120  astetta asetettu saunaan! (SAUNA_HEAT)
Sat Feb 14 20:13:55 2015 100  minuuttia saunan lämpeämiseen. Servo ei vielä kytketty. (SAUNA_ON_TIME)
Sat Feb 14 20:13:55 2015 Tätä ominaisuutta ei ole vielä tehty.(SAUNA_TIMER)
Sat Feb 14 20:14:05 2015 Device turned OFF=0
Sun Feb 15 18:26:31 2015 Device turned ON=1
Sun Feb 15 18:26:32 2015 60  astetta asetettu saunaan! (SAUNA_HEAT)
Sun Feb 15 18:26:32 2015 60  minuuttia saunan lämpeämiseen. Servo ei vielä kytketty. (SAUNA_ON_TIME)
Sun Feb 15 18:26:32 2015 Tätä ominaisuutta ei ole vielä tehty.(SAUNA_TIMER)
Sun Feb 15 18:27:08 2015 Device turned OFF=0
```

Kuva 26: Palvelimella olevan lokitiedoston sisältöä.

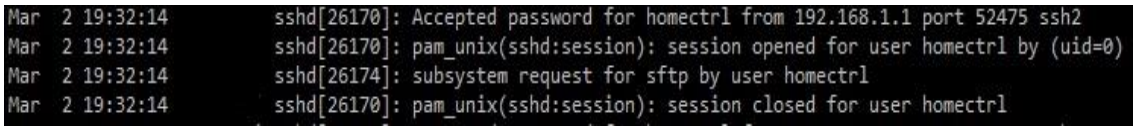
Yksi ongelmatilanne huomattiin lähetettäessä saunasovelluksella tiedostoa palvelimelle. Pääohjelman suorittaessa viimeisintä käskyä, jossa sauna haluttiin lämmittää 100-asteiseksi, ohjelma ei ottanut uusia käskyjä vastaan, ellei kiuasta sammutettu välissä. Haasteen tuo se, että lämpötilan tai saunan päälläoloajan muuttaminen ei käy saunan jo ollessa päällä. Kiuas joudutaan sammuttamaan välissä, ennen kuin sinne voidaan lähettää uudet käskyt. Tämän pienen ohjelmointivirheen vuoksi käytönvalvontajärjestelmään tehtiin viimeisen tiedoston sisällön tallennus. Se tallentaa viimeisen käskyn sisällön lastfile.log-nimiseen tiedostoon, joka on esitetty kuvassa 27. Palvelin kävisi tarkistamassa käytönvalvontatiedostoista edellisen käskyn aikaleiman ja vertaisi sitä lastfile.log-tiedoston aikaleimaan. Jos aikaleimat ovat

erilaiset, palvelin käy lukemassa tiedoston ja asettaa uudet asetukset saunaan. Tätä ominaisuutta ei kuitenkaan lähdetty toteuttamaan, koska se vaatisi palvelimen toiminnan muuttamista sen jo ollessa muuten toimiva. Tämä todennäköisesti toteutetaan palvelimelle myöhemmin.



Kuva 27: lastfile.log-tiedoston sisältö.

Jos käytönvalvontatiedoston on liian iso, sitä varten luotu skripti poistaa turhat datarivit tiedostosta. Tällä tavoin palvelimen levytilaa säästetään ja ylläpitoa helpotetaan minimoimalla käytön valvontadata. Vaihtoehtona olisi myös ottaa isot käytönvalvontatiedostot talteen pakkaamalla ja arkistoimalla ne verkkolevylle tai pilveen, josta niitä voitaisiin tarkastella myöhemmin. (28.)



Kuva 28: Palvelimen käytön valvontaa SFTP-yhteydestä.

6.5 Palvelimen komponentit

Etäohjattavan saunajärjestelmän palvelimen komponentteihin kuuluu fyysisten osien lisäksi monia eri ohjelmakomponentteja, jotka yhdistetään pääohjelmassa yhtenäiseksi kokonaisuudeksi. Palvelimen tärkein osa on minitietokone Raspberry Pi, jolla palvelin on käynnissä.

Raspberry Pissä on 16 gigatavun SD-muistikortti, jolla käyttöjärjestelmä ja muut tiedostot sijaitsevat. Raspberry Pissä on ARM-arkkitehtuurin 700 MHz:n prosessori. Näytönohjaimena toimii Broadcom VideoCore IV. Näytönohjaimessa on HDMI-liitäntä, joten tarvittaessa Raspberryyn voi yhdistää myös näytön. Välimuistia Rasbperryssä on 512 MB:n verran, joka on jaettu näytönohjaimen kanssa. USB-portteja on kaksi, ja niiden versio on 2.0. Verkkokortti on 10/100 Ethernet.

Tärkein ominaisuus, miksi Raspberry Pi valittiin etäohjattavan saunajärjestelmän palvelimen alustaksi, on se, että siinä on valmiina 26 GPIO-pinniä (General Purpose I/O), joita voi ohjelmoida joko lähettämään tai vastaanottamaan signaaleja. Etäohjattavassa saunajärjestelmässä käytössä on kumpikin ominaisuus. Lämpötila luetaan sensorista siten, että sinne lähetetään 5 V:n jännite. Sensori lähettää GPIO-pinniin takaisin signaalin, jossa se ilmoittaa lämpötilan. Tätä lämpötilatietoa käydään sen jälkeen lukemassa pinnistä. Saunan päälle- ja poiskytkemiseen käytetään myös GPIO-pinnejä. Palvelin lähettää käskyn pinniin, joka on kiinni servomoottorissa. Servomoottori on kytketty GPIO-pinniin ja saa pinnistä tietyn signaalin ja kääntyy tämän perusteella haluttuun asentoon. Tarkemmat piirin ominaisuudet on esitelty taulukossa 4.

Taulukko 4: Raspberry Pi Model B:n ominaisuudet.

Ominaisuudet	
Piiri	Broadcom BCM2835 SoC
Prosessori	700 MHz Low Power ARM1176JZFS
Näytönohjain	Kaksoisydin VideoCore IV® Multimedia Co-Processor. Tuki 1080p-resoluutiolle
Välimuisti	512 Mb SDRAM
USB 2.0	2
Videoulostulo	HDMI, komposiitti RCA
Ääniulostulo	3,5 mm stereo, HDMI
Massamuisti	SD/MMC/SDIO-korttipaikka
Verkkosovitin	10/100 Ethernet integroidun USB-hubin kautta
Muut liittimet	26-pinninen GPIO-liitin
Tehonkulutus	700 mA (3,5 W)
Paino	45 g
Käyttöjärjestelmät	Raspbian, Pidora, Arch Linux jne.

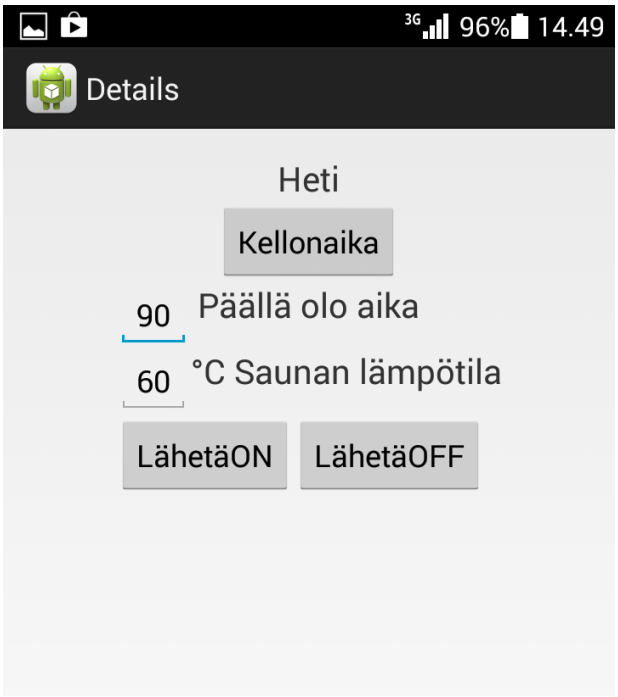
7 Android-saunasovellus

7.1 Yleistä

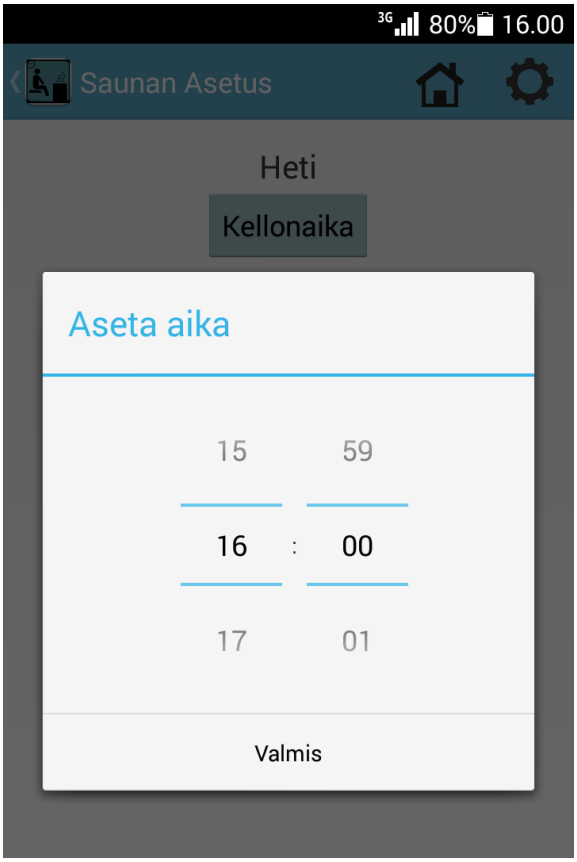
Insinöörityössä kehitettiin saunan kiuasta ohjataava Android-sovellus. Ohjelmointityökaluna käytettiin pääasiallisesti Eclipsen Java IDE:ä, jolla ei voi suoraan aloittaa tekemään Android-sovelluksia, vaan siihen pitää myös asentaa Android Developement Tools (ADT), joka tuo Eclipseen tarvittavat kirjastot ja muut ominaisuudet sovellusten tekemiseen. ADT mahdollistaa myös sovellusten

testaamisen Eclipsessä virtuaalisen Android-emulaattorin avulla, mutta tämä ominaisuus on yleensä turhauttavan hidas, joten on järkevämpää käyttää saatavilla olevia Android-puhelimia. Eclipse mahdollistaa myös helpon ongelmanratkonnan virhekonsolin logcatin ja debug-moodien avulla.

Sovellus toimii seuraavanlaisesti: Sovelluksen käynnistyessä on käyttäjän mahdollista siirtyä etusivulta joko kiukaan päälle laittamiseen tai asetuksiin tai tarkistaa saunan nykyinen lämpötila. Jos sauna halutaan laittaa päälle, voi käyttäjä valita sille omat asetukset, kuten kuvassa 29. Asetuksia ovat haluttu saunan lämpötila, saunan päälläoloaika ja kellonaika, jolloin sauna menee päälle. Kellonajan valinta näkyy kuvassa 30.



Kuva 29: Saunan asetusten valintaruutu. Ruudussa näkyvät arvot ovat oletusarvoja, ja niitä voi helposti muuttaa.



Kuva 30: Kellonaika, jolloin saunan halutaan menevän päälle, valitaan rullaamalla numeroita haluttuun suuntaan.

Tämän jälkeen asetetut tiedot lähetetään palvelimelle, joka hoitaa saunan lämmittämisen annetuilla tiedoilla. Sovellus ottaa kuitenkin talteen asetetun kohdelämpötilan, jota se pystyy vertaamaan palvelimelta saamaansa saunan nykyiseen lämpötilaan. Tätä lämpötilaa seurataan 15 minuutin välein siitä hetkestä, kun sauna menee päälle. Kun se saavuttaa kohdelämpötilan tai lähestyy 10 asteen tarkkuudella sitä, annetaan käyttäjälle ilmoitus saunan valmiudesta. Ilmoituksessa on myös mukana saunan senhetkinen lämpötila, jotta käyttäjä voi itse päättää, haluaako sinne mennä.

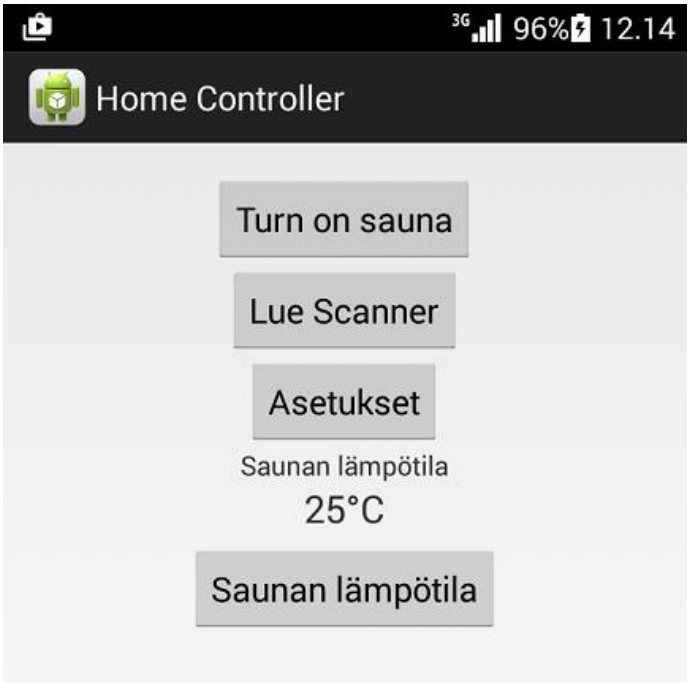
7.2 Lämpötila

Käyttäjän pitää pystyä seuraamaan saunan lämpötilaa helposti ja vastaanottamaan hälytyksiä mahdollisista vikatilanteista, joissa saunan lämpötila on korkea tai nousee ilman, että saunaa on hetkeen käytetty.

Sovelluksessa käyttäjä voi myös itse asettaa haluamansa kohdelämpötilan, johon

pyritään, kun sauna alkaa lämmetä. Jos käyttäjä ei aseta haluamaansa lämpötilaa, sovellus käyttää oletuslämpötilaa, joka on asetettu 60 asteeseen. Kun sauna saavuttaa kohdelämpötilan tai pääsee lähelle sitä, sovelluksen lämpötilaseuranta luo ilmoituksen käyttäjälle. Näin käyttäjä saa tietää, voiko saunaan vielä mennä.

Lämpötilaa voi seurata myös manuaalisesti sovelluksen päänäytöltä. Päävalikkoon on luotu nappi, jota painamalla saadaan käyttäjälle ajan tasalla olevaa lämpötilatietoa. Kuvassa 31 on manuaalisesti tarkistettu lämpötila.



Kuva 31: Saunan lämpötila tulee näkyviin pääruudulle napin painalluksella.

Kiukaan päässä olevalla sensorilla luetaan ilman lämpötilaa, joka kirjoitetaan tiedostoon palvelimella. Android-sovellus käyttää ainoastaan tämän tiedoston sisältöä lämpötilan seuraamiseen ja tarkistamiseen. Toinen vaihtoehtoinen lähestymistapa lämpötilan tarkistamiseen olisi ollut Cloud Sync, jonka avulla saisi tiedon päivittymään koska vain, ja se soveltuukin hyvin ohjelmiin, jotka tarvitsevat ympärivuorokautista yhteyttä tai päivittymistä. Kun lämpötilatiedosto on hyvin pieni ja tarkistamisväli rajattu, ei Cloud Sync ole välttämätön. Kaikki palvelimelta ladattavat tiedostot ovat kooltaan hyvin pieniä, ja ne tallennetaan mobiililaitteen muistikortille.

Koko ohjelma on toteutettu asynkronisella tehtävällä, joka ensin luo yhteyden palvelimeen, lataa lämpötilatiedoston, lukee siitä lämpötilatiedon, joka näkyy kuvassa 32, ja muuttaa sen luettavaksi.

```
7b 01 4b 46 7f ff 05 10 1d t=23687
```

Kuva 32: Tiedoston tärkein rivi, josta saadaan lämpötilatietoa käyttäjälle. Desimaalipiste lisätään kahden ensimmäisen numeron jälkeen eli tässä lämpötila on 23,687 astetta.

Tätä asynkronista tehtävää käytetään lämpötilan niin manuaalisesti tarkistamiseen kuin automaattiseen, tietyin väliajoin tapahtuvaan seurantaan. Kuvassa 33 on asynkronisen tehtävän käynnistävä metodi.

```
public int onStartCommand(Intent intent, int flags, int startId){

    asyncTask = new ReadTempTask(this);
    asyncTask.setOnResultListener(asynResult);
    asyncTask.execute();

    return START_NOT_STICKY;
}
```

Kuva 33. TempService-luokka, joka hoitaa ajastetut lämpötilan luvut ja suorittaa asynkronisen tehtävän kutsun.

Saunaa päälle laitettaessa käyttäjä voi asettaa haluamansa kohdelämpötilan, saunan päälläoloajan ja kellonajan, jolloin sauna menee päälle. Kaikki nämä valitut tiedot kirjoitetaan tiedostoon, jonka pääte on .on. Nämä tiedot luetaan palvelimella ja asetetaan saunaan. Tiedoston sisältö on esitelty kuvassa 34.

```
SAUNA_ON_TIME=0
SAUNA_TIMER=90
SAUNA_HEAT=60
```

Kuva 34: Tiedoston .ON sisältö. SAUNA_ON_TIME=0: aika minuutteina siihen, kun sauna laitetaan päälle. SAUNA_TIMER=90: aika, jonka sauna on päällä. SAUNA_HEAT=60: lämpötilaan, johon pyritään. Kaikki arvot tässä ovat oletusarvoja.

Koska valittua kohdelämpötilaa pitää pystyä tietyin väliajoin vertaamaan saunan varsinaiseen lämpötilaan, se tallennetaan SharedPreferences-tiedostoon. Lämpötilan tallennus on näytetty kuvassa 35. Tällä tavalla lämpötilatieto ei katoa tai nollaannu muuttujista ja oikea tieto saadaan aina vertailuun.

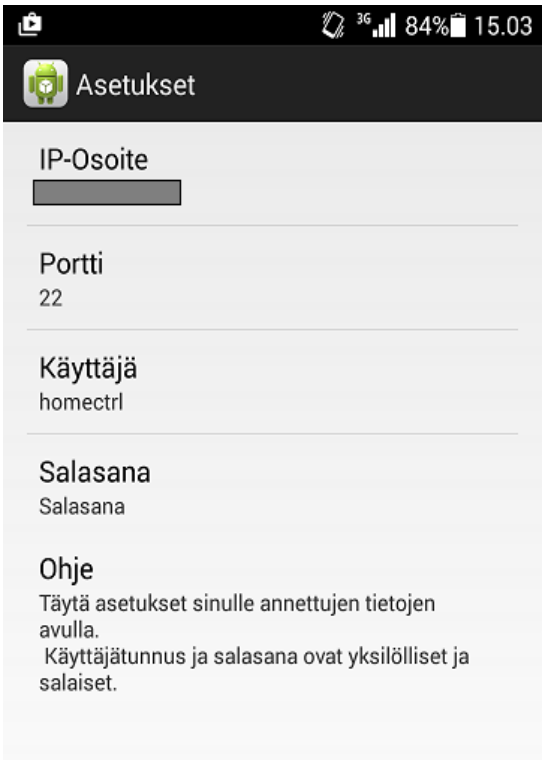
```
SharedPreferences prefs = PreferenceManager.getDefaultSharedPreferences(this);
SharedPreferences.Editor notifyText = prefs.edit();

//laitetaan haluttu lämpötila vertailua varten talteen
notifyText.putString("tempSet", heat.getText().toString());
notifyText.commit();
```

Kuva 35: Valittu kohdelämpötila laitetaan String-tyyppisenä talteen SharedPreferences-tiedostoon.

7.3 Asetukset

Asetukset-osuus on yksi tämän sovelluksen tärkeimpiä asioita, koska sen kautta pystyy asettamaan käyttäjää koskevat tiedot sovelluksessa suoraan kaikkiin luokkiin, jotka sitä tarvitsevat. Asetuksiin pääsee kätevästi käsiksi sovelluksen päänäytölle luodun napin kautta. Asetuksissa voi asettaa kaikki tiedot, joita tarvitaan, kun muodostetaan yhteys palvelinpäähän, kuten IP-osoite, käyttäjätunnus, porttinumero ja salasana. Asetuksiin on myös lisätty lyhyesti selittävä ohje, miten kaikki kohdat kuuluu täyttää ja mitä ne tarkoittavat. Asetusten näkymä on kuvassa 36.



Kuva 36: Asetukset-osio. IP-osoite on peitetty ja salasana on valmiiksi suojattu, niin että muut eivät sitä näe.

Kaikki halutut asetukset kirjoitetaan XML-tiedostoon, koska sitä on helpompi päivittää ja se on selkeälukuisempi. Tiedosto tallennetaan polkuun res/xml/ ja nimetään yleisen suosituksen mukaan aina ”preferences.xml”. Kuvassa 37 on asetukset sisältävän XML-tiedoston ensimmäinen kenttä.

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android" >

    <EditTextPreference
        android:capitalize="words"
        android:defaultValue="@string/ip_address_default"
        android:summary="@string/ip_address_summary"
        android:inputType="textCapWords"
        android:key="ip_text"
        android:maxLines="1"
        android:selectAllOnFocus="true"
        android:singleLine="true"
        android:title="@string/ip_address" />
```

Kuva 37: Asetuksetesimerkki. Ensimmäinen asetus: IP-osoite.

Asetukset-napista käynnistyy PreferencesActivity, joka puolestaan käynnistää PreferencesFragmentin. Fragmenteja käytetään aina, jos sovellus käyttää Android API 3.0:a tai uudempaa. Fragmentien käyttö lisää monipuolisuutta activityyn verrattuna.

Asetetut tiedot tallennetaan Androidin omaan SharedPreferences-tiedostoon, jossa ne pysyvät tallessa, ja näin käyttäjän ei tarvitse asettaa kaikkea uudelleen joka kerta, kun sovellus käynnistetään. Tätä tiedostoa käytetään myös joidenkin muiden tietojen tallentamiseen ja käyttämiseen.

Kuvassa 38 näkyy, kuinka koodissa asetetut tiedot haetaan muuttujiin, joita käytetään yhteyden muodostamisessa palvelimelle.

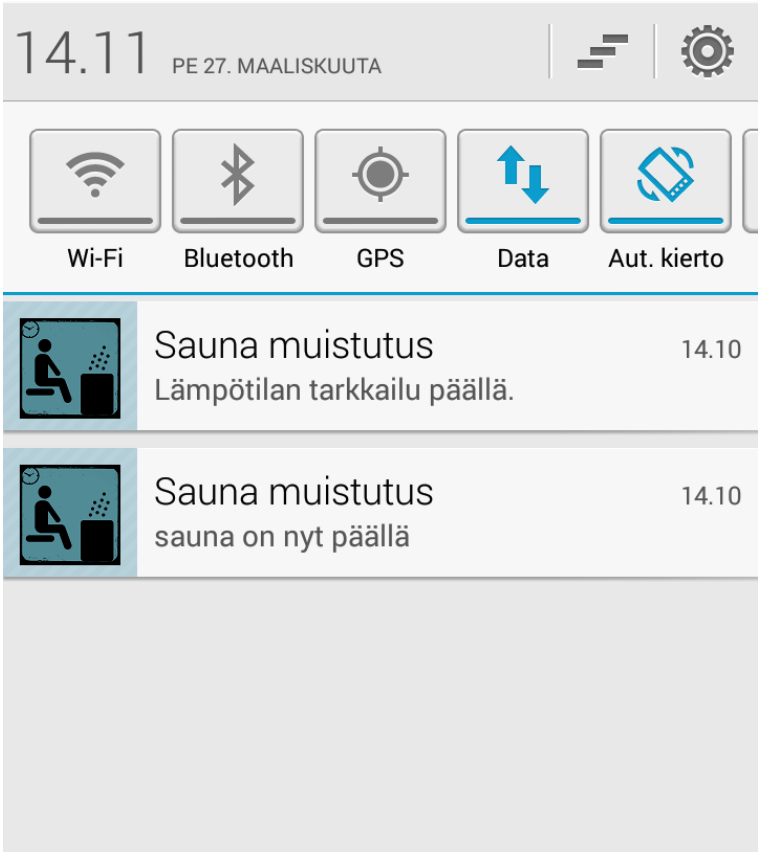
```
public void setUserInfo(){
 //Log.d("ReadTempTask", "setUserInfo");
 prefs = PreferenceManager.getDefaultSharedPreferences(c);
 String ip_test = prefs.getString("ip_text", "NA");
 String user=prefs.getString("user_text", "NA");
 int port= Integer.parseInt(prefs.getString("port_text", "NA"));
 String passwd =prefs.getString("passwd_text", "NA");
 SFTPHOST = ip_test;
 SFTPUSER = user;
 SFTPPORT = port;
 SFTPPASS =passwd;
 WORKINGDIR ="/sys/bus/w1/devices/28*";
}
```

Kuva 38: Asetetut tiedot haetaan muuttujiin.

7.4 Ilmoitukset

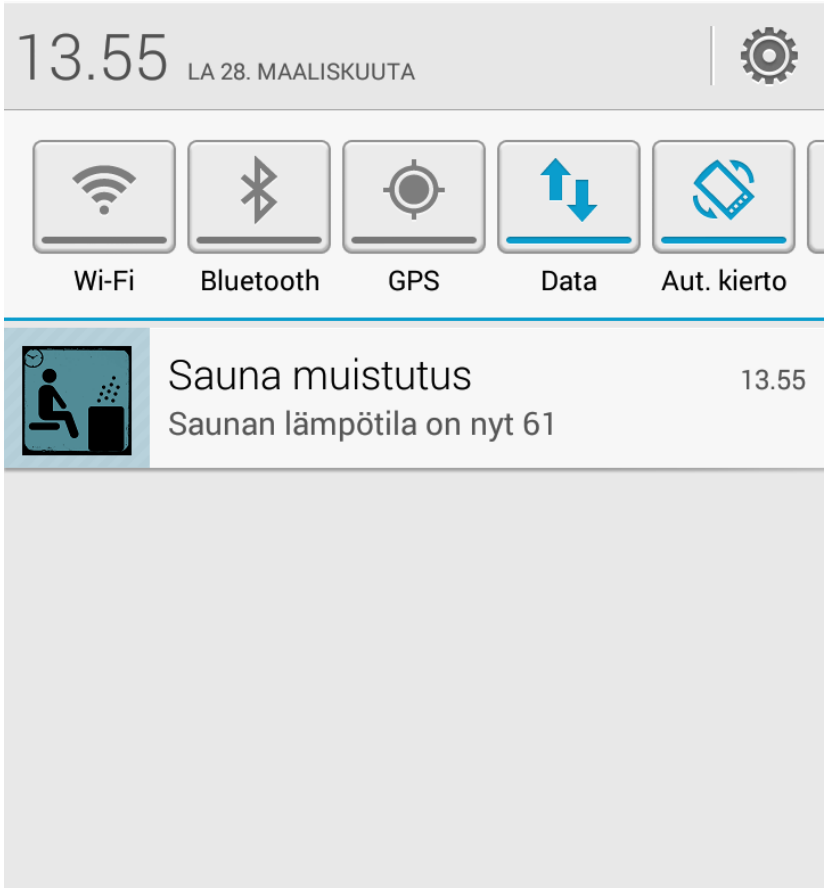
Ajastetuissa etäkäyttösovelluksissa ilmoitukset käyttäjälle ovat hyvin tärkeitä: ne pitävät heidät ajan tasalla sekä ongelmien että vaaratilanteiden välttämiseksi. Ilmoituksia luodaan sovelluksessa yleisimmin muistutukseksi käyttäjälle siitä, että sauna on päällä tai menee päälle asetetun ajan kuluttua, jos asetuksesta on jo kulunut pitkä aika. Myös lämpötilaa seurataan ilmoitusten avulla, kun käyttäjän asettama kohdelämpötila saavutetaan.

Tässä sovelluksessa ilmoitukset toimivat yhdessä Androidin oman AlarmManager-ominaisuuden kanssa. AlarmManager hoitaa hälytysten lisäämisen, poistamisen ja suorittamisen ilman, että kohdesovellus joutuisi olemaan jatkuvasti päällä. Kun käyttäjän asettama saunan päällemenoaika saavutetaan, luodaan käyttäjälle kaksi eri ilmoitusta, jotka kertovat saunan menneen päälle ja saunan lämpötilan olevan tarkkailussa, kuten kuvassa 39.



Kuva 39: Ilmoitukset saunan päälle kytkemisestä ja lämpötilantarkkailusta.

Ilmoituksen, joka kertoo saunan menneen päälle, saa pois painamalla ilmoitusta, mutta ilmoitus lämpötilatarkkailusta jää näkyviin ja päivittyy saunan lämpötilan saavuttaessa kohdelämpötilan tai hieman ennen sitä. Lämpötilan seuraamista voi jatkaa ilmoitusten kautta, kuten kuvassa 40, tai manuaalisesti sovelluksen päänäytöltä. Kuitenkin, kun ilmoitus on saanut ensimmäisen lämpötilatietonsa, saadaan lämpötilantarkkailu pois päältä painamalla ilmoitusta.



Kuva 40: Ilmoitus tarkistetusta lämpötilasta. Tarkkailu saadaan pois painamalla ilmoitusta.

Sovellus luo muutaman erilaisen hälytyksen tilanteen mukaan. Kuvassa 41 näkyy niistä ensimmäinen, joka luodaan, kun käyttäjä on asettanut kaikki haluamansa asetukset saunaan ja laittanut sen ajastuksella päälle. Valittu päälle menoaika asetetaan ensimmäisen hälytyksen ajaksi, joka suorittaa sovelluksen BroadcastReceiver-luokan.

```
Intent myIntent = new Intent(Details.this, SaunaReceiver.class);
myIntent.putExtra("id", id);

pendingIntentTime = PendingIntent.getBroadcast(Details.this, id, myIntent,0);
AlarmManager alarmManager = (AlarmManager)getSystemService(ALARM_SERVICE);
alarmManager.set(AlarmManager.RTC_WAKEUP,ca.getTimeInMillis(), pendingIntentTime);
```

Kuva 41: Ensimmäinen hälytys, joka tapahtuu, kun saunan on tarkoitus mennä päälle.

SaunaReceiver-luokka valitsee parametrina laitetun id:n perusteella kahdesta eri Servicestä, kumpi laitetaan päälle. Ensimmäisenä suoritetaan NotificationService-luokka, jossa luodaan ”Sauna on nyt päällä” -ilmoitus käyttäjälle sekä asetetaan

toistuvat hälytykset lämpötilantarkkailua varten. Kuvissa 42 ja 43 on ilmoituksen luonti ja hälytyksen eli ilmoituksen ilmaantumisen toistumisen toteutus.

```
wl.acquire(10000);
notifyText = prefs.getString("sauna_notification", "def");
NotificationCompat.Builder nt  = new NotificationCompat.Builder(this);
nt.setContentTitle("Sauna muistutus")
.setContentText(notifyText)
.setSmallIcon(R.drawable.sauna_logo1)
.setAutoCancel(true);

PendingIntent notifyPIntent =
    PendingIntent.getActivity(getApplicationContext(), 0, new Intent(), 0);
    nt.setContentIntent(notifyPIntent);


NotificationManager notificationManager = (NotificationManager)
          getSystemService(NOTIFICATION_SERVICE);



notificationManager.notify(0, nt.build());
```

Kuva 42: Ilmoituksen luominen. Siihen asetetaan otsikko, ilmoitusteksti ja logo.

```
pendingIntentTemp = PendingIntent.getBroadcast(NotificationService.this, id, myIntent,0);
tempManager = (AlarmManager)getSystemService(ALARM_SERVICE);
//asetetaan hälyt tietyn ajanjakson välein
tempManager.setRepeating(AlarmManager.RTC_WAKEUP, cal.getTimeInMillis(),6000000, pendingIntentTemp);
```

Kuva 43: Lämpötilan tarkkailua varten asetettava toistuva hälytys, johon aikaväliksi on asetettu 600 000 millisekuntia eli 10 minuuttia.

Aina ilmoitusta luotaessa täytyy myös huomioida, miten käyttäjä huomaa saaneensa ilmoituksen. Ilmoituksiin voi myös lisätä äänen tai tärinäominaisuuden. Tärkein tarvittava ominaisuus ilmoitusta luotaessa on Androidin WakeLock, joka ”herättää” puhelimen ja mahdollistaa oman sovelluksen suorittaa haluttuja komentoja ja luokkia. WakeLock suoritetaan ”acquire”-komennolla ennen ilmoituksen luomista, ja siihen asetetaan sen päälläoloaika millisekunteina, mutta se pitää aina myös muistaa vapauttaa ”release”-komennolla, jotta mobiililaitteen virta ei kulu turhaan.

Toinen ilmoitus, jonka käyttäjä saa, on ”Lämpötilan tarkkailu on nyt päällä”. Tämän on tarkoitus informoida käyttäjää siitä, että lämpötilaa seurataan, mutta myös auttaa sovellusta toimimaan oikein. Kuvassa 44 oleva ilmoitus luodaan TempService-luokassa, joka pitää huolta myös lämpötilantarkistuksen suorittamisesta.

```
NotificationCompat.Builder n  = new NotificationCompat.Builder(context);
wl.acquire(10000);
n.setContentTitle("Sauna muistutus")
.setContentText("Lämpötilan tarkkailu päällä.")
.setSmallIcon(R.drawable.sauna_logo1)
.setAutoCancel(true);
Notification note;
note=n.build();


// käytetään startForeground jotta android OS ei tuhoa hälytyksiä.
startForeground(1,note);
```

Kuva 44: Toisen ilmoituksen luominen. Tämän tilalle päivittyy lämpötilatietoa, kun kohdelämpötilaa saavutetaan.

Tätä luokkaa on tarkoitus kutsua aina 10 minuutin välein, kun tarkistus tehdään, ja Androidin käyttöjärjestelmällä on ikävä tapa optimoida itseään aina välillä poistamalla ”turhat” hälytykset ja vapauttamalla lisää resursseja mobiililaitteen käyttöön. Vaikka saunasovellus käyttää minimaalisesti resursseja, käyttöjärjestelmä poisti aina sen hälytykset. Tämä ongelma kierrettiin luomalla toinen ilmoitus ”etualalla”, jolloin Android pitää sitä tärkeänä eikä näin pyyhi sovelluksen hälytyksiä.

Ongelmat

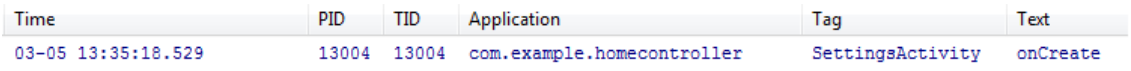
Ongelmat ja niiden ratkaiseminen ovat iso osa sovelluskehitystä, ja niiden kautta ohjelmistosta saadaan toimivampi ja käyttäjäystävällisempi. Saunasovelluksen ongelmat tuottivat välillä suuria vaikeuksia varsinkin ilmoitusten kanssa. Ongelmanratkontaan käytettiin pääasiallisesti Eclipsessä olevaa console- ja logcat-osiota. Console on Eclipsen kehitysympäristössä sijaitseva pieni ruutu, josta voi seurata, mitä tapahtuu, kun sovellusta suoritetaan. Siihen tuodaan näkyviin riveittäin tietoa, joista selviää, mitä ohjelma tekee ja mitä virheitä siinä ilmenee, esimerkkinä kuva 45.

Time	Application	Tag	Text
			0)
03-05 13:41:10.519	com.example.homecontroller	AndroidRuntime	FATAL EXCEPTION: main
03-05 13:41:10.519	com.example.homecontroller	AndroidRuntime	java.lang.IllegalStateException: Could not execute method of the ac ⏎ tivity
03-05 13:41:10.519	com.example.homecontroller	AndroidRuntime	at android.view.View$1.onClick(View.java:3628)
03-05 13:41:10.519	com.example.homecontroller	AndroidRuntime	... 11 more
03-05 13:41:10.519	com.example.homecontroller	AndroidRuntime	Caused by: java.io.FileNotFoundException: /storage/sdcard0/timouus. ⏎ on: open failed: EACCES (Permission denied)

Kuva 45: Virheilmoitukset näkyvät punaisella. Ilmoitus kertoo, että tiettyyn tiedostoon on pääsy estetty.

Consolen lisäksi käytettävänä on logcat-osio, joka myös helpottaa ongelmanratkonnassa. Logcat on samanlainen pieni Eclipsen ikkuna, josta on luettavissa rivejä, jotka sisältävät kaiken, mitä mobiililaitteessa tapahtuu, kun ohjelma suoritetaan sillä.

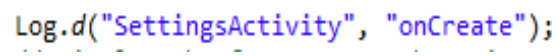
Kuvassa 46 näkyy, kuinka logcatillä saa myös hyvin seurattua, miten sovellus suorittaa tiettyjä osioitaan ja missä järjestyksessä.



Time	PID	TID	Application	Tag	Text
03-05 13:35:18.529	13004	13004	com.example.homecontroller	SettingsActivity	onCreate

Kuva 46: Logcat-ikkunan ilmoitus. Riviltä näkyy, mihin aikaan ohjelma on suoritettu.

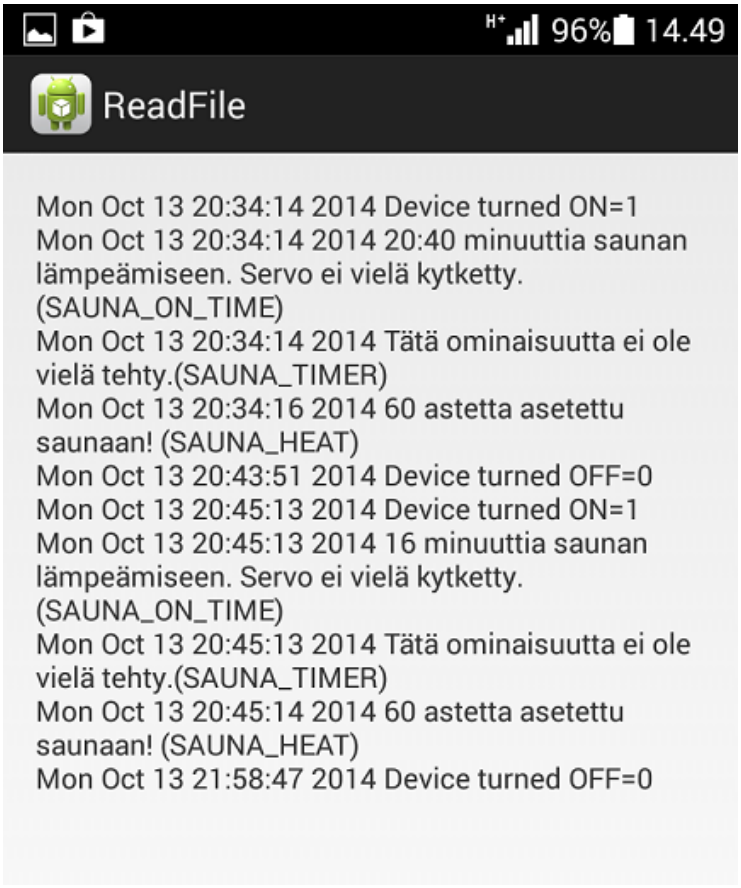
Logcat toimii kutsumalla Log-luokkaa ja lisäämällä siihen omat merkinnät tunnistusta varten, kuten kuvassa 47.



```
Log.d("SettingsActivity", "onCreate");
```

Kuva 47: Logcatin käyttö koodissa. Rivi kertoo loki-ikkunassa, kun luokassa "SettingsActivity" suoritetaan "onCreate"-metodi.

Sovellukseen lisättiin testausvaiheessa myös mahdollisuus lukea palvelimen suorittamaa käytön valvontaa puhelimella. Tämä ratkaistiin mahdollisimman yksinkertaisesti napilla, joka lataa palvelimen oman käytön valvonta -tiedoston puhelimeen ja näyttää siitä viimeiset 10 riviä. Käytön valvonnan seuraaminen osoitti aina, kun käskyt palvelimelle olivat onnistuneet tai epäonnistuneet, kuten kuvassa 48 havainnollistetaan. Se myös helpotti ongelmien ratkaisemisessa ilman, että tarvitsi olla jatkuvasti etäyhteydessä palvelimeen.



Kuva 48: Käytön valvonnan lukua puhelimella.

Alussa sovellus tuntui käyttävän liikaa muistia, koska se jäi päälle aina, kun liikuttiin pois sovelluksesta. Muistin käyttö oli tässä vaiheessa liian suuri pienelle ohjelmalle. Sovelluksella on tarkoitus ilmoittaa käyttäjälle, kun tietty asia tapahtuu. Tätä ei voida valvoa jatkuvasti jättämällä sovellusta päälle, vaan tähän käytetään Androidin omaa AlarmManageria, joka hoitaa sille määriteltyjä asioita. Näin sovellus voidaan sulkea rauhassa ilman, että mitään tuhotaan, vaikka Android-käyttöjärjestelmä yrittää kovasti tuhota sovelluksen hälytyksiä. Ohjelmaa optimoitiin vielä lisää, ja siitä saatiin vähemmän muistia käyttävä ja nopeammin toimiva. Loppuvaiheessa sovelluksen muistinkäyttö liikkui tarpeeksi pienellä välillä sen ollessa päällä.

8 Testaus

Saunajärjestelmää testattiin heti projektin alusta lähtien. Pääohjelman toteutusta alettiin kirjoittaa kesäkuusta 2014 lähtien. Pääohjelma ei siinä vaiheessa osannut muuta kuin

skannata hakemistoja. Tätä ominaisuutta testattiin monta viikkoa, ennen kuin sen sai toimimaan hyvin. Seuraavaksi testattavana oli tiedostosta lukemista ja servomoottoreiden liikuttamista. Tiedostosta lukeminen sujui ongelmitta, mutta servomoottoreiden testauksen kanssa vaikeuksia oli enemmän, ja meni todella kauan saada servojen ääriasennot testattua ja tehdä liikeradoista asteikko, joka täsmäsi sähkökiukaan säätövipujen kanssa. Ohjelmiston ensimmäisten versioiden valmistuttua oli mahdollista aloittaa kokonaisuuden testaus. Puhelimella lähetettiin käsky laittaa kiuas päälle 120 asteeseen, joka on maksimiarvo saunan lämpötilalle. Tiedosto tuli palvelimelle oikeaan hakemistoon, ja palvelin kirjoitti lokitiedostoon sovitulla tavalla, että laite on kytketty päälle. Samaan aikaan servomoottori kääntyi 120 asteen kohdalle. Tällä testillä huomattiin, että ohjelmistokokonaisuus toimii juuri niin kuin pitääkin.

Testausta tehtiin jatkuvasti rajatuilla ominaisuuksilla pienen asian muuttuessa ja koko projektikokonaisuuden testaus aina, kun uusi versio tai komponentti toteutettiin. Tällä tavoin ehkäistiin ongelmaa, jonka uusi tai päivitetty ohjelmakoodi voisi aiheuttaa rikkomalla ennestään olemassa olevaa, toimivaa toteutusta. Siinä tapauksessa, että päivitys rikkoi jotain, ongelmat pystyttiin nopeasti paikantamaan ja korjaamaan virheet ohjelmistossa. Palvelinpuolella käytössä oli erikseen testi- ja tuotantoympäristö. Testit suoritettiin aina ensin testipuolella varotoimenpiteenä, ettei suurta vahinkoa tapahdu ja että komponentit toimivat yhdessä. Sen jälkeen komponentit siirrettiin tuotannon hakemistoon, johon ei siirretty uutta toimivaa binääriä, ennen kuin oli todettu sen toimivan testiympäristössä. Android-sovelluksesta pidettiin koko ajan pääasiassa vain yhtä versiota, jota päivitettiin jatkuvasti, ja toimiviksi todetuista versioista tehtiin varmuuskopiot. Muutamaa eri testiversiota pidettiin erillään, koska ne erosivat huomattavasti perustoimivuudeltaan pääversiosta. Nämä testiversiot sisälsivät esimerkiksi pilvisynkronoinnin, ja niiden kehitys jäikin hyvin vähäiselle huomiolle liian suurien vaikeuksien tai turhiksi todettujen asioiden takia.

Android-laitteita oli käytössä vain yksi, joten kokonaisuuden testaaminen ilman molempien tekijöiden osallistumista oli hankalaa. Tätä varten kehitettiin skripti, joka luo samanlaisen tiedoston kuin Android-sovelluksella tehtäessä. Laitteen sammutusta varten toteutettiin myös toinen skripti. Molemmat skriptit näkyvät kuvissa 49 ja 50

```
@ECHO OFF
set /p heat= What is sauna heat?
set /p time= How long is sauna on?
set /p timer= How long you want to wait before sauna goes on? (min)
echo Setting %heat% to sauna.
echo Sauna is %time% minutes on.
echo Sauna offset is %timer%
echo SAUNA_HEAT=%heat% > asda.on
echo SAUNA_ON_TIME=%time% >> asda.on
echo SAUNA_TIMER=%timer% >> asda.on
echo All set. Get ready for sauna!
echo option batch abort > script.tmp
echo option confirm off >> script.tmp
echo option transfer ascii >> script.tmp
echo open sftp://username:password@server.name.fi/ >> script.tmp
echo put asda.on /path/to/dir/ >> script.tmp
echo close >> script.tmp
echo exit >> script.tmp
WinSCP.com /script=script.tmp /xmllog=logitus.txt
if %ERRORLEVEL% neq 0 goto error

echo Transfer succesfully completed!
del script.tmp
del asda.on
pause
exit 0

:error
echo ERROR transferring file!
del script.tmp
del asda.on
pause
exit 1
```

Kuva 49: Saunan lämmitys Androidin simulointiskriptillä.

```
@ECHO OFF
echo diiipadaapa > asda.off
echo Shutting sauna off!
echo option batch abort > script.tmp
echo option confirm off >> script.tmp
echo option transfer ascii >> script.tmp
echo open sftp://username:password@server.name.fi/ >> script.tmp
echo put asda.off /path/to/dir/ >> script.tmp
echo close >> script.tmp
echo exit >> script.tmp
WinSCP.com /script=script.tmp /xmllog=logitus.txt
if %ERRORLEVEL% neq 0 goto error

echo Transfer succesfully completed!
del script.tmp
del asda.off
pause
exit 0

:error
echo ERROR transferring file!
del script.tmp
del asda.off
pause
exit 1
```

Kuva 50: Saunan sammutus Androidin simulointiskriptillä.

9 Jatkokehitys

9.1 Palvelimen jatkokehitys

Palvelinpuolen jatkokehitysaiheena on kehittää tietoturvaa ja käytettyä asiakas-palvelinmallia. Nykyinen toteutus mahdollistaa kenen tahansa lähettää käskyjä palvelimelle, jos IP-osoite on tiedossa. Tätä varten yritettiin toteuttaa salasanalla kirjautumista, mutta se osoittautui yllättävän haasteelliseksi, eikä se vastannut projektille asetettuja tavoitteita, joten se siirrettiin jatkokehitysmahdollisuudeksi. (29.)

Tarkoituksena on jatkaa palvelimen kehitystä parantamalla nykyistä asiakas-palvelinmallia. Mahdollisuutena olisi muidenkin laitteiden hallinta kuin saunan sekä ominaisuus tarjota asiakasohjelmalle salasanan luonti tai vaihto, mikäli asiakas on kadottanut salasanan.

Yksi jatkokehitysmahdollisuus on myös luoda turvakytkin palvelinpuolelle. Turvakytkin toimisi niin, että lämpötilan kohotessa saunassa esimerkiksi 20 celsiusastetta saunaan asetetun lämpötilan yli, palvelin reagoisi lämpötilan ylitykseen sammuttamalla saunan kokonaan kääntämällä servot nolla-asentoon, jolloin siitä lähtisi välitön ilmoitus käyttäjälle. Näin projektissa on huomioitu muun muassa paloturvallisuus. Tuotteen kaupallistaminen varmasti vaatisi myös edellä kuvatut turvatoimet.

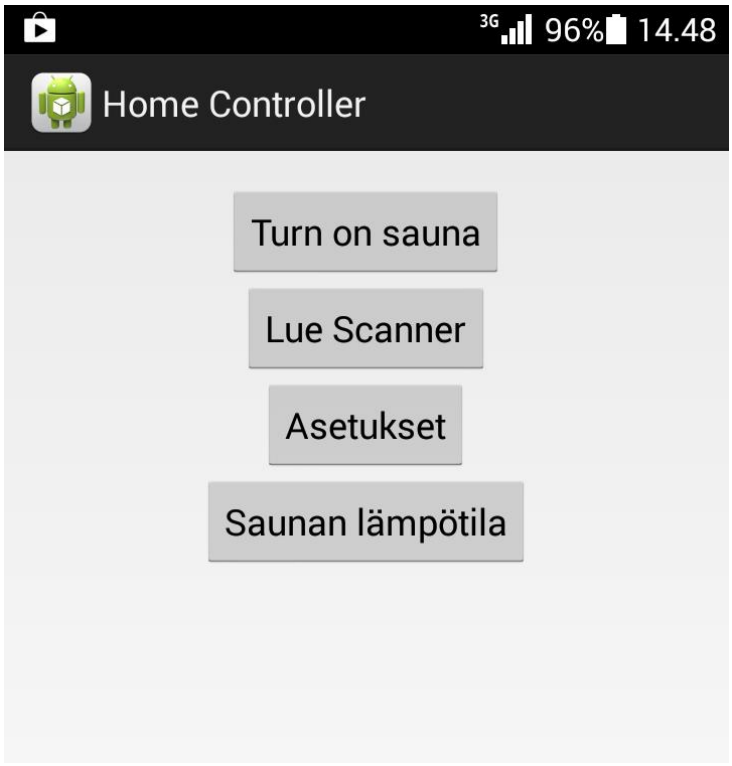
Yksi tulevaisuuden tärkeimmistä jatkokehityskohteista olisi palvelinpuolen pääprosessin edellisen tilan tarkistaminen sen uudelleen käynnistyessä. Pääohjelman kaatuessa skripti käynnistää sen uudelleen. Pääprosessi ei kuitenkaan tällä hetkellä tarkista saunan edellistä tilaa ja sitä, onko saunan tarkoitus edelleen olla päällä. Tähän on mahdollista kehittää ominaisuus, joka kävisi lukemassa käytönvalvontatiedostosta viimeiset asetukset ja palauttaisi pääprosessin siihen tilaan, missä se oli ennen kaatumista. Pääprosessi vertaisi nykyistä aikaa tiedostossa olevaan aikaleimaan ja päättelisi siitä, kuuluuko saunan olla päällä vai sammuneena. Tämän ominaisuuden toteutuksen ei pitäisi olla kovin haasteellinen, sillä se on otettu huomioon käytönvalvontaa tehtäessä.

Sähkökatkojen varalta olisi myös hyvä hankkia palvelimelle erillinen varavirtalähde ja UPS-laite. Varavirran ei tarvitsisi kestää kuin sen verran, että palvelin kerkeäisi asettamaan saunan pois päältä. Sähkön palautuessa palvelin toimisi tavanomaisesti ja

tarkistaisi edellisen tilan ja toimisi sen mukaisesti. UPS-laite takaisi palvelimelle pidempää käyttöikää, kun virransyöttö olisi tasaista. Tämä toisi myös lisää turvallisuutta, jota ei voi koskaan korostaa liikaa.

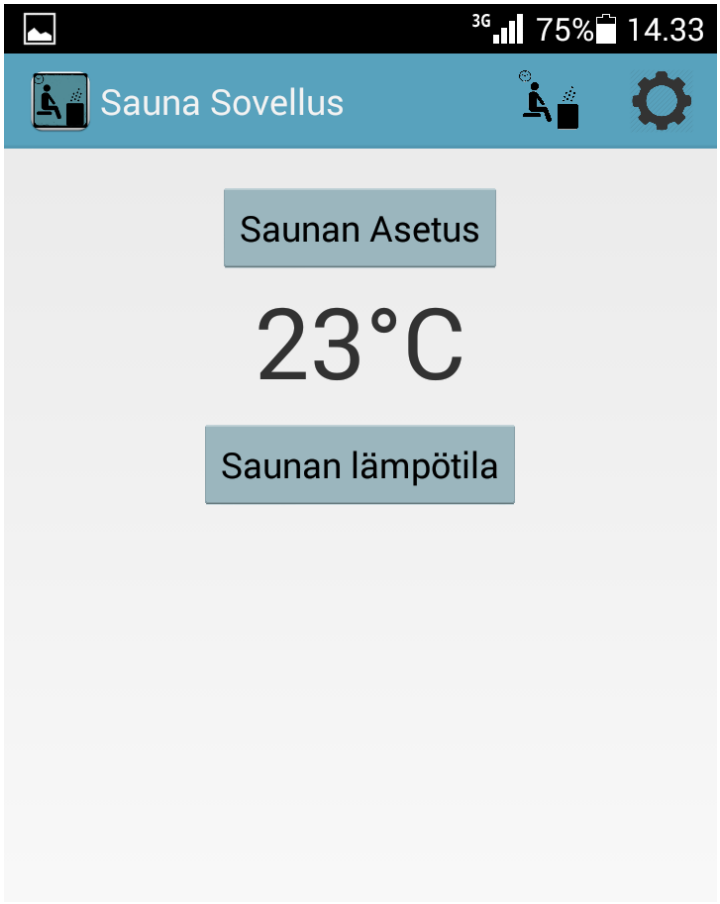
9.2 Android-sovelluksen jatkokehitys

Projektin alkaessa sovelluksesta oli suhteellisen hyvä käsitys ja näkemys siitä, millainen sen tulisi olla. Aluksi sovellukseen oli tarkoitus sisältää helpot debuggaus-mahdollisuudet, mikä näkyy kuvassa 51 ”Lue Scanner”-nappina, ja nämä ominaisuudet poistettiinkin sitä mukaa, mitä lähemmäs sovelluksen valmistumista päästiin.

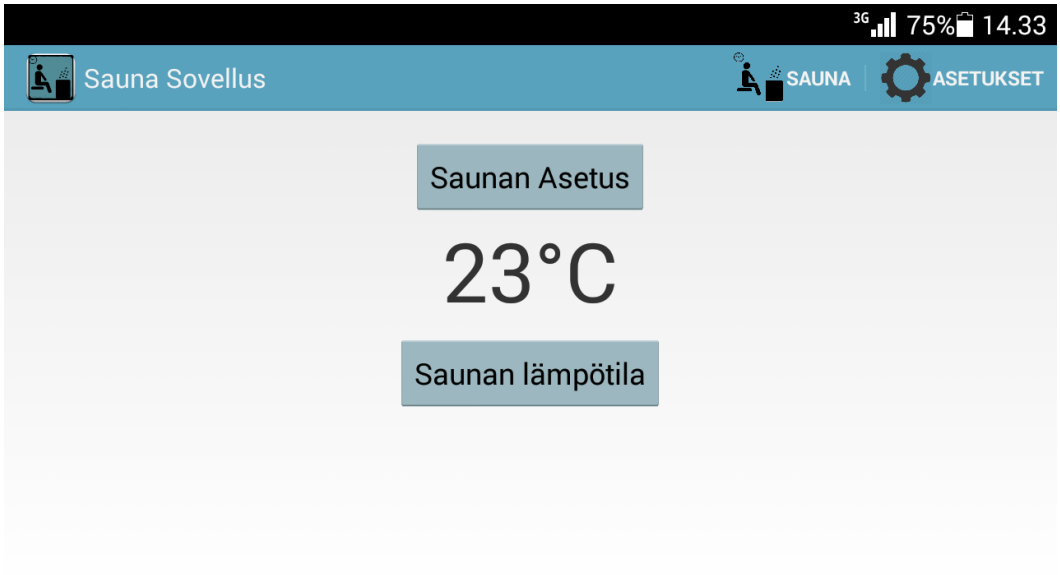


Kuva 51: Sovelluksen etusivulta pääsi käsiksi kaikkiin ominaisuuksiin helposti.

Sovelluksen graafinen olemus oli koko projektin ajan taka-alalla, eikä siihen panostettu niin paljon, kuin ehkä olisi voinut. Yksi syy tähän oli se, että koko projektia pidettiin enemmän testiversiona kuin suoraan myyntiin ja asiakkaalle menevänä valmiina tuotteena. Myöhäisemmässä vaiheessa sovelluksen ulkonäköä kuitenkin muutettiin vaihtamalla värimaailmaa ja lisäämällä kuvakkeita ja logoja. Sovelluksen uudempi ilme näkyy kuvissa 52 ja 53.



Kuva 52: Sovelluksen päänäytön uudempi ilme. Turhat debug-ominaisuudet on poistettu ja uudet navigointi-kuvakkeet on lisätty yläkulmaan logon kanssa.



Kuva 53: Sovelluksen päänäyttö horisontaalisessa asennossa. Ruudun tilankäyttö toimii hieman paremmin näin päin käytettynä ja kuvakkeiden teksiselitykset tulevat näkyviin.

Jatkokehitysideoina ovat olleet varoitukset vikatilanteista ja saunan lämpötilan ympärivuorokautinen seuranta. Lämpötilan manuaalista lukemista yksinkertaistettiin

myöhemmin toimimaan eri tavalla kuin aiemmin. Uudessa mallissa ei tarvitse erikseen ladata tiedostoa, joka sisältää lämpötilan, vaan palvelimeen otetaan suora yhteys. Tämän jälkeen sille syötetään tietty numeroarvo, jolla kerrotaan, mitä palvelimen pitää tehdä. Kun palvelin huomaa yhteydenoton, se odottaa syötettyä numeroa. Syötteen saatuaan palvelin lukee saunan lämpötilatiedon ja syöttää sen suoraan takaisin sovellukselle. Tämän jälkeen yhteys katkaistaan ja lämpötila voidaan esittää suoraan käyttäjälle. Palvelimella on myös aikakatkaisuominaisuus vikatilanteita varten, jolloin mitään syötettä ei tulisikaan sovellukselta. Tällöin palvelin katkaisee yhteyden automaattisesti.

Mahdollisuus olisi myös luoda sovellukseen ominaisuus, johon voisi syöttää kiukaan mallin tai vastaavanlaisen tiedon, jonka avulla voisi laskea saunalle arvioidun sähkönkulutuksen. Sovelluksessa on vielä pieniä parantamisen mahdollisuuksia. Osan ominaisuuksista pystyisi toteuttamaan eri tavalla, ja tämä mahdollisesti parantaisi sovelluksen suorituskykyä. Sovelluksen ulkonäköä voisi muuttaa mukavammaksi ja kiinnostavammaksi. Ohjeita sovelluksen käytöstä voisi lisätä jokaiseen näkymään, joko suoraan tekstinä ruudulle tai nappia painamalla pienenä ponnahdusikkunana.

9.3 Mekaniikan jatkokehitys

Tulevaisuudessa mekaniikkaa voisi parantaa siltä osin, että servomoottorit korvattaisiin hieman tehokkaammilla moottoreilla, joissa on tehoa paljon enemmän. Tämä vaatisi myös uuden virtalähteen hankkimista, jotta uudet tehokkaammat servomoottorit saisivat tarpeeksi virtaa toimiakseen. Virtalähde tulisi sijaitsemaan saunatiloissa, joten sen vedenpitävyydestä tulisi myös huolehtia. Yhtenä jatkokehitysideana on muutos mekaanisiin komponentteihin. Tässä ratkaisussa rakennettaisiin erillinen osa, jossa on pelkästään servomoottorit ja langaton vastaanotin. Palvelin lähettäisi sitten käskyjä servomoottoreille, aivan kuten nykyisessä ratkaisussa, mutta käskyt toimitettaisiin langattomasti servoille. Näin vältyttäisiin kaapeleiden vetämiseltä saunaan. Kaapelit on vaikea kytkeä saunatiloihin ilman, että joudutaan porailemaan reikiä ja piilottamaan niitä listojen alle. Kaapelit voivat jopa olla vaarallisia mahdollisten sähköiskujen vuoksi.

10 Yhteenveto

Insinöörityössä oli tarkoituksena tehdä suomalaisesta saunakulttuurista helpompaa siltä osin, että saunaan menemiskynnys olisi mahdollisimman alhainen ja saunan lämmitys olisi mahdollisimman vaivatonta. Pitkän työmatkan jälkeen on mukava laittaa sauna päälle jo kotimatkalla, jolloin kotiin palatessa on valmis sauna odottamassa.

Insinöörityön tekeminen oli välillä todella hidasta. Haasteita oli ensinnäkin palvelinpuolella todella paljon, koska palvelinpuolella on niin monta eri komponenttia, joiden tulee aina toimia yhdessä. Yhden komponentin päivittäminen saattoi rikkoa toisen. Ongelman ratkaisuun saattoi välillä mennä useita viikkoja. Onneksi kuitenkin aloitimme työn tekemisen ajoissa.

Valitsimme Androidin mieluummin kuin Windows Phonen tai Iphonen, koska Android tarjoaa monipuolisemmat mahdollisuudet ja suuremman joustavuuden sovellusten suhteen. Android oli meille myös osittain valmiiksi tuttu, ja testaukseen oli käytettävissä oma Android-puhelin. Ennen tätä projektia meillä oli vain vähän kokemusta Android-sovellusten tekemisestä, ja lähdimmekin tekemään tätä oppimismielellä.

Aloimme suunnitella sovellusta tyhjän päältä katselematta esimerkkisovelluksia, koska halusimme tehdä kaiken itse alusta asti. Tarkoituksena oli tehdä mahdollisimman selkeä ja helppokäyttöinen käyttöliittymä, jossa olisi kuitenkin monipuoliset toiminnot.

Koska Android-käyttöjärjestelmä on suosituin niin käyttäjien kuin kehittäjienkin mobiililaitteissa, on tämänkaltaisille etäohjattaville järjestelmille hyvät mahdollisuudet markkinoilla. Tietenkin saunan etäkäyttöön soveltuvana se sopii Pohjoismaihin, mutta samanlaista konseptia voi soveltaa moniin muihin laitteisiin. Android-sovelluksen tekeminen tällaisesta aiheesta oli erittäin kiinnostavaa ja motivoivaa mutta kokemattomille välillä hyvin hankalaakin. Projektin lopputulosta pitää ajatella kokonaisena pakettina, joka myytäisiin asiakkaalle. Asiakas lataisi mobiilisovelluksen, ja sitä pidettäisiin päivitettynä joko Googlen Play -kaupan tai omien palvelimien kautta. Mekaaninen puoli olisi sitten joko mieluiten itse asennettavissa tai vaatisi asentajan käynnin.

Insinöörityön lopputulos täyttää mielestämme työn alussa asetetut kriteerit, ja suurin osa asetetuista tavoitteista saavutettiin. Työn tarkoitus oli helpottaa saunassa käymistä

ja tehdä siitä entistä vaivattomampaa. Saimme osoitettua, että on mahdollista rakentaa toimiva ja saunomista helpottava laitteisto vähäisin kustannuksin. Isommalla budjetilla on mahdollista vielä parantaa lopputulosta. Sen aiomme tulevaisuudessa tehdä. Tämä oli vasta alkua suuremmalle projektille.

Lähteet

1 Etäkäyttö. 2014. Verkkodokumentti. PcWorld. <http://www.pcworld.com/article/2687241/5-ways-to-use-your-pc-remotely.html> Luettu 18.1.2015.

2 SSH. 2009. Verkkodokumentti. Tampereen teknillinen yliopisto Tieto- ja sähkötekniikan tiedekunta. <http://www.cs.tut.fi/lintula/software/ssh/teoria.shtml> Luettu 23.10.2015

3 Raspberry Pi. Verkkosivusto. Raspberry Pi. <http://www.raspberrypi.org> Luettu 20.12.2014.

4 Raspberry Pi käyttöjärjestelmä arvostelu. Verkkodokumentti. TechRadar. <http://www.techradar.com/news/software/operating-systems/raspberry-pi-operating-systems-5-reviewed-and-rated-1147941> Luettu 8.10.2015.

5 Raspberry Pi kotiteatteri käyttöliittymä. Verkkodokumentti. SoftArchive. <https://softarchive.unblocked.la/blogs/islamayman/kodi_final.1789401.html> Luettu 8.10.2015.

6 Raspberry Pi myyntiluvut. Verkkodokumentti. Raspberry Pi. <https://www.raspberrypi.org/blog/five-million-sold/> Luettu 15.10.2015.

7 Raspberry Pi Arcade. Verkkodokumentti. Cnet. <http://www.cnet.com/uk/how-to/25-fun-things-to-do-with-a-raspberry-pi/> Luettu 20.8.2015.

8 Androidin alku. 2014. Verkkodokumentti. IPWatchdog. <http://www.ipwatchdog.com/2014/11/26/a-brief-history-of-googles-android-operating-system/id=52285/> Luettu 10.2.2015.

9 Android versioiden historiataulukko. Verkkodokumentti. Socialcompare. <http://socialcompare.com/en/comparison/android-versions-comparison> Luettu 10.2.2015.

10 Ensimmäinen Android-puhelin. Verkkodokumentti. Androidcentral. <http://www.androidcentral.com/review-android-t-mobile-g1> Luettu 10.2.2015.

11 Google Chrome engine. 2012. Verkkodokumentti. Arstechnica. <http://arstechnica.com/gadgets/2012/02/chrome-finally-brings-modern-web-standards-to-android/ > Luettu 10.2.2015.

12 Android Flash -tuki. 2010. Verkkodokumentti. New York Times Blog. <http://bits.blogs.nytimes.com/2010/04/27/googles-andy-rubin-on-everything-android/?_r=0>. Luettu 10.2.2015.

13 ART korvaa Dalvikin. 2014. Verkkodokumentti. Liliputing. <http://liliputing.com/2014/10/whats-new- android-5-0-lollipop.html> Luettu 15.2.2015.

14 Android versioiden tilastot. Verkkodokumentti. Statista. <http://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/> Luettu 17.3.2015.

15 Android yleisesti. Verkkodokumentti. Android developer blog. <http://developer.android.com/about/index.html> Luettu 12.11.2014.

16 Android kurssit. Verkkodokumentti. WCCFtech. <http://wccftech.com/free-android-mobile-development-bundle/> Luettu 12.10.2015.

17 Android huijaus. 2013. Verkkodokumentti. Digitoday. <http://www.digitoday.fi/tietoturva/2013/12/10/taskulamppu-katsoi-sinne-minne-ei-saanut-android-kayttajia-petettiin/201317081/66> Luettu 12.10.2015.

18 Facebook-sovelluksen oikeudet. Verkkodokumentti. Mbnet. <http://www.mbnet.fi/artikkeli/ajankohtaiset/android_turvalliseksi> Luettu 15.10.2015.

19 Android oikeudet. Verkkodokumentti. Androidauthority. <http://www.androidauthority.com/android-app-permissions-explained-642452/> Luettu 12.10.2015.

20 Androidin muokkaus. Verkkodokumentti. Cnet. <http://www.cnet.com/how-to/twelve-ways-to-customize-your-android-device/> Luettu 15.3.2015.

21 Android widgetit. Verkkodokumentti. Androidcentral. <http://www.androidcentral.com/what-is-a-widget> Luettu 15.3.2015.

22 Android yleisesti. Verkkodokumentti. About tech. <http://google.about.com/od/socialtoolsfromgoogle/p/android_what_is.htm> Luettu 20.10.2014.

23 Android auto. Verkkodokumentti. Android. <http://www.android.com/auto/> Luettu 15.3.2015.

24 Android ohittaa Symbianin. Verkkodokumentti. The Guardian. 2011. <http://www.theguardian.com/technology/2011/jan/31/android-symbian-smartphone-sales> Luettu 16.3.2015.

25 Älypuhelinten myyntitilasto. Verkkodokumentti. Idc. <http://www.idc.com/prodserv/smartphone-os-market-share.jsp> Luettu 16.3.2015.

26 Android-järjestelmän päivitys. Verkkodokumentti. How to Geek. <http://www.howtogeek.com/129273/why-your-android-phone-isnt-getting-operating-system-updates-and-what-you-can-do-about-it/> Luettu 16.3.2015.

27 Android-laitteiden lähetysten kasvu. Verkkodokumentti. Statista. <http://www.statista.com/statistics/241947/global-shipment-forecast-of-smartphones-using-android-os/> Luettu 17.3.2015.

28 Harvia sähkökiukaan mekaaninen ohjaus. Verkkodokumentti. Harvia. <http://www.harvia.fi/content/fi/35/10691/Kiukaan%20pikak%C3%A4ytt%C3%B6ohje%20%28mekaaninen%20ohjaus%29.html> Luettu 20.1.2015.

29 Rasbian käyttöjärjestelmä. Verkkosivusto. Rasbian yhteisö. <https://www.raspbian.org/> Luettu 25.6.2015.

30 Verkkosivusto dy.fi. Verkkodokumentti. dy.fi. <http://www.dy.fi/> Luettu 18.1.2015.

31 Stevens W. Richard, Fenner Bill, Rudoff Andrew M. 2004. UNIX Network Programming. Volume 1. Boston: Pearson Education.