Antti Pääkkönen

ALUSTARIIPPUMATON MOBIILISOVELLUS BLUETOOTH LOW ENERGY -KOMMUNIKOINTIIN

Insinöörityö
Kajaanin ammattikorkeakoulu
Tekniikan ala
Tietotekniikan koulutusohjelma
Kevät 2014

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ
TIIVISTELMÄ

Koulutusala Tekniikka ja liikenne	Koulutusohjelma Tietotekniikan koulutusohjelma
Tekijä(t) Antti Pääkkönen	
Työn nimi Alustariippumaton mobiilisovellus Bluetooth Low Energy -kommunikointiin	
Vaihtoehtoiset ammattiopinnot Ajoneuvojen tietojärjestelmät	Toimeksiantaja Exéns Development Oy, Petri Ingalsuo
Aika Kevät 2014	Sivumäärä ja liitteet 58

Tämän insinöörityön tavoitteena oli suunnitella ja toteuttaa BLE (Bluetooth Low Energy) -teknologiaa hyödyntävä mobiilisovellus Android-pohjaiselle älypuhelimelle. Työn toteuttamiseen kuului graafisen käyttöliittymän sekä tämän avulla ohjattavien toiminnallisuuksien ohjelmointi. Insinöörityön toimeksiantajana toimi kajaanilainen sulautettuja laitteita sekä järjestelmiä valmistava yritys Exéns Development Oy.

Työ suunniteltiin toteutettavaksi HTML5-sovelluksena ja JavaScript-ohjelmointikielellä. Tavoitteena oli saada aikaiseksi mahdollisimman laitteisto- ja käyttöjärjestelmäriippumaton sovellusratkaisu, joka tulevaisuudessa voidaan ottaa käyttöön myös muilla laitteisto- ja käyttöjärjestelmäalustoilla. Työn alkuvaiheissa selvitettiin tarkemmin HTML5- ja JavaScript-ohjelmointikielten soveltuvuus työn suorittamiseen. Selvityksessä mahdollisesti ilmenneiden toteutustavan esteiden vuoksi täytyi harkita myös vaihtoehtoisia toteutustapoja mobiilisovellukselle.

Tässä työssä esitellään teoriatasolla Android-mobiilikehitystä, laitteistoriippumatonta PhoneGap-mobiilikehitystä ja Bluetooth-teknologiaa. Työssä vertaillaan natiivin eli laitteistokohtaisen Android-mobiilikehittämisen sekä PhoneGapilla toteutetun laitteistoriippumattoman mobiilikehittämisen eroja toisiinsa.

Työn lopputuloksena saatiin toimiva, graafisen käyttöliittymän omaava mobiilisovellus Android-älylaitteille, joiden käyttöjärjestelmäversio on vähintään BLE-tuen tarjoama 4.3. Käyttöliittymän kautta saadaan etsittyä BLE-laitteita, paritettua laitteet keskenään, muodostettua yhteys laitteisiin automaattisesti sekä lähettämään ohjauskomento kohdelaitteelle.

Kieli	Suomi
Asiasanat	Mobiilisovellus, PhoneGap, BLE, Android, HTML5, JavaScript
Säilytyspaikka	☒ Verkkokirjasto Theseus ☐ Kajaanin ammattikorkeakoulun kirjasto

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

THESIS
ABSTRACT

School Engineering	Degree Programme Information Technology
Author(s) Antti Pääkkönen	
Title Cross-platform Mobile Application for Bluetooth Low Energy Communication	
Optional Professional Studies Vehicle Information Systems	Commissioned by Exéns Development Oy, Petri Ingalsuo
Date Spring 2014	Total Number of Pages and Appendices 58

The aim of this Bachelor's thesis was to design and implement a mobile application for Android mobile devices exploiting BLE (Bluetooth Low Energy) technology. The implementation included programming the graphical user interface for the application and the functionality controlled by it. This thesis was commissioned by Exéns Development Oy.

The work was planned to be done in HTML5 and JavaScript programming languages. The goal was to create a cross-platform solution, because the application is supposed to be taken on other hardware and operating system platforms in the future. Therefore, the suitability of HTML5 and JavaScript programming languages to develop the software was researched at the beginning of the work. Due to the potential obstacles of research, alternative ways of implementing the project had to be considered.

Also, the thesis introduces the theories of Android mobile development, cross-platform mobile development with PhoneGap and theories of Bluetooth technology. This thesis compares differences between native (hardware-specific) Android development and cross-platform development with PhoneGap.

As a result, a functional application was developed for Android mobile devices that are running at least with version 4.3, which has the BLE support for device. BLE devices can be found via application, devices can be paired with each other, a connection can be established between devices automatically and the control command can be transmitted to the target device.

Language of Thesis	Finnish
Keywords	Mobile, Application, PhoneGap, BLE, Android, HTML5, JavaScript
Deposited at	☒ Electronic library Theseus ☐ Library of Kajaani University of Applied Sciences

ALKUSANAT

Haluan kiittää tämän insinöörityön mahdollistamisesta Exéns Development Oy:tä, joka etsi työn suorittajaa Kajaanin ammattikorkeakoululta. Yritys tarjosi hyvät ja erittäin joustavat puitteet insinöörityön toteutukselle sekä yrityksen tiloissa että etätyöskentelymahdollisuutena. Erityisesti haluan kiittää yhteyshenkilönä toiminutta Petri Ingalsuota, jonka kanssa työskentely oli helppoa ja miellyttävää omista alkuvaikeuksista huolimatta. Haluan kiittää yliopettaja Pentti Romppaista insinöörityön ohjauksesta ja ystäviäni, jotka ovat kannustaneet minua työn suorittamisen aikana.

Kajaanissa, 4.5.2014

Antti Pääkkönen

SISÄLLYS

SYMBOLILUETTELO

ACL	Asynchronous Connection Oriented link, Bluetooth-teknologiassa käytetty kanavatyyppi
ADT	Android Developer Tools, Android-kehitystyökalut Eclipse-sovellusympäristöön
API	Application Programming Interface, ohjelmointirajapinta eri ohjelmien tai toimintojen välille
ATT	Attribute Protocol, palvelupohjainen arkkitehtuuri BLE-teknologiassa
BLE	Bluetooth Low Energy, Bluetoothin vähävirtainen tiedonsiirto-teknologia
CSS	Cascading Style Sheets, web-sivuilla ulkoasun muotoiluun käytettävä merkintäkieli
GATT	Generic Attribute Profile, luo yhteiset toiminnot ja puitteet datan lähetykselle sekä tallennukselle BLE-teknologiassa
HTML	Hypertext Markup Language, web-sivuilla käytetty merkintäkieli
SCO	Synchronous Connection Oriented link, Bluetooth-teknologiassa käytetty kanavatyyppi
SDK	Software Development Kit, kehitystyökalu sovellusten luomiseen
SIG	Special Interest Group, lyhyen kantaman radioteknologioiden kehittämiseen perehtynyt ryhmittymä
SQL	Structured Query Language, kyselykieli tietokantajärjestelmiin
VM	Virtual Machine, ohjelmallisesti toteutettu virtuaalikone
XML	Extensible Markup Language, tiedon kuvaamisessa käytetty merkintä-kieli

1 JOHDANTO

Mobiilikehitys on viime vuosien aikana ollut nopeaa, ja kilpailu markkinoilla eri laitevalmistajien välillä on kiivasta. Eri laitteistoille on tarjolla paljon pelejä ja sovelluksia erilaisia käyttötarkoituksia ajatellen. Näiden molempien kehittäminen on tullut avoimeksi kaikille mobiilikehittämisestä kiinnostuneille ja tarjolla on alustakohtaisesti kattavia kehitysmahdollisuuksia. Uudet teknologiat sisältävät myös mielenkiintoisia kehityskohteita, jotka tarjoavat uusia haasteita sovelluskehittäjille.

Exéns Development Oy on vuonna 1990 rekisteröity kajaanilainen tuotekehitys- ja valmistusyritys, joka kehittää sulautettuja laitteita sekä järjestelmiä. Exénsin keskeisimpinä osaamisalueina ovat eritoten elektroniikkaan, ohjelmistoihin ja mekaniikkaan liittyvät ratkaisut, joita täydentävät asiakasorganisaatioille tarjottavat tukipalvelut. Yrityksellä on Kajaanin toimipisteen lisäksi toimintaa myös Oulussa, ja se työllistää yhteensä noin 20 henkilöä. [1.] [2.]

Tämän insinöörityön tavoitteena oli toteuttaa Exéns Development Oy:lle mobiilisovellus, jolla pystytään kommunikoimaan vähävirtaisen BLE (Bluetooth Low Energy) -laitteen kanssa. Toteutettavalla sovelluksella on tarkoitus pystyä luomaan yhteys kohdelaitteisiin automaattisesti, lähettämään näille ohjauskomentoja ja lukemaan laitteilta saatavaa tietoa. Mobiilisovellus on osa Exéns Development Oy:n asiakasorganisaatiolle toteutettavaa suurempaa kokonaisuutta.

Työn tavoitteena oli tutkia mahdollisia laitteistoriippumattomia toteutustapoja ja lopulta toteuttaa mobiilisovellus Android-älylaitteelle. Lähtökohtana oli saada kyseinen sovellus toteutettua tulevaisuudessa myös iOS- ja Windows Phone -älypuhelimille. Työssä tutkittiin laitteistoriippumatonta PhoneGap-sovelluskehitystä ja vertailtiin tätä teoriatasolla Android-natiivisovellusten toteuttamiseen. Lopulta toteutettiin mobiilisovellus Android-laitteelle, jolla pystytään yhdistämään tavoiteltavaan kohdelaitteeseen ja lähettämään laitteelle tietoa BLE-yhteyden kautta.

2 ÄLYPUHELINALUSTAT

Älypuhelin on laite, joka mahdollistaa normaalien puhelimen toimintojen lisäksi laitteen käyttämisen taskukokoisena tietokoneena. Älypuhelin sisältää samanlaisia toimintoja kuin tietokoneet, mahdollistaen sähköpostien, internet-yhteyksien ja muiden vastaavien toimintojen käyttämisen kompaktissa koossa. [3.] Älypuhelinten lisäksi on toteutettu myös puhelin- ja tekstiviestiominaisuuksia vailla olevia mobiililaitteita, joista käytetään useimmin nimitystä Tablet PC [4].

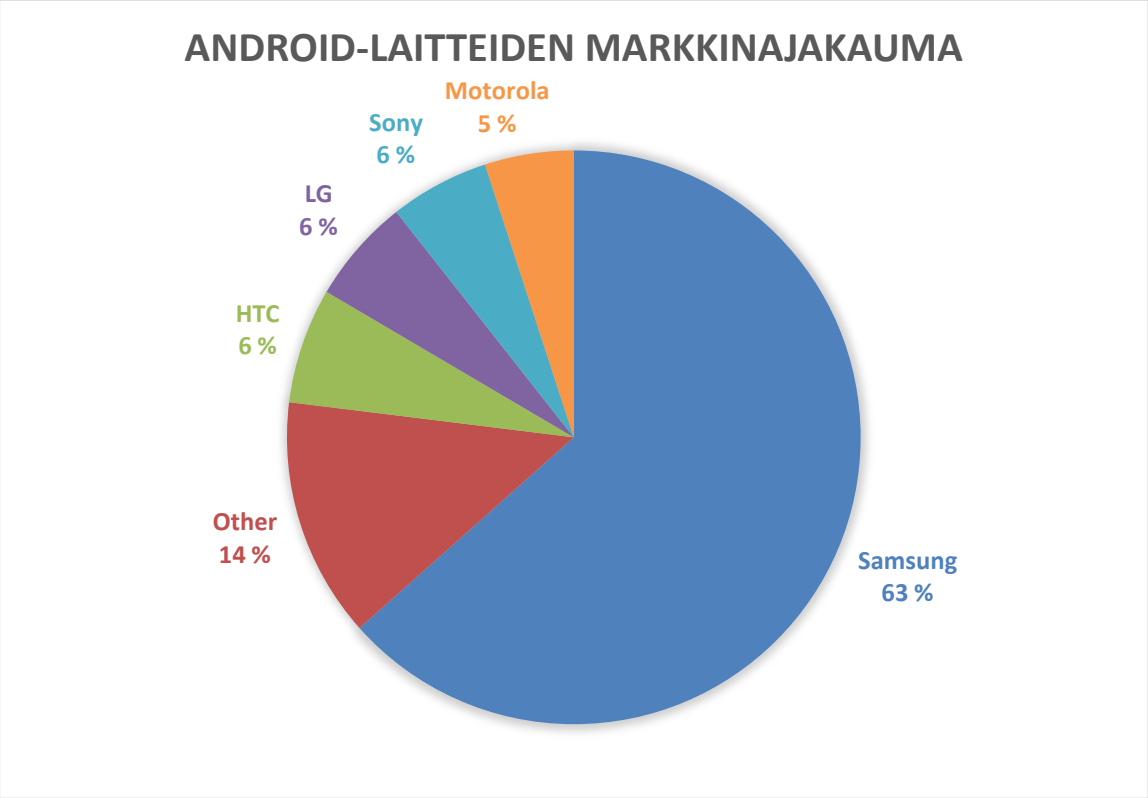
Markkinoilta löytyy paljon kilpailua älypuhelin- ja mobiililaitteiden valmistajien välillä. Laitevalmistajat eivät taistele keskenään vain paremman tekniikan välillä, vaan taistelu keskittyy pitkälti laitteissa käytettävien käyttöjärjestelmien välisiin eroihin. Suosituimpia sekä käyttäjäryhmittäin suurimpia älypuhelinten käyttöjärjestelmiä ovat Android, iOS ja Windows, joista Androidilla on suurin käyttäjäkunta usean eri valmistajan älylaitteen kautta. Taulukossa 1 on esitelty älypuhelinten käyttöjärjestelmien markkinajakauma vuosilta 2012 ja 2013, josta nähdään Androidin markkinaosuuden olevan huomattavasti kilpailijoitaan suurempi. Taulukossa on esitelty koko vuoden jakaumien lisäksi myös viimeisen vuosineljänneksen tulokset. [5.]

Taulukko 1. Älypuhelinten käyttöjärjestelmien markkinajakauma vuosilta 2012 ja 2013 [5].

Älypuhelinten markkinajakauma käyttöjärjestelmittäin	**Q4 2012**	**2012**	**Q4 2013**	**2013**
Android	70,3 %	69,8 %	78,4 %	78,9 %
Apple iOS	22,0 %	19,4 %	17,6 %	15,5 %
Microsoft	2,7 %	2,7 %	3,2 %	3,6 %
Muut	5,0 %	9,1 %	0,7 %	2,0 %

2.1 Android

Android-käyttöjärjestelmän kehitys alkoi vuonna 2003 yhdysvaltalaisen Andy Rubin johdosta. Hän perusti kumppaneittensa kesken yhtiön Android Inc., jonka päätavoitteena oli kehittää ohjelmistoalusta digitaalisille kameroille. Parempien markkinoiden vuoksi kehityksen suuntausta muutettiin kohti älypuhelimia. Android Inc. kehitti Android-käyttöjärjestelmää aina vuoteen 2005 saakka, jolloin yritys myytiin Googlelle. Vuonna 2007 julkaistiin ensimmäinen Android-käyttöjärjestelmä, jolloin Google lähti mukaan jatkuvasti laajentuville mobiilimarkkinoille. [6, s. 11–12.] Tällä hetkellä suurin Android-käyttöjärjestelmää käyttävistä laitevalmistajista on Samsung, jonka markkinaosuus on nähtävissä kuvasta 1 [7].



Kuva 1. Android-laitteiden markkinajakauma marraskuussa 2013 [7].

Androidin ohjelmistoalusta pohjautuu Linux-ytimeen, ja sen kehittyminen sekä suosio perustuvat pitkälti avoimen lähdekoodin hyödyntämiseen. Käyttöjärjestelmä on siis avoin ja maksuton, ja sen lähdekoodit on julkaistu avoimella Apache-lisenssillä. [6, s. 11.]

Käyttöjärjestelmä tarjoaa sovelluskehittäjille tehokkaat kehitystyökalut. Sovellusten kehittäminen voidaan toteuttaa Java-ohjelmointikielellä, mutta natiivisovellusten kehittäminen on mahdollista myös C++-ohjelmointikielellä. Android-sovelluskehitys on

suhteellisen helppoa eritoten Javan ansiosta, joten markkinoilta löytyy paljon nimenomaan Androidille suunniteltuja sovellusratkaisuja. [6, s. 12.]

2.2 iOS

iOS-käyttöjärjestelmä on yhdysvaltalaisen älypuhelin- ja tietokonevalmistajan Applen mobiililaitteissa käytetty käyttöjärjestelmä. Apple julkaisi ensimmäisen iOS-pohjaisen älypuhelimensa vuonna 2007, jolloin myös kapasitiivisesti toimivien kosketusnäytöllisten älypuhelimien kehittäminen sai kunnollisen lähtölaukauksensa. [8.]

Käyttöjärjestelmän toiminta perustuu Apple Mac -tietokoneissa käytettyyn OS X -käyttöjärjestelmään. Sen toiminta pohjautuu Mach-ytimeen sekä Darwin-käyttöjärjestelmään, vaikka iOS:n sovellusten toiminta perustuukin laitteistoriippumattomaan UNIX-pohjaiseen järjestelmään. [9.]

iOS on hyvin suosittu sen helppokäyttöisyyden, sulavan ja ennen kaikkea laajan sovelluskaupan App Storen ansiosta. Apple julkaisee itse vain hyvin vähän sovelluksia, mutta sovelluksia kehittävät paljon kolmannen osapuolen tekijät Applen tarjoamilla kehitystyökaluilla. [10.] Sovelluskehitys tapahtuu käyttämällä Objective-C-ohjelmointikieltä ja vaatii tätä varten Intel-pohjaisen Mac OS-tietokoneen. [11].

2.3 Windows Phone

Windows Phone on Microsoftin mobiililaitteisiin kehittämä käyttöjärjestelmä, jonka ensimmäinen versio julkaistiin vuonna 2010. Windows Phone 7 -käyttöjärjestelmäversiota käyttävät älypuhelimissaan muun muassa Nokia, HTC, LG ja Samsung [12]. Uudempaa Windows Phone 8 -käyttöjärjestelmää löytyy pääasiassa Nokian, HTC:n, Samsungin ja Huawein laitteista. Nokia on markkinajakaumaltaan selkeästi suurin Windows Phone -käyttöjärjestelmää käyttävistä mobiililaitteiden valmistajista vuoden 2013 viimeisellä neljänneksellä kattaen noin 80 % osuuden kaikista Windows-puhelimista. [13.]

Windows Phone tunnettiin ennen nimellä Windows Mobile, ja sen toiminta perustui Windows CE -käyttöjärjestelmään [14]. Käyttöjärjestelmän kehittyessä sen toimintaa muutettiin, ja nykyään se perustuu mobiiliystävällisempään Windows NT -käyttöjärjestelmän

ytimeen [15]. Sovelluskehitys Windows Phonelle toteutetaan useimmin joko C#- tai C++-ohjelmointikielellä, joista jälkimmäistä käytetään yleensä graafisiin sovellusympäristöihin. [16].

Microsoft-laitteet eroavat muista mobiilikäyttöjärjestelmistä Metro-ulkoasunsa avulla, jossa käyttöjärjestelmä sisältää laajan aloitusnäytön. Aloitusnäyttö sisältää suuria kuvakkeita, joista päästään käsiksi laitteen sovelluksiin ja ominaisuuksiin. [17.]

3 NATIIVI ANDROID-SOVELLUS

3.1 Yleistä Androidista

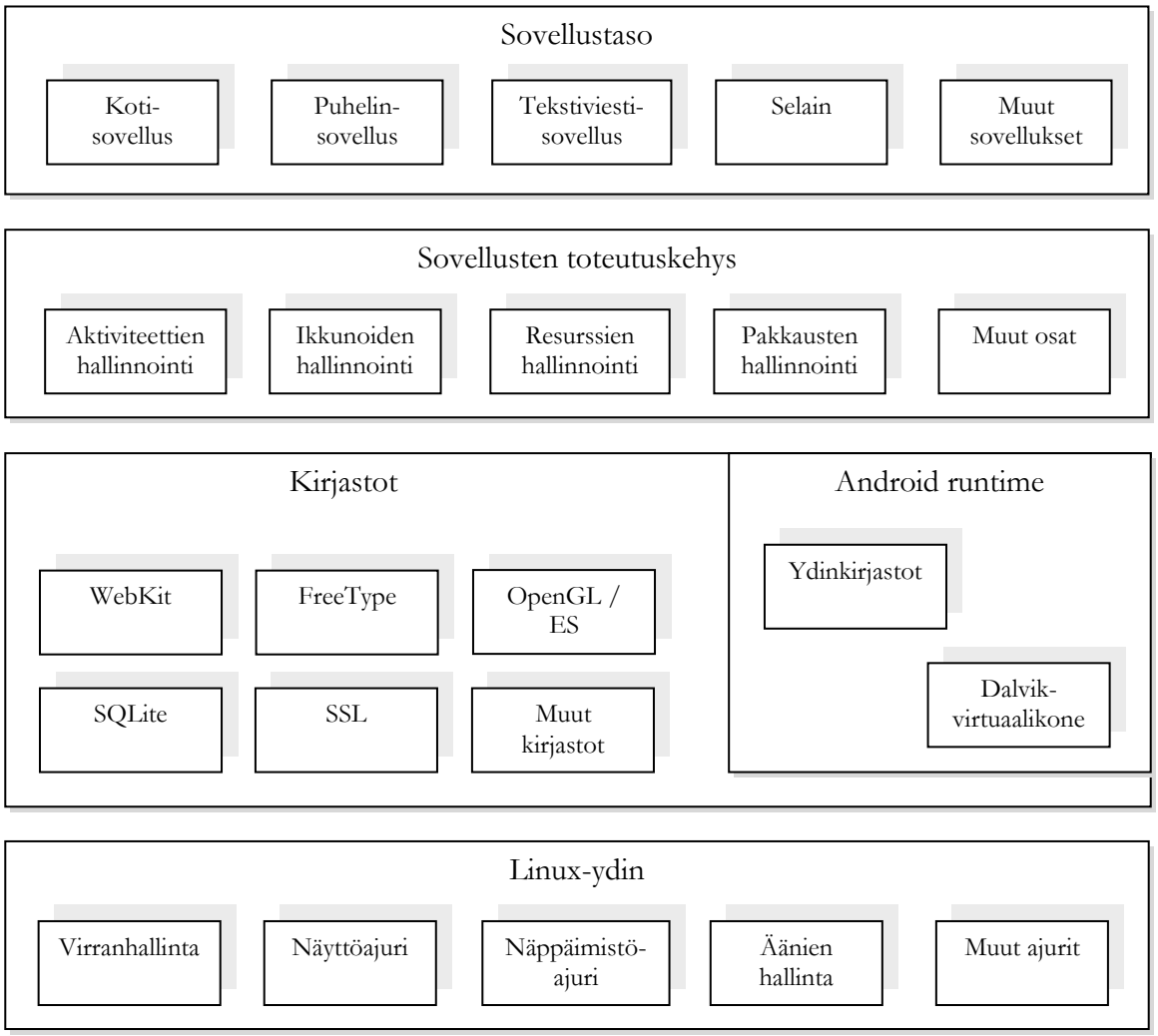
Google oli yksi OHA:n (Open Handset Alliance) perustajajäsenistä, kun joukko matkapuhelinalan yrityksiä perusti yhteenliittymän vuonna 2007. OHA:n tavoitteena oli luoda kuluttajien tarpeita vastaavia avoimia standardeja. Sen ensimmäisenä tuloksena oli Android, joka lisensoitiin kokonaan avoimen Apache-lisenssin alaiseksi. Ensimmäinen käyttöjärjestelmäversio julkaistiin marraskuussa 2007, mutta Android otettiin käyttöön matkapuhelimessa vasta lokakuussa 2008, kun T-Mobile-operaattori julkaisi Androidilla varustetun T-Mobile G1 -puhelimensa. Androidista on tämän jälkeen ilmestynyt uusia versioita tiheään tahtiin. Taulukossa 2 on esitelty Androidin merkittävimmät versiopäivitykset. [6, s. 11–12.]

Taulukko 2. Androidin versiohistoria [6, s. 12], [18].

Versio	**API-taso**	**Nimi**	**Julkaisu**	**Joitakin uusista ominaisuuksista**
1.0	1		23.9.2008	Ensimmäinen versio
1.1	2		9.2.2009	Virhekorjauksia ja pieniä parannuksia
1.5	3	Cupcake	30.4.2009	Softkeyboard, videot, pienoisohjelmat
1.6	4	Donut	15.9.2009	Päivitetty Market ja haku, WVGA
2.0/2.1	5-7	Eclair	26.10.2009	HTML5, optimointia
2.2.x	8	Froyo	20.5.2010	USB tethering, Flash-tuki, optimointia
2.3.x	9-10	Gingerbread	6.12.2010	Käyttöliittymäpäivitys, WXGA, NFC
3.x	11-13	Honeycomb	22.2.2011	Versio tablet-laitteille
4.0.x	14-15	Ice Cream Sandwich	19.10.2011	Paranneltu käyttöliittymä, widgettien koko muutettavissa, voice input engine
4.1 - 4.3	16-18	Jelly Bean	27.6.2012	Optimoitu nopeammaksi, monen käyttäjän tuki, BLE-tuki Android 4.3 (API18)
4.4	19	KitKat	Syksy 2013	Kevyempi käyttöliittymä, muistinkäytön pienentäminen, akunkesto, kuvanmuokkain

3.2 Arkkitehtuuri ja toimintaperiaate

Androidin ohjelmistoarkkitehtuuri noudattaa kuvan 2 mukaista rakennetta.



Kuva 2. Android-käyttöjärjestelmän arkkitehtuuri [6, s. 14].

Arkkitehtuuri on toteutettu viidellä eri tasolla.

- **Sovellustaso** sisältää laitteen mukana tulleet perussovellukset, yhteystiedot ja Android-kaupasta ladatut sovellukset.

- **Sovellusten toteutuskehys** mahdollistaa sovellusten toteuttamisen. Kehittäjille on tarjottu sovellustason perussovelluksissa käytössä oleva täysi API-tuki (Application programming interface).

- **Kirjastot** sisältävät Androidin komponenttien käyttämät C++- ja C-kirjastot. Taso tarjoaa esimerkiksi SQLite-tietokannan tiedon tallentamiseen ja jakamiseen, web-kirjaston ja kirjastot videon ja äänen toistamiseen sekä tallentamiseen.

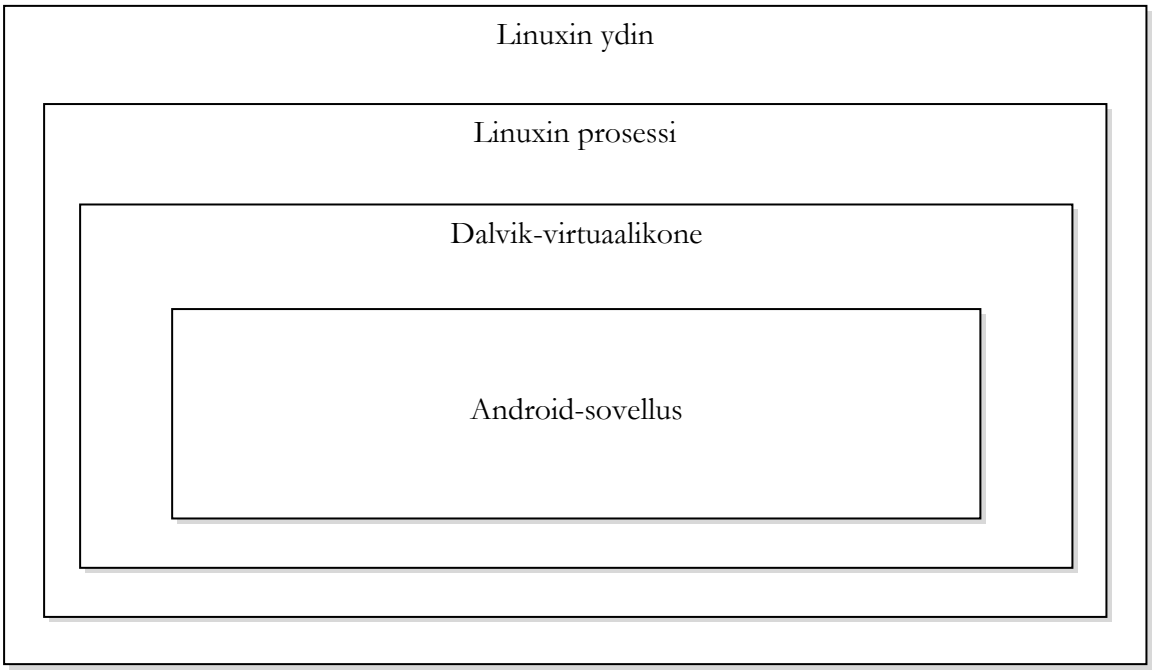
- **Android Runtime** sisältää ydinkirjastot Java-ohjelmille sekä näiden suorittamiseen tarkoitetun Dalvik-virtuaalikoneen. Tämä taso on saatavilla toisen tason (kirjastot) kautta.

- **Linux-ydin** on Linux-käyttöjärjestelmä, jonka päälle Android on rakennettu. Se sisältää perustiedot järjestelmän toiminnasta kuten prosessinhallinnasta, muistinhallinnasta ja laitehallinnasta. Ydin hoitaa kaikki asiat joissa Linux on yleisesti ottaen hyvä, kuten verkkoon liittymisen ja se tarjoaa myös laajat valikoimat laiteohjaimia. [6, s. 13.], [19.]

Androidissa käytettävistä sovelluksista suurin osa on toteutettu Javalla. Sen lisäksi Androidille voidaan toteuttaa ohjelmia muun muassa C++- tai C-ohjelmointikielillä Android NDK:n (Native Development Kit) avulla. [6, s. 12].

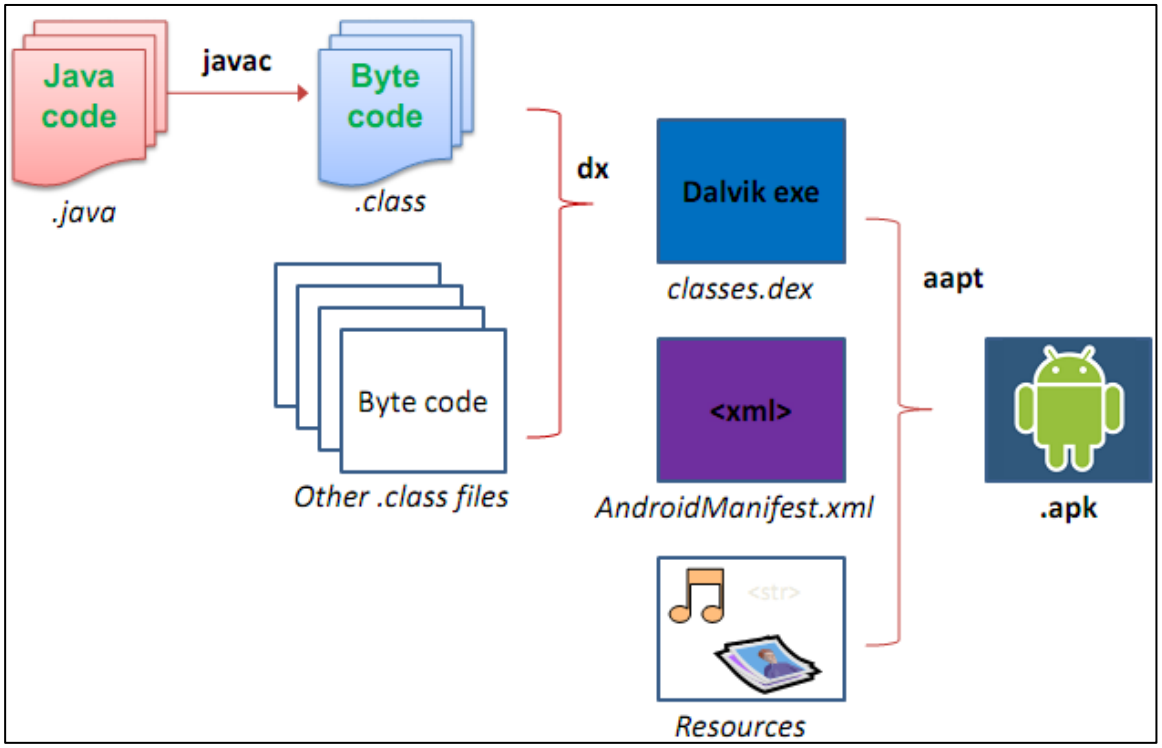
Javan bittikoodi ei ole tuettu suoraan Androidin puolesta. Voidakseen suorittaa kieltä Androidissa täytyy Java-luokat esikääntää Dalvik executables (.dex) -tiedostoiksi. Ohjelmat suoritetaan Dalvik VM (Dalvik Virtual Machine) -virtuaalikoneessa, joka on suunniteltu nimenomaan Androidille. Dalvik-virtuaalikoneita voi laitteessa olla käytössä samanaikaisesti useita, koska jokainen laitteen Java-sovelluksista ajetaan omassa virtuaalikoneessaan. [6, s. 12–13].

Dalvik VM käyttää Linux-ytimen ominaisuuksia kuten muistinhallintaa sekä säikeistämistä, jotka ovat ominaisia asioita Java-ohjelmoinnissa. Kuvassa 3 on esitelty tasoittain esimerkki virtuaalikoneen ja järjestelmän toiminnasta. Pohjimmaisena löytyy Linux-ydin, johon on luotu oma prosessi. Tämän sisällä toimii Dalvik-virtuaalikone, joka suorittaa Android-sovellusta. [20.]



Kuva 3. Dalvik VM [20].

Android-sovellus muodostuu kuvan 4 mukaisesta rakenteesta. Rakenne sisältää sovelluksen sisällön, joka on pakattu APK (Android application package file) -muotoon.



Kuva 4. Kuvakaappaus APK-tiedoston rakenteesta [21].

Java-ohjelmointikielellä toteutetut tiedostot esikäännetään Dalvik-virtuaalikoneelle sopivaan tiedostomuotoon [6, s. 12–13]. Dalvik executables -tiedostojen lisäksi pakettiin lisätään AndroidManifest.xml-tiedosto, joka esittelee Android-käyttöjärjestelmälle sovelluksen tärkeimmät tiedot [22]. Toteutettavaan sovellukseen voidaan vielä lisätä muita lähdetiedostoja, joita voivat olla esimerkiksi kuvia ja yhteystietoja. Näistä kaikista saadaan muodostettua Android application package file (APK), joka asennetaan lopulta laitteelle toimivaksi sovellukseksi. [21.]

3.3 Ohjelmointikielet

3.3.1 Java

Java on Sun MicroSystemsin vuonna 1995 kehittämä oliopohjainen ohjelmointikieli, joka suunniteltiin alun perin sulautettujen järjestelmien verkottamiseen. Java mahdollistaa laitteistoriippumattoman ohjelmointiratkaisun, ja se toimii kaikissa digitaalisissa laitteissa käyttöjärjestelmästä riippumatta. Sitä hyödynnetään maailmanlaajuisesti erilaisissa alustaratkaisuissa, muun muassa mobiililaitteissa sekä pöytätietokoneiden ohjelmistoissa ja se on nykyään yksi tämän hetken suosituimmista ohjelmointikielistä. [23, s. 2–6 .]

Ohjelmointikielenä Java on kohtalaisen yksinkertainen ja ennen kaikkea käyttäjäystävällinen. Se on kehitetty C++-ohjelmointikielestä, jonka koetaan yleensä olevan monimutkainen. Javalla on haettu ratkaisua C++-kielen rinnalle helppokäyttöisempänä ja silti tehokkaana ohjelmointikielenä. Sen tärkeimpiä ominaisuuksia ovat:

- oliopohjaisuus
- järjestelmäriippumattomuus
- turvallisuus
- automaattinen muistin vapauttaminen
- tuki säikeistykselle
- tuki verkko-ohjelmoinnille

- tuki graafisen käyttöliittymän ohjelmoinnille
- sisäänrakennettu virheiden käsittely
- tuki kansainväliselle merkistölle
- oma komponenttimalli (JavaBeans). [23, s. 6–11.]

Java-ohjelman kehittäminen vaatii Java SDK:n (Software Development Kit). Se sisältää muun muassa kielikääntäjän, tulkin ja muita työkaluja, joita käytetään sovelluksen tuottamisessa [24]. Valmis sovellus suoritetaan JVM-virtuaalikoneella (Java Virtual Machine), joka toimii yhdyskäytävänä sovelluksen ja suoritettavan laitteistoalustan välillä [23, s. 13].

3.3.2 C++

C++ on Bjarne Stroustrupin vuonna 1979 aloittama ohjelmointikielen kehitysprojekti, jonka tarkoituksena oli parantaa C-ohjelmointikielen toimintaa. Se tunnettiin alun perin nimellä C-kieli luokilla, mutta sai nykyisen muotonsa vuonna 1983. Kieltä kutsutaan C-kielen laajennukseksi, jonka vuoksi C++-koodia voidaan kirjoittaa C-kielen tyylillä tai oliopohjaisesti ollen näin tehokas esimerkki hybridistä ohjelmointikielestä. [25.]

C++ on yleiskäyttöinen olio-ohjelmointikieli ja sitä pidetään yleisesti ottaen suosituimpana ohjelmointikielenä sen monikäyttöisyyden ja tehokkuuden vuoksi. Sitä hyödynnetään muun muassa järjestelmien ja sovellusten ohjelmointiin, ajureiden tekemiseen, asiakaspalvelin-sovelluksiin, peliohjelmointiin ja sulautettujen laitteiden ohjelmointiin. [25.]

3.3.3 XML

XML (Extensible Markup Language) on yksinkertainen ja joustava SGML-kielestä (Standard Generalized Markup Language) johdettu merkkauskieli. Se suunniteltiin alun perin laajamittaisten sähköisten julkaisuiden haasteisiin. XML on World Wide Web Consortiumin (W3C) virallisesti suosittelema standardi. Kielellä voidaan kuvata ja järjestää tietoja ihmisille ja tietokoneille helposti ymmärrettävässä muodossa. [26.]

XML on hyvin samankaltainen tekstimuotoisen kirjoitustyylinsä puolesta HTML-kielen (Hypertext Markup Language) kanssa. Molemmat kielet sisältävät myös merkkaussymboleita kuvaamaan web-sivun tai tiedoston sisältöä. Erot kielten välillä kuitenkin määrittyvät nimenomaan sivun sisällön kuvaamisessa. XML kuvaa tiedon rakennetta ilman ennalta määrättyjä koodeja, kun HTML puolestaan kuvaa sisältöä suoraan ennalta määrätyn koodin kautta. Molempia näistä voidaan käyttää samassa web-sovelluksessa samanaikaisesti, jolloin XML voi esimerkiksi tuoda sisältöä HTML-sivulle. [26.]

XML:n eduiksi lasketaan nimenomaan kielen helppo ymmärrettävyys, helppokäyttöisyys ja se on täysin yhteensopiva Javan kanssa. Kieli on hyvin laajennettavissa avainsanojen, tunnisteiden ja muiden tietueiden puolesta. Kielen avulla voidaan luoda niin sanottu oma kuvauskieli, joka sisältää joukon sääntöjä ja tunnisteita tietojen kuvaamista varten. Näitä voivat olla esimerkiksi postinumero tai henkilön sukunimi, jotka viittaavat suoraan tiedon tyyppiin. [27.]

4 PHONEGAP

4.1 Yleistä PhoneGapista

PhoneGap on avoimen lähdekoodin sovelluskehys alustariippumattomien natiivien mobiilisovellusten kehittämiseksi hyödyntämällä HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) ja JavaScript web-teknologioita [28, s. 3]. Sen kehittäminen aloitettiin vuonna 2008 kanadalaisen ohjelmistoyrityksen Nitobin toimesta, jonka ajatuksena oli luoda helpompi tie alustariippumattomalle mobiilikehitykselle. Projektin alkuperäisenä tavoitteena oli kehittää iPhone-älypuhelimelle rajapinta, jolla saataisiin laitteen web-verkkoselaimen kautta käytettyä laitteen natiiveja toiminnallisuuksia, kuten kameraa tai kiihtyvyysanturia. Mobiililaitteiden omat web-verkkoselaimet eivät itsessään pääse käsiksi laitteen natiiviin toiminnallisuuteen, joten toiminnallisuuden saavuttaminen olisi suuri harppaus eteenpäin web-ohjelmoinnin saralla. [28, s. 4.]

iPhone-tuen valmistuttua projektista julkaistiin nopeasti tuet myös Androidin ja BlackBerryn käyttöjärjestelmille. Ajan saatossa PhoneGap on laajentanut laitteistotuen käytännössä kaikille laitteistopohjille. Sen kehittäminen jatkuu koko ajan varmistaakseen ohjelmointirajapinnan käytön laitteiden natiivitoiminnallisuuksien ja web-selaimen välillä tulevaisuudessa. [28, s. 4.]

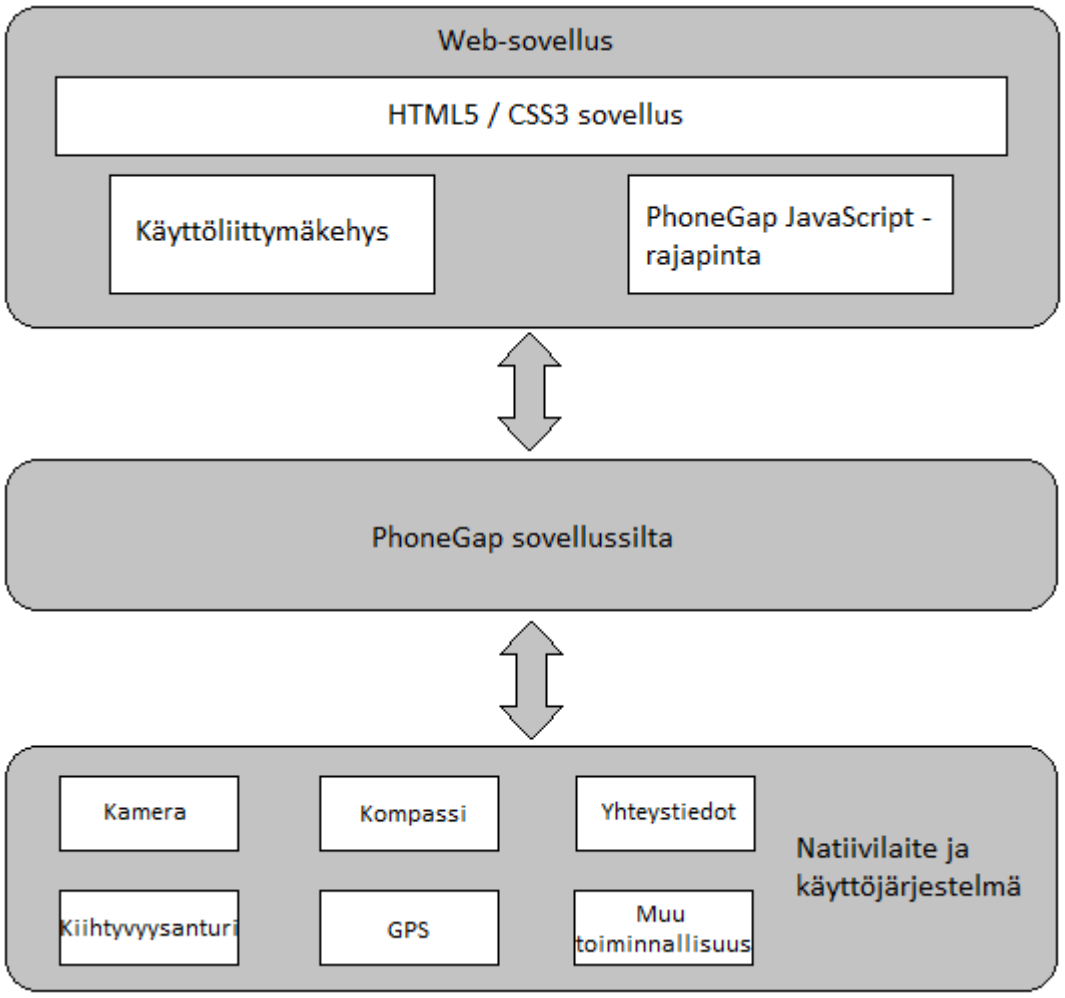
Nykyään PhoneGapin kehityksestä vastaa Adobe Systems. PhoneGapin ohella puhutaan nimestä Apache Cordova, joka monesti sekoittaa näiden kahden merkitykset keskenään. Apache Cordova on PhoneGapin JavaScript-kirjasto, joka mahdollistaa laitteen natiivien sovellusten käyttämisen. Lyhyesti sanottuna Cordova on JavaScript-moottori, joka antaa PhoneGapille virtaa toimiakseen. [29.]

PhoneGapin käyttöä suositaan sen monipuolisuuden vuoksi. Jo valmiiksi olemassa oleva web-sovellus saadaan PhoneGapin avulla helposti käännettyä natiiviksi mobiilisovellukseksi, jonka ansiosta pystytään hyödyntämään mobiililaitteen natiivitoiminnallisuutta. Toteutettaessa sovellus PhoneGapilla kerralla usealle eri mobiililaitteelle ei mobiilisovelluksen kehittäjän tarvitse toteuttaa omaa koodia näille jokaiselle erikseen. Web-tekniikoilla kirjoitetun koodin ja PhoneGap-käännöksen jälkeen ohjelma on toimintavalmis kaikille tavoitelluille laitteistoille. Tämän ansiosta kehittäjän ei välttämättä tarvitse opetella

itselleen uusia, ennalta tuntemattomia ohjelmointikieliä, kuten Javaa tai Objective-C:tä. [28, s. 5.]

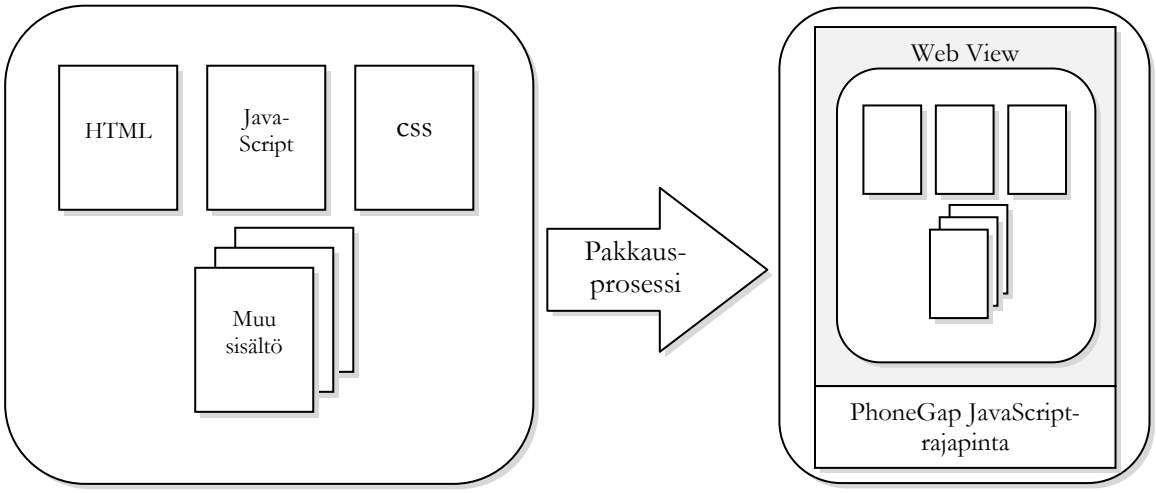
4.2 Arkkitehtuuri ja toimintaperiaate

PhoneGap-sovelluskehyksen arkkitehtuuri koostuu käytännössä kolmesta eri tasosta, jotka on esitelty kuvassa 5. Ylimpään tasoon kuuluvat web-tekniikoilla toteutetut käyttöliittymäkomponentit, joilla kehittäjä luo sovellukseen ulkoasun ja toiminnallisuuden. PhoneGapin JavaScript-rajapinta keskustelee toisella tasolla olevan sovellussillan kanssa, jonka avulla päästään käsiksi laitteen natiiviin toiminnallisuuteen. Kolmannessa tasossa löytyy laitteen käyttöjärjestelmä ja laitteen natiivit toiminnallisuudet, joihin toteutettu web-sovellus saa yhteyden PhoneGapin sovellussillan kautta. [30, s. 182.]



Kuva 5. PhoneGapin arkkitehtuuri [30, s. 182].

Mobiilisovelluksen kehittäjän toteuttaman web-sovelluksen kääntämisessä PhoneGap tarjoaa työkalut, joilla sovellus saadaan toimimaan jokaisella tuetulla alustalla. Kuva 6 havainnollistaa prosessin toimintaa. [28, s. 6.]

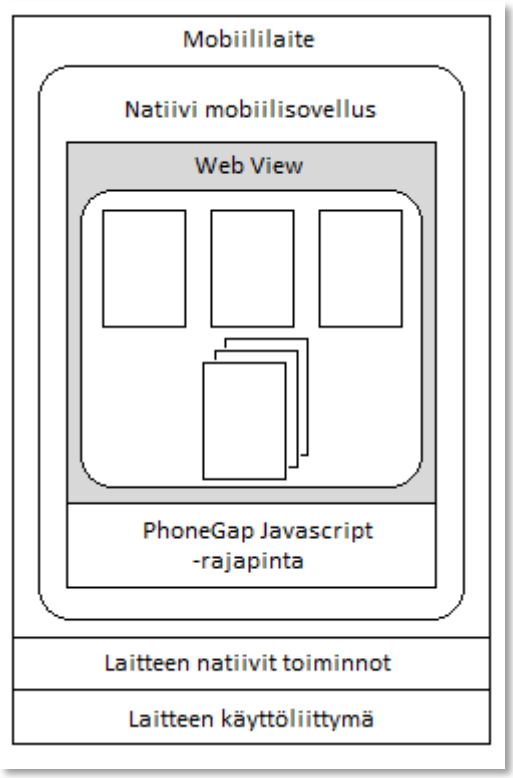


Kuva 6. PhoneGap-sovelluksen paketointiprosessi [28, s. 6].

PhoneGap paketoi web-tekniikoilla toteutetun sovelluksen mobiililaitteen natiivisovellukseksi. Luodussa sovelluksessa käyttäjä on vuorovaikutuksessa koko näytön kattavan näkymän, Web View -komponentin kanssa. Käytännössä tämä komponentti mahdollistaa toteutetun näkymän näytöllä laitteen web-selaimen kautta, jonka avulla toteutetaan vuorovaikutus luodun mobiilisovelluksen ja käyttäjän välillä. [28, s. 6–7.]

Toteutettu sovellus käyttäytyy kuin mikä tahansa web-sovellus. Se pystyy avaamaan muita html-sivuja, jotka ovat saatavilla joko sovelluksen omista hakemistoista tai jopa internetissä. Täten toteutettu sovellus pystyy hakemaan itselleen sisältöä myös suoraan internetistä. [28, s. 7.]

Tyypillisesti laitteen toimintoihin ja komponentteihin päästään käsiksi vain natiivisovelluksen kautta. Päästäkseen käsiksi näihin toimintoihin web-selaimen kautta, tarvitsee tämä yhteyden toimintoihin web-selaimen ulkopuolelta. PhoneGap auttaa tässä tarjoamalla JavaScript-rajapinnan laitteen eri toimintoja varten. Kuva 7 havainnollistaa rajapinnan toimintaa. [28, s. 7–8.]



Kuva 7. PhoneGap-sovelluksen ja natiivien toimintojen vuorovaikutus [28, s. 8].

PhoneGapin tarjoamaa rajapintaa käyttämällä sovellus pystyy kutsumaan laitteen natiivia toimintoa JavaScriptin avulla. JavaScriptilla toteutetulla kutsulla haetaan haluttua natiivitoimintoa, kuten kameraa. Rajapinta siis mahdollistaa laitteen natiivilla kielellä kirjoitetun toiminnon käyttämisen yksinkertaisen JavaScript-koodin avulla. Alla on esimerkki, jota käytetään älylaitteen kameraan pääsemiseksi ja kuvan ottamiseksi toteutetulla PhoneGap-sovelluksella. [28, s. 8.]

```
navigator.camera.getPicture( onSuccess, onFail);
```

Yllä oleva koodi kutsuu laitteen natiivilla ohjelmointikielellä kirjoitettua toimintoa. Vaikka kutsu palauttaakin arvoja, joita täytyy vielä käsitellä ennen sovellukseen viemistä, toimii tämä hyvänä esimerkkinä PhoneGapilla toteutetun laitteistoriippumattoman mobiilikehityksen tehokkuudesta. Yhdellä yksinkertaisella JavaScript-kutsulla saadaan siis toteutettua haluttu natiivi toiminto niin, ettei mobiilikehittäjän tarvitse perehtyä taustalla tapahtuvaan mobiililaitteen natiiviin toimintaan. [28, s. 8–9.]

PhoneGap mahdollistaa tällä hetkellä taulukon 3 mukaisten rajapintojen käyttämisen laitteistokohtaisesti. Kuten kuvasta voidaan nähdä, tiettyjä natiivisovelluksia ei pystytä

käyttämään kaikilla mobiililaitteiden käyttöjärjestelmillä. Tämä johtuu yleensä joko puhelimen teknisistä tai sovelluskohtaisista rajoituksista. [31].

Taulukko 3. PhoneGapin tukemat natiivitoiminnot käyttöjärjestelmittäin [31].

	iPhone / iPhone 3G	iPhone 3GS and newer	Android	Blackberry OS 6.0+	Blackberry 10	WebOS	Windows Phone 7 + 8	Symbian	Bada
Accelerometer	✓	✓	✓	✓	✓	✓	✓	✓	✓
Camera	✓	✓	✓	✓	✓	✓	✓	✓	✓
Compass	x	✓	✓	x	✓	✓	✓	x	✓
Contacts	✓	✓	✓	✓	✓	x	✓	✓	✓
File	✓	✓	✓	✓	✓	x	✓	x	x
Geolocation	✓	✓	✓	✓	✓	✓	✓	✓	✓
Media	✓	✓	✓	x	✓	x	✓	x	x
Network	✓	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Alert)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Sound)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Notification (Vibration)	✓	✓	✓	✓	✓	✓	✓	✓	✓
Storage	✓	✓	✓	✓	✓	✓	✓	x	x

4.3 PhoneGap-liitännäiset

PhoneGap-liitännäiset (Plug-Ins) mahdollistavat pääsyn laitteen natiivitoimintoihin, joille PhoneGap itsessään ei tarjoa suoraa tukea ohjelmointirajapintana. Tosin myös kaikki aiemmin mainitut, taulukossa 3 esitellyt natiivitoiminnot, ovat toteutettu liitännäisinä. Kuten huomataan, PhoneGap ei ole tukenut suoraan rajapintaa esimerkiksi laitteen kalenterille. Mikäli halutaan toteuttaa ominaisuuksia, joita sovelluskehys ei itsessään valmiiksi tue, täytyy nämä toteuttaa liitännäisen kautta. [32.]

Liitännäisillä toteutetaan JavaScript-rajapinnan avulla toiminnallisuus usealle eri mobiililaitteelle, mutta jokaisen toiminnallisuuden tausta täytyy toteuttaa laitteen oman natiivin ohjelmointikielen avulla. Täten sovellukseen pystytään lisäämään myös omaa kustomoitua natiivitoiminnallisuutta, joka voi erota tavallisesta natiivitoiminnosta. [32.]

Usein liitännäisiä on toteutettu kolmannen osapuolen johdosta. Liitännäisillä on voitu toteuttaa esimerkiksi viivakoodiskanneri, jolla pystytään älylaitteen kameran avulla tutkimaan viivakoodilta saatavaa tietoa. Sovelluskehittäjä voi itse toteuttaa liitännäisen omaan sovellukseensa kirjoittamalla haluamaansa ominaisuutta varten laitteen natiivikielellä toteutettavan koodin sekä JavaScript-koodin taikka käyttää valmista, toisen aiemmin luomaa

liitännäistä sovelluksessaan. PhoneGap tarjoaakin kattavan valikoiman valmiita liitännäissovelluksia, joita on helppo liittää mukaan PhoneGap-projektiin. [33.]

4.4 Tietojen tallentaminen

Suurin osa nykyaikaisista HTML5-yhteensopivista web-selaimista tarjoaa web-sovelluksille mahdollisuuden lukea tai kirjoittaa selaimen paikalliseen muistiin (local storage) tai vaihtoehtoisesti paikalliseen SQL (Structured Query Language) -tietokantaan. Kumpikaan näistä ominaisuuksista ei ole suoria HTML-funktioita, mutta näitä voidaan käyttää selaimessa käytettävässä JavaScript-koodissa. [28, s. 315.]

PhoneGap-sovellus pystyy helposti hyödyntämään selaimen tarjoamaa tallennuskapasiteettia. Mikäli PhoneGap-sovellus on luotu vanhalle mobiililaitteelle eikä tämän selain suoraan tue HTML5-tiedostomuotoa tai muuta tallennusmahdollisuutta, on PhoneGap mahdollistanut tämän toiminnon käyttämisen PhoneGap-liitännäisenä. [28, s. 315]

4.4.1 Web Storage

Web Storage on HTML5-standardiin perustuva tiedontallennusmenetelmä. HTML5 Web Storage mahdollistaa tiedon tallentamisen selaimen muistiin kahdella eri tavalla. Session storagen avulla voidaan tallentaa tietoa sessiokohtaisesti, jolloin tieto pysyy web-selaimen välimuistissa niin pitkään kunnes selain suljetaan. Local storage mahdollistaa tiedon tallentamisen muistiin ilman tiedon häviämistä selainta suljettaessa. Tällöin tallennettu tieto on tallessa niin pitkään kuin sitä tarvitaan eikä tuhoudu ennen erillistä muistin tyhjentämistä. [28, s. 316]. Web Storage -tallennusmenetelmien käyttöä rajoittaa monissa tilanteissa tallennuskapasiteetin suuruus, joka voi olla maksimissaan viisi megatavua [34].

Local storage on PhoneGapin tukema tiedontallennusmenetelmä. Se on hyödyllinen esimerkiksi sovelluksissa, joissa tiedon täytyy pysyä tallessa vaikka sovellus suljettaisiin. Hyviä käyttökohteita local storagen käyttöön ovat muun muassa erilaiset sovellusasetukset, joiden täytyy olla saatavilla sovellusta käynnistettäessä. [28, s. 316].

Local storagen toiminta perustuu tiedon tallentamiseen avain/arvo-menetelmällä, jossa tallennettava arvo laitetaan muistiin tietyn avaimen avulla. Arvo saadaan tallennettua kirjoittamalla JavaScript-koodia esimerkiksi alla olevalla tavalla. [28, s. 316].

```
window.localStorage.setItem(”avain”, ”arvo”);
```

Komennossa syötetään avain, joka toimii niin sanotusti osoitteena tallennettavalle arvolle. Arvo sisältää haluttavan tallennettavan tiedon, joka voi olla esimerkiksi muuttujan arvo tai käsin syötetty merkkijono. Vastaavasti tietoa haettaessa saadaan oikea arvo haettua muistista annetulla avaimella. [28, s. 316–317].

```
window.localStorage.getItem(”avain”, ”arvo”);
```

4.4.2 Web SQL Database

Web SQL (Structured Query Language) on yksinkertainen web-sovelluksissa käytettävä tietokanta, jonka määrittelyn ja kehittämisen toteutti W3C (Web Applicatins Working Group). Sen toiminta perustuu SQLite-relaatiotietokantajärjestelmään. Web-sovelluksessa tiedon tallentaminen tapahtuu HTML-verkkosivulla JavaScript-koodin avulla. Tietokannan etuihin luetaan Web Storageen nähden suurempi tallennuskapasiteetti, jota voidaan kasvattaa jopa satoihin megatavuihin käyttäjän niin halutessa. [35.]

Web SQL Database ei ole enää W3C:n ylläpitämä tiedontallennusmenetelmä. Ryhmä ei halunnut jatkaa tämän kehittämistä vuoden 2010 jälkeen, koska suurin osa tietokantatoteuttajista on toteuttanut oman tallennusjärjestelmänsä SQLiten avulla. W3C halusi lähteä toteuttamaan tiedontallennusmenetelmää sellaisten erilaisten toteutusten kautta, jotka eivät pohjautuisi SQLite-relaatiotietokantaan. [35.]

4.5 PhoneGap-ohjelmointikielet

4.5.1 HTML5

HTML (Hypertext Markup Language) on verkkosivujen tekemiseen käytetty merkintäkieli. Se on vakiinnutti paikkansa web-sivujen toteuttamiseen käytettävänä tekniikkana vuoden 1999 jälkeen. HTML5 on viimeisin versio merkintäkielestä, ja se julkaistiin vuonna 2007. Julkaisun alkuvaiheissa selaintuki kielelle oli heikkoa, mutta nykyään HTML5-tuki on tarjolla kaikilla yleisimmillä selaimilla. Kieli on alustariippumaton, ja se on suunniteltu toimimaan kaikilla laitteilla. [36.]

Kieli on suunniteltu tuottamaan monipuolista sisältöä web-sivuille ilman ylimääräisiä liitännäisiä. Sen ansiosta sivustoille voidaan lisätä upotettua graafista toimintaa kuten videoita ja muuta multimediaa niin, että se näkyy suoraan sivustolla. Aikaisemmilla versioilla tämä oli mahdotonta ja vaati aina videon nähdäkseen siirtymisen videon alkuperäiselle web-sivulle. [36.]

HTML5 on periaatteessa toteutettu yhdessä CSS- ja JavaScript-ohjelmointikielien kanssa. Se ei siis koostu pelkästään HTML-kielestä, vaan HTML5-kielellä toteutetut graafiset web-sivut sisältävät kaikkia kolmea ohjelmointikieltä. HTML5-kieltä käytetäänkin enemmän muotisanana näille nykyaikaisille web-teknologioille. Näiden yhteistyön tuloksena on helposti toteutettavissa olevia, graafisesti upeita ja käytettävyydeltään helppoja web-sivuja, joille voidaan toteuttaa hyvinkin monimutkaisia toiminnallisuuksia. [37.]

4.5.2 JavaScript

JavaScript on Netscapen kehittämä oliopohjainen ohjelmointikieli, joka on julkaistu vuonna 1995 [38]. Tarkemmin määriteltynä se on komentosarjakieli (skriptikieli), jota käytetään dynaamisten ja interaktiivisten toimintojen luomiseksi web-sivuille. Vaikka kieli vaikuttaa nimensä, ominaisuuksiensa ja rakenteidensa puolesta pitkälti samalta kuin Java-ohjelmointikieli, on se kehitetty itsenäisesti eikä näillä kielillä ole yhteyttä toisiinsa. [39.]

Kieli on hyvin erilainen kuin muut suositut olio-pohjaiset ohjelmointikielet eikä sisällä muun muassa luokkia kuten Java- tai C++-ohjelmointikielet. Se tukee useita valmiiksi kieleen

lisättyjä objekteja sekä muita oliopohjaisia ominaisuuksia, joita sovelluskehittäjät voivat käyttää hyväkseen sovelluksia toteuttaessaan. [40.]

JavaScript on tulkittava kieli, jolle löytyy tuki käytännössä kaikista web-selaimista. Käyttäjän selain tulkitsee kielellä kirjoitetun komennon, analysoi sen välittömästi ja suorittaa kirjoitetun toimenpiteen. Nykyaikaisissa sovelluksissa JavaScript-koodia voidaan tulkita JIT-kääntäjällä (Just-In-Time). Koodin suorituksen aikana selain voi päättää ajaa osan JavaScript-koodista JIT-kääntäjän läpi paremman suorituskyvyn aikaansaamiseksi. Tämän toiminnon ansiosta JavaScript voi toimia huomattavasti nopeammin, jolloin pystytään toteuttamaan monimutkaisempia ja suorituskykyisempiä web-sovelluksia. [40.]

Kielen käyttö on laajentunut nopeasti ja se on nykyään yksi suosituimmista ohjelmointikielistä. Suuri suosio perustuu pitkälti monipuolisista käyttökohteista web-teknologioissa, laajennettavuudesta, suhteellisen kevyestä toiminnasta ja helppokäyttöisyydestä. [41.]

4.5.3 CSS3

CSS (Cascading Style Sheets) on web-sovellusten käyttöön tarkoitettu kieli, jonka kehittäminen alkoi vuonna 1997 [42]. Kielellä toteutetaan web-sivustolle ulkoasuun liittyviä ominaisuuksia, kuten värejä, elementtien muotoja, taustakuvia ja fontteja. Sillä myös määritellään web-sivulle istutettavien elementtien sijainnit, eli miltä sivuston kokonaisuus ja rakenne tulee lopulta näyttämään. CSS ei tarvitse toimiakseen HTML-pohjaa, vaan sillä voidaan toteuttaa myös XML (Extensible Markup Language) -pohjaisen toteutuksen ulkoasu. [43.]

CSS-koodia ei tarvitse lisätä suoraan HTML-sivulle, vaan se voidaan tuoda sivulle erillisellä kutsulla. CSS:n erottaminen kokonaan HTML:stä omaan tiedostoonsa helpottaa web-sivujen ylläpitoa, mahdollistaa ulkoasujen jakamisen muiden käytettävien sivujen kesken ja sivuja pystytään räätälöimään tarkemmin erilaisiin ympäristöihin. [43.]

Kielen viimeisin versio on nimeltään CSS3. Sen myötä käyttöön saatiin moduulit, joilla pystytään toteuttamaan huomattavasti nopeammin toimivaa CSS-koodia. CSS3 on käytännössä täysin samanlaista aikaisempien versioiden kanssa, mutta sitä kirjoittaessa

voidaan tyylit jakaa pienempiin ”paloihin”, joissa pystytään määrittelemään tyylejä entistä yksityiskohtaisemmin ja helpommin. [44.]

4.6 jQuery-kirjastot

jQuery on web-tekniikoilla toteutetun sivuston tai sovelluksen käyttöön tarkoitettu avoimen lähdekoodin JavaScript-kirjasto, jonka ensimmäinen versio esiteltiin vuonna 2006 [45]. Se on tarkoitettu helpottamaan sivuston ulkoasun luomista valmiiden JavaScript-komentojen avulla. Kirjasto sisältää useimmin monia rivejä JavaScript-koodia vaativia komentoja, joita voidaan käyttää web-sivulla yksinkertaisen kutsun avulla. HTML-sivustolle lisättävällä kutsulla voidaan luoda esimerkiksi näppäin, jota painamalla voidaan toteuttaa haluttuja toimintoja. jQuery mahdollistaa myös monimutkaisten asioiden, kuten AJAX:n (Asynchronous JavaScript And XML), helpomman käyttämisen web-sivustoilla. [46.]

jQuery Mobile on kosketusnäytöllisille laitteille optimoitu käyttöliittymäkirjasto. Se käyttää toiminnassaan jQueryn JavaScript-kirjastoa, joka on yhteensopiva sekä mobiililaitteiden että työpöytäselaimien kanssa. jQuery Mobile on alustariippumaton, ja sen tarkoituksena on yksinkertaistaa ja parantaa mobiilikäyttöön tarkoitettujen web-toteutusten kehittämistä integroimalla HTML5, CSS3, jQuery ja jQuery UI -teknologiat yhdeksi, varmatoimiseksi ja tehokkaaksi sovelluskehykseksi. jQuery UI mahdollistaa yhdessä CSS- ja JavaScript-kirjastojen avulla käyttäjäystävällisen ja siistin ulkoasun luomisen web-sivuille. [47.]

4.7 PhoneGap Build -pilvipalvelu

Adoben tarjoamalla PhoneGap Build -pilvipalvelulla on mahdollista kääntää sovellus suoraan verkossa kaikille tavoitelluille laitealustoille ilman alustakohtaisten kehitysympäristöjen asentamista. Tämä pienentää tarvittavien resurssien määrää sekä vähentää suurien laitehankintojen hankkimista. PhoneGap Build -palvelulla voidaan kääntää rajattomasti sovelluksia julkaisemalla sovelluksen lähdekoodit julkisesti, mutta yhden sovelluksen ilmaiseksi näitä julkaisematta. 9,99 dollaria kuukaudessa kustantavalla palvelun jäsenyydellä voidaan julkaista 25 sovellusta kuukaudessa, joiden lähdekoodeja ei tarvitse julkisesti julkaista. [48.]

5 BLUETOOTH

5.1 Yleistä Bluetoothista

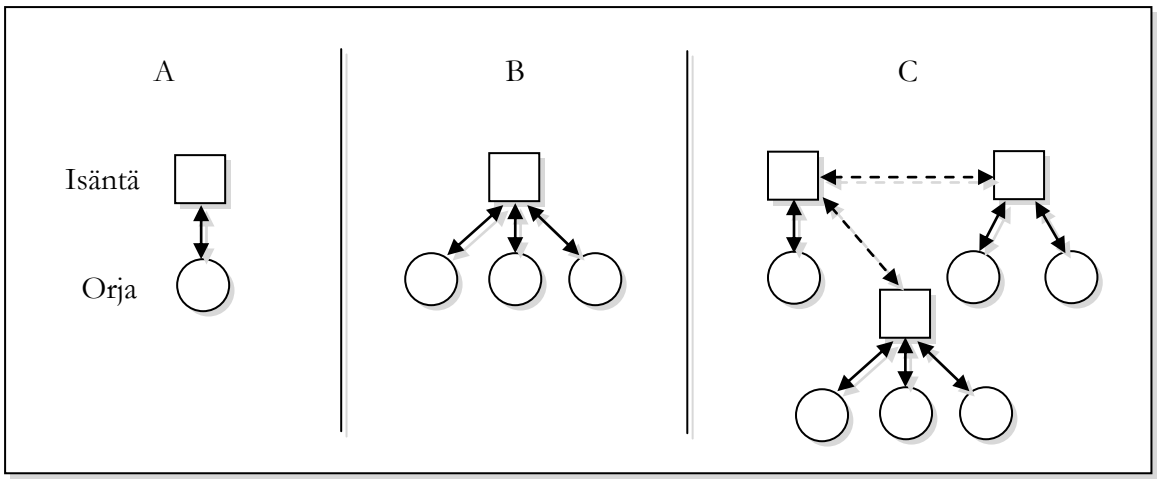
Bluetoothin kehittäminen alkoi vuonna 1998, kun elektroniikkajätit Ericsson, IBM, Intel, Nokia ja Toshiba perustivat Bluetooth SIG (Special Interest Group) -ryhmittymän lyhyen kantaman radioteknologian kehittämiselle. Tavoitteena oli toteuttaa langaton järjestelmä kiinteiden tai liikkuvien laitteiden välille kiinteiden kaapeliyhteyksien korvaamiseksi. Tekniikan lähtökohtina oli pieni tehonkulutus, luotettavuus, halpuus ja yksinkertaisuus. [49, s. 325.]

Tekniikan ensimmäinen versio 1.0 esiteltiin kesäkuussa 1999, joskin tätä ennen oli salaisia versioita eikä niitä julkaistu. Vuosien varrella on esitelty useita eri julkaisuja, mutta vuoden 2004 version 2.0 myötä tapahtui merkittävin muutos, kun EDT (Enhanced Data Rate) mahdollisti siirtonopeuden kolminkertaistamisen (3 Mbit/s) aikaisempiin versioihin nähden. [49, s. 325.] Bluetooth-yhteyden kantama voi parhaimmillaan olla noin 100 metriä, mutta kantaman kasvaessa myös laitteiden lähetystehontarve lisääntyy [49, s. 328]. Nykyään Bluetooth on julkaistu versionumerolla 4.0, jonka päällimmäisenä tarkoituksena on mahdollistaa kommunikointi erittäin vähävirtaisten laitteiden kanssa [50].

5.2 Bluetoothin toimintaperiaate

5.2.1 Arkkitehtuuri

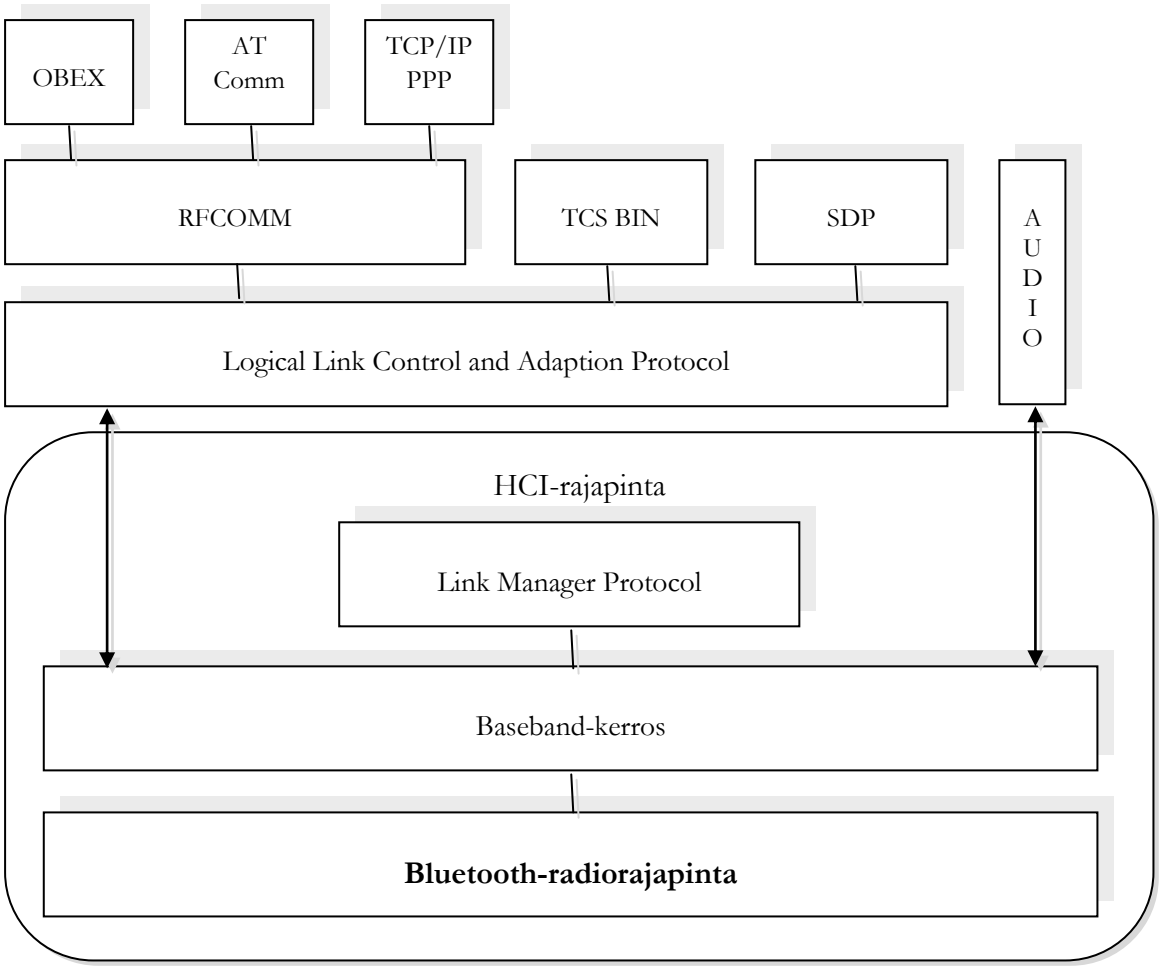
Bluetoothin toiminta perustuu master-slave-periaatteeseen, jossa yksi laite toimii verkon isäntänä ja muut yhdistetyt laitteet orjina. Yhden verkon laitteista on aina oltava isäntänä, eikä se eroa laitteena millään tavalla verkon muista laitteista. Yhdistyneiden laitteiden verkkoja kutsutaan myös nimellä pikoverkko. Kuvassa 8 on esimerkki Bluetooth-verkon topologioista. [49, s. 325–326.]



Kuva 8. Bluetooth-verkon topologioita [49, s. 326].

Bluetooth-pikoverkossa voi olla vain kahdeksan aktiivista laitetta kerrallaan, mutta se voi näiden lisäksi sisältää paljon isäntälaitteen herätettävissä olevia passiivisia laitteita. Kuvan 8 osassa C on nähtävillä verkkojen välistä toimintaa, jossa yksi laitteista toimii yhdyskäytävänä verkkojen välillä. Tämä laite voi toimia joko orjana toisessa ja isäntänä toisessa verkossa, mutta se voi toimia myös pelkkänä orjana molemmissa verkoissa. Vaikka laite on yhteydessä kahteen eri verkkoon samanaikaisesti, voi se keskustella vain toisen verkon kanssa kerrallaan. Tämä tilanne voidaan ohittaa rakentamalla siltoja, jotka sisältävät useita eri Bluetooth-moduuleja. [49, s. 326.]

Bluetooth on spesifioitu protokollapinon mukaisesti, joka kattaa isäntälaitteen lisäksi sekä fyysisen kerroksen että siirtoyhteyskerroksen toiminnot. Havaintoesimerkki Bluetooth-protokollapinosta on esitetty kuvassa 9. Isäntälaitteena toimiva laite ohjaa tietoliikennettä laitteistoriippumattoman HCI-rajapinnan (Host Controller Interface) kautta. [49, s. 327–328.] Tämä rajapinta määrittelee Bluetooth-laitteen ohjaustavan ja mahdollistaa muun muassa eri komentojen käyttämisen. Rajapinta tarjoaa sovelluskehittäjälle mahdollisuuden käyttää ja testata kaikkia Bluetooth-laitteen laiteominaisuuksia. [49, s. 336–337.]



Kuva 9. Bluetooth-protokollapino [49, s. 327].

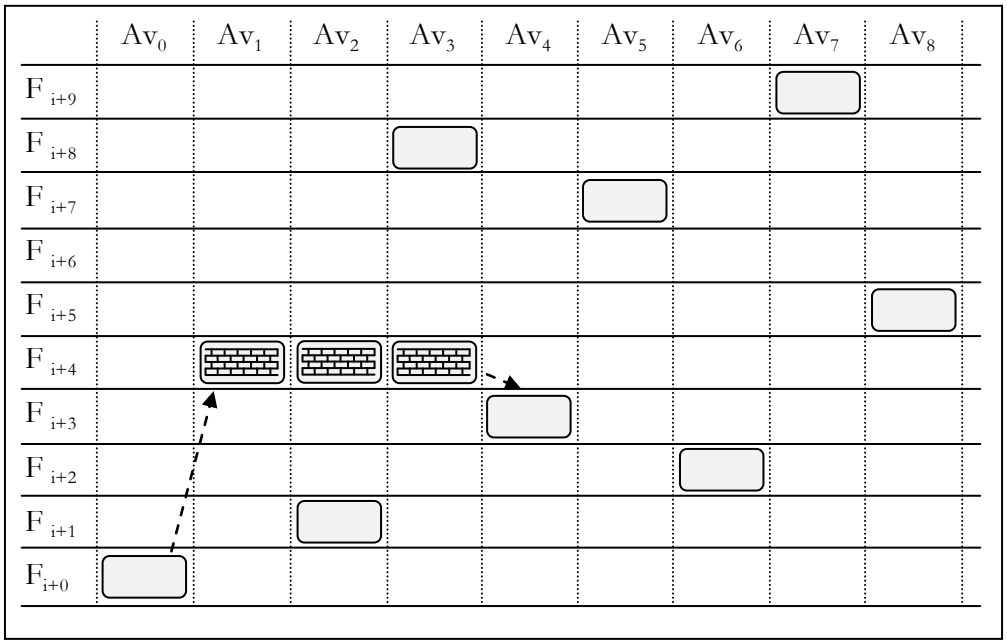
OSI 1 (Open System for Interconnection) -kerroksella sijaitsee radiotie ja osa Baseband-kerroksesta, joka kuuluu osittain myös OSI 2 -kerrokseen. Jälkimmäisen kerroksen toiminnot sisältyvät LMP- (Link Manager Protocol), L2CAP- ja Baseband-kerroksiin. OSI-kerrosmallissa kukin kerroksista käyttää yhtä alemman kerroksen palvelua ja tarjoaa palveluja kerrosta ylemmäs. Bluetooth ei tätä täysin puhdasoppisesti noudata, vaan OSI 1- ja OSI 2 -kerrosten välinen raja kulkee keskellä Baseband-kerrosta. [49, s. 327.]

Kerrosarkkitehtuurin perusperiaatteisiin kuuluu myös yhden kerroksen keskusteleminen naapurikerroksen kanssa. Tätä Bluetooth ei myöskään noudata, vaan se sallii joidenkin toimintojen osalta, kuten puheen siirron osalta, kerrosarkkitehtuurin palvelujen ohittamisen. [49, s. 327.]

5.2.2 Fyysinen kerros

Bluetooth toimii 2,4 GHz:n ISM-alueella (Industrial Scientific and Medical). Se on Euroopassa jaettu 79 eri kanavaan, joiden väli on yksi MHz. Tiedonsiirto tapahtuu taajuusmoduloinnin avulla, jossa 1-bitti ilmaistaan 140–175 kHz:iä kanavan keskitaajuuden yläpuolella sijaitsevalla taajuudella. 0-bitti ilmaistaan vastaavasti taajuudella, joka sijaitsee 140–175 kHz:n verran kanavan keskitaajuuden alapuolella. Vaikka Bluetooth-tekniikan käytössä on sama taajuusalue ja kanavajako kuin IEEE 802.11 (Institute of Electrical and Electronics Engineers) -verkossa, pystyy Bluetooth-verkon AFH-toiminto (Adaptive Frequency Hopping) välttämään radiotaajuuksien sekoittumisen keskenään. [49, s. 328.]

Bluetooth-teknologian tiedonsiirto perustuu taajuushyppelyyn, jossa isäntä-laite vaihtaa lähetystaajuutta aina tietyn aikavälin jälkeen [49, s. 116–117]. Bluetooth-tekniikassa aikavälin pituus on 625 μs, jolloin hyppelytaajuudeksi saadaan 1600 hyppyä sekunnissa. Bluetooth-verkon taajuushyppelyssä sanoman pituuden ylittäessä yhden aikavälin, jatkuu sanoman tiedonsiirto samalla taajuudella. Lähetystaajuus vaihtuu vasta sen jälkeen suunnitellulle taajuudelle, kun sanoma on siirretty loppuun asti. Asiaa on havainnollistettu kuvassa 10, jossa aikavälit on ilmoitettu vaakasuorassa tasossa ja taajuudet pystysuorassa tasossa. [49, s. 328.]



Kuva 10. Taajuushyppely Bluetooth-verkossa [49, s. 329].

Kuvassa 10 nähdään, että samalla taajuudella oleva sanoman jatkuva siirtäminen koskee vain isäntää ja orja-laitetta, joka joko vastaanottaa tai lähettää sanomaa. Muut laitteet eivät tästä taajuuden säilymisestä ole tietoisia, vaan jatkavat taajuushyppelyä alkuperäisen kaavan mukaisesti.

5.2.3 Kanavat

Bluetooth-verkossa voidaan laitteiden välille luoda kahdentyyppisiä kanavia. Synkronisen kanavan SCO (Synchronous Connection Oriented link), eli yhteydellisen kanavan, tehtävänä on palvella jatkuvan ja tasaisen bittivirran vaativia sovelluksia. Asynkronisen kanavan ACL (Asynchronous Connection Oriented link), eli yhteydettömän kanavan, on tarkoitus toimia silloin, kun isäntälaitteen ei tarvitse palvella synkronisia kanavia. [49, s. 329–331.]

Synkroninen yhteys

Synkronisen yhteyden tyypillisenä tiedonsiirtona tapahtuu puheen siirtäminen. Tällaisten yhteyksien ylläpitämiseksi Bluetooth-verkossa on SCO-yhteyksien eri osapuolelle annettava siirtoaikaa säännöllisin väliajoin. Tämän vuoksi aikaväleistä sovitaan yhteyden muodostamisen yhteydessä, mutta isäntälaite voi tästä huolimatta lähettää pyyntöjä orja-laitteelle muinakin aikoina. Synkroninen yhteys on aina kaksisuuntainen, koska isäntälaitteen pyytäessään orjalaitteelta tulevaa dataa, lähettää se samalla SCO-linkin dataa orjalaitteelle. [49, s. 330.]

Synkronisessa siirrossa käytetään sanomia, jotka kaikki ovat bruttona hyötykuormaltaan 240 bittiä. Sanomat eroavat databittien määrän puolesta toisistaan vain erilaisten suojauskoodaamisten kautta. [49, s. 330.]

SCO-kanava ei rajoitu vain yksinkertaisen datan siirtämiseen, vaan sillä voidaan siirtää myös monimuotoista tietoa. Näitä ovat esimerkiksi DV-sanomat (Data & Voice), jolloin sanomassa siirretään puhetta suojaamattomana ja dataa suojatulla koodauksella. Mikäli suojausta lisätään myös puheen siirtoon, kasvaa Bluetooth-verkon kapasiteetin tarve huomattavasti ja verkko tukkeutuu mahdollisesti jo yhdestä kanavasta. [49, s. 330–331.]

Asynkroninen yhteys

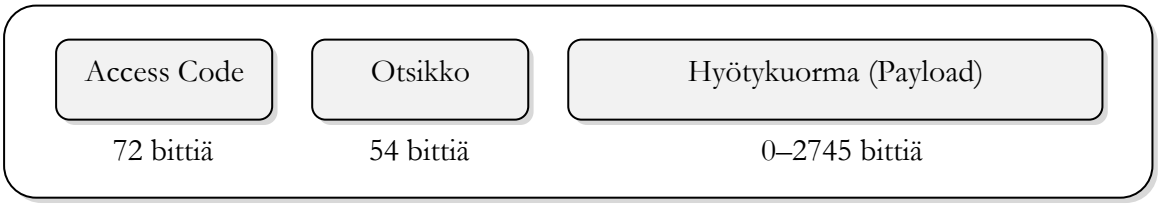
Isäntälaite lähettää ACL-yhteydellä verkossa oleville orjalaitteille datasanomia tai kyselysanomia, joihin orjalaitteet voivat vastata datasanomilla. ACL-kanavalla voidaan lähettää kysely yhdelle ja tietylle laitteelle, joka tunnistaa oman laitteen osoitteen saadusta sanomasta. Mikäli laite ei pysty tulkitsemaan saatua osoitetta, ei tämä pysty siirtämään tietoa verkkoon. [49, s. 331.]

ACL-kanavalla isäntä voi lähettää levitysviestejä kaikille verkossa oleville laitteille. Isäntä lähettää viestejä omalla osoitteellaan, jolloin laitteet tunnistavat isännän lähettämän sanoman. Kanavan liikenne muodostuu isännän tekemistä lähetyksistä ja orjalaitteiden sanomiin lähettämistä vastauksista. Siirtyvä data voi olla joko ohjaustietoa tai dataa. [49, s. 331.]

5.2.4 Sanomat

Bluetooth-sanomat siirretään radiotiellä aikaväleissä, joiden pituus on 625 µs. Data siirtyy vuorosuuntaisesti, jossa isäntä lähettää sanomansa parillisilla aikaväleillä ja orjalaite vastaa parittomin aikavälein. Tätä tekniikkaa kutsutaan nimellä TDD (Time Division Dublex). Sanoma voi jatkua aikavälistä toiseen, mutta sen kesto voi olla enimmillään viisi aikaväliä. [49, s. 333.]

Bluetooth-verkossa sanomat noudattavat kuvan 11 mukaista formaattia. Sanoma koostuu kiinteämittaisista tiedoista Access Code ja otsikko, joita seuraa itse sanoman hyötykuorma. Tämän pituus voi vaihdella eri sanomissa 0–2745 bitin välillä. [49, s. 333.]



Kuva 11. Bluetooth-sanomamuoto [49, s. 333].

Access Code (AC) muodostuu neljästä alkutahdistusbitistä, 64 bitin opetus- ja synkronointikentästä sekä neljästä häntäbitistä. Samassa pikoverkossa toimivat laitteet käyttävät samanlaista AC-kenttää. Otsikko sisältää tiedot sanoman vastaanottajan ja tyypin tunnistamiseksi. Verkko sallii enintään kahdeksan aktiivista laitetta, koska vastaanottajan

tunnus on aina kolmen bitin mittainen. Hyötykuormassa kulkee kaikki kolme perussanomaa, joita ovat hallintasanomat, SCO-kanavan sanomat sekä ACL-kanavan sanomat. [49, s. 334.]

5.2.5 Tietoturva

Jokaisella verkkolaitteella on yksilöllinen MAC-osoite (Media Access Control), joka on määritelty valmistajien puolesta laitekohtaisesti. [51]. Bluetooth-verkon tietoturva perustuu siirrettävän datan salaukseen ja laitteiden autentikointiin eli todentamiseen, joiden hallintaan käytetään laitteen MAC-osoitetta. Tämän lisäksi tietoturva käyttää autentikoinnissa luotua satunnaislukua sekä kahta samassa tilanteessa luotua 128-bittistä avainta. Salauksessa käytettävän avaimen pituus voi vaihdella 8–128 bitin välillä, jolloin tietoturvaa voidaan tarvittaessa parantaa suunnittelematta laitetta uudestaan. Pituuden vaihteluväliin vaikuttaa myös eri maiden salakirjoitusta koskevat säännökset. Salausavain muuttuu jokainen kerta salausta aktivoidessa, vaikka itse todentamisessa käytetty avain on aina kiinteä. [49, s. 335–336.]

Laitteiden todentaminen perustuu haaste-vastaus -menetelmään. Menetelmässä toinen osapuoli lähettää satunnaisluvun, johon vastaanottaja soveltaa yhteisesti sovittua algoritmia omalla avaimellaan ja palauttaa tämän jälkeen salakirjoituksella tuotetun vastauksen lähettäjälle. Avainten täsmätessä voi todentamista pyytänyt laite hyväksyä toisen kohteen verkkoon. Tämän epäonnistuessa käytetään eksponentiaalisesti kasvavaa odotusaikaa, jolloin tunkeilijan yritys arvata oikea avain tehdään vaikeaksi kasvavien aikavälien avulla. [49, s. 336.]

5.3 Bluetooth Low Energy (Bluetooth Smart)

Bluetooth 4.0 on viimeisin versio Bluetooth-teknologiasta. Bluetooth SIG julkaisi version kesäkuussa 2010 ja sen merkittävimpiin uusiin ominaisuuksiin lukeutui vähävirtaisen tiedonsiirron mukaan ottaminen teknologiassa [52]. Tämä teknologia tunnetaan nimellä Bluetooth Smart, mutta siitä käytetään yleisesti nimitystä Bluetooth Low Energy (BLE) [53].

BLE mahdollistaa erittäin vähävirtaiset yhteydet ja perustiedonsiirron laitteiden välillä, joiden toteuttamiseen aiemmin vaikeuttivat rajat joko virrankulutuksessa, laitteen fyysisissä kokorajoitteissa tai monimutkaisuudessa yhdistää laitteita muihin langattomiin standardeihin

[53]. Nyt teknologia mahdollistaa esimerkiksi pienillä kolikkoparistoilla varustettujen laitteiden kuten kellojen ja lelujen yhdistämisen langattomasti Bluetooth-verkkoon. Monissa tapauksissa se tarkoittaa sitä, että laitteita voidaan käyttää osana verkkoa jopa yli vuoden ajan ilman akun tai pariston uusimista. [50.]

BLE-teknologia on hyväksytty Windows 8 -standardiksi vähävirtaisille HID-laitteille (Human Interface Device). Uusissa Windows 8 -mobiililaitteissa on Bluetooth 4.0 -versio, joten kytkettävien HID-laitteiden kuten langattomien hiirten ja näppäimistöjen asentaminen onnistuu ilman erillistä ajureiden asentamista. Kuluttajat hyötyvät samalla pitkäkestoisista oheislaitteiden akuista eikä mobiililaitteen USB-porttiliitäntöjä tarvitse täyttää ylimääräisillä Bluetooth-telakoilla. [53.]

BLE on sovellusystävällinen teknologia, ja se tukee kaikkia suurimpia käyttöjärjestelmiä. Teknologia tarjoaa joustavan kehitysarkkitehtuurin luoda sovelluksia, joilla voidaan tuoda arkipäiväiset laitteet verkkoon muiden laitteiden kanssa kommunikoidakseen. Esimerkiksi Bluetooth Smart -teknologialla varustetut sykemittarit voidaan yhdistää teknologian avulla älypuhelimeen, jonka tarkoitukseen räätälöidyllä sovelluksella voidaan tarkastella sykemittarilta saatavaa tietoa. [54.]

5.3.1 BLE-teknologian etuja ja haittoja

Monet BLE-teknologian ominaisuudet ovat periytyneet klassisesta Bluetooth-teknologiasta. BLE käyttää samaa taajuushyppelyä kuin klassinen sekä sen tietoturva on toteutettu samanlaisilla salauksilla. Tämän vuoksi BLE-teknologia on hyvin helppo ottaa käyttöön uusissa Bluetooth-teknologian käyttökohteissa. [55.]

BLE omaa hyvien virransäästöominaisuuksiensa lisäksi myös tietynlaisia heikkouksia. Bluetooth-teknologia on alun perin suunniteltu jatkuvaan langattomaan tiedonsiirtoon, jota käytetään muun muassa äänen siirtämiseen. Tämä vaatii monissa tapauksissa suuria tiedonsiirtonopeuksia. Klassisella Bluetooth-teknologialla pystytään siirtämään suuria määriä dataa yhtäjaksoisesti käytännössä 2 Mb/s nopeudella, kun taas BLE-teknologian tiedonsiirtonopeus jää noin 100 kbit/s tasolle virransäästöominaisuuksien johdosta. [55.]

BLE-teknologia säästää virtaa pitämällä laitetta lepotilassa ja herättää laitteen lepotilasta vasta kun yhteys aloitetaan. Virrankulutus pysyy alhaisena myös alhaisten yhteysaikojen ansiosta,

koska yhteyttä pidetään yllä muutaman millisekunnin mittaisina jaksoina. Korkein virrankulutus voi olla noin 15 mA, ja keskimääräinen virrankulutus saadaan laskettua jopa yhden (1) μA:n tasolle. Täten se on ideaalinen toimimaan sovelluksissa, joissa siirretään pieniä määriä dataa eikä tiedonsiirtonopeudella ole niin paljoa merkitystä. [55.]

Kuten klassisessa Bluetoothissa, BLE perustuu master-slave-periaatteeseen. Tosin BLE-teknologiassa orjalaitteiden määrä voi olla hyvinkin suuri ja se riippuu käytettävän muistin koosta. Orjalaite voi myös itse ilmoittaa sillä olevan jotakin lähetettävää tietoa tätä skannaaville laitteille, jolloin skannaavien laitteiden ei tarvitse lähettää erillisiä kyselyitä tietoa hankkiakseen. [55.]

5.3.2 BLE sovelluskehityksessä

BLE eroaa klassisesta Bluetoothista myös ohjelmiston rakenteen puolesta. BLE:n rakenteessa kaikki parametrit sisältävät tilan, johon päästään käsiksi käyttämällä palvelupohjaista atribuuttiarkkitehtuuria. Atribuutit esitetään ominaisuuksina, jotka kuvaavat esimerkiksi signaalin arvoa tai esitystapaa. Nämä ominaisuudet mahdollistavat toiminnallisuuksien käyttämisen erilaisissa käyttösovelluksissa saumattomasti ja yhteensopivasti eri laitevalmistajien välillä. [55.]

BLE käyttää palvelupohjaista ATT (Attribute Protocol) -arkkitehtuuria ja kaikki viestintä tapahtuu GATT-profiilin (Generic Attribute Profile) kautta [56]. GATT luo yhteiset toiminnot ja puitteet datan lähetykselle sekä tallennukselle. Se myös määrittää kaksi eri roolia, palvelin ja asiakas, joita käytetään hyväksi palvelupohjaisessa viestinnässä [57]. Sovellus tai toinen profiili käyttää GATT-profiilia, jolloin asiakas ja palvelin voivat olla vuorovaikutuksessa keskenään. [56].

Palvelin sisältää useita eri atribuutteja. GATT-profiili määrittelee, kuinka ATT käyttää kyseisiä atribuutteja löytääkseen, lukeakseen, kirjoittaakseen ja saadakseen viitteitä. Viitteitä käytetään esimerkiksi skannatessa BLE-laitteita ja laitteesta saatavilla olevien tietojen käsittelyyn. Nämä ominaisuudet tukevat palvelupohjaista arkkitehtuuria ja näitä käytetään siten, kuin ne on ennalta profiiliin määritelty. GATT siis mahdollistaa sovelluskehittäjän käyttöön profiilissa määriteltyjä palveluja ja ominaisuuksia sovelluskehityksessä. [56.]

GATT-arkkitehtuuri tekee profiilien luomisesta ja toteuttamisesta kohtalaisen helppoa. Monet uudet profiilit ovat jatkuvasti kehitteillä ja profiilien helppo käyttöönottaminen erilaisissa sovelluksissa ja sulautetuissa laitteissa edistää profiiliarkkitehtuurin kasvua huomattavasti. [56.]

6 KEHITYSYMPÄRISTÖN VALINTA

Insinöörityön tavoitteena oli suunnitella mobiilisovellus Android-pohjaiselle älypuhelimelle. Työn toteuttamiseen kuului graafisen käyttöliittymän ja tämän avulla toteutettavien toiminnallisuuksien ohjelmointi. Toiminnallisuuksiin sisältyvät käyttöliittymässä tapahtuvien toimintojen suorittaminen, joista tärkeimpänä sovelluksessa tapahtuva Bluetooth Low Energy -teknologian käyttäminen.

Sovellus on tulevaisuudessa tarkoitus toteuttaa Androidin lisäksi myös iPhone- ja Windows Phone -älypuhelinalustoille. Tämän vuoksi työn alkuvaiheisiin kuului selvittää, pystytäänkö sovellusta toteuttamaan käyttämällä alustariippumattomia web-teknologioita vai täytyykö se toteuttaa laitteen natiivilla ohjelmointikielellä.

Alustariippumattoman sovelluskehyksen tarjoaa PhoneGap, jolla pystytään luomaan natiivin ulkoasun ja toiminnallisuuden omaavia sovelluksia web-tekniikoilla. Laitteen omalla natiivikielellä kirjoitettavalla koodilla saadaan varmasti toteutettua sovellus, joka pystyy käyttämään hyväksi laitteesta löytyvää tekniikkaa ongelmattomasti.

Pystytäänkö web-pohjaisella PhoneGap-sovelluksella käyttämään hyväksi laitteen natiivia toiminnallisuutta? Entä mahdollistaako toteutustapa BLE-teknologian käyttämisen sovelluksessa?

6.1 Teknologioiden eroavaisuudet

Natiivi mobiilisovellus on toteutettu juuri tiettyyn laitteeseen suunniteltuja ohjelmointikieliä käyttäen. Tämän ansiosta se on yleisesti erittäin tehokas, kykenee raskaisiin graafisiin toimintoihin ja kykenee käyttämään laitteen omia toimintoja sekä komponentteja ongelmitta. Vaikka sovellus voi olla hyvä näiden ominaisuuksien puolesta, voidaan sen suurimmaksi heikkoudeksi mainita toteutusvaihe. Vaikka kehittäjä ymmärtäisikin toteutettavan sovelluksen natiivikieltä tai mahdollisesti haluaisi opetella tätä, ei sitä pystytä hyödyntämään eri alustaratkaisuissa. Laajennettaessa sovellus käyttöön myös muille laitealustoille syntyy lisäkustannuksia ja aikaa vaaditaan sovelluksen toteuttamiseen paljon. [58.] [59.]

Android-natiiviohjelmoinnin etuihin kuuluu sen avoimuus. Avoimella Apache-lisenssillä julkaistua Androidin lähdekoodia voi jokainen muokata haluamakseen, jolloin myös sovellusten toteuttamismahdollisuudet monipuolistuvat. Vaikka Android on tietoturvaltaan hyvin suojattu, mahdollistaa avoimuus luonnollisesti myös haittaohjelmien helpomman toteuttamisen käyttöjärjestelmälle.

Taulukossa 4 on esitelty natiiviohjelmoinnin merkittävimpiä etuja sekä heikkouksia.

Taulukko 4. Natiivin sovelluskehityksen edut ja heikkoudet [58] [59].

Edut	Heikkoudet
Paras suorituskyky. Pääsy saatavilla oleviin laitteen ominaisuuksiin aina mahdollista ja nopeasti.	Suuri budjetti ja resurssit usean alustan ratkaisujen toteuttamiseksi ja näiden ylläpitämiseksi.
Sovellus saadaan varmasti toteutettua myös taustalla toimivaksi ratkaisuksi.	Sovelluksen saaminen sovelluskauppaan voi olla vaikeaa ja kestää pitkään.
Laitekohtaiset sovelluskaupat tarjoavat toteutukselle varman paikan kaupasta, jolloin sovelluksen turvallisuus kaupan osalta todennettu.	Eri käyttöjärjestelmäversioissa voi olla erilaisia toteutusratkaisuja sovellukselle. Kehittäminen ja ylläpitäminen ovat täten vaikeaa.
Alustakohtaisella kielellä toteutetulle sovellukselle on aina tarjolla kehitykseen soveltuvat työkalut.	Ei takeita sovelluksen suosiosta, ellei sovellusta saatavilla usealle eri alustalle.

Web-tekniikoilla toteutettu mobiilisovellus auttaa ratkaisemaan ongelman sovelluksen kehittämiseksi eri laitealustojen välillä. PhoneGapilla tai vastaavalla toteutetut hybridisovellukset, tarkemmin sanottuna natiivisti asennettavat web-sovellukset, ovat yleisesti ottaen myös nopeampia kehittää. Kielet ovat helposti opittavia eikä sovelluksia toteuttaessa tarvitse kiinnittää huomiota toisille käyttöjärjestelmille laajentamiseen, koska sama koodi käy suoraan jokaiselle PhoneGapin tukemalle laitealustalle.

Web-tekniikoilla pystytään nykyään toteuttamaan myös hyvin käyttäjäystävällisiä ja kohtalaisen kevyesti toimivia ratkaisuja. Moni suosii web-ulkoasun luomaa grafiikkaa, koska sen ulkoasu ja toiminnallisuus ovat käyttäjälle ystävällistä sekä toteuttaminen hyvin yksinkertaista esimerkiksi jQueryn tarjoaminen UI-kirjastojen avulla.

PhoneGapin käyttämien web-tekniikoiden huonot puolet näkyvät lähinnä raskaiden sovellusten toteuttamisessa ja laitealustojen natiivitoimintoihin käsiksi pääsemisessä. HTML5, CSS ja JavaScript tarjoavat kohtalaisen kevyen toteutusratkaisun, eikä näillä pystytä toteuttamaan kovinkaan raskaita graafisia sovelluksia. Laitteen natiivitoiminnot tosin ovat hyvin saatavilla PhoneGapin API-rajapinnan avulla, mutta eivät ole tuettuja kaikkien PhoneGapin tarjoamien laitealustojen välillä. Laajennettaessa sovellusta eri laitealustoille ei perussovelluksen runkoa tarvitse kirjoittaa jokaiselle erikseen, vaan laitteen natiivitoiminnallisuuteen liittyvän koodauksen uudelleenkirjoittaminen on riittävää. [58.] [59.]

Koska PhoneGap käyttää hyväkseen laitteen web-selainta, on se merkittävässä roolissa sovelluksen toiminnan kannalta. Laitealustat käyttävät usein erilaisia oletusselaimia web-sivujen näyttämiseksi, jolloin sovellusten toiminnassa saattaa syntyä eroja eri laitealustojen välillä. HTML5-tuki löytyy käytännössä kaikista nykyaikaisista selaimista, mutta selainkohtaiset erot tulkita kirjoitettu koodi vaikuttavat paljon toteutuksen lopputulokseen. Selainta pidetään yleisesti ottaen myös hitaana yksinkertaistenkin asioiden suorittamisessa, mutta monissa tapauksissa erot natiiviin toteutukseen verrattuna ovat marginaalisen pienet. [60.]

Taulukossa 5 on esitelty PhoneGap-sovelluskehityksen merkittävimpiä etuja sekä heikkouksia.

Taulukko 5. PhoneGap-sovelluskehityksen edut ja heikkoudet [58] [59].

Edut	Heikkoudet
Nopea kehittäminen ja helppo laajennettavuus.	Suorituskyvyltään heikompi natiiviin sovellukseen verrattuna.
Mahdollisuus käyttää laitteen natiivitoimintoja API-rajapinnan kautta.	Ei suoraan täyttä tukea laitteen kaikkiin natiivitoimintoihin liitännäisten muodossa.
Web-teknologioiden käyttö ei välttämättä vaadi laitteen natiivikielen opiskelemista.	Joitakin natiivitoimintoja ei tuettu kaikilla laitealustoilla.
Tuki usealle eri laitealustalle.	Rajalliset ominaisuudet natiiviin verrattuna.
Mahdollisuus toteuttaa sovellus taustalla toimivaksi ratkaisuksi.	Pienet eroavaisuudet eri laitevalmistajien välillä aiheuttavat eroja sovelluksen toiminnassa.
Julkaisumahdollisuus sovelluskaupoissa.	
Sovelluksen nopea testaaminen ja käyttöönottaminen.	
Tarjottavat liitännäiset usein vapaasti käytettävissä ja ylläpidettyjä yhteisöjen puolesta.	
Voidaan käyttää hyväksi muun muassa jQueryn tarjoamia UI-kirjastoja käyttäjäystävällisen ulkoasun luomiseksi	
Web-teknologiat parhaiten standardoituja	

PhoneGap tarjoaa Androidille hyvin kattavan API-rajapinnan laitteen natiivitoimintoihin pääsemiseksi. Vaikka rajapinta ei sisällä suoraa tukea laitteen Bluetooth-ominaisuuden hyödyntämiseksi, voidaan teknologiaa käyttää hyväksi liitännäisen avulla. Plugin Registryn sivuilta on tarjolla kolmannen osapuolen kehittämiä liitännäisiä PhoneGap-sovelluksien käyttöön. Sivulta löytyy myös ruotsalaisen ohjelmointiryhmän Evothingsin kehittämä BLE-liitännäinen Android- ja iOS-laitteille vähävirtaisen BLE:n käyttämiseksi PhoneGap-sovelluksissa. Teknologian käyttäminen on siis mahdollista eikä vaadi sovelluskehittäjältä oman liitännäisen toteuttamista. Koska liitännäisellä pystytään etsimään vain BLE-laitteita, se ei sovellu aikaisemmilla Bluetooth-versioilla varustettujen laitteiden etsimiseen ja niiden kanssa kommunikoimiseen. [61.]

6.2 Tulevaisuus ja jatkokehitys

Web-teknologioita voidaan pitää tulevaisuuden ratkaisuna. Näitä voidaan käyttää PhoneGap-tyylisen hybridisovelluksen lisäksi myös suoraan web-palvelimella sijaitsevan sovelluksen toteuttamiseksi. Vielä tällä hetkellä teknologioilla ei pystytä saavuttamaan alustakohtaisen natiivisovelluksen suorituskykyä, mutta jatkuvan kehityksen ansiosta voi tulevaisuudessa olla saavutettavissa hyvinkin tehokkaita web-sovelluksia. Suurimpana kynnyksenä suorituskyvyn suhteen voidaan tällä hetkellä pitää laitteiden web-selaimia, jotka rajoittavat tekniikoiden tehokkaampaa hyödyntämistä.

PhoneGap-sovelluskehitys sopii siis tulevaisuuden haasteisiin erinomaisesti. Juuri tällä hetkellä se mahdollistaa pääsyn laitteen natiiviin toimintoon, mihin älylaitteiden web-selaimet eivät itsessään kykene. Tarjoamalla lähes samat mahdollisuudet natiivikielellä toteutettuun sovellukseen verrattuna toimii se kovana haastajana sovellusten toteutustavan valinnassa.

Mobiilisovellusta ei pidä katsoa kertaluontoisena ratkaisuna, vaan pitkän aikavälin toteutuksena. Sovellus voi hyvinkin olla käytössä vuosien ajan sen julkaisuhetkestä, joten sen helppo kehittäminen sekä päivittäminen ovat tärkeitä osa-alueita sovelluksen toteutusta. Natiivikielellä toteutettua sovellusta joudutaan monesti uusien käyttöjärjestelmä- ja laiteversioiden mukana päivittämään, jolloin syntyy paljon lisätyötä ja resurssien tarve kasvaa. Web-tekniikoiden kehittyessä PhoneGapilla toteutettu sovellus voidaan jalostaa tulevaisuudessa helposti erilaisiin ratkaisuihin, joten jatkokehitys näillä tekniikoilla on helpompaa ja näin myös edullisempaa.

Sovelluksen kehittämisen kannalta on käytännössä sama, kummalla teknologialla sovellusta tällä hetkellä lähdetään toteuttamaan. Molemmat toteutusmahdollisuudet tarjoavat omat hyvät ja huonot puolensa sekä samat mahdollisuudet toiminnallisuuksien osalta. Kehittäjän kannalta ratkaisevaa onkin se, kummalla tekniikalla toteutettua sovellusta on helpompaa jatkokehittää.

7 MOBIILISOVELLUKSEN TOTEUTUS

PhoneGap valittiin tähän projektiin, koska se tarjoaa laitteistoriippumattoman lähestymisratkaisun mobiilisovelluksen kehittämiseksi ja on täten helposti laajennettavissa muille laitealustoille. Se myös mahdollistaa pääsyn laitteen natiiviin toiminnallisuuteen, vaikkakin hieman monimutkaisemman toteutustavan kautta Android-natiivikehitykseen verrattuna. Koska sovelluksen ei tarvitse suoriutua rasittavista graafisista tai suurta laskentatehoa vaativista toiminnoista, sopii web-teknologioiden käyttäminen projektiin mainiosti.

7.1 Kehitysympäristön pystyttäminen

PhoneGap tarjoaa hyvät ohjeet kehitysympäristöjen pystyttämiseksi eri alustoilla. Ohjeistus on saatavilla PhoneGapin tarjoaman dokumentaation kautta, joka löytyy sivustolta http://docs.phonegap.com/en/3.3.0/guide_platforms_index.md.html#Platform%20Guides.

Tässä työssä PhoneGapin Android-kehitysympäristö otettiin käyttöön 64-bittisellä Windows 7 -käyttöjärjestelmällä. PhoneGap tarjoaa selkeän ohjeistuksen kehitysympäristön pystyttämiseen ja käyttämiseen myös Linux- ja Mac OS -käyttöjärjestelmissä. [62.]

PhoneGapin uusimmat versiot vaativat The Command-Line Interfacen (CLI) käyttöä PhoneGap-projekteissa. Tämän hyödyntämiseen täytyy sovelluskehittäjän ladata Node.js -sovellusalusta, joka mahdollistaa npm-paketinhallintajärjestelmän käyttämisen Windowsin komentorivillä (Command Prompt) tai vastaavasti Linuxin terminaalissa. Tällöin komentorivin kautta pystytään muun muassa luomaan projekteja, lisäämään tähän halutut laitteistoalustat sekä suorittaa toteutettua sovellusta kohdelaitteella tai emulaattorilla. [63.]

PhoneGap-sovelluskehityksessä tarvitaan myös Android SDK -työkaluja. Nämä sisältävät tärkeät API-kirjastot ja kehitystyökalut sovellusten toteuttamista varten. Android Developers tarjoaa sovelluskehittäjille myös suoraan täydellistä kehitysympäristöpakettia, joka sisältää SDK-työkalujen lisäksi myös Eclipse IDE -ohjelmointiympäristön sekä sen sisälle rakennetun ADT (Android Developer Tools) -liitännäisen. [64.]

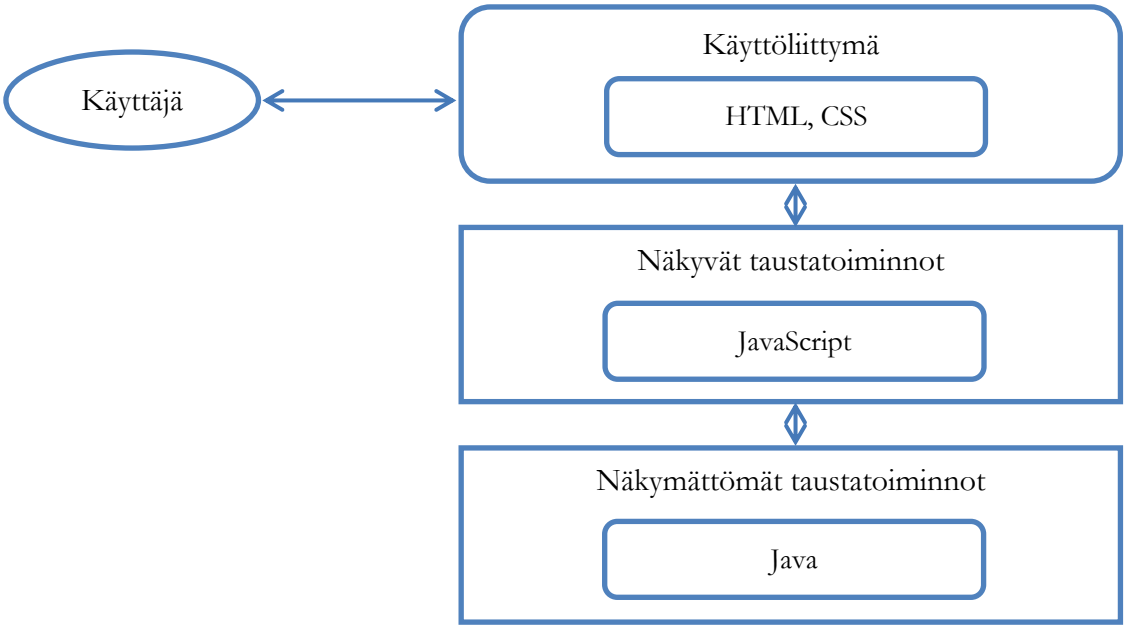
Java Development Kit (JDK) ja Apache Ant ovat tarpeellisia PhoneGap-projektissa. JDK sisältää työkalut sovellusten kehittämistä varten sekä Java Runtime Environmentin (JRE) sovellusten suorittamiseksi. [65.] Apache Ant on Java-kirjasto ja komentorivityökalu, jolla pystytään ajamaan sovelluksen build-tiedostoissa määriteltyjä prosesseja [66].

Kehitysympäristön lisäksi sovelluksen toteuttamiseen tarvittiin työkalu, jolla pystyttäisiin luomaan grafiikkaa sovelluksen ulkoasuun. Graafisia elementtejä ovat muun muassa logot ja kuvat, joiden toteuttamisessa käytettiin Adobe Photoshop CS2 -kuvankäsittelyohjelmistoa.

Tarvittavien kehitystyökalujen asentamisen jälkeen aloitettiin PhoneGap-sovelluksen kehittäminen asentamalla tietokoneelle Apache Cordova JavaScript-kirjasto hyödyntämällä asennettua npm-paketinhallintajärjestelmää. Työkalu asennettiin komentoriville kirjoitettavalla **npm install -g cordova** -komennolla, jonka jälkeen Cordovaa voitiin käyttää PhoneGap-sovelluksessa tarvittavana moottorina. Cordovan asentamisen jälkeen työkalulla luotiin uusi PhoneGap-projekti, lisättiin Android-laitealustat mukaan projektiin ja tätä alettiin kehittämään Eclipse IDE -ohjelmointiympäristössä. [62.]

7.2 Sovelluksen rakenne

Rakenteeltaan sovellusta lähdettiin kehittämään siten, että sen jatkokehittäminen olisi mahdollisimman helppoa ja lisäominaisuuksien mukaan ottaminen vaivatonta. Perusrakenne koostuu käyttöliittymästä, siinä tapahtuvasta vuorovaikutuksesta käyttäjän kanssa ja taustalla tapahtuvista toiminnallisuuksista. PhoneGap-sovellus toteutettiin täten usealla eri tasolla, joihin pystytään helposti lisäämään lisää toiminnallisuutta myöhäisemmissä vaiheissa. Rakenne on yksinkertaisuudessaan esitelty kuvassa 12.



Kuva 12. Sovelluksen perusrakenne.

Tasoista ensimmäinen koostuu käyttöliittymätasosta, joka sisältää HTML-, CSS- ja myös JavaScript-ohjelmointikielillä toteutettua graafista ulkoasua. Käyttöliittymätasolla tapahtuu käyttäjän ja sovelluksen välillä tapahtuva vuorovaikutus. Ulkoasun peruselementtien lisäksi käyttäjälle välittyy käyttöliittymän kautta toisen tason toimintoja, jotka ovat näkyvää taustatoiminnallisuutta. Nämä on toteutettu JavaScript-ohjelmointikielellä, ja ne ovat vuorovaikutuksessa kolmannen tason kanssa. Vuorostaan nämä näkymättömät toiminnallisuudet on kirjoitettu laitteen natiivilla Java-ohjelmointikielellä ja ne sisältävät esimerkiksi laitteen Bluetooth-teknologiaan liittyvät toiminnot. Näkyvä toiminnallisuus välittää tästä taustalla tapahtuvasta toiminnasta informaation käyttöliittymään, jonka kautta käyttäjä on tietoinen tapahtuneesta toiminnasta.

7.3 Sovelluksen toteuttaminen

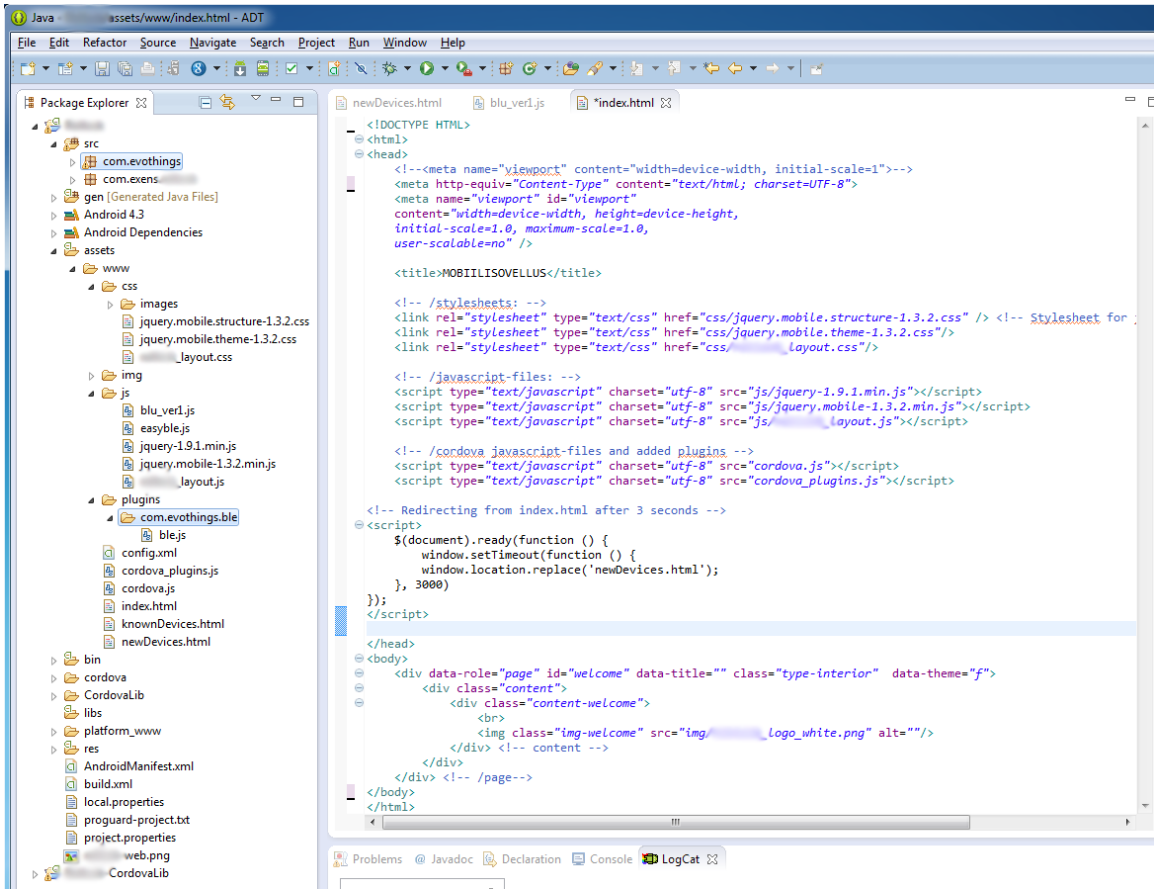
Sovelluksen toteuttaminen aloitettiin ulkoasun perusrakenteen luomisesta. Sitä lähdettiin toteuttamaan mainostoiminnon suunnitteleman ulkoasun pohjalta, jota oli mahdollista muokata eikä ollut täten ”orjallisesti” noudatettavissa. Suunnitellun ulkoasun toteuttamisessa syntyi käytännön tasolla pieniä esteitä, joita täytyi muokata web-teknologian sallimissa rajoissa. Tärkeimpinä ominaisuuksia toteutuksessa oli yksinkertainen ulkoasurakenne, joka on käytettävyydeltään helppo ja miellyttävän tuntuinen. Ulkoasun ja toiminnallisuuksien

luomisen apuna käytettiin jQuery Mobilen tarjoamia UI-kirjastoja, jotka sisältävät valmiita komponentteja mobiililaitteille toteutettavan ulkoasun luomiseksi.

Sovelluksen ulkoasun perusrakenteen jälkeen täytyi sovellukseen toteuttaa toiminnallisuus sovelluksen käyttöliittymän ja laitteen BLE-teknologian välille. PhoneGap-sovelluskehityksessä tämä tarkoittaa joko olemassa olevan PhoneGap-liitännäisen käyttämistä tai itse liitännäisen alusta saakka itse kirjoittamista. Bluetooth-teknologian hyödyntäminen on mahdollista PhoneGap-sovelluksissa, mutta työn alkuvaiheissa kunnollista liitännäistä vähävirtaisen BLE-teknologian käyttämiseen ei tahtonut löytyä.

Taustatutkimusten kautta löytyi Plugin Registryn sivuilta pätevä BLE-liitännäinen Android- ja iOS-laitteille vähävirtaisen BLE:n käyttämiseksi PhoneGap-sovelluksissa. Tämä on saatavilta sivulta http://plugreg.com/plugins ja se on julkaistu nimellä Evothings BLE API. Se sisältää valmiin Java-koodin laitteen natiiviin BLE-teknologiaan käsiksi pääsemiseksi sekä JavaScript-kirjaston PhoneGap-sovelluksen toiminnallisuuden luomiseksi. [61.]

PhoneGap-liitännäisen mukaan ottaminen projektiin tarvittiin myös GIT-työkalu, joka saatiin ladattua osoitteesta http://git-scm.com/downloads. Työkalun asentamisen jälkeen projektiin lisättiin BLE-plugin komentorivin komennolla **cordova plugin add https://github.com/evothings/cordova-ble.git**. Komento lisää olemassa olevaan projektiin Java-lähdekoodin ja JavaScript-kirjastot omiin hakemistoihinsa. Kuvan 13 vasemmalla laidalla on esitelty näiden hakemistojen lisäksi myös koko PhoneGap-sovelluksen hakemistorakenne. Samassa kuvassa on nähtävillä myös etusivu index.html, joka aukeaa PhoneGap-sovelluksissa ensimmäisenä näkymänä. [33].



Kuva 13. PhoneGap-sovelluksen hakemistorakenne ja index.html-tiedoston sisältö Eclipse-projektissa.

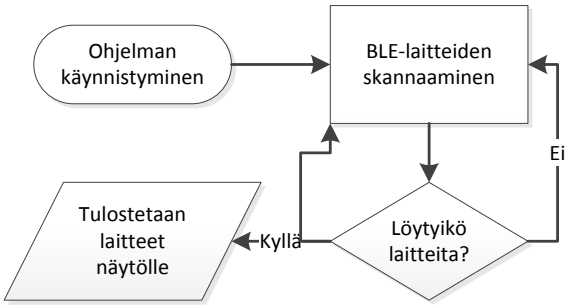
Koska BLE on käytettävissä vain Androidin käyttöjärjestelmäversioista 4.3 eteenpäin, täytyi tämä huomioida myös AndroidManifest.xml-tiedostossa. Tämä määrittelee sovellukselle kaikki tärkeimmät tiedot, joihin kuuluvat myös tavoiteltavat API-versiot. Jotta sovellus olisi yhteensopiva vain BLE-tuen tarjoamien käyttöjärjestelmäversioiden kanssa, täytyi AndroidManifest-tiedostoa muokata kuvan 14 mukaiseen muotoon. BLE-tuki on tarjolla Androidille API-versiosta 18 eteenpäin, joten se määriteltiin Androidin käyttämäksi minimialustaksi. Tiedostoon on myös BLE-liitännäisen asennuksen aikana automaattisesti määritelty sovelluksen lupa päästä käsiksi laitteen Bluetooth-ominaisuuteen.

```
<?xml version='1.0' encoding='utf-8'?>
<manifest
    android:hardwareAccelerated="true"
    android:versionCode="1" android:versionName="0.0.1"
    android:windowSoftInputMode="adjustPan"
    package="com.exens.[redacted]"
    xmlns:android="http://schemas.android.com/apk/res/android">
    <supports-screens
        android:anyDensity="true"
        android:largeScreens="true"
        android:normalScreens="true"
        android:resizeable="true"
        android:smallScreens="true"
        android:xlargeScreens="true" />
    <uses-permission
        android:name="android.permission.INTERNET" />
    <application
        android:debuggable="true"
        android:hardwareAccelerated="true"
        android:icon="@drawable/[redacted]"
        android:label="@string/app_name">
        <activity
            android:configChanges="orientation|keyboardHidden|keyboard|screenSize|locale"
            android:label="@string/app_name"
            android:name="[redacted]"
            android:theme="@android:style/Theme.Black.NoTitleBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
    <uses-sdk android:minSdkVersion="18" android:targetSdkVersion="19" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
</manifest>
```

Kuva 14. AndroidManifest.xml-tiedoston sisältö.

Evothingsin BLE-liitännäisen JavaScript-tiedosto ei suoraan toimi siltana laitteen natiivitoiminnon ja sovelluksen välillä, vaan se toimii enemmänkin dokumentaationa liitännäisen käyttämiseksi. Tämän vuoksi koko JavaScript-rajapinta täytyi kirjoittaa kokonaan itse. JavaScript-koodissa kutsutaan funktiokutsuilla natiivipuolelle toteutettua Java-koodin toimintoa, jonka kautta sovellus pääsee käsiksi BLE-toiminnallisuuteen.

JavaScript-rajapinta toteutettiin siten, että sovellusta käynnistettäessä aloitetaan BLE-laitteiden skannaaminen. Kun sovelluksen etusivu käynnistyy, toimii ohjelma kuvan 15 mukaisella logiikalla.

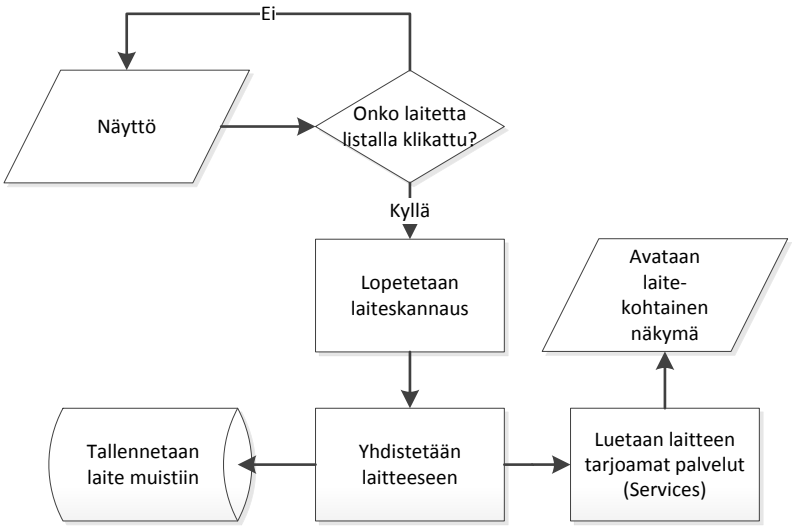


Kuva 15. Sovelluksen ja BLE-teknologian vuorovaikutus sovelluksen käynnistyessä.

BLE-laitteita skannataan jatkuva-aikaisesti. Laitteen löytyessä tämä näytetään uutena elementtinä käyttöliittymässä. Elementti sisältää tärkeimmät tiedot laitteesta, kuten sen nimen, MAC-osoitteen sekä signaalin vahvuuden. Kuvassa 16 on esimerkki laiteskannauksen tuloksesta, jossa sovellus on löytänyt kaksi eri BLE-laitetta. Samassa on myös nähtävillä itse sovelluksen perusulkoasua.

Kuva 16. Laiteskannauksen tulokset.

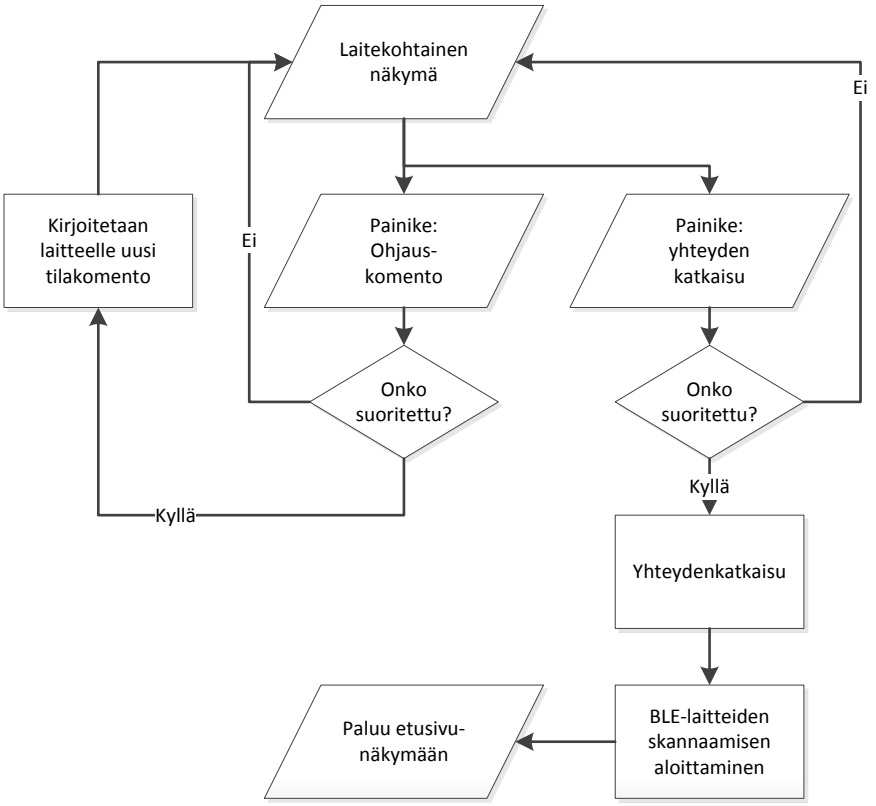
Laitteen näyttävälle elementille on määritelty toiminnallisuus, joka toteutetaan käyttäjän toimenpiteen kautta. Kun näyttöliittymässä näkyvän laitteen päällä klikataan käyttäjän toimesta, aloitetaan kyseiseen laitteeseen yhdistäminen. Toiminnon seurauksena avautuu sovelluksessa uusi näkymä, missä näkyy laitteen tärkeimpien tietojen lisäksi sen yhteyden tila sekä laitteelle kohdistettuja toimintoja. Yhdistämisvaihe noudattaa kuvan 17 mukaista logiikkaa.



Kuva 17. Sovelluksen toiminnallisuus laitteeseen yhdistettäessä.

BLE-laitteiden skannaaminen keskeytetään laitetta yhdistettäessä kohdelaitteeseen yhdistämisen mahdollistamiseksi. Laitteeseen yhdistettäessä laitteen uniikki MAC-osoite tallennetaan sovelluksen muistiin PhoneGapin tarjoaman local storagen avulla. Muistia hyödynnetään tulevaisuudessa siten, että sovelluksella on mahdollisuus yhdistää aikaisemmin tunnettuun ja yhdistettyyn laitteeseen automaattisesti laitteen löytyessä.

Laitekohtainen näkymä aukeaa aina laitteeseen yhdistettäessä. Siinä on nähtävillä laitteen yhteyden tila, sen nimi, MAC-osoite ja laitekohtaiset painikkeet. Näkymästä pystytään lähettämään suoritettava ohjauskomento laitteelle tai katkaisemaan luotu yhteys Android-laitteen ja yhdistetyn BLE-laitteen välillä. Laitekohtaisen näkymän toiminnallisuus on havainnollistettu kuvassa 18.



Kuva 18. Yhdistetyn laitteen toiminnallisuus.

Välittömästi laitteeseen yhdistämisen jälkeen luetaan laitteen tarjoamat palvelut (Services). Käytännössä tämä tarkoittaa BLE-profiilien laitteelle määrittelemiä palveluita, joita se tarjoaa sitä etsiville BLE-laitteille. Nämä palvelut sisältävät ominaisuuksia (Characteristics), jotka sisältävät laitteessa olevaa tietoa tai tietyn ominaisuuden tilaa. Näitä tiloja ei pystytä muuttamaan ennen kuin laitteen palvelut ovat sovelluksen tiedossa. Kaikki laitteen Services- ja Characteristics-tiedot ovat saatavilla tietyn ennalta määritellyn UUID:n (Universally Unique Identifier) kautta.

Muuttamalla BLE-laitteen Servicen sisällä olevan Characteristicsin tilaa, pystytään esimerkiksi ohjaamaan tähän ominaisuuteen määritellyn ledin arvoa. Sovelluksessa Services-tietojen lukemisen jälkeen Characteristicsin tilaa voidaan vaihtaa kirjoittamalla tämän UUID:n sisältämäksi arvoksi joko 1 tai 0. Sovelluksessa käytetty UUID on 128-bittinen merkkijono, joka on ennalta määritelty yhdistettyyn BLE-laitteeseen.

Kuvassa 19 on esimerkkikoodi JavaScriptilla toteutetusta Characteristics-tilan muuttamisesta. Avautuva näkymä sisältää painikkeet laitteelle suoritettavien ohjauskomentojen toteuttamiseksi, jotka sisältävät toiminnot tilan vaihtamiseksi. Mikäli jompaakumpaa

painiketta painetaan, siirrytään Characteristicsin kirjoittamiseen ja viedään painikkeen määrittelemä arvo (1 tai 0) ja kirjoitetaan tämä ominaisuuden uudeksi tilaksi.

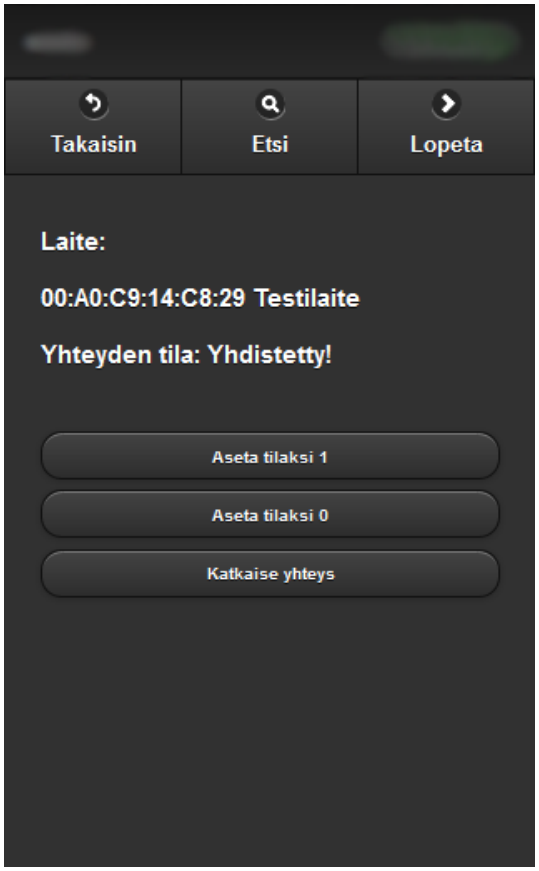
```
// Aseta laitteen ominaisuuden tilaksi PÄÄLLE
app.deviceWriteOn = function()
{
    app.device && app.device.writeDataArray(new Uint8Array([1]));
};

//Aseta laitteen ominaisuuden tilaksi PÄÄLLE
app.deviceWriteOff = function()
{
    app.device && app.device.writeDataArray(new Uint8Array([0]));
};


// Kirjoitetaan asetettu arvo laitteen Characteristicsiin
app.addMethodsToDeviceObject = function(device)
{
    console.log('kirjoitetaan data... ');
    device.writeDataArray = function(uint8array)
    {
        device.writeCharacteristic(
                '00005678-0000-1000-8000-00805f9b34fb',
                uint8array,
                function()
                {
                    console.log('writeCharacteristic success!');
                },
                function(errorcode)
                {
                    console.log('writeCharacteristic error: ' + errorcode);
                }
        );
    }
};
```

Kuva 19. Esimerkkikoodi laitteen Characteristicsin sisältämän tilan muuttamisesta.

Laitenäkymästä pystytään myös katkaisemaan yhteys yhdistettyyn laitteeseen. Yhteyttä katkaistaessa aloitetaan BLE-laitteiden skannaaminen uudestaan ja palataan sovelluksen aloitusnäkymään, josta on nähtävillä skannatut laitteet. Esimerkkikuva yhdistetyn laitteen näkymästä on esitetty kuvassa 20.



Kuva 20. Yhdistetyn laitteen näkymä sekä näkymässä suoritettava toiminnallisuus.

7.4 Testaaminen

Sovellusta testattiin alkuvaiheissa ADT:n sisältämän emulaattorin avulla. Sillä pystytään mallintamaan oikeaa Android-laitetta ja sen perustoiminnallisuuksia. Emulaattori ei kuitenkaan pysty toteuttamaan minkäänlaisia laitteen natiivitoimintoja, koska sitä käytetään sovelluskehittäjän tietokoneella eikä ole täten yhteydessä minkäänlaiseen fyysiseen Android-laitteeseen. Se kuitenkin soveltui erinomaisesti perusulkoasun testaamiseen ja yksinkertaisten komentojen suorittamiseen, joita sovelluksen käyttöliittymältä vaaditaan.

Itse sovelluksen natiivitoiminnallisuuksia, kuten BLE-teknologiaa, pystyttiin testaamaan vain oikealla Android-laitteella, josta löytyy vähintään BLE-tuen vaatima Android-käyttöjärjestelmä. Tämä laite oli käytettävissä yrityksen tiloissa, jossa pystyttiin testaamaan sovelluksen toimintaa Android-laitteen ja BLE-laitteen välillä.

Testaamista Android-laitteella toteutettiin suoraan Eclipsen kautta. ADT sisältää LogCat-toiminnon sovelluksen testiajamisen (debug) yhteyteen. Tämä kerää lokiviestejä Android-laitteelta ja näyttää nämä Eclipsen käyttöliittymässä. Sen avulla nähtiin reaaliaikaisesti sovelluksessa tapahtuvia asioita ja täten sovelluksen toiminnan kehittäminen oli helpompaa.

Testaaminen sisälsi lähinnä BLE-teknologiassa tapahtuvia toimintoja, joita pystyttiin kehittämään LogCatin ulosannin avulla. Esimerkiksi skannattuun BLE-laitteeseen yhdistettäessä nähtiin koodiin lisätyn tulostuskomennon avulla mitä yhdistettäessä tapahtui. Täten ongelmatilanteiden ratkominen näitä ilmaantuessa oli helpompaa kuin suoritettaessa sovellusta Android-laitteella ilman ADT:n LogCat-toimintoa.

Testatessa todennettiin myös local storagen toimivuus. Kun tietty laite paritettiin ensimmäistä kertaa, tallennettiin kyseisen laitteen osoite local storagen muistiin. Sovellus käynnistettiin uudestaan, jolloin haettiin tallennettujen laitteiden osoitteet muistista. Kun tämä entuudestaan tunnettu laite löytyi laitteiden skannaamisen yhteydessä, yhdistettiin laitteeseen automaattisesti.

8 YHTEENVETO

Mobiilisovellusmarkkinoilla syntyy väistämättä uusia toteutus- ja kehitysmahdollisuuksia sovellusten toteuttamiseksi. Laitteistoriippumattomat web-sovellukset valtaavat markkina-alaa, vaikka nämä ovatkin haastajan asemassa natiivisovelluksiin verrattuna. PhoneGap-sovelluskehys tarjoaa mainiot olosuhteet natiivinomaisen mobiilisovelluksen kehittämiseksi web-teknologioita hyödyntämällä. Käyttämällä tämän tarjoamaa JavaScript-rajapintaa laitteen toiminnallisuuden ja web-selaimen välillä pystytään luomaan natiivisovelluksen omaisia ja ulkoasultaan web-ratkaisujen tyylisiä mobiilisovelluksia mobiililaitteen ominaisuuksia hyödyntäen.

Tämän opinnäytetyön tavoitteena oli toteuttaa HTML5-, CSS- ja JavaScript-tekniikoita käyttäen mobiilisovellus Android-laitealustalle, jolla pystytään toteuttamaan kommunikointi toisen laitteen kesken Bluetooth Low Energy -teknologiaa käyttämällä. Lopputuloksena oli yksinkertainen PhoneGap-sovelluskehyksellä toteutettu mobiilisovellus Android-laitteelle, jota on helppo lähteä laajentamaan web-tekniikoiden kautta.

Androidin natiivikielellä kirjoitettuna lopputulos sovelluksen osalta olisi voinut olla sama, ellei jopa parempi kuin nyt saavutettu tulos, koska BLE-teknologiaa on yleisesti hyödynnetty natiiviratkaisuissa PhoneGap-toteutustapaa enemmän. Tällä toteutustavalla sovelluksen alustariippumattomuus vuorostaan olisi kärsinyt.

Käytetty BLE-liitännäinen sisältää iOS-laiteympäristöissä tarvittavat, Objective-C-kielellä kirjoitetut .h- ja .m-tiedostot Androidin Java-koodin lisäksi, jolloin toteutus on näiden osalta toimintavalmis molemmille alustoille. Sovellus on myös helposti käännettävissä iOS:lle Adoben tarjoaman PhoneGap Build -pilvipalvelun avulla. Ainoastaan Windowsille laajennettaessa täytyy sovellukseen toteuttaa tämän vaatima natiivipuolen ohjelmointi erikseen.

Lähtökohtaisesti PhoneGap-sovelluskehyksellä lähdettiin kokeilemaan, mihin tämä loppujen lopuksi käytännössä kykenee. Vaarallisin yhtälö tällä tavalla toteutettuna olisi voinut olla toimimaton sovellusratkaisu, jolloin toteutustapaa olisi mahdollisesti joutunut muuttamaan natiivin sovelluskehityksen suuntaan. Projektin aikataulutuksen olisi voinut toteuttaa entistä paremmin, jolloin mahdollisen esteen vuoksi olisi vaihtoehtoisen toteutustavan ehtinyt

vieläkin toteuttamaan. Heikon suunnittelun vuoksi myös työn suorittaminen venyi hieman tavoitellun aikataulun ylitse.

Työ oli heti tutustumisvaiheessa hyvin mielenkiintoinen. Mobiilikehitys on kiinnostava aihealue ja eritoten uudet sovellusympäristöt sekä teknologiat mielenkiintoisia työhön kuuluneita osia. Projektin aloittamiseen ja sen suorittamiseen vaikuttaneet ulkoiset seikat aloitusvaiheessa vaikuttivat projektin käynnistymiseen hieman negatiivisesti, jolloin myös paineet työn suorittamisen osalta kasvoivat aikataulun venyessä. Työn tarjoajan joustavuus ja hieno yhteistyö auttoivat näiden purkamisessa, joten lopputuloksena saatiin toteutettua kehitettävälle sovellukselle tämän perusrunko ensimmäisten toiminnallisuuksien osalta. Vaikka tavoitteet työn etenemisen suhteen olivatkin kokonaisuuden puolesta hieman korkeammalla, onnistui työ toteutetulta osaltaan mainiosti.

LÄHTEET

1 Exens Development. Ideasta tuotteeksi. Luettu 17.4.2014. [WWW-dokumentti]. <http://www.exens.com/index.php?p=31>

2 Kauppalehti. Exens Development Oy. Luettu 17.4.2014. [WWW-dokumentti]. <http://www.kauppalehti.fi/yritykset/yritys/exens+development+oy/08234549>

3 techopedia. Smartphone. Luettu 2.3.2014. [WWW-dokumentti]. <http://www.techopedia.com/definition/2977/smartphone>

4 techopedia. Mobile Device. Luettu 2.3.2014. [WWW-dokumentti]. <http://www.techopedia.com/definition/23586/mobile-device>

5 Strategy Analytics. Global Smartphone Shipments in 2013. Päivitetty 29.1.2014. [WWW-dokumentti]. <http://blogs.strategyanalytics.com/WSS/post/2014/01/29/Android-Captured-79-Share-of-Global-Smartphone-Shipments-in-2013.aspx>

6 Harju Jukka. Android-ohjelmoinnin perusteet. Helsinki: Books on Demand, 2013. 192 s. ISBN: 978-952-286-612-7

7 Android community. Samsung dominates Android scene with 63 percent of market Share. Päivitetty marraskuu 2013. [WWW-dokumentti]. <http://androidcommunity.com/samsung-dominates-android-scene-with-63-percent-of-market-share-20131112/>

8 Hongkiat.com. A Look Into: The History of IOS And Its Features. Luettu 7.3.2014. [WWW-dokumentti]. <http://www.hongkiat.com/blog/ios-history/>

9 iOS Developer Library. iOS App Programming Guide. Luettu 10.3.2014. [WWW-dokumentti]. <https://developer.apple.com/library/ios/documentation/iphone/conceptual/iphoneosprogrammingguide/TheiOSEnvironment/TheiOSEnvironment.html>

10 About.com. What is the App Store. Luettu 11.3.2014. [WWW-dokumentti]. <http://ipod.about.com/od/iphonesoftwareterms/g/app_store_def.htm>

11 Code Project. Getting Started with iPhone and iOS Development. Päivitetty 19.7.2010. [WWW-dokumentti]. <http://www.codeproject.com/Articles/88929/Getting-Started-with-iPhone-and-iOS-Development>

12 engadget.com. Microsoft confirms Windows Phone 7 manufacturers: ASUS, Dell, HTC, LG, and Samsung all on board. Päivitetty 22.7.2010. [WWW-dokumentti]. <http://www.engadget.com/2010/07/22/microsoft-confirms-windows-phone-7-manufacturers-asus-dell-ht/>

13 Paul Thurrott's supersite for Windows. Windows Phone Device Stats: December 2013. Päivitetty 30.12.2013. [WWW-dokumentti]. <http://winsupersite.com/windows-phone/windows-phone-device-stats-december-2013>

14 wedopedia. The History of Microsoft Operating Systems. Päivitetty 27.1.2012. [WWW-dokumentti]. <http://www.webopedia.com/DidYouKnow/Hardware_Software/history_of_microsoft_windows_operating_system.html>

15 ZDNet. Microsoft's Windows Phone 8 finally gets a 'real' Windows core. Päivitetty 20.6.2012. [WWW-dokumentti]. <http://www.zdnet.com/blog/microsoft/microsofts-windows-phone-8-finally-gets-a-real-windows-core/12975>

16 Microsoft Windows Phone. Getting started with developing for Windows Phone. Päivitetty 7.1.2014. [WWW-dokumentti]. <http://msdn.microsoft.com/library/windowsphone/develop/ff402529(v=vs.105).aspx>

17 webopedia. Windows Phone. Luettu 11.3.2014. [WWW-dokumentti]. <http://www.webopedia.com/TERM/W/windows_phone.html>

18 CNET.com. Google Android 4.4 KitKat Preview - KitKat promises Android for all (hands-on). Päivitetty 7.12.2013. [WWW-dokumentti]. <http://www.cnet.com/products/google-android-kitkat/>

19 tutorialspoint. Android Architecture. Luettu 7.4.2014. [WWW-dokumentti]. <http://www.tutorialspoint.com/android/android_architecture.htm>

20 IBM developerWorks. Introduction to Android development. Luettu 8.4.2014. [WWW-dokumentti]. <http://www.ibm.com/developerworks/opensource/library/os-android-devel/>

21 Just a moment! Android Started – Note 2: Android file .apk decompile. Lisätty 4.6.2013. [WWW-dokumentti]. <http://justamomentgoose.wordpress.com/2013/06/04/android-started-note-2-android-file-apk-decompile/>

22 Android Developers. App Manifest. Luettu 8.4.2014. [WWW-dokumentti]. <http://developer.android.com/guide/topics/manifest/manifest-intro.html>

23 Peltomäki Juha, Malmirae Pekka. Java-ohjelmoinnin peruskirja. Jyväskylä, Docendo 2002. 5. painos. s. 553. ISBN: 951-846-038-8

24 techopedia. Java. Luettu 8.4.2014. [WWW-dokumentti]. <http://www.techopedia.com/definition/3927/java>

25 techopedia. C++ Programming Language. Luettu 9.4.2014. [WWW-dokumentti]. <http://www.techopedia.com/definition/26184/c-programming-language>

26 SearchSOA. XML (Extensible Markup Language). Päivitetty Toukokuu 2007. [WWW-dokumentti]. <http://searchsoa.techtarget.com/definition/XML>

27 IBM. Advantages of XML. Luettu 8.4.2014. [WWW-dokumentti]. <https://publib.boulder.ibm.com/infocenter/iseries/v5r4/index.jsp?topic=%2Frzamj%2Frzamjintroadvantages.htm>

28 Wargo John M. PhoneGap Essentials: Building Cross-Platform Mobile Apps. Upper Saddle River, NJ: Addison-Wesley, 2012. 359 s. ISBN 978-0-321-81429-6

29 PhoneGap Blog. PhoneGap, Cordova, and what's in a name? Päivitetty 19.3.2012. [WWW-dokumentti]. <http://phonegap.com/2012/03/19/phonegap-cordova-and-what%E2%80%99s-in-a-name/>

30 Palmieri, M. Singh, I. Cicchetti, A. Comparison of cross-platform mobile development tools. 16th International Conference on Intelligence in Next Generation Networks, 2012. s. 179–186. E-ISBN: 978-1-4673-1525-8

31 PhoneGap.com. Supported Features. Luettu 11.3.2014. [WWW.dokumentti]. <http://phonegap.com/about/feature/>

32 PhoneGap Documentation. Plugin Development guide, versio 3.3.0. Luettu 11.3.2014. [WWW-dokumentti]. <http://docs.phonegap.com/en/3.3.0/guide_hybrid_plugins_index.md.html#Plugin%20Development%20Guide>

33 Adobe PhoneGap Build. Plugins. Luettu 11.3.2014. [WWW-dokumentti]. <https://build.phonegap.com/plugins>

34 w3schools.com. HTML5 Web Storage. Luettu 31.3.2014. [WWW-dokumentti]. <http://www.w3schools.com/html/html5_webstorage.asp>

35 W3C Working Group Note. Web SQL Database. Päivitetty 18.11.2010. [WWW-dokumentti]. <http://dev.w3.org/html5/webdatabase/>

36 w3schools.com. HTML5 introduction. Luettu 5.4.2014. [WWW-dokumentti]. <http://www.w3schools.com/html/html5_intro.asp>

37 TechRadar - Technology News and Reviews. HTML5: what is it? Päivitetty 31.12.2011. [WWW-dokumentti]. <http://www.techradar.com/news/internet/web/html5-what-is-it-1047393>

38 w3schools.com. JavaScript Introduction. Luettu 5.4.2014. [WWW-dokumentti]. <http://www.w3schools.com/js/js_intro.asp>

39 Web Teacher. JavaScript Tutorial. Luettu 5.4.2014. [WWW-dokumentti]. <http://www.webteacher.com/javascript/index.html>

40 JavaScripter.net FAQ. What is JavaScript? Luettu 5.4.2014. [WWW-dokumentti]. <http://www.javascripter.net/faq/whatisja.htm>

41 readwrite.com. Why JavaScript Will Become The Dominant Programming Language Of The Enterprise. Lisätty 9.8.2013. [WWW-dokumentti]. <http://readwrite.com/2013/08/09/why-javascript-will-become-the-dominant-programming-language-of-the-enterprise#awesm=~oAAnWK5POJZfRi>

42 About.com. What is CSS? What are Cascading Style Sheets? Luettu 5.4.2014. [WWW-dokumentti]. <http://webdesign.about.com/od/beginningcss/a/aa021607.htm>

43 W3C. HTML & CSS. Luettu 5.4.2014. [WWW-dokumentti]. <http://www.w3.org/standards/webdesign/htmlcss>

44 About.com What is CSS3? An Introduction to the Modularization of Cascading Style Sheets (level 3). Luettu 5.4.2014. [WWW-dokumentti]. <http://webdesign.about.com/od/css3/a/aa061206.htm>

45 techopedia. jQuery. Luettu 5.4.2014. [WWW-dokumentti]. <http://www.techopedia.com/definition/3977/jquery>

46 w3schools.com. jQuery Introduction. Luettu 5.4.2014. [WWW-dokumentti]. <http://www.w3schools.com/Jquery/jquery_intro.asp>

47 HTMLGoodies. Introduction To jQuery Mobile. Luettu 5.4.2014. [WWW-dokumentti]. <http://www.htmlgoodies.com/html5/tutorials/introduction-to-jquery-mobile.html#fbid=5Jm0nBT4KWT>

48 Adobe PhoneGap Build. Package mobile apps in the cloud. Luettu 8.5.2014. [WWW-dokumentti]. < http://html.adobe.com/edge/phonegap-build/faq.html>

49 Granlund Kaj. Tietoliikenne. Jyväskylä: Docendo, 2007. s. 474. ISBN: 978-951-0-32821-7

50 Bluetooth.com. Technical Information: About Bluetooth Low Energy Technology. Luettu 12.3.2014. [WWW-dokumentti]. <http://www.bluetooth.com/Pages/low-energy-tech-info.aspx>

51 Gnome help. Mikä MAC-osoite on? Luettu 24.3.2014. [WWW-dokumentti]. <https://help.gnome.org/users/gnome-help/stable/net-macaddress.html.fi>

52 Pocketnow. Here's why you should be thrilled about Bluetooth 4.0. Päivitetty 22.5.2013. [WWW-dokumentti]. <http://pocketnow.com/2013/05/22/bluetooth-4-0-devices>

53 CSR. Bluetooth® Smart. Luettu 29.3.2014. [WWW-dokumentti]. <http://www.csr.com/products/technology/low-energy>

54 Bluetooth. Bluetooth Smart Technology: Powering the Internet of Things. Luettu 29.3.2014. [WWW-dokumentti]. <http://www.bluetooth.com/Pages/Bluetooth-Smart.aspx>

55 Medical Electronics Design. Bluetooth Low Energy vs. Classic Bluetooth: Choose the Best Wireless Technology For Your Application. Päivitetty 8.6.2012. [WWW-dokumentti]. <http://www.medicalelectronicsdesign.com/article/bluetooth-low-energy-vs-classic-bluetooth-choose-best-wireless-technology-your-application>

56 Bluetooth Developer Portal. Bluetooth Smart (Low Energy) Technology. Luettu 29.3.2014. [WWW-dokumentti]. <https://developer.bluetooth.org/TechnologyOverview/Pages/BLE.aspx>

57 Bluetooth Developer Portal. Generic Attribute Profile (GATT). Luettu 30.3.2014. [WWW-dokumentti]. <https://developer.bluetooth.org/TechnologyOverview/Pages/GATT.aspx>

58 Java Code Geeks. Native vs. Mobile Web vs. Hybrid applications. Julkaistu 5.12.2013. [WWW-dokumentti]. <http://www.javacodegeeks.com/2013/12/native-vs-mobile-web-vs-hybrid-applications.html>

59 About.com. The Pros and Cons of Native Apps and Mobile Web Apps. Luettu 2.5.2014. [WWW-dokumentti]. <http://mobiledevices.about.com/od/additionalresources/qt/The-Pros-And-Cons-Of-Native-Apps-And-Mobile-Web-Apps.htm>

60 sealed abstract. Why mobile web apps are slow. Julkaistu 9.7.2013. [WWW-dokumentti]. <http://sealedabstract.com/rants/why-mobile-web-apps-are-slow/>

61 PlugReg. Evothings BLE API by evothings. Luettu 24.1.2014. [WWW-dokumentti]. <http://plugreg.com/plugin/evothings/cordova-ble>

62 PhoneGap Documentation. Android Platform Guide. Luettu 13.4.2014. [WWW-dokumentti].
<http://docs.phonegap.com/en/3.3.0/guide_platforms_android_index.md.html#Android%20Platform%20Guide>

63 PhoneGap Documentation. The Command-Line Interface. Luettu 13.4.2014. [WWW-dokumentti].
<http://docs.phonegap.com/en/3.3.0/guide_cli_index.md.html#The%20Command-Line%20Interface>

64 Android Developers. Get the Android SDK. Luettu 13.4.2014. [WWW-dokumentti].
<http://developer.android.com/sdk/index.html>

65 Oracle. Java SE Development Kit Downloads. Luettu 13.4.2014. [WWW-dokumentti].
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

66 Apache Ant. Welcome. Luettu 13.4.2014. [WWW-dokumentti].
<http://ant.apache.org/>