Tuomas Mäkinen

Bluetooth laitteiden seurantasovellus Androidille

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikka

Insinöörityö

15.9.2012

Metropolia

Tekijä Otsikko Sivumäärä Aika	Tuomas Mäkinen Bluetooth seurantasovellus Androidille 42 sivua 19.5.2013
Tutkinto	insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Ohjelmistotekniikka
Ohjaaja	yliopettaja Jarkko Vuori

Tämän insinöörityön tarkoitus oli tutkia sopiiko Bluetooth tekniikka käyttäjien seurantaan Android käyttöjärjestelmään perustuvissa puhelimissa sekä hyödyntää käyttöjärjestelmän rajapinnoista löytyviä tunnistukseen käytettäviä ominaisuuksia. Tavoitteen mukaisesti luotaisiin sovellus, jolla kyetään seuraamaan puhelimen lähellä olevia Bluetooth laitteita ja määrittelemään niiden sijainti. Työssä tutkitaan onko seuranta perustuen Bluetooth teknologiaan järkevää ja käytännöllistä.

Sovellus ohjelmointiin Javalla, joka on yksi mahdollisista vaihtoehdoista tehtäessä sovelluksia Android-puhelimille. Sovelluksen piti pystyä toimimaan niin, että puhelin olisi itsenäinen yksikkö, joka lähettäisi tietoa lähellä olevista laitteista sekä luonnollisesti toimisi käyttäjän itsensä ohjaamana. Sovellukseen saatiin kaikki ne ominaisuudet mitkä sille oli alussa määritelty.

Lopputulokseen työssä päädyttiin ja siitä huomattiin, että Bluetooth sopii seurantaan erinomaisesti. Sovellus myös pystytään rakentamaan sellaiseksi, että käyttäjä pystyy ohjaamaan sovellusta ilman paikalla oloa sen tuottaessa tietoa käyttäjälleen. Android käyttöjärjestelmälle ohjelmointiin on helppoa päästä sisälle, esimerkkien sekä aktiivisen kehittäjäyhteisön avuin. Työ tehtiin itsenäisesti tutkimus- ja kehitystyönä sen oli tarkoitus opettaa uuden mobiilialustan erikoisuuksia ja tuoda sen ominaisuudet helpommin lähestyttäviksi.

Avainsanat	Bluetooth, Android, Eclipse, Java, Paikannus

Author Title Number of Pages Date	Tuomas Mäkinen Bluetooth based monitoring software for Android 42 pages + 2 appendices 19 May 2013
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Software Development
Instructor	Jarkko Vuori, Principal Lecturer

The main goals of this bachelor's thesis were to analyze if Bluetooth is capable of monitoring users in Android-based mobile devices and to exploit the functionalities of the operating systems' frameworks. The main criteria for the application were that it could track new Bluetooth devices and be able to identify their location using coordinates. This thesis determines if using Bluetooth for this kind of purpose is practical and feasible.

The application was programmed with Java which is one of the possible choices when developing applications for Android phones. The application should work independently in the phone and it should send data from nearby devices around, as well as work operated by the user. The application was programmed with all the possible features which were originally engineered.

The conclusion drawn in this bachelor thesis is that Bluetooth suits this kind of use well. Application can be built to produce information for its user remotely. Developing Android proved to be easy to learn with the help of the developer community and through basic examples. This thesis was done as an independent for study and its goal was to introduce new special features of the mobile environment and to make it easier to approach.

Keywords	Bluetooth, Android, Eclipse, Java, Location

Sisällys

Lyhenteet

ACL	*Asynchronous Connection-Less*. Normaali radiolinkki datan lähetykseen.
ADT	*Android Development Tools*. Kehittäjätyökalut.
Activity	Android käyttöjärjestelmän suoritettava luokka.
Android	Ohjelmistoyhtiö Googlen oma käyttöjärjestelmä.
Bluetooth	Radiotekniikka joka mahdollistaa lyhyen kantaman tiedonsiirron.
Eclipse	Yleinen sovelluskehitysympäristö.
ETSI	*European Telecommunications Standards Institute.* Telealan eurooppalainen standardointijärjestö.
GFSK	*Gaussian Frequency Shift Keying*. Taajuusmodulointimenetelmä joka käyttää gaussilaista filtteriä muokatakseen pulssin muotoa.
iOS	Applen oma käyttöjärjestelmä.
Java	Oraclen hallinnoima ohjelmointikieli.
JNI	*Java Native Interface.* Rajapinta, jonka avulla C kielistä koodia voidaan suorittaa Java virtuaalikoneessa.
LMP	*Link Manager Protocol*. Ydin protokolla vastaa yhteyden muodostamisesta.
L2CAP	*Link Layer Control and Adaption Protocol.* Ydin protokolla hoitaa kantataajuusprotokollan sovittamisen yhteen ylemmän tason kanssa.
NDK	*Native Development Kit*. Kirjastot C-kielelle, jota tarvitaan koodiin kääntämiseen prosessorille.

NFC	*Near Field Communication*. Lyhyen kantaman radiotekniikkaan perustuva menetelmä kantama noin 20 senttimetriä.
RFCOMM	*Radio Frequency communication*. Sarjakaapelia emuloiva protokolla.
SCO	*Synchorouns connection-oriented*. Radiolinkki äänen lähetystä varten.
SDK	*Software Development Kit*. Kirjasto ohjelmistokehitykseen.
SDP	*Service Discovery Protocol*. Protokolla, jonka avulla etsitään uusia palveluita.
TCS	*Telephony Control Protocol*. Protokolla, jonka avulla siirretään data- ja äänipuheluja.
UUID	Universal Unique Identifier Bluetooth järjestön tarjoama uniikki tunnistenumero.

1 Johdanto

Radiotekniikan historia juontaa juurensa 1800-luvulle, vaikka ensimmäiset langattomien viestien lähetykset onnistuivat vasta vuonna 1897 ensimmäisten morse-viestien yhteydessä. Tekniikat ovat kehittyneet noista ajoista ja laitteet ovat pienentyneet. Bluetooth on yksi tekniikka, joka perustuu langattomien viestien lähettämiseen kahden laitteen välillä. Ensimmäinen tekniikkaa käyttävä matkapuhelin tuotiin markkinoille vuonna 2000. [1.]

Bluetooth on yksi tämän päivän avoimia standardeja liittyen langattomaan tiedonsiirtoon eri laitteiden välillä. Se oli pitkään yksinvaltias suhteessa nopeuteen ja välimatkaan. Tekniikka löytyy tänä päivänä melkein jokaisesta teknisestä laitteesta (televisiot, puhelimet, yms.).

Puhelimien käyttöjärjestelmissä oli pitkään vain yksi vaihtoehto, ja se oli Symbian. Vuonna 2007 tapahtui muutos, Google julkisti oman Android-mobiililaitteille tarkoitetun käyttöjärjestelmänsä, ja Apple julkisti oman iOSinsa. Nopeasti uudet tulijat syrjäyttivät vanhan hallitsijan, joka oli hallinnut yli vuosikymmenen mobiililaitteita. Aluksi Nokialta valtikan vei Apple iOS, joka oli ollut kehitteillä pitempään kuin muut uudet kilpailijat ja oli näin siis valmiimpi markkinoille käyttöjärjestelmänä ja sovellustensa puolesta. Vuonna 2010 tilanne muuttui kun Androidista tuli mobiilimaailman hallitsija ja suurimman markkinaosuuden haltija.

Suurin syy, joka on helpottanut iOSin ja Androidin yleistymistä ja niiden sovellusvalikoiman kasvua verrattuna Symbianiin, on sovelluskehityksen helppous. Se on saanut harrastelijat innostumaan molemmista alustoista. Yhtenä erona kehittämisessä ja sen helppoudessa on se, että ainoastaan Applen omalla Mac-tietokoneella kehittäminen onnistuu iOSille. Androidille kehittäminen onnistuu mitä tahansa käyttöjärjestelmää käyttävällä tietokoneella.

Tämän insinöörityön tarkoituksena on tutkia, sopiiko Bluetooth-tekniikka käyttäjien seurantaan Android-käyttöjärjestelmään perustuvissa puhelimissa, sekä hyödyntää käyttöjärjestelmän rajapinnoista löytyviä tunnistukseen käytettäviä ominaisuuksia. Tavoitteena on luoda sovellus, jolla kyetään seuraamaan puhelimen lähellä olevia Bluetooth-laitteita ja määrittelemään niiden sijainti. Työssä tutkitaan, onko seuranta perustuen Bluetooth teknologiaan järkevää ja käytännöllistä.

2 Bluetooth radiotekniikka

Bluetooth on radiotekniikka, jota alettiin kehittää, kun ruotsalainen Ericsson alkoi tutkia erilaisia menetelmiä langattomaan kommunikointiin lähietäisyydellä. Bluetooth on saanut nimensä tanskalaisen viikinkikuninkaan Harald Sinihampaan mukaan. [2.]

Kehitystyö alkoi vuonna 1994. Jotta tästä menetelmästä tulisi laajalle levinnyt, se vaati avoimen standardin luomista ja niin syntyi Bluetooth SIG (Special Interest Group). Sen tarkoituksena oli hallinnoida Bluetoothin kehitystä. Se perustettiin helmikuussa 1998, ja siihen kuului viisi yritystä (Ericsson, Nokia, IBM, Toshiba ja Intel). Vuoden 1998 loppuun mennessä järjestö oli kasvanut melkoisesti, kun siihen liittyi 400. yritys. [3.]

Bluetooth on lyhyen kantaman radiotekniikkaan perustuva langaton tiedonsiirtotekniikka, jonka tarkoituksena oli korvata kaapelit erilaisten laitteiden välisessä tiedonsiirrossa. Se käyttää tiedonsiirtoon 2,4 GHz:n taajuudella toimivaa ISM (Industry, Medical, Science)-kaistaa ja hyödyntää GFSK-taajuussiirtokoodausta. Gaussian Frequency Shift Keying on taajuusmodulointimenetelmä, joka hyödyntää gaussialaista filtteriä suodatukseen muokatakseen pulssin muotoa. Filtterillä ei ole ollenkaan ylitystä ja se minimoi nousu ja laskuajan. Koska ISM-taajuuksia käyttää muutkin laitteet aina mikroaaltouuneista langattomiin modeemeihin, päätettiin, että Bluetoothin tiedonsiirtoon käytettäisiin taajuushyppelyä hajaspektritekniikan sijasta, koska se on tehonkulutukseltaan parempi. Bluetooth käyttää taajuushyppelyyn 1600 hyppyä sekunnissa, mikä tarkoittaa, että taajuus vaihtuu näennäissatunnaisesti 625 mikrosekunnin välein. [3.]

Kuva 1. Bluetooth-laitteita.

Kuvassa yksi on esitelty yleisimpiä Bluetooth-laitteita. Bluetooth on suunniteltu perinteiset kaapelit korvaavaksi yleiseen standardiin perustuvaksi arkkitehtuuriksi, jonka ominaisuuksia ovat vähäinen virrankulutus ja lyhyt kantama. Kantamaa pystytään lisäämään tehoja kasvattamalla, mikä vaikuttaa virrankulutukseen lisäävästi.

Bluetoothin kantamat vaihtelevat käytettyjen lähetinluokkien mukaan. Kantama vaihtelee Class 1 -luokan ja Class 3 -luokan välillä aina noin sadasta metristä noin yhteen metriin. Jos lähettimessä ei ole virransäätöominaisuutta laite tulee Class 1–luokan laitteen toimia luokan 2 tai 3 tavoin. [3.]

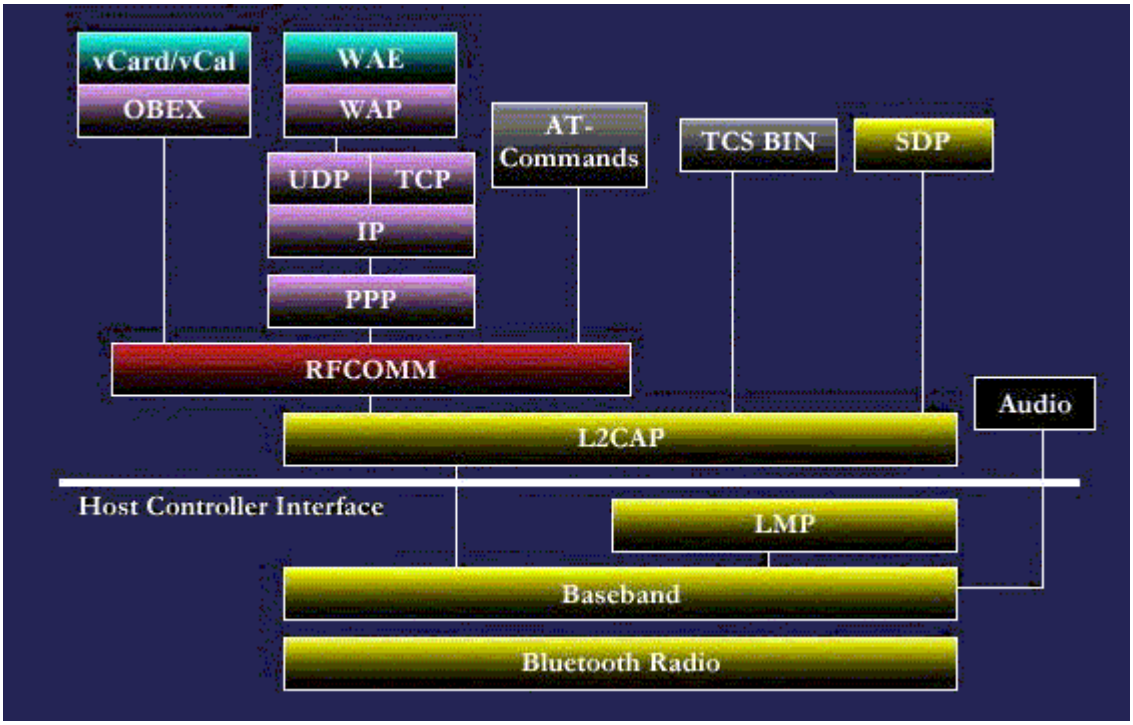
Tekniikan kehittyessä tiedonsiirtonopeudet ovat kasvaneet aikaisemmista versioista nykypäivänä käytettyyn versioon huimasti, nopeus on kasvanut 1 Mb:stä/s 24 Mb:iin/s. [4.]

Protokollat ja arkkitehtuuri

Bluetooth-laitteiden pitää pystyä tunnistamaan toisensa ja osata selvittää paritettavien laitteiden ominaisuudet. Pariutuminen tarkoittaa, että kahden laitteen välille luodaan yhteys. Laitteiden teknisten eroavien ominaisuuksien vuoksi ne on jaettu luokkiin, joista

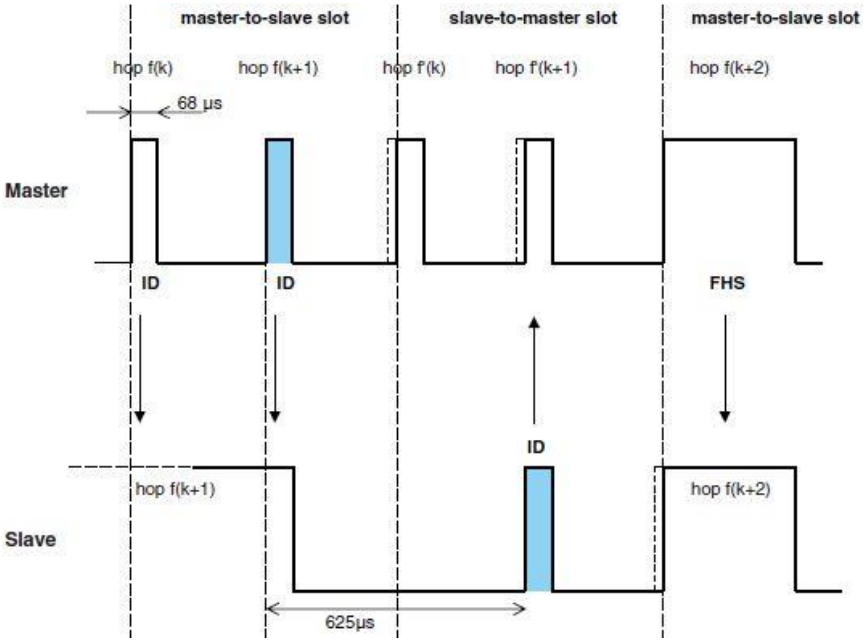
kukin tukee tiettyä joukkoa ominaisuuksia jolloin kommunikointiin laitteiden välillä pystytään käyttämään tiettyjä ohjelmistomoduuleita. [4.]

Muutamia näistä protokollista arkkitehtuuri kuitenkin määrittelee. Tarkoituksena on tässä esitellä tärkeimmät protokollat. Ydinprotokollat ovat baseband, LMP, L2CAP sekä SDP ja kaksi protokollaa, jotka korvaavat kaapelin ovat RFCOMM ja TCS.



Kuva 2. Bluetooth järjestelmän käyttämät protokollat. [5].

Baseband eli Kantataajuuskaista muodostaa fyysisen radiotaajuuslinkin Bluetooth-laitteiden muodostavan pikoverkkojen välille. Linkki muodostaa laitteiden välille yhteyden, jonka avulla tiedonsiirto tapahtuu. Bluetooth-radiotekniikka perustuu siihen, että paketit lähetetään määriteltyyn aikaan ja määritellyillä taajuuksilla. Kuvassa 3 on esitetty Bluetoothin paketinlähetys. Se muodostaa kaksi erilaista fyysistä linkkiä, jotka vastaavat tiettyjä kantataajuuspaketteja, SCO (Synchronous Connection-Oriented) ja ACL (Asynchronous Connectionless). SCO-paketit voivat sisältää dataa sekä ääntä, kun taas ACL-paketit vain dataa. [6, s. 7.]



Kuva 3. Aika-taajuusjako. [7].

Kuvassa 3 esitetään kahden laitteen välillä tapahtuva tunnistus. Isäntätila kuvaa toisia laitteita hakevaa laitetta, kun taas orja on haettava laite. Ensin isäntä lähettää kyselyn orjalle, joka odottaa ennen oman yksilöllisen tunnistenumeronsa lähettämistä yhden aikajakson 625 mikrosekuntia. Sen jälkeen lähetetään tunnistenumero isännälle, joka lähettää takaisin paketin muodostaen orjalaitteelle lähetystilan. Orjalaitteen lähettäessä paketin takaisin isännällä tämäkin siirtyy lähetystilaan. Kuvassa oleva sininen palkki on kellon kohdistussignaali, jonka avulla synkronoidaan orjalaite toimimaan isäntälaitteen kanssa samassa aika-avaruudessa.

Link Manager Protocol(LMP) on vastuussa linkin asettamisesta kahden Bluetooth-laitteen välille. Se hoitaa myös turvallisuuden linkkien välille käyttämällä todennusta sekä tiedon salausta. Sen tehtävänä on hoitaa eri virtatilojen hallinta ja kontrolloida eri toimintatapoja Bluetooth-laitteille sekä laitteiden yhteystilojen ollessaan liitettynä piko-verkkoon. [6, s. 7.]

Link Layer Control and Adaption Protocol(L2CAP) sovittaa ylemmän kerroksen protokollat kantataajuuskaistan avulla Bluetooth-laitteille. Se tarjoaa yhteyden ja yhteydettömän tilan data palvelut ylemmille kerroksille. Sen toiminnot ovat: datan multipleksaus, pakettien segmentointi ja uudelleen järjestäminen, yksisuuntaisen lähetyksen käsittely

sekä palvelun laadun takaaminen ylemmille kerroksille. L2CAP käytetään isännän yhteyksiin ACL linkin yli. [6, s. 8.]

Service Discovery Protocol (SDP) on tärkeä osa Bluetooth-viitekehystä. Nämä palvelut tarjoavat pohjan kaikille käytetyille malleille. Käyttämällä SDP:tä laitteen tiedot, mahdolliset palvelut ja laitteen ominaisuudet selvitetään yhdistyshetkellä. Selvitykseen muiden laitteiden tarjoamista palveluista protokolla käyttää uniikkia tunnistenumeroa nimeltään UUID, jonka muotona on 128-bittinen luku josta käytetään lyhyempää 16-bittistä versiota. [6, s. 8.]

Radio Frequency Communication (RFCOMM) on sarjaporttia emuloiva protokolla, joka perustuu ETSI 07.10 määritelmään. ETSI on eurooppalainen standardointijärjestö. Se tarjoaa tiedonsiirtokapasiteettia ylemmän luokan kerroksille jotka käyttävät sarjaporttia tiedonsiirtoon. [6, s. 8.]

Telephony Control Protocol(TCP) protokolla koostuu kahdesta eri osasta binäärisestä sekä AT-komennoista.

Binäärinen osa on bittiorientoitunut protokolla, joka määrittelee puheluiden kontrollisignaalit laitteiden välillä. AT-komentoja käytetään mobiililaitteiden ja modeemien ohjaamiseen, kun laitteita on monia. [6, s. 8.]

3 Android

Android (kuvassa neljä) on Linuxiin perustuva käyttöjärjestelmä, joka perustuu monoliittiseen ytimeen. Se tarkoittaa, että kaikki käyttöjärjestelmätoiminnot ovat samassa osoiteavaruudessa. Tämän ydinmallin edut ovat nopeus ja helppo kehitysmalli, haittoina taas ylläpitämisen monimutkaistuminen ja se että yhdenkin vakavan virheen ohjelmointi toiminnoissa aiheuttaa käyttöjärjestelmän kaatumisen. [8.]

Kuva 4. Android käyttöjärjestelmän tunnus.

Android-käyttöjärjestelmä suunniteltiin käytettäväksi kosketusnäyttölaitteille, kuten puhelimille sekä taulutietokoneille. Alun alkaen sitä alkoi kehittää Android Inc., jonka Google osti vuonna 2005. [10.]

Android-käyttöjärjestelmä on julkaisunsa jälkeen noussut käytetyimmäksi mobiililaitteiden käyttöjärjestelmäksi. Sen markkinaosuus on tällä hetkellä 70 prosenttia kaikista puhelimissa käytetyistä käyttöjärjestelmistä. [9.]

Taulukko 1. Matkapuhelinten markkinaosuudet vuoden 2012 lopussa. [9.]

Operating System	4Q12 Unit Shipments	4Q12 Market Share	4Q11 Unit Shipments	4Q11 Market Share	Year over Year Change
Android	159,8	70,1 %	85	52,9 %	88,0 %
iOS	47,8	21,0 %	37	23,0 %	29,2 %
BlackBerry	7,4	3,2 %	13	8,1 %	-43,1 %
Windows Phone / Windows Mobile	6	2,6 %	2,4	1,5 %	150,0 %
Linux	3,8	1,7 %	3,9	2,4 %	-2,6 %
Others	3	1,3 %	19,5	12,1 %	84,6 %
Total	227,8	100 %	160,8	100 %	41,7 %

Android on avoimen lähdekoodin järjestelmä, jota Google julkaisee Apache-lisenssin alla. Tämä lisenssi mahdollistaa ohjelmistojen vapaan muokkauksen ja julkaisun laitteiden valmistajille, operaattoreille ja innokkaille ohjelmistokehittäjille.

Android Inc.:in perustivat lokakuussa 2003 Andy Rubin, Rich Miner, Nick Sears ja Chris White, koska he halusivat tehdä laitteita, jotka ovat lähempänä ja tietoisempia käyttäjästään. Alun perin heillä oli tarkoitus tehdä kehittynyt käyttöjärjestelmä vain digitaalisille kameroille. Kameramarkkinoiden ollessa liian pienet he päättivät muuntaa käyttöjärjestelmän toimivaksi myös puhelimiin. [10.]

Vuoden 2005 elokuussa Google osti Android Inc.:in, jonka seurauksena yrityksestä tuli Googlen tytäryhtiö perustajien jäädessä töihin yritykseen. Uudella työnantajallaan Rubinin tiimi alkoi kehittää Androidista Linux-pohjaista käyttöjärjestelmää. [10.]

Loppuvuonna 2007 Google julkaisi Android-käyttöjärjestelmän ensimmäisen version yhteistyökumppaneidensa kanssa. Ensimmäinen versio oli nimeltään Cupcake. Silloin myös kerrottiin, että he aikovat luoda avoimen standardin mobiileille laitteille. Loka-kuussa vuonna 2008 ensimmäinen Android-käyttöjärjestelmää käyttävä laite julkaistiin. Kuvassa 5 ensimmäinen Android-laite. [10.]

Kuva 5. Ensimmäinen Android-laite HTC Dream. [11.]

Vuonna 2010 Google julkaisi oman Nexus-sarjansa, mikä tehtäisiin aina yhteistyössä valitun yhteistyökumppanin kanssa. Nexus-laitteina on julkaistu niin puhelimia kuin tabletteja.

Googlen perustaja Larry Page julkaisi maaliskuussa 2013 tiedotteen, että Androidin yksi aikakausi päättyy, kun Android-käyttöjärjestelmän kehitysjohtaja Andy Rubin siirtyy muihin tehtäviin Googlella. [12.]

Eri versiot ja niiden ominaisuudet

Tähän mennessä Google on julkaisut yhdeksän eri versiota käyttöjärjestelmästään. Jokainen niistä on saanut jonkun herkun nimen, kuten Cupcake, Donut, Ice Cream Sandwich ja Jelly Bean. Kuvassa kuusi on esitelty kuvina kaikki tähän mennessä ilmestyneet käyttöjärjestelmäversiot.

Kuva 6. Android eri käyttöjärjestelemäversiot.

Ensimmäisen Android-version julkaisussa käyttöjärjestelmä sai yleisen ilmeen, joka säilyi aina versioon 3.0 saakka. Mukana olivat myös widgetit (työpöydän pienoisohjelmat), monta aloitusruutua, yläreunan ilmoituspalkki, joka näytti laitteeseen saapuneet puhelut, viestit ja sähköpostit. Nämä olivat uusia ominaisuuksia, joita mikään käyttöjärjestelmä ei aikaisemmin ollut tarjonnut. [13.]

Kuva 7. Ruutukaappaus pienoisohjelmista.

Kuvassa 7 on ruutukaappaus puhelimen työpöydältä, jossa esitetään kaksi eri widget-pienoisohjelmaa.

Seuraava suurempi parannus käyttöjärjestelmään tapahtui versiossa 2.0/2.1 Eclair. Käyttöjärjestelmän ydin (kernel) päivitettiin ja laitteet nopeutuivat huomattavasti. Tämän version myötä Bluetooth 2.1:n tuki tuli Android-laitteille mahdollistaen Bluetoothin kytkemisen päälle ja pois sekä yhteydenluomisen ulkoisiin laitteisiin RFCOMM-moduulin avulla. [13.]

Seuraava versio käyttöjärjestelmästä oli Froyo. Sen mukana julkaistiin ensimmäinen Nexus–laite, jonka HTC kehitti yhdessä Googlen kanssa tuodakseen käyttäjille paremman käyttökokemuksen Androidista. Käyttöjärjestelmäpäivitys nopeutti laitteita ja mahdollisti soittamisen Bluetoothin välityksellä, mobiilverkkojen jaon ja Adobe Flash tuen. [13.]

Gingerbread-versiossa oli muuta uutta, tuki NFC -sirulle, verkkopuheluille (VOIP), uusi roskienkeräys sekä tuki erilaisille sensoreille kuten gyroskoopille. Versiossa laitteisiin tuli myös sovelluskehittäjätila, mikä helpotti sovellusten testaamista. [13.]

Honeycomb 3.0 oli uudistus, joka saapui helmikuussa 2011. Se oli ensimmäinen vain tablet-tietokoneille suunniteltu päivitys, jossa oli tuki useammalle prosessoriytimelle.

Versiossa oli uusi käyttöliittymän ulkoasu sekä kehittyneempi virtuaalinen näppäimistö, jok oli suunniteltu tableteille. [13.]

Lokakuussa 2011 julkaistiin seuraava versio nimeltään Ice Cream Sandwich. Käyttöjärjestelmässä Linux-ydin oli jo päivitetty versioon 3.0.1, mikä mahdollisti moniydintuen puhelimille ja suoraa tuen rinnakkainajolle. Mukana tuli myös monia muita uusia ominaisuuksia, kuten suora Wi-Fi, mahdollisuus sammuttaa taustalla suoritettavia sovelluksia, Android Beam, joka mahdollisti kahden laitteen välillä tapahtuvan lyhyen kantaman tiedonsiirron käyttämällä NFC-sirua, sekä mahdollisuus nauhoittaa Full HD tasoista kuvaa. Kuten aina aikaisempienkin päivitysten kanssa, käyttöliittymän graafinen ilme oli parantunut. [13.]

Viimeisin versio Androidista on nimeltään Jelly Bean, joka julkaistiin vuoden 2012 kesäkuussa. Suurimmat uudistukset olivat Google Now-hakukone, Android Beamin tuki Bluetoothille, monikanavaääni, tuki useille kielille sekä mahdollisuus kytkeä langattomat yhteydet päälle painamalla kuvakkeita pitkään. [13.]

Android-ohjelmointi

Android-sovelluskehitys on helppo aloittaa Googlen julkaisemien ja aktiivisesti päivitettyjen työkalujen avulla. Kehitys on mahdollista melkein kaikilla yleisimmillä käyttöjärjestelmillä ja ohjelmointikielissäkin on valinnan varaa, mikä taas edesauttaa sovelluksen kehittämistä eri järjestelmille. Yleisesti sovelluskehitystä tehdään Googlen omilla Java-kirjastoilla. Niiden dokumentointi on selkeää ja esimerkkien avulla ymmärtäminen on tehty aloittelijoillekin helpoksi. Käytännössä katsoen Androidia kirjoitetaan Javalla, mutta se ei ole Javaa. Ohjemointisyntaksi vastaa Javan omaa, mutta molemmilla on omat luokkakirjastonsa. Tästä Google on joutunut edesvastuuseen, kun Oracle haastoi Googlen oikeuteen ja väitti, että se on liiaksi Javaa. Java perustuu JVM-prosessivirtuaalikoneeseen, joka perustuu pinopohjaiseen arkkitehtuurin. Android käyttää omaa Dalvik-prosessivirtuaalikonetta, joka taas perustuu rekisteripohjaiseen arkkitehtuuriin. Tässä työssä käytetään Java-nimitystä, kun puhutaan Android SDK-ohjelmoinnista. Kehittäjäyhteistyön ollessa vahvaa lisäavun löytyminen sovelluskehittäjien tietopankeista, kuten Stack Overflowsta tai Android Developersista, on varmaa.

Android-käyttöjärjestelmällä varustetuille puhelimille kehittäminen on mahdollista kaikilla yleisimmillä käyttöjärjestelmillä. Windowseista tuettuja ovat XP, Vista, 7 ja 8. Mac OS X versioista tuettuja ovat 10.4.8 ja sitä uudemmat versiot, tosin ainoastaan Intelin piirisarjalla varustetut. Linux-kehitys onnistuu myös, kunhan käyttöjärjestelmän jakelu tukee 32-bittisen ohjelman suoritusta.

Kehitysympäristöistä ainakin Eclipse, Netbeans ja IntelliJ Idea toimivat Googlen tarjoamien asennettavien lisäosien kanssa. Tässä työssä tehty sovellus on kehitetty Eclipsellä, sen ollessa Netbeansin kanssa tutuin kehitysympäristö. Eclipsen käyttäminen kehitykseen vaatii vähintään version 3.4 sekä JDK version 5 tai 6 sekä Googlen oman lisäosan ADT pluginin, jonka avulla pystytään luomaan käyttöliittymä sovellukselle ja projektit Eclipsessä. [14.]

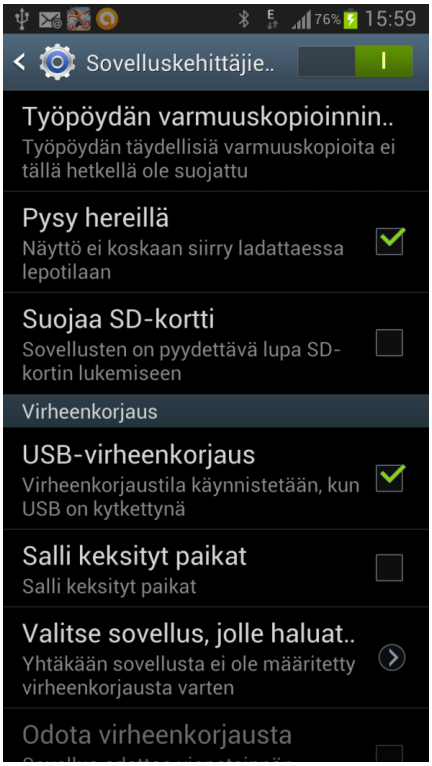
Java kehitys

Eclipsen rinnalle tarvitsi asentaa ADT-liitännäinen. Sen päätehtävänä on mahdollistaa projektien luonti, käyttöliittymän suunnittelu, liittää Android Framework APIit ja SDK-ohjelmistoihin sekä mahdollistaa suoritettavan projektin luonti Eclipsessä. [15.]

Ohjelmistokehitys vaati myös Android SDK:n asennettuna tietokoneelle, josta Eclipse osaa hakea oikeat kirjastot. Tämän avulla pystytään testaamaan sovelluksen toimintaa ja tarkistamaan, että ohjelma toimii virheettömästi (debugging). SDK sisältää myös ohjelmointiin tarvittavat kirjastot, dokumentaation sekä esimerkkisovellukset ja kurssit. Myös virtuaalinen emulaattori sisältyy pakettiin. Emulaattorilla ei tosin pysty emuloimaan kaikkia mahdollisia puhelimen lisälaitteita. [16.]

Kehityksen aloittaminen Eclipsessä vaatii vielä yhden lisäasennuksen, joka on nimeltään Platforms. Se on niin sanottu kääntäjä, joka valmistaa koodin tiettyä Android käyttöjärjestelmän versiota varten. Käytännössä .class-tiedosto käännetään Androidin Dalvik-virtuaalikoneen muotoon .apk-tiedostoksi, jotta se voidaan suorittaa laitteessa. [17.]

Sovelluksen testaaminen on mahdollista myös suoraan laitteessa, puhelimessa tai tablet-tietokoneessa. Silloin laitteen asetuksista aktivoidaan sovelluskehittäjätila.



Kuva 8 : Ruutukaappaus sovelluskehittäjätilasta

Tämän tilan ollessa päällä sovelluksia pystyy testaamaan suoraan laitteessa, mikä on huomattavasti emulaattoria nopeampi testausväline. Kuvassa kahdeksan on ruutukaappaus sovelluskehittäjätilasta. Se myös antaa suoran palautteen siitä, toimiiko ohjelma odotetulla tavalla.

C / C++ kehitys

Androidille kehittäminen onnistuu kirjoittamalla koodia myös C tai C++ kielillä. Tämä tosin vaatii erilaisen kirjaston, kuin Java-kehittäminen, nimeltään NDK. Native Development Kit on C-kielistä koodia varten asennettava kirjasto. NDK-koodatut tiedostot käännettään JNI-tiedostoiksi, jolloin suurin osa Androidin perustoiminnallisuudesta ei tule valmiina. Java Native Interface tiedostot määrittelevät sen miten C-kielinen koodi toimii Java-kielisen koodin rinnalla. Tämä mahdollistaa sen, että ohjelmoija pystyy kirjoittamaan oman Activity-luokan ja sille ominaiset elinkaarimetodit.

C- tai C++-kielillä kehittäminen on huomattavasti vaikeampaa kuin Java-kehitys. C- tai C++-kehityksessä ei ole roskienkerääjää, joka hoitaa käyttämättömien muuttujien poistamisen muistista. Eri kehityskirjastot vaikuttavat myös siihen, että viimeisimmät Android-versiot eivät ole tuettuja C- ja C++-kehityksessä.

Yleisesti ottaen Android ei hyödy kovin paljoa lisää tehokkuudessa käyttämällä näitä kieliä. Toki se tuo mukanaan lisää joustavuutta ohjelmointiin. Ohjelmat, jotka vaativat vähän muistia mutta enemmän prosessoritehoa, ovat näille kielille ominaisia kohteita, kuten pelit.

Avustettu sovelluskehitys

Androidille on olemassa myös muunlaisia sovelluskehitysympäristöjä, jotka eivät vaadi kehittäjältä niin paljon ohjelmointikokemusta kuin kaksi edellä mainittua tapaa kehittää sovelluksia. Siksi ne esitellään vain vaihtoehtona eikä niihin ole sen enempää tutustuttu.

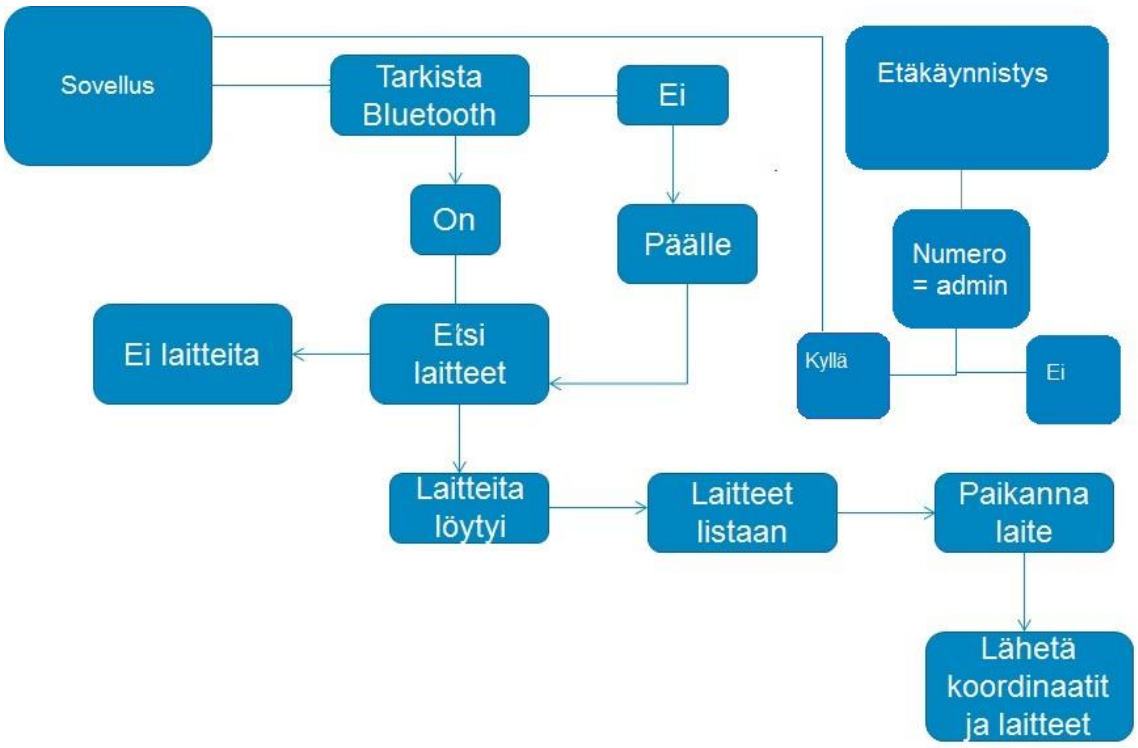
Google App Inventor on sovellus, jonka avulla pystytään luomaan yksinkertaisia sovelluksia drag-and-drop-tyylillä. Komponentteja vedetään vain ruudulle eikä toiminnallisuudesta tarvitse itse huolehtia ohjelma hoitaa sen puolestasi.

HyperNext Android Creator on edellä mainitun kaltainen sovellus, joka on tarkoitettu aloitteleville ohjelmoijille, jotka eivät tiedä sen enempää Javasta kuin SDK-kirjastoista. Se perustuu Applen kehittämään HyperCard-teknologiaan, jonka perusajatuksena on se, että se käsittelee ohjelmaa korttipinona, josta vain yksi kortti on näkyvissä kerrallaan. Ohjelmoinnissa käytetään englannin kaltaista kieltä, joka käyttää omia luokkiaan, jotka ovat Androidin alaluokkia. Niiden avulla ohjelma luo oman graafisen ilmeensä sekä toiminnallisuuden. [18.]

4 Seurantasovellus

Insinöörityön tarkoituksena oli tutustua Android-käyttöjärjestelmälle ohjelmointiin sekä tutustua Bluetooth-teknologiaan ja sen mahdollisiin käyttötarkoituksiin sovelluksissa. Yksi tarkoituksista oli työssä myös se, että siitä saataisiin toimiva sovellus, jota pystyttäisiin hyödyntämään käytännössä.

Idea ohjelmalla oli pystyä tarkistamaan verkon yli onko huonetilassa toisia henkilöitä ja näyttämään kartalla tämä tieto. Sovelluksen asentaminen laitteeseen mahdollistaa esimerkiksi myös sellaisen asian tarkastelun, onko lapsi saapunut ajoissa kotiin. Laitteita olisi tarkoitus etsiä Bluetoothin avulla. Rakenne selkiytyi jo hyvin varhaisessa vaiheessa. Oli siis hyödynnettävä Bluetoothia, paikannusta sekä puhelimen etäkäynnistystä.



Kaavio 1. Ohjelmaidean vuokaavio.

Kaaviossa 1 esitetään sovelluksen tapahtumajärjestyksessä. Käynnistyessään sovellus tarkistaa, onko Bluetooth käynnissä, koska haettaessa toisia laitteita asetuksen pitää olla päällä ja näkyvä muille laitteille. Tämän jälkeen tulee etsiä toisia laitteita ja niitä löytäessään lisätä ne listaan. Listaan lisäämisen tapahduttua tulee paikantaa oma laite web-sovellusta varten, jonka jälkeen laitteet mitkä olivat tulleet tunnistetuiksi, lähetetään web-sivulle, jossa ne näytetään. Laitteiden yksilölliset MAC-osoitteet, päivämäärä ja kellonaika sekä koordinaatit lähetetään palvelimelle. Tarkoituksena on muodostaa loki, jonka avulla käyttäjä pystyy seuraamaan tilassa löydettyjen laitteiden näkyvyyttä ja esiintymistiheyttä.

Ohjelman rakenteesta tuli selkeä ja suoraviivainen. Silti sovelluksesta tuli laaja, sen käyttäessä monia erilaisia tekniikoita ja rajapintoja, joiden yhteen sovittaminen vaati testaamista sekä suunnittelua.

4.1 Toteutus

Sovellus toteutettiin käyttäen Eclipse kehitysympäristöä Windows 7 käytöjärjestelmällä, johtuen siitä että kyseinen kehitysympäristö oli jo valmiiksi asennettuna. Ainoastaan Android ADT, SDK ja Platforms jouduttiin asentamaan koneelle, jotta Eclipse pystyy käyttämään Android-sovelluskehitystyökaluja ja emuloimaan sovellusta.

Sovellus on kehitettty testaamalla suoraan Android-puhelimesssa Samsung Galaxy S III:ssa. Sovellusta ei olisi voinut testata emulaattorilla, koska se ei olisi pystynyt emuloimaan kaikkia käytettäviä laitteita. Bluetooth laitteiden löytäminen tai niiden välinen kommunikointi ei onnistu emulaattorilla. Puhelimen tarkemmat ominaisuudet ja tiedot löytyvät valmistaja Samsungin web-osoitteesta. [19.]

Puhelimessa ohjelman suorittaminen vaatii koneelle asennettavat USB-laiteajurit, jotka löytyvät Samsungin omalta lataussivustolta. Ilman ajureita Eclipse ei löydä puhelinta eikä pysty etsimään virheitä sovelluksesta, vaikka se olisi yhdistetty tietokoneeseen USB-kaapelin avulla.

Android-sovelluskehitys on helppo aloittaa, koska tiiviin harrastelijayhteisön vuoksi lisäosien asentaminen ja oman projektin luonti oli helppoa. Esimerkit, jotka tulevat SDK:n

mukana, ovat selkeitä ja tarpeeksi laajoja esimerkkejä joissa on hyvin hyödynnetty eri rajapintoja.

Sovellus on ohjelmoitu Javalla, vaikka C tai C++ olisi voinut olla myös vaihtoehtona. Sovelluksen kehityksen aloittamisen yhteydessä punnittiin Javan ja C-kehityksen välillä päätyen kuitenkin Javaan, koska se tarjosi perustoiminnallisuuden helpommin. C-kehityksessä olisi pitänyt miettiä ohjelman tekemät JNI kutsut, jotka olisivat monimutkaistaneet sovelluksen rakennetta.

4.2 Sovelluksen rakenne

Sovellus koostuu kahdesta erillisistä suoritettavasta luokasta ja yhdestä vastaanottaja-luokasta. Se myös luo erillisiä säikeitä riippuen siitä, kuinka ohjelmaa aiotaan käyttää.

Sovellus on mahdollista käynnistää normaalilla tavalla niin kuin käyttöjärjestelmän muutkin sovellukset suoritetaan tai sitten sen käynnistäminen onnistuu lähettämällä tekstiviesti puhelimeen.

Ohjelman pääluokan käynnistyessä tarkistetaan, onko Bluetooth-yhteys käytössä. Jos se ei ole, niin yhteys laitetaan päälle. Pääohjelma hoitaa kaiken Bluetooth-toiminnan tarkistamalla, löytyykö laitteita, ja löytäessään lisää ne listaan. Kun pääohjelma ei löydä enempää laitteita, se käynnistää toisen suoritettavan ohjelman asettaen pääohjelman lepotilaan.

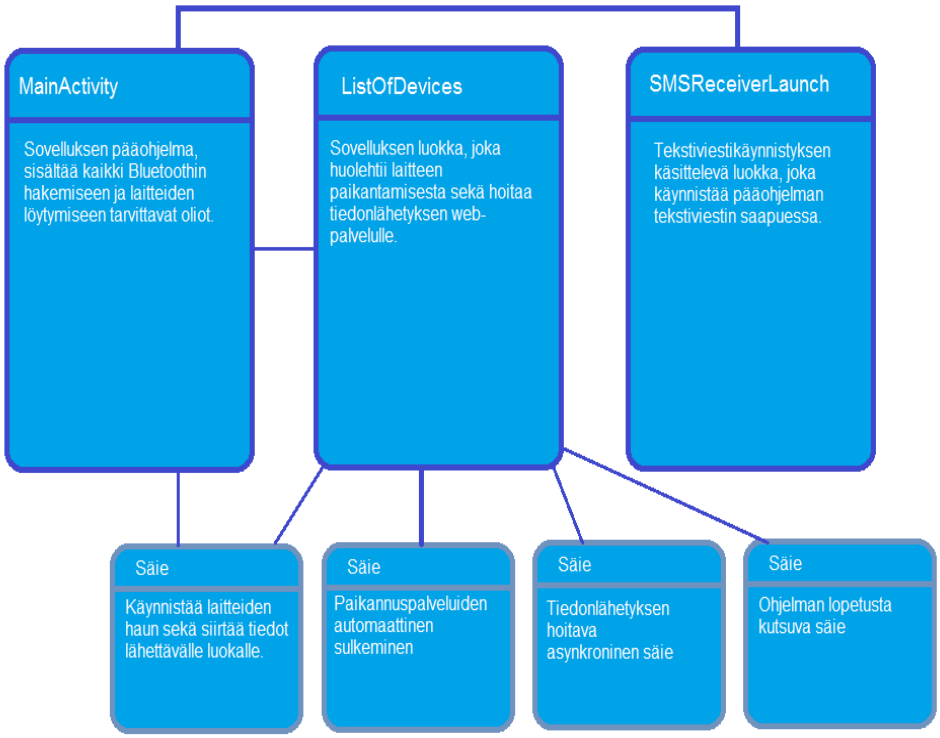
Käynnistyessään toinen suoritettava luokka hyppää suoraan puhelimen asetuksiin paikannuspalveluihin ja kysytään, haluaako käyttäjä paikannuksen käytössä vai ei. Tämä ohjelma edellyttää jonkun paikannuspalvelun olevan päällä. Jos ohjelmaa käytetään etäluennalla, oletetaan, että käyttäjä on jättänyt paikannuksen käyttöön. Tällöin ohjelma sulkee automaattisesti asetuksien paikannuspalvelut.

Toinen suoritettavista luokista saa käynnistyessään käytettäväkseen listan laitteista, jotka asetetaan näkymään näytöllä olevaan listanäkymään. Sen avulla ruudulla pystytään näyttämään erilaisia objekteja niin, että lista on vieritettävissä alas- tai ylöspäin. Samalla paikannuspalvelut paikantavat isäntälaitetta, minkä jälkeen koordinaatit tallen-

netaan ruudulla näkyvään kahteen tekstikenttään, jotta käyttäjän käyttäessään laitetta itse pystyy näkemään koordinaatit.

Tämän tapahduttua laite käynnistää asynkronisen lähetysluokan ja lähettää tiedot web-osoitteeseen, jossa php:lla tehty koodi tallentaa tiedot tekstitiedostoon luoden lokitiedoston. Sovelluksen mukana olisi tarkoitus käyttää karttapalvelua, jossa löydetyt laitteet näytettäisiin kartalla nastoina, joissa tietona olisi päivämäärä ja laite.

Kaaviossa 2 on esitelty seuransovelluksen luokkarakenne sekä ohjelman vaativa säikeistys.

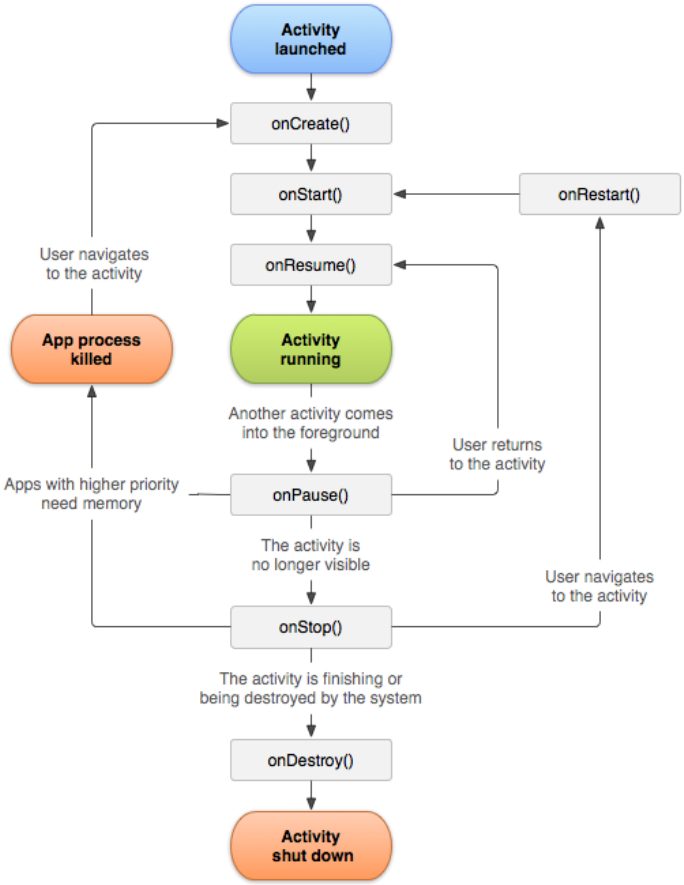


Kaavio 2. Sovelluksen luokkarakenne sekä säikeistys.

4.3 Android-luokat

Androidissa on mahdollista määritellä tavallisia luokkia samalla tavalla kuin perus-Javassakin, mutta niitä ei ole mahdollista suorittaa yksinään sovelluksina. Androidissa

suoritettavaa luokkaa kutsutaan nimellä Activity, ja se tarjoaa perustoiminnallisuuden automaattisesti sekä myös niin sanotut elinkaarimetodit. Activity onkin yhtä aikaa työn kohteena ja työn tekijänä. Activity hoitaa tiedonvälityksen toisten Activityjen välillä käyttämällä hyväkseen Intent-luokkaa. Intent-luokan olio hoitaa toisen ohjelmaosion käynnistyksen ja loppuun asti viemisen. Jollain abstraktiotasolla Android-sovelluksia voi verratta web-sovelluksiin, sillä jokaiselle Activitylle on määritelty oma käyttöliittymä ja uniikki nimi niin kuin web-sivullekin. Käyttäjä voi liikkua www-sivulta toiselle seuraamalla linkkejä, kun taas Android-sovelluksissa toiminnallisuus tapahtuu Intent-olion avulla. [20, s.77.]



Kuva 9. Android Activityn elinkaari. [21.]

Activity elinkaari on esitelty yksinkertaisesti kuvassa yhdeksän. Elinkaarimetodeja ovat onCreate, onStart, onResume, onRestart, onPause, onStop sekä onDestroy. Englanninkieliset nimet ovat hyvin kuvaavia kullekin metodille. onCreate-metodi toteutetaan aina sovelluksen käynnistyessä, joten siinä on tärkeää luoda käyttöliittymä sekä alustaa lis-

tat. Tämän jälkeen kutsutaan onStartia, joka kutsuttaessa tuo ruudun näkyviin. onPause tulee kutsutuksi, kun aikaisempaa Activityä kutsutaan onResume-metodin muodossa, jolloin tallennetaan dataa pysyvään muistiin tai laitetaan prosessoriteho sekä virtaa vaativat toiminnot pois. onStop-metodikutsu tulee yleensä siinä vaiheessa, kun toinen Activity tulee suoritusvuoroon tai kyseistä Activityä suljetaan. Tämän tapahtuessa Activity ei ole enää näkyvä käyttäjälle. Ohjelman jo sulkeutuessa kutsutaan onDestroyta, joka hoitaa ohjelman muistintyhjennyksen. [21.]

Jokaiselle ohjelmalle tarvitaan AndroidManifest.xml–tiedosto, joka on esitelty aina tämän nimisenä. Sen pitää sijaita juuressa, jotta käyttöjärjestelmä osaa hakea tiedostosta ohjelman suorittamista varten uniikin tiedostonimen. Tähän tiedostoon pitää esitellä kaikki ohjelman osat, kuten Activityt ja vastaanottajaluokat. [20, s.90.]

Kaikki sovelluksessa tarvittavat oikeudet paikannuspalveluihin, tekstiviestin vastaanottoon, Internetyhteyden luontiin sekä Bluetoothin käyttämiseen tuli esitellä tiedostoon. Sovellukselle piti sen luontivaiheessa määritellä minimiohjelmointirajapinta, johon vaikuttavat käytetyt toiminnot. Tämän sovelluksen miniversiovaatimus on Froyo.

Ohjelman käynnistävä MainActivity tuo kutsuttaessa onCreate-metodia käyttöliittymän, joka vastaa suunnittelun työkalun näkymää. Tälle luokalle suunniteltiin yksinkertainen käyttöliittymä, jossa on kolme painiketta. Metodissa pitää luoda myös kaikki kolme nappulaa kutsumalla Button-luokan olioita. Näin napit tuodaan ruudulle ja niihin yhdistetään xml-tiedostossa vastaavat määritykset. Jotta käytetyt napit reagoisivat koskетukseen, niille pitää asettaa myös Listener-luokan oliot.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
            listItems = new ArrayList<String>();
            startsearchButton = (Button)findViewById(R.id.search);
            discoverDeviceButton = (Button)findViewById(R.id.discover);
            changeActivityButton = (Button)findViewById(R.id.change);
            startsearchButton.setOnClickListener(new OnClickListener(){
            @Override
            public void onClick(View v){
                        startBluetooth();
            }
            });

}
```

Esimerkkikoodi 1. Napit ja tapahtumankäsittelijä.

Esimerkkikoodissa 1 esitellään, miten yhdistetään käytetyt napit suunniteltuun näkymään ja asetetaan yhdelle napeista kuuntelijaolio. Kuuntelijaolio toteuttaa painalluksen jälkeen yleisen metodin onClick(View v), johon on tässä esimerkkitapauksessa määritetty toteutettavaksi startBluetooth-metodi.

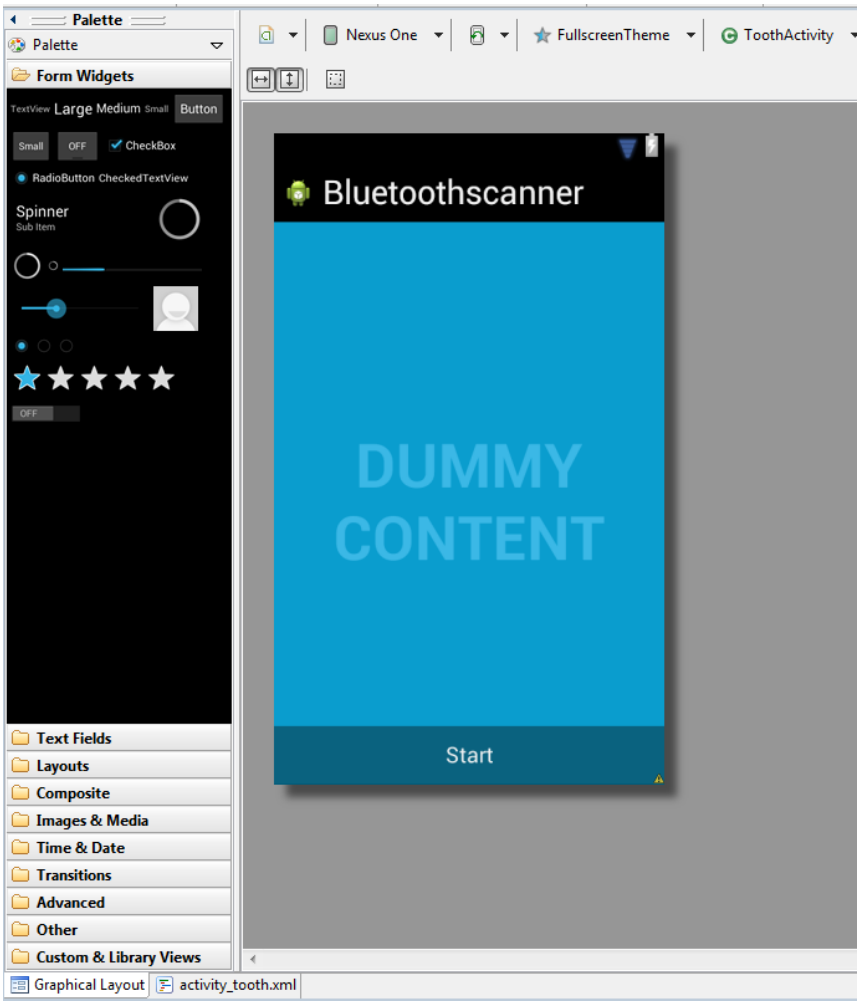
Koska ohjelman halutaan toimivan ilman käyttäjän interaktiivisuutta, luodaan myös säie, joka käynnistää sovelluksen toiminnot automaattisesti tietyn ajan kuluttua.

Seuraavaan Activityyn siirrytään, kun Bluetooth-yhteyksien etsintä päättyy. Kutsumalla metodia changeActivity, jossa annetaan tiedot löytyneestä listasta seuraava suoritettavalle osalle.

ListOfDevicesActivity-luokan käynnistyessä luodaan uusi näkymä, joka on erilainen kuin edeltäjänsä. Näkymässä on yksi painike, ListView-näkymä sekä kaksi tekstikenttää. ListView alustetaan listalla, joka saadaan parametreina luokan käynnistyessä. Koska ohjelma käyttää paikannusta, täytyy luoda paikannuksesta huolehtiva olio sovellukselle. Tietojen lähettämistä varten palvelimelle luodaan säie, joka ajastetaan nukkumaan kunnes tietty aika on kulunut, jonka jälkeen kutsutaan lähetysfunktiota.

4.4 Sovelluksen graafinen ilme

Ohjelmaan haluttiin tehdä yksikertainen käyttöliittymä, jotta sen käyttäminen olisi helppoa. Sovelluksen toimintamallin takia se ei vaadi graafiselta ulkoasulta paljoa. Käyttöliittymän värimaailma rakennettiin käyttöjärjestelmälle ominaisten piirteiden mukaan. Ohjelman käyttöliittymän värimaailman ollessa yksinkertainen lisättiin kaikille painikkeille samanlainen räätälöity ulkoasu. Android-sovelluskehitystyökalujen asennuksen yhteydessä Eclipseen tulee oma käyttöliittymän rakennustyökalu. Työkalun käyttöönotto oli pienen harjoittelun jälkeen vaivatonta ja hyvin virtaviivaista.



Kuva 10. Eclipsen käyttöliittymä työkalu ruutukaappaus.

Kuvassa 10 olevan työkalun avulla on helppoa suunnitella käyttöliittymä, siihen vedetään sivussa olevasta valikosta erilaisia komponentteja, joiden käyttäminen vaatii ohjelmakoodiin niiden lisäämisen ja luomisen sekä toiminnallisuuden rakentamisen. Valittuja komponentteja, esimerkiksi nappulaa, on mahdollista myös muokata omanlaisikseen xml-tiedostosta. Jokaiselle Activitylle generoituu automaattisesti oma xml-tiedosto, josta käyttöliittymän muokkaus on mahdollista, tai se voi käyttää jo olemassa olevaa asettelua.

Android-käyttöjärjestelmä lataa sovelluksen käynnistyessä xml-tiedostosta tarvittavat tiedot, jotta käyttöjärjestelmä osaa rakentaa oikeanlaisen käyttöliittymän kullekin Activitylle.

4.5 Bluetoothin käyttäminen

Bluetoothin käyttäminen sovelluksessa vaatii sitä, että puhelimessa on Bluetooth-sensori. Bluetooth-sensori on integroitu piiri puhelimessa, joka on suunniteltu esitellyn arkkitehtuurin mukaan, jotta se osaisi toimi Bluetoothille ominaisilla tavoilla. Käyttäminen vaatii, että se on kytketty puhelimen asetuksista, tai jos sitä ei ole tehty ohjelma kysyy käyttäjältä, halutaanko kyseinen sensori kytkeä. Kun sensori on aktivoitu, ohjelma ilmoittaa sen käyttäjälle.

```
public void discoverDevices(){
mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
mBCReceiver = new BroadcastReceiver(){
            public void onReceive(Context context, Intent intent){
                                                String action = intent.getAction();
            if(BluetoothDevice.ACTION_FOUND.equals(action)){
                        BluetoothDevice device =
                        intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
                        tToast(device.getName() + device.getAddress());
                        String s = new String(device.getName() + device.getAddress());
                        createList(s);
                        }
            }
            };
IntentFilter intFilter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
registerReceiver(mBCReceiver,intFilter);
mBluetoothAdapter.startDiscovery();
}
```

Esimerkkikoodi 2 : Bluetooth laitteiden etsintä

Toisten laitteiden etsintä aloitetaan luomalla vastaanottajaluokan olio, jolle annetaan sensori (esimerkkikoodi 2). Olioon on määritelty, millä perusteilla laite on löytynyt. Sensorille määritellään niin sanottu toiminto, joka etsii samanlaisia toimintoja eli avoimia Bluetooth-yhteyksiä. Sen jälkeen on mahdollista suorittaa etsintä. Tiedot löytyneistä laitteista siirretään listaan, johon laite lisätään MAC-osoitteensa muodossa, josta etsimismetodi myös tunnistaa laitteet.

4.6 Sijainnin määritys

Sijainnin määrittäminen voidaan tehdä matkapuhelinverkon ja avoimien Wi-Fi-yhteyksien avulla tai käyttämällä tarkempaa GPS-paikannusta. Näiden yhdistelmällä saadaan tarkempi sijainninmääritys. Ohjelmointirajapinta edellyttää paikannuspalveluiden käyttämistä ohjelmassa. Niiden tulee olla kytketty asetuksista tai olleessaan poissa käytöstä ohjelma käynnistää asetukset, josta käyttäjä voi valita haluamansa sensorit päälle. Jotta käyttöjärjestelmä pystyy hyödyntämään palveluita, tulee paikannusoliolle antaa sisältöolio (Context). Sisältönä on tieto siitä, mitä paikannusta tulee käyttää kussakin tapauksessa.

Ohjelmaan on kirjoitettu metodeja, jotka tarkistavat, mitkä sensorit ovat käytössä, tai onko niiden tila on muuttunut edellisestä kerrasta.

Kutsuttaessa paikannusmetodia tarkistetaan ensin sensorin viimeisin sijainti, jonka jälkeen ohjelma tarkentaa sijaintia etsimällä uusia päivityksiä sensoreille. Koordinaatit lisätään ruudulla oleviin tekstikenttiin, jossa ne ovat leveyspiirin ja pituuspiirin muodossa. Näiden avulla pystytään määrittelemään sijainti

4.7 Tiedonlähetys

Ohjelman suorituksen kuormituksen vähentämiseksi http-yhteyden luonti tehdään erillisessä asynkronisessa luokassa. Se luo uuden säikeen, joka hoitaa yhteydenoton taustaajona. Asynkronisen tehtäväluokan olio luodaan ja kutsutaan, kun sovellus on paikantanut sijainnin. Jotta pystytään ottamaan yhteys ulkoiseen web-palveluun, valmiiseen tietokantaan tai web-sivulla toimivaan sovellukseen, tämän luokan käyttäminen on välttämätöntä. Tämä tulee kyseeseen tilanteessa, jossa tarvitaan tiedontallennusta tai ohjelman pitää hyödyntää Internetiä.

```
public void postData(String deviceNames) {
HttpClient httpclient = new DefaultHttpClient();
HttpPost httppost = new
HttpPost("http://users.metropolia.fi/~tuomasm/c/sijainti/teho3.php");
            try {
            List<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(4);
            nameValuePairs.add(new BasicNameValuePair("Date",
            java.text.DateFormat.getDateTimeInstance()
            .format(Calendar.getInstance().getTime())));

            nameValuePairs.add(new BasicNameValuePair("Device", deviceNames));
            nameValuePairs.add(new BasicNameValuePair("Latitude",
            Double.toString(location.getLatitude())));
            nameValuePairs.add(new BasicNameValuePair("Longitude",
            Double.toString(location.getLongitude())));
            httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
            HttpResponse response = httpclient.execute(httppost);
                        if(response != null){
                                    System.out.println("call ok");
                        }
            } catch (ClientProtocolException e) {
            } catch (IOException e) {
            }
}
```

Esimerkkikoodi 3. Yhteyden muodostus.

Yhteyden muodostaminen tapahtuu esimerkkikoodissa 3 luomalla asiakasohjelmaolio, joka alustetaan lähetysoliolla. Sille annetaan osoitteeksi koulun palvelimelle omassa käyttäjähakemistossa sijaitseva php-koodi. Olioon kirjoitetaan ennen yhteydenottoa tieto päivämäärästä ja kellonajasta, löytyneet laitteet sekä koordinaatit. Tämän jälkeen ohjelma suorittaa lähetyksen ja onnistuessaan tulostaa ruudulle näkyviin viestin siitä. Mikäli suoritus epäonnistuu, poikkeukset käsitellään ohjelmassa.

Php-koodin tarkoitus on yksinkertainen; se luo tiedoston ja lisää sinne sovellukselta saamansa tiedot. Tietoja ei päällekirjoiteta vaan tarkoituksen mukaisesti luodaan aina uusia rivejä, jotta saadaan aikaiseksi lokitiedosto. Koodin on tarkoitus simuloida palvelimella toimivaa sovellusta, joka tilastoi laitteiden näkymistä tietyillä aikaväleillä. Palvelimelle olisi hyvä olla vielä erillinen karttasovellus, jossa laitteiden esiintyminen eri paikoissa näytettäisiin.

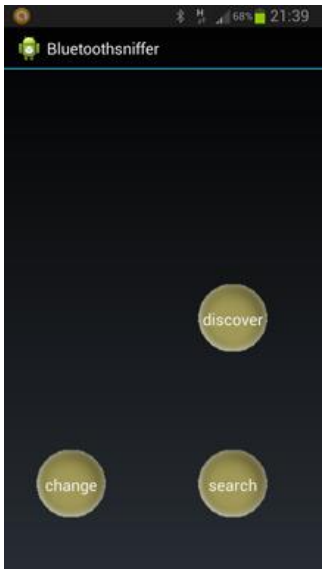
4.8 Tekstiviestikäynnistys

Jotta sovellus toimisi suunnitellulla tavalla, täytyy ohjelman pystyä käynnistymään niin, että käyttäjän ei tarvitse olla itse paikalla. Tätä varten ohjelmaan on luotu erillinen vastaanottajaluokka, joka toimii taustalla itsenäisesti. Luokan toiminta on kovin yksinkertainen eikä vaadi suuria resursseja laitteelta. Se osaa käynnistää sovelluksen saadessaan tekstiviestin. Puhelimen saadessa viestin numeroa verrataan pääkäyttäjän puhelinnumeroon, ja jos ne täsmäävät, kutsutaan pääohjelmaa, joka saa suoritusvuoron.

Tekstiviestikäynnistys-luokka on luokka, jonka pitää periä Broadcast Receiver-luokka, joka on Androidin yleinen lähetys- ja vastaanottajaluokka. Luokan pitää pystyä toteuttamaan myös tekstiviestin vastaanotto ja sillä pitää olla mahdollisuus verrata kahta numeroa toisiinsa puhelinnumeroina. Tämä toteutetaan PhoneNumberUtils-olion avulla, jolla pystytään vertaamaan numeroita keskenään, minkä jälkeen luodaan uusi Intent-olio, jonka avulla käynnistetään MainActivity-luokka.

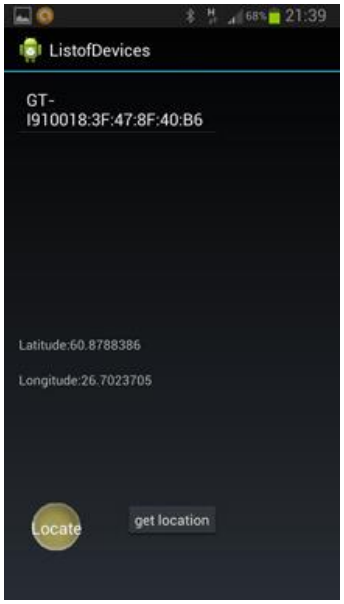
4.9 Valmis sovellus

Sovelluksella päästiin haluttuun lopputulokseen eli saatiin toimiva sovellus, joka on käytettävissä Android-puhelimissa ja -tableteissa. Haasteina työn tekemisessä oli täysin uuden arkkitehtuurin opettelu sekä uusien kehitystyökalujen käyttäminen. Ohjelmalle asetetut vaatimukset, kuten lähellä olevien laitteiden etsiminen, isäntälaitteen paikantaminen, tiedonsiirto web-palveluun ja lokikirjan luonti, olivat kaikki valmiissa sovelluksessa. Alun perin vähän ongelmana ollut etäkäyttö saatiin valmiiseen sovellukseen toimimaan, ja sitä on testattu myös käytännössä.



Kuva 11. Ruutukappaus sovelluksen alkuvalikosta.

Sovelluksen alkuvalikko kuvassa 11 on yksinkertainen, mutta sovelluksen toimintaperiaatteiden mukainen. Ulkoasuun on lisätty omat räätälöidyt painikkeensa, jotka tuovat ohjelmalle persoonallisen ilmeen. Puhelimen laukaiseminen tekstiviestillä tuo ruudun näkyviin ja alkaa automaattisesti etsiä lähistöllä olevia laitteita.



Kuva 12. Paikannuksen ja lähetyksen hoitava tila.

Sovelluksen toinen näkymä eroaa alkuvalikosta asettelullaan sekä toiminnallisuudellaan. Kuvassa 12 nähdään erillisen listanäkymän, johon on tuotu löydetty laite sekä koordinaattikentät. Käyttäjää varten tulostuvat pikaviestit kertovat, kun paikannus on tapahtunut sekä yhteys ja lähetys on hoidettu onnistuneesti. Kuvassa 12 esimerkkitulostuksena näkyy onnistunut koordinaattien haku.

5 Yhteenveto

Avoimen lähdekoodinsa ja aloittamisen helppouden takia Android-ohjelmistokehitys on jokaisen harrastajan ulottuvilla. Käyttöjärjestelmälle tehtävä kehitys tulee varmasti lisääntymään, koska ohjelmoinnissa käytetään Javaa, joka on yleisesti käytetyin ohjelmointikieli. Osaajia riittää, ja mobiilimarkkinoiden kasvaessa ja viedessä markkinoita työpöytäympäristöiltä tulee entistä suurempi tarve tehdä aina mukana pidettäville laitteille uusia sovelluksia.

Android-sovelluskehityksen aloittaminen oli helppoa, koska tunsin etukäteen Javaa ja sen eri muotoja. Ohjelmointi tapahtuu samanlaisella syntaksilla, mutta tutustuminen Activity-luokkaan ja sen elinkaarimetodeihin oli välttämätöntä ohjelman toimivuuden takaamiseksi. Koska kehitysympäristö oli minulle entuudestaan tuttu saatoin aloittaa työn tekemisen verrattain nopeasti. Alussa suurimmat vaikeudet olivat graafisen ilmeen suunnitteluun käytettävästä lisäosasta johtuvia.

Kirjastojen käyttäminen oli samanlaista kuin Javassa, mikä ei tuottanut ongelmia. Vaikeaa oli hahmottaa eri elinkaarimetodien käyttämistä, mitä tarvitsee tehdä milloinkin. Ongelmia tuli siinä vaiheessa, kun sovellusta sammutettiin tai siirryttiin Activity-luokasta toiseen. Vääränlainen ohjelmointi aiheutti usein sovelluksen kaatumisen. Ongelmista selvittiin paneutumalla syvemmin eri olioiden toimintaan ja siihen, milloin ne tulee vapauttaa työmuistista.

Työssä käytettävät paikannuspalvelut ja niiden käyttäminen aiheuttivat seuraavaksi ongelmia työn edetessä. Paikannuspalveluiden käyttäminen vaatii jokaisella käyttökerralla oikeutuksen puhelimen asetuksista. Asetukset avataan joka kerralla, kun alustetaan Location Manager -olio valmiiksi paikannusta varten. Manuaalinen sulkeminen onnistuu helposti käyttäjälle, mutta etäkäytössä sitä ei ole mahdollista tehdä. Etäkäyttöä varten rakennettiin oikeutuksen kysely erillisessä säikeessä, jolloin asetukset suljetaan, jos käyttäjä ei ole valitsemassa eri paikannuspalveluita päälle. Tämä tosin vaatii sen, että käyttäjä on asettanut vähintään Wi-Fin ja kännykkätorniasetukset käyttöön tai GPS-paikannuksen. Ratkaisu ei ole täydellinen mutta tarpeeksi toimiva, koska Google poisti automaattisen asetusten käyttämisen Android-version 2.2 jälkeen.

Työssä tarkoituksena oli opetella Android-käyttöjärjestelmälle ohjelmointia sekä tutustua Bluetoothiin. Työssä selvitettiin myös Bluetoothin mahdollisia ominaisuuksia käyttäjien seuraamiseen. Bluetoothin ominaisuudet tällaisessa sovelluksessa ovat mainiot ja toimivuus moitteeton. Laitteiden Bluetooth-ominaisuuden pitää olla kytketty, jotta niitä voidaan löytää sovelluksella. Sovellusta suunnitellessa ajattelin mahdollista käyttötarkoitusta niin, että se olisi osana automaattista kellokorttijärjestelmää.

Työhön voisi tehdä palvelimella pyörivän lisäosan, joka piirtäisi kaavioita laitteen esiintymistiheydestä tietyssä paikassa ja siitä kuinka kauan laite on tietyssä paikassa sekä mihin aikaan laite yleensä on paikassa. Palvelimella voisi myös olla Googlen Javascript Apin tarjoaman karttapalvelusovelluksen, jossa näytettäisiin laitteet kartalla ja niiden esiintymisuseus.

Työssä suurimpana apuna olivat varmastikin Internetin kehittäjäyhteisöt, joista etsimällä sai avun suurimpaan osaan ongelmista. Myös Googlen oma dokumentointi on selkeää ja josta löytyvät yksinkertaiset esimerkit tuovat monet asiat hyvin ilmi.

Työn tekeminen oli mielenkiintoista ja hyödyllistä sen lisätessä osaamista uuden käyttöjärjestelmän ja siihen ohjelmoinnin suhteen. Käyttöjärjestelmään ja sen toimintaan perehtyminen synnytti työn edetessä kysymyksiä, joihin vastauksen löytäminen oli merkittävää myös muun ohjelmointityön kannalta. Mielenkiintoista oli myös perehtyä syvemmin Bluetoothin teknologiaan ja sen tarjoamiin mahdollisuuksiin. Työ tehtiin itse tutkimus- ja opiskelutyönä, mikä oli palkitsevaa aina onnistumisten ja epäonnistumisten myötä.

Lähteet

1 CNN. 2000 June 8. Ericsson demos first Bluetooth phone. Verkkodokumentti. <http://archives.cnn.com/2000/TECH/computing/06/08/bluetooth.phone.idg/>.Luettu 23.5.2013.

2 Bluetooth-got-its-name. 2008. Verkkodokumentti. <http://www.eetimes.com/electronics-news/4182202/Tech-History-How-Bluetooth-got-its.name>.Luettu 23.5.2013.

3 Bluetooth. 2010. Verkkodokumentti. Wikipedia. <http://en.wikipedia.org/wiki/Bluetooth>.Luettu 23.5.2013.

4 Radio Electronics. Verkkodokumentti. <http://www.radio-electronics.com/info/wireless/bluetooth/radio-interface-modulation.php>.Luettu

5 Palowireless. Verkkodokumentti. <http://www.palowireless.com/infotooth/images/tutorial_images/spec_stack.gif>. Luettu 23.5.2013.

6 Mettälä, Riku. 1999. Bluetooth Protocol Architecture. <http://www.bluetooth.org/>.Luettu 23.5.2013.

7 WikiDot. 2011. Verkkodokumentti. <http://kcchao.wikidot.com/bluetooth-baseband>. Luettu 23.5.2013.

8 Android operating system. 2013. Verkkodokumentti. Wikipedia. <http://en.wikipedia.org/wiki/Android_(operating_system)>.Luettu 23.5.2013.

9 Android and iOS Combine for 91.1% of the Worldwide Smartphone OS Market in 4Q12 and 87.6% for the Year, According to IDC. 14th February 2013. IDC. Verkkodokumentti. <http://www.idc.com/getdoc.jsp?containerId=prUS23946013>.Luettu 23.5.2013

10 Andro Love. 2013. Verkkodokumentti. <http://androlove.com/history-of-android.html>.Luettu 23.5.2013.

11 Tudo Cellulat. 2013. Verkkodokumentti <http://www.tudocelular.com/new_files/images/global/HTC-Dream_39663_1.jpg>. Luettu 23.5.2013.

12 Google's Android chief Andy Rubin steps down. 13th March 2013. Verkkodokumentti. Reuters. <http://www.reuters.com/article/2013/03/13/us-google-android-idUSBRE92C0XC20130313>.Luettu 23.5.2013.

13 Android version history. 2013. Verkkodokumentti. Wikipedia. <http://en.wikipedia.org/wiki/Android_version_history>. Luettu 23.5.2013

14 Guidry Mike. Programming for the Android Platform. 2013 .Verkkodokumentti. <http://eagle.phys.utk.edu/guidry/android/minimumMachinesAndSoftware.html>. Luettu 23.5.2013.

15 Android Developers ADT Plugin. 2013. Verkkodokumentti, <http://developer.android.com/tools/sdk/eclipse-adt.html>.Luettu 23.5.2013.

16 Android Developers SDK. 2013. Verkkodokumentti. <http://developer.android.com/sdk/index.html>. Luettu 23.5.2013.

17 Android Developers Platforms. 2013. Verkkodokumentti. <http://developer.android.com/tools/revisions/platforms.html>. Luettu 23.5.2013.

18 Android Software Development. 2013. Verkkodokumentti. Wikipedia. <http://en.wikipedia.org/wiki/Android_software_development>. Luettu 23.5.2013

19 Samsung Galaxy S3. 2013. Verkkodokumentti. <http://www.samsung.com/hk_en/consumer/mobile/mobile-phones/smartphone/GT-I9300MBDTGY-spec>. Luettu 23.5.2013

20 Mednieckс, Z. Dornin, L. Blake, G. Nakamura, Meike & Masumi. 2012. Java Programming for the New Generation of Mobile Devices Android Programming. O'Reilly Media Inc.

21 Android Developers. 2013 Verkkodokumentti. <http://developer.android.com/reference/android/app/Activity.html>. Luettu 23.5.2013.