

Opinnäytetyö (AMK)

Tietotekniikka

Sulautetut ohjelmistot

2015

Elias Harri, Vesa Välimäki

SISÄTILAPAIKANNUSJÄRJESTELMÄN KÄYTETTÄVYYS KONTAMINOITUNEESSA TILASSA



TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

Elias Harri, Vesa Välimäki

SISÄTILAPAIKANNUSJÄRJESTELMÄN KÄYTETTÄVYYS KONTAMINOITUNEESSA TILASSA

Työssä tutkittiin sisätilapaikannusjärjestelmän käytettävyyttä tiloissa, joissa voi sattua kontaminaatiovahinko. Työ koostui sisätilapaikannusjärjestelmän laitteiston asennuksesta, määrittelemisestä ja sovelluksen kehittämisestä. Laitteistoon sisältyvät tag, lokaattoreita, kytkin ja palvelinkone. Tag kiinnitetään kontaminaatiomittariin, jotta sen sijaintia voidaan seurata. Tag kommunikoi lokaattorien kanssa käyttäen BLE-yhteyttä.

Työnteko aloitettiin asentamalla sisätilapaikannusjärjestelmän laitteisto haluttuun tilaan. Asennus toteutettiin hyödyntäen Quupan tuottamia asennusohjemateriaaleja. Laitteet määriteltiin tarkkailtavan alueen teknisten tietojen perusteella käyttäen Quuppa Site Planner -sovellusta, palvelinkoneella. Asennuksen jälkeen tagin sijaintia pystyttiin seuraamaan HTTP-kutsujen avulla. HTTP-kutsuilla kysytään tagin sijainti palvelimelta, joka palauttaa tiedot JSON-datana.

Seuraavaksi luotiin sovellus, jolla lähetettiin HTTP-kutsuja palvelimelle ja eriteltiin vastaanotettu JSON-data haluttuihin .Net-tietoluokkiin. Halutusta objektista luettiin tagin paikkatiedot ja hyödynnettiin niitä piirtämällä sen sijainti toimitilan pohjapiirustukseen. Täten visualisoitiin paikkatiedot halutulla tavalla.

Tagia seurattaessa todettiin muutamia ongelmia, kuten sijainnin tarkkuuden puute ja viive paikkatiedon päivittymisessä. Kyseinen viive tapahtui palvelimen ja laitteiston välisessä kommunikoinnissa. Tämä häytti kontaminaatiomittarin sijainnin seurantaa, jolloin laitteistolla ei päästy haluttuun tarkkuuteen. Järjestelmän asennus oli myös hankalaa ja sen siirtäminen kontaminoituneesta tilasta toiseen voi osoittautua vaikeaksi. Tällöin järjestelmä ei soveltunut haluttuihin käyttötarkoituksiin.

Työtä voisi jatkokehittää pienentämällä laitteiston välistä viivettä, parantamalla sijainnin tarkkuutta ja nopeuttamalla sisätilapaikannusjärjestelmän asennusta ja käyttöönottoa.

ASIASANAT:

sisätilapaikannus, Bluetooth, kontaminaatio

Elias Harri, Vesa Välimäki

INDOOR POSITIONING SYSTEMS FEASIBILITY IN A CONTAMINATION ENVIRONMENT

The object of this study was to determine whether Quuppa indoor positioning system is usable in a contamination environment. The indoor positioning system equipment consists of a tag, locators, a switch and a server. The tag was connected to a contamination meter, so that its position could be observed. The tag communicated with the locators using BLE.

The indoor positioning system was installed in a working environment. The work was started with calibrating and configuring the system using Quuppa Site Planner on the server. After this, the position data of the tag could be obtained by sending a HTTP request to the server. By making a certain HTTP request, the server returns wanted data in JSON format.

After this, the next step was the programming of software that would make the HTTP request and deserialize the returned JSON data into .Net classes. Data from .Net objects could then be used to get the position of the tag. The software then visualizes the position of the tag by drawing the position on a blueprint of the working environment.

Whilst observing the tags movements there arose some problems, such as the lack of accuracy in the position and latency formed in the communication between the system and the server. As such the positioning of the contamination meter is hindered somewhat and the accuracy required is not met. The installation of the system was also very hard which is why moving it between locations would be problematic. Therefore the system did not meet the requirements set in the study.

Future development should be focused on reducing the latency between the devices, improving the accuracy of the positioning and making the indoor positioning system setup easier and faster.

KEYWORDS:

positioning, deserializing

SISÄLTÖ

SANASTO	6
1 JOHDANTO	6
2 KÄYTETYT TEKNOLOGIAT JA TEORIA	8
2.1 Bluetooth	8
2.1.1 BLE	9
2.1.2 GATT	9
2.2 Sisätilapaikannus	11
2.2.1 HAIP	12
2.2.2 RTLS	13
2.3 JSON	13
2.4 Säteily	14
3 LAITTEET JA ASENNUS	16
3.1 Sisätilapaikannusjärjestelmä	16
3.1.1 Palvelin	18
3.1.2 Quuppa Site Planner -ohjelma	18
3.1.3 Tag	22
3.2 Kontaminaatiomittari ja säteilylähde	22
3.3 Järjestelmän käyttömahdollisuudet	23
4 SOVELLUS	24
4.1 Ohjelmointi	24
4.2 Sovelluksen käyttö	28
4.3 Tulosten visualisointi ja dokumentointi	29
5 YHTEENVETO	31
LÄHTEET	32

LIITTEET

Liite 1. Tagin JSON-datan esimerkki.

Liite 2. Ohjelman toimintaa kuvaavat UML-kaaviot.

KUVAT

Kuva 1. BLE-yhteyden toimintaperiaate.	10
Kuva 2. Tiedonhaku tagista. (Townsend 2015.)	10
Kuva 3. GATT-profiili. (Townsend 2015.)	11
Kuva 4. Sisätilapaikannuslaitteiston kytkentä.	16
Kuva 5. Sisätilapaikannusjärjestelmän asennus toimintatilassa.	17
Kuva 6. Lokaattorien sijainti pohjapiirustuksessa.	19
Kuva 7. Lokaattorin tunnistusikkuna.	20
Kuva 8. Lokaattorin kalibrointi-ikkuna.	21
Kuva 9. Kuva tagista.	22
Kuva 10. Sovellusikkuna.	29
Kuva 11. Ulostulot yhdistettynä.	30

KOODIT

Koodi 1. .Net-luokat.	25
Koodi 2. JSON-datan erittelymetodi.	26
Koodi 3. Paikkasijainnin suodatusmetodi	27
Koodi 4. Piirtotoiminto.	28

TAULUKOT

Taulukko 1. Bluetooth-versioiden nopeudet. (Bluetooth 2015.)	8
Taulukko 2. Sisätilapaikannusteknologioiden vertailu. (Quuppa 2013, 7.)	13

SANASTO

.NET	.NET Framework. Microsoftin kehittämä ohjelmistokomponenttikirjasto, jota Microsoft Visual Studio-ympäristössä kehitetyt ohjelmistot voivat käyttää.
.NET-luokat	.NET-luokat tai .NET-tietoluokat ovat .NET-ympäristössä olevia CSharp-kielellä kirjoitettuja luokkia, joita ohjelmoinnissa käytetään.
AoA	Angle of Arrival. Radioaaltojen tulokulmaan perustuva paikannustekniikka.
BLE	Bluetooth Low Energy. Vähän virtaa käyttävä Bluetooth-yhteys.
EDR	Enhanced Data Rate. Bluetooth-versiossa 2.0 esitelty nopeampi tiedonsiirtotekniikka.
GATT	Generic Attribute Profile. BLE-laitteen ominaisuudet sisältävä profiili.
HAIP	High Accuracy Indoor Positioning. Quupan kehittämä sisätilapaikannustekniikka.
HS	High Speed. Bluetooth-versiossa 3.0 esitelty tiedonsiirtotekniikka, jossa käytetään WLAN-yhteyttä nopeampaan tiedonsiirtoon.
HTTP	Hypertext Transfer Protocol. Internetprotokolla, joka on suunniteltu tiedonsiirtoa varten. Verkkoselaimet ja verkkosivut käyttävät, joko tätä tai turvallisempaa HTTPS-protokollaa.
IPS	Indoor Positioning System. Sisätilapaikannusjärjestelmä.
JSON	Java-Script Object Notation. Yksinkertainen avoimen standardin tiedostomuoto tiedonvälitykseen.
RJ45-kaapeli	Yleisin liitintyyppi parikaapelille.
RTLS	Real-Time Locating System. Reaaliaikainen paikannusjärjestelmä.
UHF	Ultra High Frequency. Korkean taajuuden radioaaltoja.
WLAN	Wireless Local Area Network. Langaton lähiverkko.

1 JOHDANTO

Sisätilapaikannusjärjestelmien kehitys alkaa olla siinä vaiheessa, että niiden avulla on mahdollista tehdä käytännön sovelluksia. Tässä työssä perehdytään yhteen sellaiseen sovellukseen.

Tämän työn tarkoituksena on tutkia Quupan valmistaman Bluetooth angle-of-arrival -kategorian sisätilapaikannusjärjestelmän käytettävyyttä kontaminoituneissa tiloissa. Tutkimuksessa halutaan myös selvittää, onko järjestelmä käyttökelpoinen mahdollisille asiakkaille. Asiakkaiksi ajatellaan käyttäjiä, joiden tiloissa on tapahtunut kontaminaatiovahinko ja tilat tulisi puhdistaa. Työhön kuuluvat sisätilapaikannusjärjestelmän asennus, tagista saatavan tiedon hyödyntäminen ja laitteistosta saatavan tiedon esittäminen visuaalisesti.

Sisätilanpaikannusjärjestelmän asennus aloitetaan asentamalla lokaattorit mitattuihin sijainteihin. Lokaattorit yhdistetään kytkimeen, joka ottaa myös yhteyden palvelinkoneeseen. Palvelinkoneella käynnistetään Quuppa Site Planner sovellus, jolla voidaan määrittää lokaattorien tarkat sijainnit sisätilassa. Lokaattorit tunnistetaan ja tarkennetaan käyttäen siihen tarkoitettua tarkennuslokaattoria. Seuraavaksi sovelluksessa lisätään tag, jonka konfiguraatio tehdään tarkennuslokaattorin avulla. Lopuksi projekti synkronoidaan pilvipalveluun, josta palvelin lataa sen käyttöönsä. Tällöin tagin sijaintia voi seurata palvelimen kautta.

Palvelimeen otetaan yhteys HTTP-kutsulla valmista rajapintaa käyttäen, minkä avulla saadaan tagin tiedot JSON muodossa. HTTP-kutsua muokkaamalla saadaan palvelimelta palautettua vain halutut tiedot. Sovellus tehdään käyttäen CSharp-kieltä, koska kontaminaatiomittarille tehty sovellus käyttää kyseistä ohjelmointikieltä. Sovelluksessa tehdään haluttu HTTP-kutsu ja eritellään palautettu JSON-data .Net-tietoluokkiin.

Toimintatilasta piirretään pohjapiirustus, jota käytetään sovelluksessa. Tämä helpottaa tagin sijainnin visualisointia toimintatilassa. Sovellus piirtää tagin sijainnin pohjapiirustuskuvaan. Pohjapiirustuskuva tallennetaan tiedostoon myöhempää tarkistusta varten, lisäksi tallennetaan kaksi lokitiedostoa tuloksista.

Luvussa 2 käydään läpi eri teknologioita, joita työssä käytettiin. Luvussa 3 kerrotaan sisätilapaikannusjärjestelmän laitteistosta ja sen asennuksesta. Elias Harri raportoi luvut 2 ja 3. Vesa Välimäki raportoi osan luvuista 2, 3 ja luvun 4, jossa käydään läpi tehty sovellus, sen toimintoja, sekä tulosten visualisointi.

2 KÄYTETYT TEKNOLOGIAT JA TEORIA

2.1 Bluetooth

Bluetooth on lyhyen matkan UHF-radiotekniikalla toteutettu tiedonsiirtoprotokolla kahden laitteen välillä. Se käyttää radiotaajuuksia 2,4–2,485 GHz. Bluetoothia hallinnoi Bluetooth Special Interest Group (SIG). Valmistajan tulee huolehtia siitä, että Bluetooth-laite täyttää SIG-standardin vaatimukset määräyksiltään ja testaustuloksiltaan. (Bluetooth 2015.)

Kun Bluetooth-laite on löydettävissä, se lähettää seuraavat laitetiedot pyydetessä: nimi, luokka, listan palveluista sekä tekniset tiedot (esimerkiksi laitteen ominaisuudet, valmistaja, käytössä oleva Bluetooth-versio). (Bluetooth 2015.)

Taulukosta 1, nähdään miten siirtonopeudet ovat kehittyneet Bluetooth-versioiden myötä. Huomioitavaa on se, että nopeudet ovat teoreettisia maksiminopeuksia, joihin ei päästä käytännön sovelluksissa. Bluetoothin versiossa 2.0 lisättiin valinnainen EDR-ominaisuus, jolla saavutetaan nopeammat siirtonopeudet. HS-nopeudet toteutetaan käyttäen WLAN-yhteyttä Bluetoothin lisäksi. (Bluetooth 2015.)

Taulukko 1. Bluetooth-versioiden nopeudet. (Bluetooth 2015.)

Versio	Siirtonopeus
1.2	1 Mbit/s
2.0 + EDR	3 Mbit/s
3.0 + HS	24 Mbit/s
4.0	24 Mbit/s

2.1.1 BLE

BLE tunnetaan myös nimellä Bluetooth Smart, joka on Nokian vuonna 2006 kehittämä uuden sukupolven Bluetooth-tekniikka. Se liitettiin Bluetooth-standardiin version 4.0 mukana vuonna 2010. (Bluetooth Low Energy 2015; Bluetooth SIG, Inc. 2015.)

BLE:n suurin hyöty verrattuna aikaisempiin Bluetooth-versioihin on sen huomattavasti pienempi virran tarve. BLE-tekniikalla varustetut laitteet pystyvät toimimaan useita kuukausia tai jopa vuosia pelkällä nappiparistolla. Tämä on saatu aikaan kehittämällä single-mode-tila, jossa laite on normaalisti lepotilassa, kunnes kontrolleri herättää sen suorittamaan halutun tehtävän. (Bluetooth Low Energy 2015; Bluetooth SIG, Inc. 2015.)

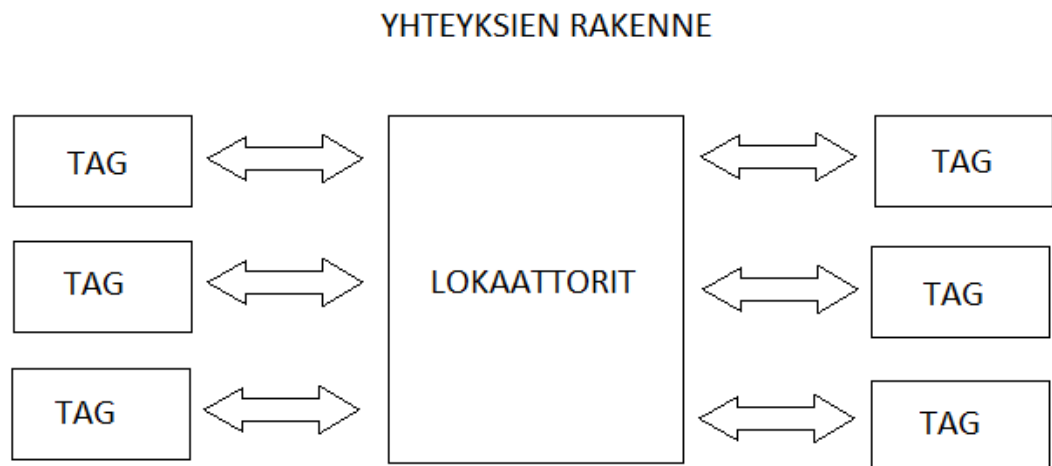
BLE toimii samalla taajuusalueella kuin aikaisemmatkin versiot eli 2,4 – 2,4835 GHz. Se eroaa kuitenkin käyttämällä 40:tä 2 MHz:n kanavaa verrattuna aikaisempaan versioon, jossa käytetään 79:ää 1 MHz:n kanavaa. Sen siirtonopeus on 1 Mbit/s, lähetysteho 10 mW ja viive lepotilasta aktiiviseksi noin 6 ms. BLE soveltuu pienen virrantarpeensa vuoksi moniin erilaisiin käyttötarkoituksiin. Näitä ovat esimerkiksi terveyteen, urheiluun, sensoreihin ja paikannukseen liittyvät sovellukset ja laitteet. (Bluetooth Low Energy 2015; Bluetooth SIG, Inc. 2015.)

Tässä työssä BLE liittyy läheisesti sisätilapaikannusjärjestelmään ja varsinkin tageihin, joiden liikettä pystytään seuraamaan. (Bluetooth Low Energy 2015; Bluetooth SIG, Inc. 2015.)

2.1.2 GATT

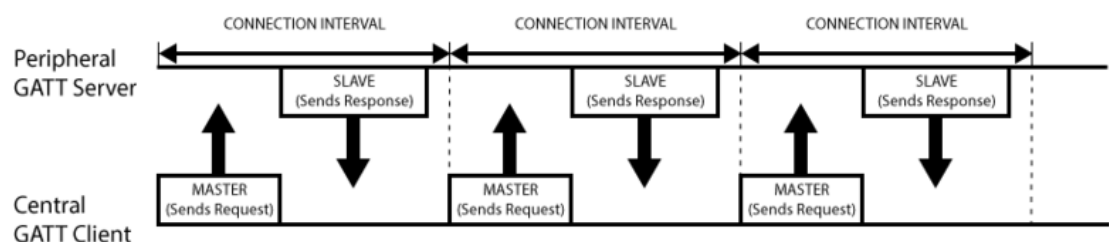
Kaikki BLE-laitteet käyttävät GATT:a (Generic Attribute Profilea). GATT määrittelee BLE-yhteyden osapuolet. BLE-yhteydet ovat eksklusiivisia eli oheislaitteet, esimerkiksi tagit, pystyvät yhdistämään vain yhteen keskuslaitteeseen samanai-

kaisesti, esimerkiksi lokaattoriin. Kun oheislaite on muodostanut yhteyden keskuslaitteeseen, se lopettaa itsensä mainostamisen, jolloin muut keskuslaitteet eivät voi enää nähdä sitä tai muodostaa yhteyttä siihen, kunnes aikaisempi yhteys on lopetettu. (Kuva 1.) (Townsend 2015.)



Kuva 1. BLE-yhteyden toimintaperiaate.

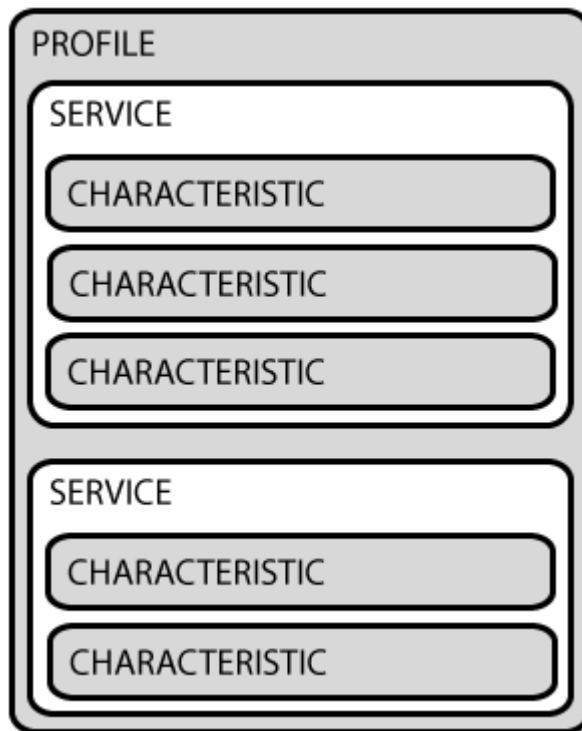
Tagit toimivat palvelimena, koska ne lähettävät tietonsa lokaattoreille. Toimeksiantaja eli lokaattori aloittaa kaikki tiedonsiirrot. Tag määrittää yhteysaikavälin ajan eli sen, kuinka usein lokaattori kysyy tagilta uusia tietoja. Tämä aikaväli ei välttämättä toteudu, jos lokaattori kyselee samanaikaisesti muilta tageilta tietoja. Tässä työssä se ei ole ongelma, koska käytössä on vain yksi tag. (Kuva 2.)



Kuva 2. Tiedonhaku tagista. (Townsend 2015.)

BLE-oheislaitteiden palvelut (services) on tapana eritellä tietologisiin kokonaisuuksiin. Palvelut sisältävät tiettyjä tietolohkoja, joita kutsutaan erityispiirteiksi

(characteristics). Palvelulla voi olla yksi tai useampia erityispiirteitä ja jokaisella palvelulla on oma tunnisteensa. Tämä tunniste on 16-bittinen SIG:n määrittelemille palveluille tai 128-bittinen mukautetuille palveluille. (Kuva 3.) (Townsend 2015.)



Kuva 3. GATT-profiili. (Townsend 2015.)

Erityispiirteet ovat alimman tason GATT-tapahtumia, joissa on yksi datapiste. Datapisteessä voi kuitenkin olla useampia arvoja esimerkiksi x-, y- ja z-koordinaatit. Erityispiirteillä on myös oma tunniste, kuten palveluillakin. Se voi olla 16- tai 128-bittinen käyttötarpeen mukaan. (Townsend 2015.)

2.2 Sisätilapaikannus

IPS eli sisätilapaikannusjärjestelmä on ratkaisu, jolla voidaan paikantaa ihmisiä ja esineitä sisätiloissa. Sisätilapaikannukseen ei vielä ole olemassa standardeja, mutta siitä huolimatta saatavilla on useita kaupallisia ratkaisuja, esimerkiksi Quupan valmistamat tuotteet. (Indoor positioning system 2015.)

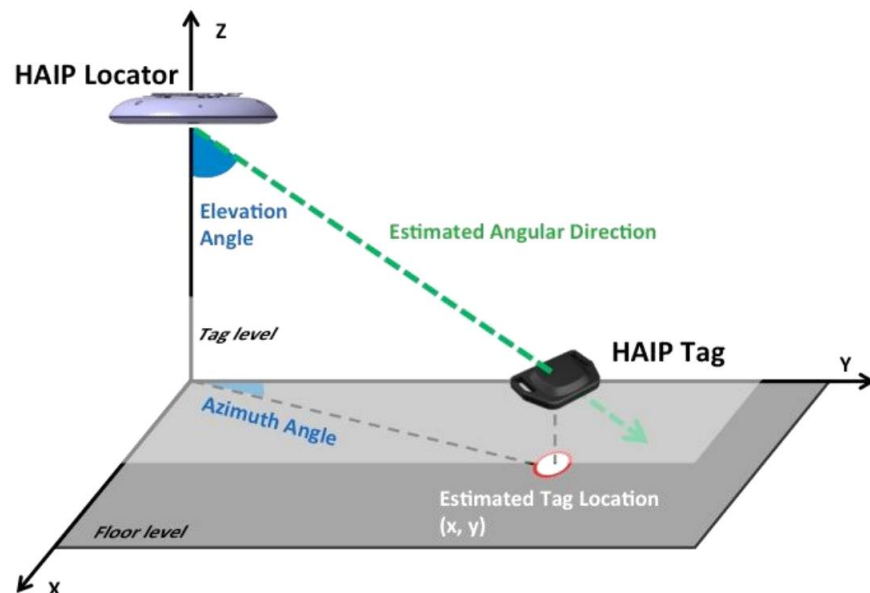
Satelliittien hyödyntämisen sijaan, sisätilapaikannussysteemit käyttävät erilaisia ratkaisuja kuten Wifi-paikannus ja BLE-majakka, joissa paikannus tapahtuu laskemalla laitteen sijainti sen lähettämän signaalin voimakkuuden perusteella. (Indoor positioning system 2015.)

2.2.1 HAIP

HAIP eli korkean tarkkuuden sisätilapaikannus on BLE:hen perustuva sisätilapaikannusratkaisu. Se eroaa aikaisemmista ratkaisuista siinä, että lokaattori mittaa saapuvan signaalin tulokulman, joka parantaa paikannustarkkuutta huomattavasti. HAIP:lla päästään noin 0,1–1,0 metrin tarkkuuteen. HAIP ei ole vielä osa virallista Bluetooth-standardia, mutta sen pitäisi tulla seuraavaan versioon (Bluetooth SiG 2014, 13). (Quuppa 2013, 3.)

Positioning principle

One Locator offers accurate 2D position



Kuva 4. HAIP:n toimintaperiaate. (Quuppa 2013, 5.)

HAIP-tekniikalla on useampia etuja verrattuna aikaisempiin sisätilapaikannusteknologioihin. Taulukosta 2 nähdään vertailua eri teknologioista.

Taulukko 2. Sisätilapaikannusteknologioiden vertailu. (Quuppa 2013, 7.)

	WiFi	Indoor GPS	Bluetooth	Bluetooth 4.0	HAIP
Technology	Triangulation radio fingerprinting RSSI (signal strength)	Triangulation/ time differential	Proximity RSSI	Proximity RSSI / Triangulation	Angle of arrival (departure)
Developed for positioning	No	Yes	No	Partially	Yes
Power consumption	High	High	High	Medium - Low	Low
Accuracy	5m-20m	5m	5m-20m	5m-20m	0.5m-1m **
Use cases					
- Mobile centric positioning	Good	Good	Average	Good	Excellent
- Tracking mode	Poor (power consumption)	Poor (needs com channel)	Poor	Poor	Excellent (server mode)
Functionality level	Map and Route	Map, Route, Navigation	Map, Route	Map, Route	Map, Route Navigation, tracking
Deployment					
Infrastructure	existing WiFi infra needs upgrade	To be deployed	To be deployed	To be deployed	To be deployed
Penetration in smartphones	Good	Not available in Phones and no roadmap/support	Available	Started	Pending standardization
Cost of ownership					
- Mobile centric positioning	Medium (infra upgrade needed)	High	High	Medium	Medium (infra needed)
- Tracking (tags)	High	High	High	Low	Low

2.2.2 RTLS

Reaaliaikaisella paikannusjärjestelmällä tarkoitetaan paikannusjärjestelmää, jolla pystytään reaaliaikaisesti paikantamaan esineitä tai ihmisiä rajatulla alueella. RTLS:n sisältyy radioaaltoja hyödyntäviä paikannusteknologioita, kuten HAIP. Lisäksi siihen sisältyy optiikkaan ja akustiikkaan perustuvia teknologioita. (Real-Time Locating System 2015.)

RTLS:n toiminta perustuu tagien paikannukseen majakoiden avulla. Majakat vastaanottavat tageilta signaalin, jonka perusteella tagin paikannus tapahtuu. (Real-Time Locating System 2015.)

2.3 JSON

JSON on suosittu tiedon välitysmuoto, jota ohjelmoijat käyttävät siirtääkseen tietoa palvelimen ja ohjelman välillä. JSON muodostuu, kun JavaScript-muuttujat ja niiden arvot yhdistetään yhdeksi luettavaksi tiedostoksi. (Severance 2012, 6.)

JSON on ohjelmointikielestä riippumaton, ja sen tietoja pystytään lukemaan millä tahansa ohjelmointikielellä. Lukua varten on kehitetty valmiit parserit eri ohjelmointikielille, mikä helpottaa JSON-tiedon käyttöä. JSON:ssa voidaan käyttää numeroita, stringejä, boolean-arvoja, listoja ja null-arvoa. (JSON 2015.)

Liitteessä 1 esitellään dataa JSON-muodossa. Data koostuu tiedoista, joita tagista ja sen sijainnista saadaan palautettua (Quuppa 2015c, 4).

Tässä työssä sen avulla saadaan monenlaista tietoa tageista. Näistä tiedoista tarpeellisimmat ovat aikaleima ja tagin x- ja y-koordinaatit. Tagin tiedot saadaan valmista rajapintaa käyttäen HTTP-kyselyllä palvelimelta.

2.4 Säteily

Säteily on joko ionisoimatonta sähkömagneettista aaltoliikettä tai ionisoivaa hiukkassäteilyä. Ionisoiva säteily on säteilyä, jolla on riittävästi energiaa irrottamaan säteilyn kohteeksi joutuvan aineen atomeista elektroneja tai rikkomaan aineen molekyylejä. Työssä on käytetty palovaroitinta, jonka säteily luokitellaan ionisoivaksi säteilyksi. Palovaroittimessa oleva Amerikium-241-säteilylähde säteilee alfa-säteilyä ja sen puoliintumisaika on 432,2 vuotta. (STUK 2014.; Amerikium 2015.)

Alfa- ja beetasäteily ovat hiukkassäteilyä. Atomin ytimeistä lähtee suurella nopeudella alfa- tai beetahiukkanen. Alfahiukkanen muodostuu kahdesta protonista ja kahdesta neutronista. (STUK 2014.)

Alfahiukkanen ei pysty läpäisemään ihmisen ihoa tai paperiarkkia. Alfa-säteily voi olla vaarallista vain, jos alfasäteilyä lähettäviä radioaktiivisia aineita joutuu elimistöön esimerkiksi hengitysilman mukana. Beetahiukkaset ovat läpäisykykyisempiä ja pystyvät tunkeutumaan esimerkiksi ihoon. Beetasäteilyä lähettävät aineet ovat vaarallisia iholla tai päästessään elimistöön. (STUK 2014.)

Radioaktiivisen aineen aktiivisuus kertoo kuinka monta ydinmuutosta siinä tapahtuu sekunnin aikana. Aktiivisuuden yksikkö on becquerel. Yksi becquerel tarkoittaa

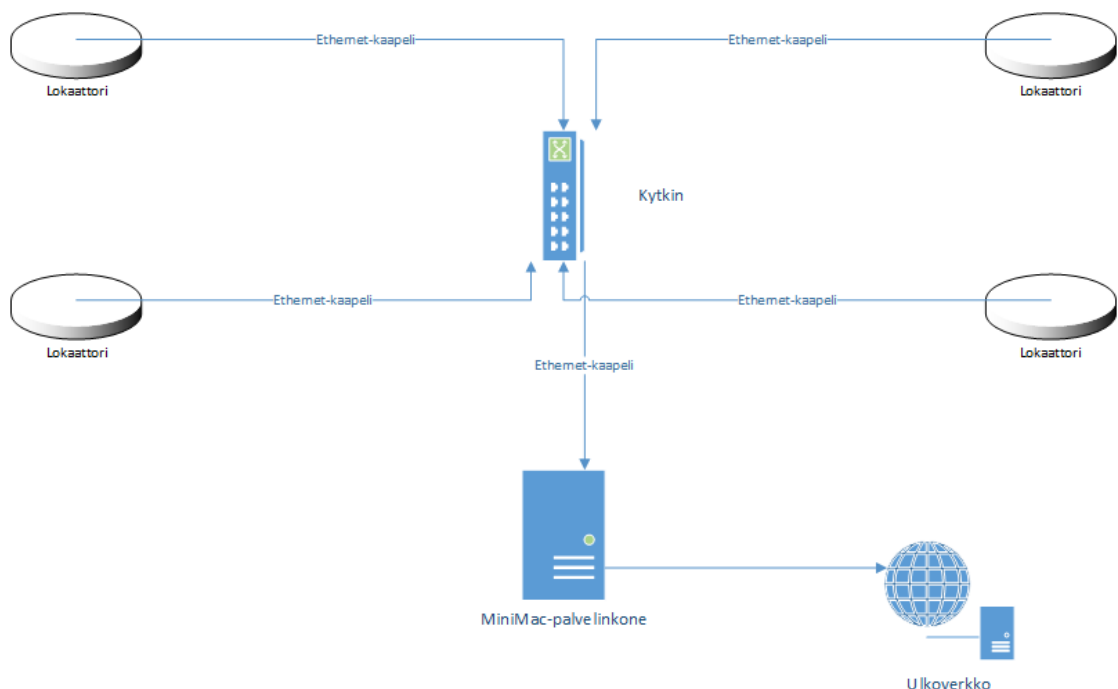
taa, että radioaktiivisessa aineessa tapahtuu yksi ydinmuutos sekunnissa. Puoliintumisaika kertoo ajan, jossa radioaktiivisen aineen aktiivisuus puolittuu. (STUK 2014.)

3 LAITTEET JA ASENNUS

Laitteistona on kontaminaatiomittari, sisätilapaikannuslaitteisto, kannettava tietokone sekä säteilylähde. Laitteiden asennus aloitettiin sisätilapaikannusjärjestelmällä, koska sen piti olla toimintakunnossa ennen kuin tagia pystyttiin seuraamaan.

3.1 Sisätilapaikannusjärjestelmä

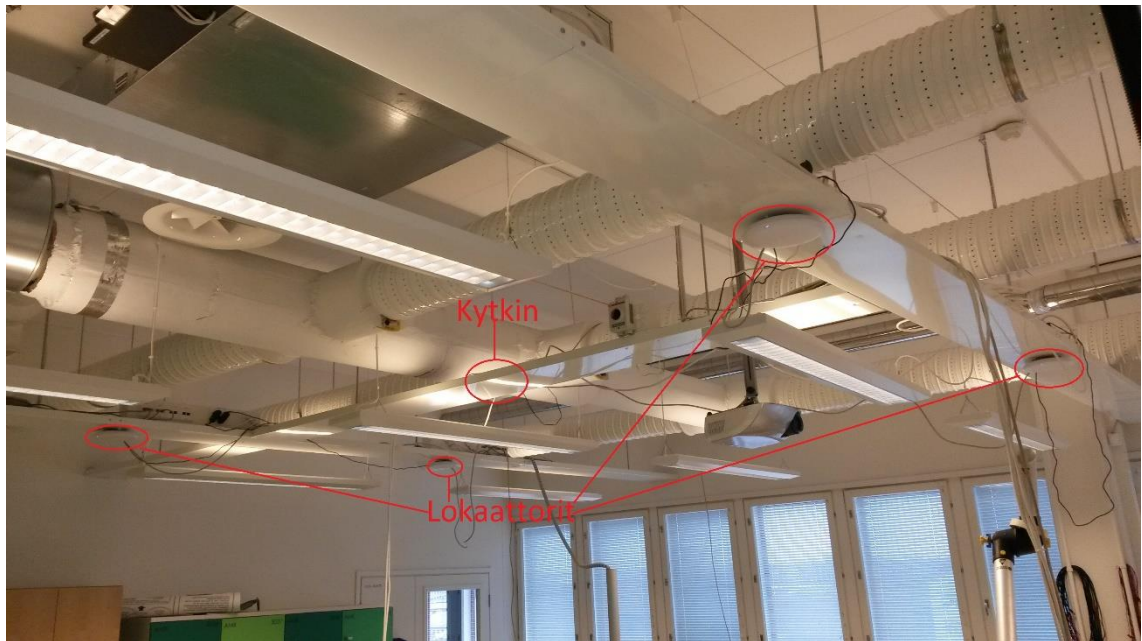
Työssä käytetty sisätilapaikannusjärjestelmä on suomalaisen Quupan kehittämä. Käytössä on Quuppa Demokit, johon kuuluvat palvelimena toimiva Apple mini-Mac-tietokone, neljä lokaattoria, yksi tarkennuslokaattori, yksi seurattava tag ja Zyxelin valmistama verkkokytkin. MiniMac-tietokone on kiinni koulun DC-verkossa sekä kytkimessä. Jokaisella lokaattorilla on oma virtajohtonsa, ja lokaattorit ovat kiinni kytkimessä RJ-45-kaapeleilla. Lokaattorit seuraavat tagin sijaintia Bluetooth Smartin avulla. (Kuva 4.)



Kuva 5. Sisätilapaikannuslaitteiston kytkentä.

Quupan kehittämä sisätilapaikannusjärjestelmä pohjautuu HAIP-teknologiaan, jonka kehittivät Quupan perustajat. Sisätilapaikannusjärjestelmän tarkkuudeksi annetaan 0,1–1 m, myös nopeilla nopeuksilla. (Quuppa 2015a.) Quupan järjestelmillä pystytään seuramaan rajatonta määrää tageja ja muita BLE-laitteita (Quuppa 2015b).

Huoneesta, johon järjestelmä asennettiin, ei ollut saatavilla kunnollista pohjapiirustusta, josta olisi nähnyt huoneen mitat. Ongelman ratkaiseminen aloitettiin piirtämällä jäljennös pohjapiirustuksesta, jolla saatiin seinien oikeat mittasuhteet. Tämän jälkeen mitattiin huoneen yhden seinän pituus. Mitatun seinän avulla QuuppaSitePlanner-ohjelma osasi laskea huoneen pinta-alan. Lokaattorit asennettiin, katossa oleviin metallilevyihin, magneettien avulla (Kuva 5).



Kuva 6. Sisätilapaikannusjärjestelmän asennus toimintatilassa.

Toimintatilassa sisätilapaikannusjärjestelmällä ei saatu hyväksyttävän tarkkuuden paikannustietoja. Kyseisessä tilassa onnistutaan demonstroimaan laitteiston käyttöä, mutta tuntemattomien syiden takia tagin antamat koordinaatit saattoivat heitellä melko paljon. Syiden selvittämiseksi laitteisto siirrettiin viereiselle käytävälle, jossa tarkkuusongelmat jatkuivat. Tämän jälkeen laitteisto siirrettiin takaisin alkuperäiseen tilaan, mutta lokaattorit asennettiin hieman eri kohtiin. Tämä auttoi

ja tagin antamat koordinaatit olivat tarkempia ja aikaisemman kokeilun kaltaisia heittoja ei enää tullut. Laitteiston parhaan toimivuuden takaamiseksi vaaditaan toimintatila, jossa lokaattorien ja tagin väliseen liikenteeseen vaikuttavat häiriötekijät ovat minimoitu. Tutkimuksessa käytetyssä tilassa mahdollisina häiriötekijöinä olivat esimerkiksi metallipinnat, sähkölaitteet ja langattomat verkot.

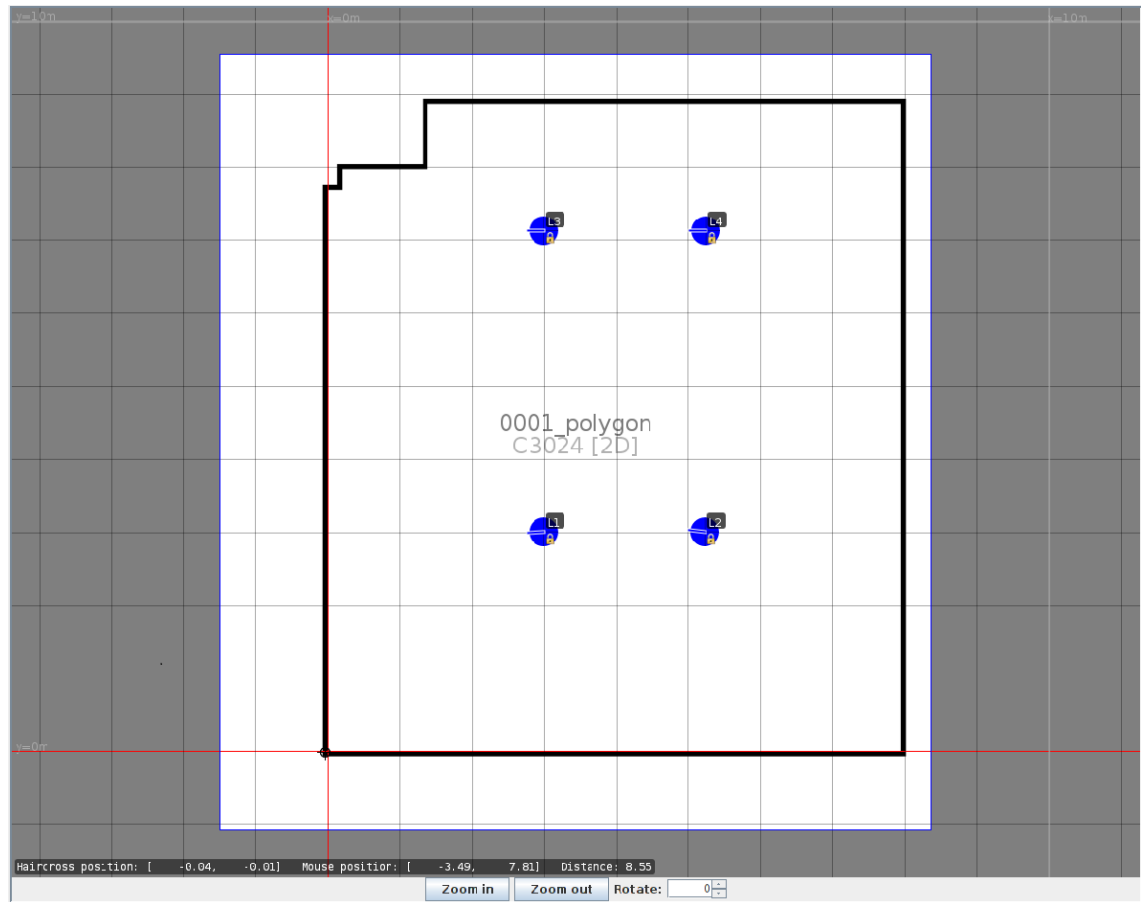
Aluksi tarkennuslokaattoria ei saatu toimimaan ollenkaan, mutta tämä johtui siitä, että tarkennuslokaattori oli rikkoutunut ennen työn aloittamista. Quupalta saatiin uuden tarkennuslokaattorin, jonka jälkeen kaikki toimi moitteettomasti.

3.1.1 Palvelin

Palvelimena toimii MiniMac-tietokone, johon on asennettu Ubuntu-käyttöjärjestelmä. Tietokone yhdistettiin ulkoverkkoon, jotta palvelin on käytettävissä omalla kannettavalla tietokoneella.

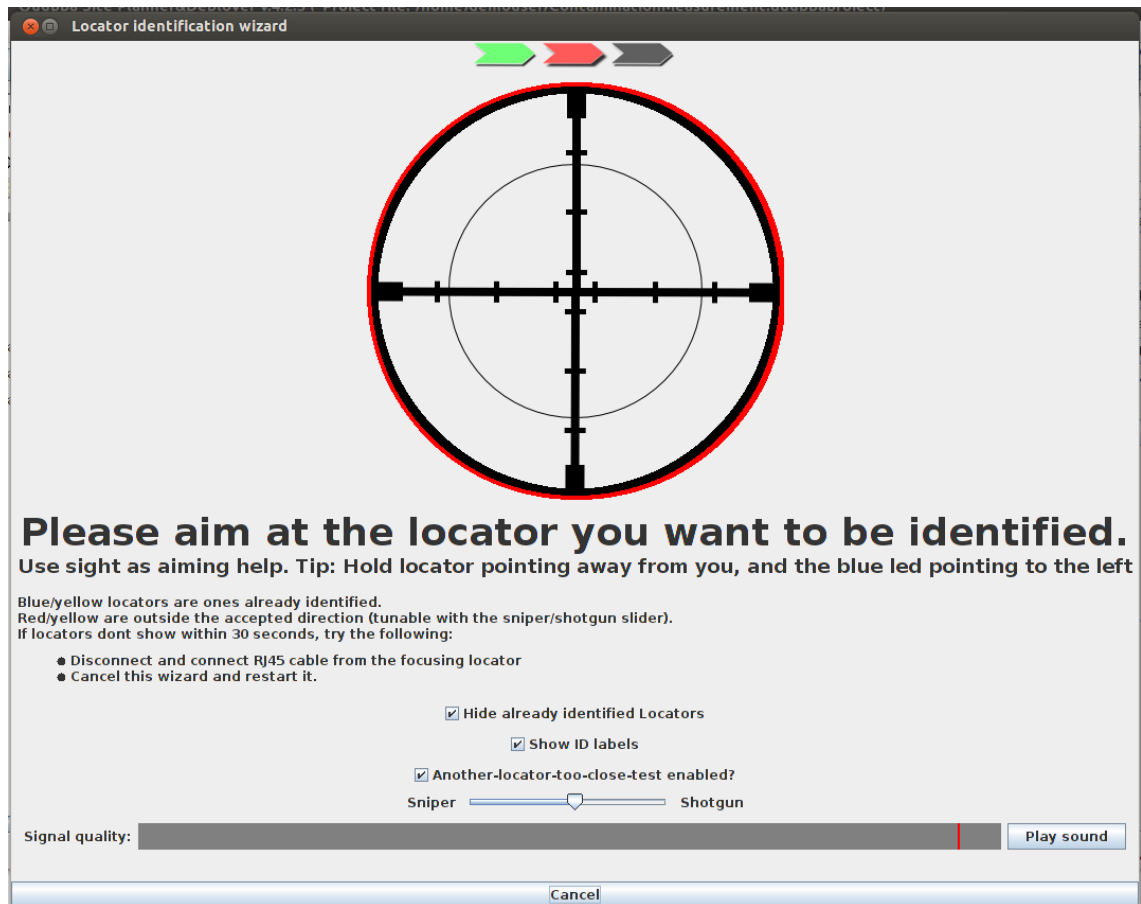
3.1.2 Quuppa Site Planner -ohjelma

Quuppa Site Planner on ohjelma, jota käytetään lokaattorien tunnistamiseen ja kalibrointiin sekä lokaattorien asettamiseen oikeille paikoilleen pohjapiirustukseen. Asennettujen lokaattorien etäisyydet seinistä mitattiin, jotta ne saatiin asetettua oikeille kohdilleen ohjelmassa (Kuva 6).



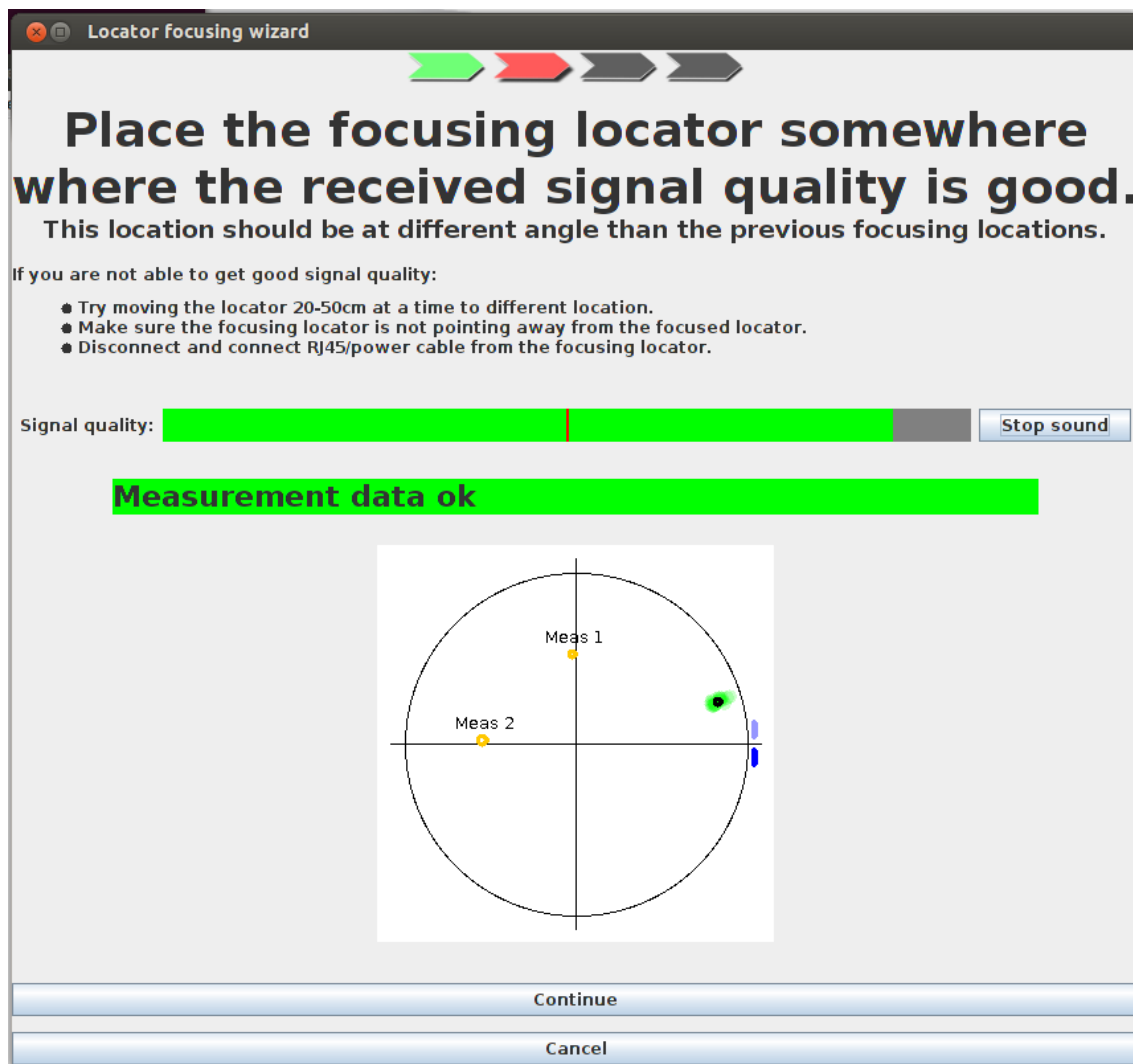
Kuva 7. Lokaattorien sijainti pohjapiirustuksessa.

Seuraavaksi lokaattorit tunnistettiin tarkennuslokaattorilla, jolla varmistettiin että oikea lokaattori on oikeassa paikassa pohjapiirustuksessa (Kuva 7).



Kuva 8. Lokaattorin tunnistusikkuna.

Tämän jälkeen lokaattorit kalibroitiin yksi kerrallaan tarkennuslokaattorilla ottamalla lukemat kahdesta eri suunnasta, tällä päästiin noin 1°:n tarkkuuteen (Kuva 8).



Kuva 9. Lokaattorin kalibrointi-ikkuna.

Kun lokaattorit oli kalibroitu, valmis projekti lähetettiin Quupan pilvipalvelimelle hyväksyttäväksi. Tämän jälkeen Applen MiniMacissa toimivalle palvelimelle laddattiin hyväksytty projekti pilvipalvelimelta. Lopuksi palvelin asetettiin seurantaan, joka mahdollistaa tagien seuraamisen.

3.1.3 Tag

Tag on pieni paristokäyttöinen laite. Sisätilapaikannusjärjestelmään kuuluu useampia tageja, mutta tässä työssä tarvitaan vain yhtä. Seurattava tag pitää lisätä Quuppa Site Planneriin, jonka jälkeen sen tietoja on mahdollista saada HTTP-kyselyllä. Tag aktivoidaan painamalla siinä olevaa pientä nappia. Tag kommunikoi lokaattorien kanssa käyttäen BLE-yhteyttä. Jokaisella tagilla on oma ID, jota käytetään tagien tunnistamiseen Quuppa Site Plannerissa. (Kuva 9.)



Kuva 10. Kuva tagista.

3.2 Kontaminaatiomittari ja säteilylähde

Kontaminaatiomittari yhdistetään USB-kaapelilla tietokoneeseen. Kontaminaatiomittarin sijainnin seuraamiseksi siihen kiinnitettiin teipillä sisätilapaikannusjärjestelmään kuuluva tag. Säteilylähteenä käytetään ionisoivasta palohälyttimestä saatua amerikiium 241-lähdettä. Palohälytin valittiin säteilylähteeksi, koska amerikiiumin lähettämä alfa-säteily ei ole vaarallista. Jopa alfa-säteilyn radioaktiiviset aineet ovat kuitenkin vaarallisia, jos niitä joutuu kehon sisään.

3.3 Järjestelmän käyttömahdollisuudet

Käytännön esimerkkinä työn järjestelmän käyttökohteesta voisi ottaa Pariisin, jossa on 1900-luvun alkupuolella käytetty radioaktiivista radiumia useissa eri käyttötarkoituksissa. Pariisilaiset käyttivät radiumia esimerkiksi huulipunissa ja suihkulähteissä. Lisäksi radiumia käytettiin maalien valmistuksessa, tällä saatiin hohtavia maaleja. (Rose & Douet 2012.)

Pariisilaisista taloista löytyy vieläkin jäänteitä radium-maaleista ja Ranskan valtio on budjetoanut ANDRA:lle (Ranskan radioaktiivijätteen kansalaisjärjestö) 4 miljoonaa euroa vuosittain talojen puhdistukseen. Taloista mitataan säteilyarvoja ja niistä riippuen ANDRA ostaa ja tuhoaa talon, tai puhdistaa talon. (Rose & Douet 2012.)

Työn lopputulokseksi syntynyttä järjestelmää voisi hyödyntää Pariisissa olevissa kohteissa. Järjestelmällä saataisiin helposti selville säteilylähteen sijainti, jolloin talojen puhdistus olisi kätevää ja kustannustehokasta.

4 SOVELLUS

Sovellus toteutettiin Microsoftin Visual Studio 2013 Ultimate -versiolla. Ohjelmointikielenä käytettiin CSharp-kieltä. Sovelluksen pohjana on kontaminaatiomittarille aikaisemmin tehty ohjelma. Työssä päätettiin käyttää aikaisempaa ohjelmaa, koska se suorittaa kontaminaatiomittarilta tarvittavat operaatiot, jotka uuden ohjelman olisi tullut suorittaa. Alkuperäinen ohjelma mittaa alfa- ja beetasäteilyä ja kirjoittaa saatuja arvoja aikaleiman kanssa tekstitiedostoon. Sovelluksen tarkoituksena on tutkia tagista saatua tietoa ja visualisoida se käyttäjälle sopivaan muotoon.

Kontaminaatiomittaria varten luotu sovellus oli kehitetty .Net Framework 3.5 -ympäristössä. Sovellusta kehittäessä vanha ohjelma päivitettiin .Net Framework 4.5 -ympäristöön, jotta tiettyjä toimintaan vaikuttavia ominaisuuksia, kuten Tuple-luokkaa, voitiin käyttää (Microsoft 2015a).

Sovelluksen toiminnasta tehtiin Microsoft Visio -ohjelmalla UML-kaaviot. UML-kaavioista nähdään sovelluksen tekevät toiminnot ja se, miten sovellusta tulisi käyttää. (Liite 2.)

4.1 Ohjelmointi

Ohjelmointi aloitettiin muodostamalla tagin JSON-datasta tarvittavat .Net-luokat, joihin data voitaisiin eritellä. Tämä tehtiin kopioimalla JSON-data leikepöydälle ja

liittämällä se Visual Studiossa projektiin Paste Special -toiminnolla. Visual Studio luo liitetystä koodista tarvittavat luokat (Koodi 1.) (Paranjape 2012.)

Koodi 1. .Net-luokat.

```
//Luodaan luokat joiden objekteihin JSON data saadaan deserialisoitua
1 reference
public class Tag
{
    1 reference
    public List<double> position { get; set; }
    0 references
    public string id { get; set; }
    0 references
    public List<double> covarianceMatrix { get; set; }
    0 references
    public string areaName { get; set; }
    0 references
    public string color { get; set; }
    0 references
    public List<double> smoothedPosition { get; set; }
    0 references
    public string name { get; set; }
    0 references
    public object coordinateSystemName { get; set; }
    0 references
    public List<object> zones { get; set; }
    1 reference
    public long positionTS { get; set; }
    0 references
    public string coordinateSystemId { get; set; }
    0 references
    public string areaId { get; set; }
    1 reference
    public double positionAccuracy { get; set; }
}
2 references
public class RootObject
{
    1 reference
    public List<Tag> tags { get; set; }
    0 references
    public long responseTS { get; set; }
    0 references
    public string version { get; set; }
}
```

Tämän jälkeen hyödynnettiin Newtonsoftin JSON.Net-lisäosaa, joka asennettiin Visual Studion omaa manage NuGet package -toiminnolla. JSON.Net helpottaa JSON-datan lataamista ja erittelyä. (Mikhail-T 2014.) Seuraavaksi luotiin Tuple-luokan Contaminationmeter-niminen metodi, joka JSON.Net-lisäosan avulla lataa

JSON-dataa palvelimelta ja erittelee sen .Net-luokan objekteihin (Dimitrov 2014) (Koodi 2).

Koodi 2. JSON-datan erittelymetodi.

```
1 reference
public Tuple<long,double,int[]> Contaminationmeter()
{
    //Ladataan verkkosivulta haluttu JSON data string muodossa
    string jsonString = downloader.DownloadString(serviceUri);
    //Kirjoitetaan logitiedosto jota voidaan myöhemmin tutkia
    if (writerOpen == true)
    {
        writer.WriteLine(jsonString);
    }
    //Deserialisoidaan JSON data .Net objekteihin
    RootObject xy = JsonConvert.DeserializeObject<RootObject>(jsonString);
    foreach (var item in xy.tags)
    {
        double[] position = item.position.ToArray();
        postTS = item.positionTS;
        posAcc = item.positionAccuracy;
        double[] _position = new double[position.Length];
        for (int i = 0; i < position.Length; i++)
        {
            _position[i] = position[i] * 100;
        }
        intArray = new int[_position.Length];
        for (int i = 0; i < intArray.Length; ++i)
        {
            intArray[i] = (int)_position[i];
        }
        //Kirjoitetaan logitiedosto vain halutuilla tiedoilla
        using (StreamWriter writer_ = new StreamWriter("TagData.txt", true))
        {
            writer_.WriteLine("Tag position: " + position[0] + ", " + position[1] + " (m)\tPosition Timestamp: "
                + postTS + " (epoch time)\tPosition Accuracy: " + posAcc + " (m)" + "Alpha Radiation level: " + beta.Text);
            writer_.Close();
        }
    }
    //Palautetaan halutut tiedot muuttujiin
    return new Tuple<long,double,int[]>(postTS, posAcc, intArray);
}
```

Contaminationmeter-metodi palauttaa paikkasijainnin arvot int-taulukkoon. Paikkasijainnin arvot ladataan luotuun Filter-nimiseen metodiin, jossa ne suodatetaan

halutulla tavalla ja lisätään muuttujiin x3 ja y3. Suodatus toteutuu lisäämällä paikkasijainti arvoja Queue-luokan instanssiin, josta valitaan kolme keskimmäistä arvoa ja lasketaan niiden keskiarvo (Microsoft 2015b). (Koodi 3.)

Koodi 3. Paikkasijainnin suodatusmetodi

```
1 reference
private void Filter()
{
    //Suodatetaan paikkatietoja halutulla tavalla
    xCord.Enqueue(intArray[0]);
    yCord.Enqueue(intArray[1]);
    if (xCord.Count == 9 && yCord.Count == 9)
    {
        int[] x = xCord.ToArray();
        int[] y = yCord.ToArray();
        Array.Sort(x);
        Array.Sort(y);
        int[] x2 = new int[3];
        int[] y2 = new int[3];
        for (int i = 3; i < 6; i++)
        {
            x2[i - 3] = x[i];
            y2[i - 3] = y[i];
        }
        x3 = (x2.Sum() / x2.Length);
        y3 = (y2.Sum() / y2.Length);
        xCord.Dequeue();
        yCord.Dequeue();
        testi = true;
    }
}
```

Lisäksi luodaan uusi formi nimeltä tagfollow, jossa visualisointi tapahtuu. Tagfollow-formiin asetetaan taustakuva, jossa paikkasijaintia halutaan seurata. Sovelluksessa on luotu backgroundworker-luokan instanssi, jossa kutsutaan aikaisempia metodeja, Contaminationmeter ja Filter. Lisäksi backgroundworker tekee

piirto-operaation, joka hyödyntää paikkasijaintimuuttujia (Microsoft 2015c). Paikatiedot ladataan Point-muuttujaan, joita piirtotoiminto hyödyntää. (Koodi 4.)

Koodi 4. Piirtotoiminto.

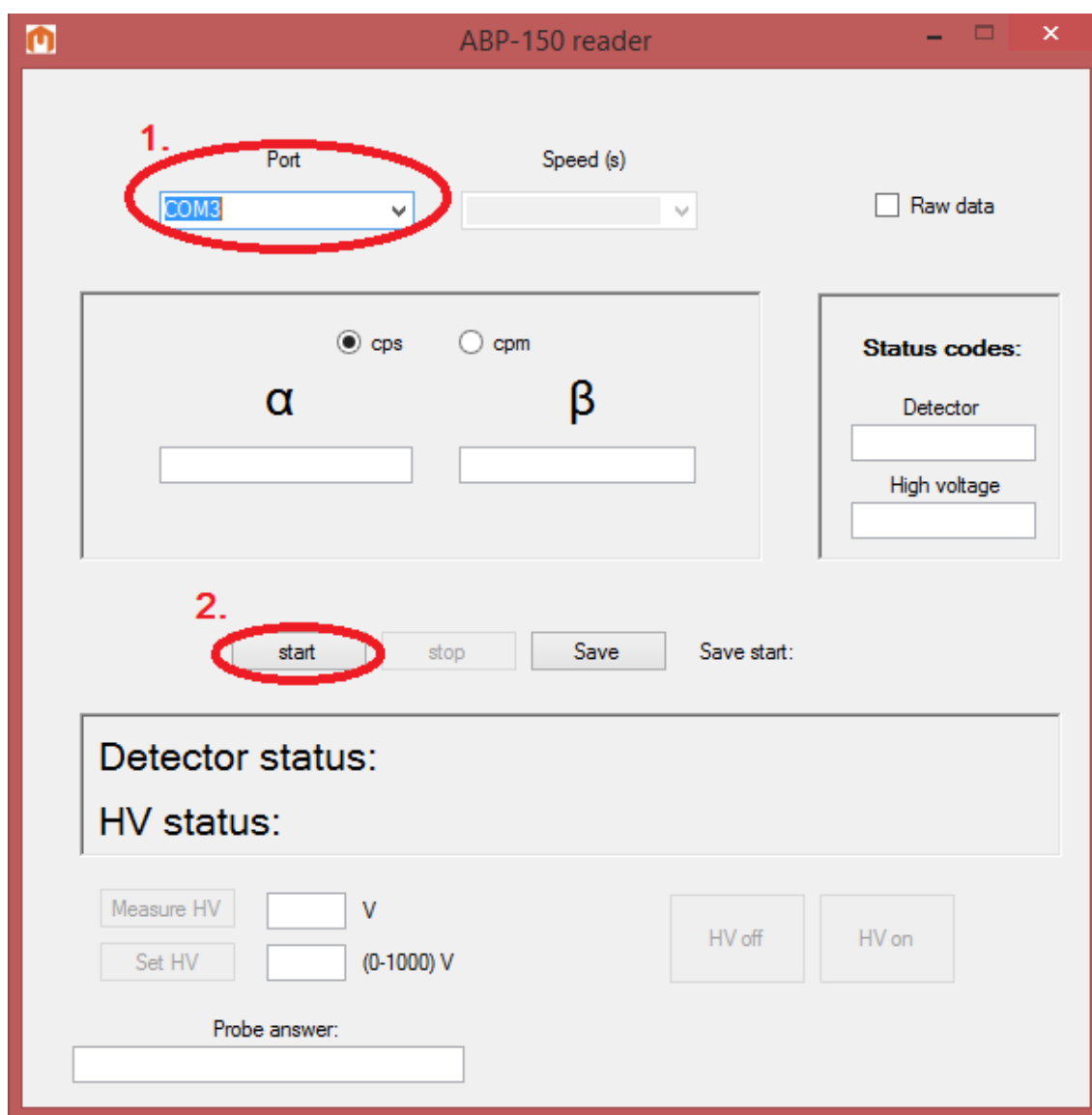
```
//Kutsutaan luotuja metodeja
Contaminationmeter();
Filter();
alfaActually = BitConverter.ToSingle(floatValsBeta, 0);
Image bg = background.Clone() as Image;
//Testataan onko kordinaatti sijainti suodatettu
if (testi == true)
{
    Point test = new Point(x3 - meterlogo.Width / 2, (bg.Height - y3) - meterlogo.Height / 2);
    Point test_ = new Point(x3 - taglogo.Width / 2, (bg.Height - y3) - taglogo.Height / 2);
    Rectangle rekt = new Rectangle(test, taglogo.Size);
    //Piirretään halutut tiedot tagfollow formin taustakuvaan
    using (Graphics g = Graphics.FromImage(bg))
    {
        GetRekt(alfaActually, rekt, g);
        g.DrawImage(meterlogo, test);
        tagfollow.BackgroundImage = bg.Clone() as Image;
        background = bg as Bitmap;
        tagfollow.Invalidate();
    }
}
//Ei-suodatettu sijainti piirretään ennen suodatusta
else
{
    Point test = new Point((intArray[0]) - meterlogo.Width / 2, (bg.Height - intArray[1]) - meterlogo.Height / 2);
    Point test_ = new Point(intArray[0] - taglogo.Width / 2, (bg.Height - intArray[1]) - taglogo.Height / 2);
    Rectangle rekt = new Rectangle(test, taglogo.Size);
    using (Graphics g = Graphics.FromImage(bg))
    {
        GetRekt(alfaActually, rekt, g);
        g.DrawImage(meterlogo, test);
        tagfollow.BackgroundImage = bg.Clone() as Image;
        background = bg as Bitmap;
        tagfollow.Invalidate();
    }
}
```

Operaatiot toteutetaan 20 ms:n välein, jonka määrää Timer-objekti. Timer-objekti luodaan projektin alussa muiden tarvittavien muuttujien ja objektien ohessa.

4.2 Sovelluksen käyttö

Sovelluksen käyttö aloitetaan käynnistämällä se, minkä jälkeen ikkuna avautuu näytölle. Sovellusikkunasta tulee valita COM-portti, jossa kontaminaatiomittari on kiinni. Sovellus tarkistaa mittarin ajurista portin ja ehdottaa vain kyseistä porttia. Testausvaiheessa portti oli COM3. Seuraavaksi tulee painaa sovellusikkunan

start-painiketta, tämän jälkeen sovellus aloittaa toimintansa (Kuva 10). Sovellus avaa toisen ikkunan, jossa visualisointi tapahtuu.

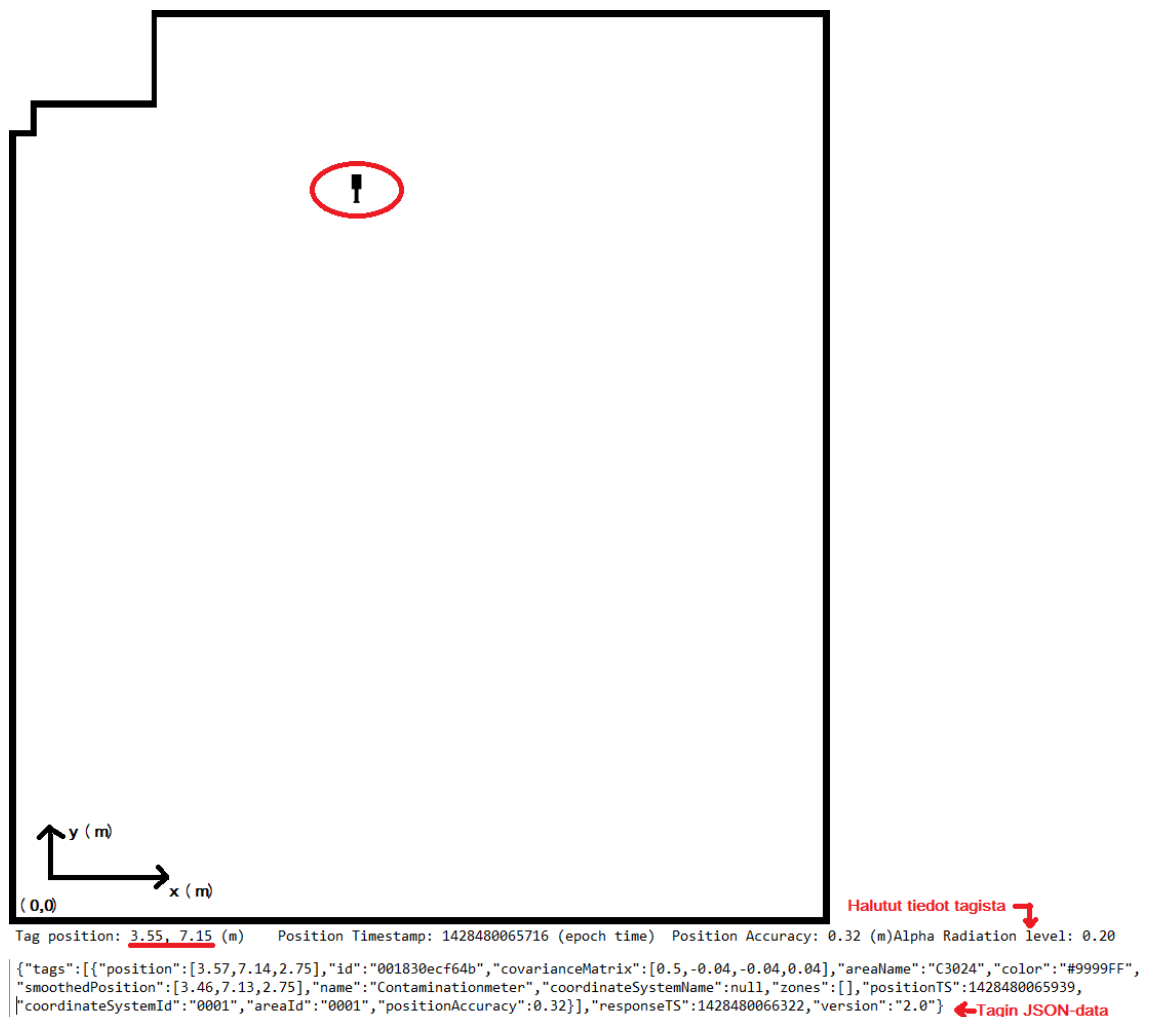


Kuva 11. Sovellusikkuna.

4.3 Tulosten visualisointi ja dokumentointi

Sovelluksesta saadaan ulostulona kuva, jossa näkyy tagin kulkema reitti ja säteilyn voimakkuus eri värein voimakkuudesta riippuen. Lisäksi ulostulona saadaan

kaksi lokitiedostoa, joista toinen kirjoittaa JSON-datan tekstinä ja toinen kirjoittaa halutut muuttujat käyttäjäystävälliseen muotoon. (Kuva 11.)



Kuva 12. Ulostulot yhdistettynä.

Kuvassa 10 nähdään tagin ja mittarin sijainti huoneessa sekä säteilyn määrä kyseisessä kohdassa. Säteilyn määrän ollessa erittäin pieni, kuten esimerkin alle 10 partikkelia sekunnissa, piirto-operaatio ei värjää sijaintia.

Laitevalmistajan sivuilla järjestelmälle oli ilmoitettu viiveeksi noin 100 ms:a (Quuppa 2015). Sovellusta käytettäessä kuitenkin huomattiin noin 500–1000 ms:n viive kuvan päivittymisen ja mittarin sijainnin välissä. Kyseinen viive esiintyi jo palvelinkoneella, mistä pääteltiin sen syntyvän laitteiston ja palvelimen välisessä liikenteessä.

5 YHTEENVETO

Opinnäytetyössä tutkittiin Quupan sisätilapaikannusjärjestelmän soveltuvuutta tilassa, jossa on tapahtunut radioaktiivinen kontaminaatio. Työ osoitti, että järjestelmän tarkkuus oli valmistajan esittämien arvojen mukainen, tämä ei kuitenkaan riittänyt. Myös liiallinen viive haittaa käytettävyyttä, jolloin järjestelmä ei sovellu nopeutta vaativiin käytännön tilanteisiin. Lisäksi laitteiston asennus uuteen tilaan vie jonkin verran aikaa ja konfigurointi vaatii ammattitaitoa, jota potentiaalisilla asiakkailta ei välttämättä ole. Paras vaihtoehto olisi, että järjestelmä asennettaisiin pysyvästi tarvittaviin tiloihin.

Työlle asetetut tavoitteet saavutettiin, ja pienten alkuvaikeuksien jälkeen laitteisto toimi halutulla tavalla. Tutkimuksen aikana mahdollisia häiriötekijöitä olivat tilassa olevat metallipinnat, langattomat verkot ja matkapuhelinverkot. Häiriöiden aiheuttajasta ei kuitenkaan saatu varmuutta.

Järjestelmä toimisi paremmin työn aiheutta ajatellen, jos lokaattorit olisivat helpompia asentaa, esimerkiksi ne olisivat liikuteltavia lattialla olevia tolppia. Toinen helpottava asia olisi, jos lokaattorit osaisivat automaattisesti mitata tilan, jossa ne ovat.

Työtä voisi jatkokehittää yrittämällä pienentää laitteiston välistä viivettä, ja kokeilemalla järjestelmän toimivuutta helpommin asennettavalla sekä liikuteltavalla sisätilapaikannusjärjestelmiä.

LÄHTEET

- Amerikium 2015. Wikipedia. Viitattu 7.4.2015 <http://fi.wikipedia.org/wiki/Amerikium>
- Bluetooth 2015. Wikipedia. Viitattu 10.2.2015 <http://en.wikipedia.org/wiki/Bluetooth>
- Bluetooth Low Energy 2015. Wikipedia. Viitattu 10.2.2015 http://en.wikipedia.org/wiki/Bluetooth_low_energy
- Bluetooth SIG, Inc. 2015. Bluetooth Smart. Viitattu 10.2.2015 <http://www.bluetooth.com/Pages/low-energy-tech-info.aspx>
- Bluetooth SIG, Inc. 2014. Working Group Updates. Viitattu 8.6.2015
- Dimitrov, D. 2014. json newtonsoft : Deserialize Object containing a list of string. Viitattu 11.3.2015 <http://stackoverflow.com/questions/22580141/json-newtonsoft-deserialize-object-containing-a-list-of-string>
- Indoor positioning system 2015. Wikipedia. Viitattu 8.6.2015 http://en.wikipedia.org/wiki/Indoor_positioning_system
- JSON 2015. Wikipedia. Viitattu 11.2.2015 <http://fi.wikipedia.org/wiki/JSON>
- Microsoft 2015a. Tuple Class. Viitattu 6.4.2015 <https://msdn.microsoft.com/en-us/library/system.tuple.aspx>
- Microsoft 2015b. Queue<T> Class. Viitattu 14.4.2015 <https://msdn.microsoft.com/en-us/library/7977ey2c.aspx>
- Microsoft 2015c. Graphics Class. Viitattu 24.3.2015 <https://msdn.microsoft.com/en-us/library/system.drawing.graphics%28v=vs.110%29.aspx>
- Mikhail-T 2014. Use C# to get JSON Data from the Web and Map it to .NET Class => Made Easy! Viitattu 10.3.2015 <http://www.codeproject.com/Tips/397574/Use-Csharp-to-get-JSON-data-from-the-web-and-map-i>
- .Net Framework 2015. Wikipedia. Viitattu 8.4.2015 http://fi.wikipedia.org/wiki/.NET_Framework
- Paranjape, A. 2012. 'Paste JSON As Classes' in ASP.NET and Web Tools 2012.2 RC. Viitattu 5.3.2015 <http://blogs.msdn.com/b/webdev/archive/2012/12/18/paste-json-as-classes-in-asp-net-and-web-tools-2012-2-rc.aspx?Redirected=true>
- Quuppa 2013. Locating Anything Anywhere. 3-7. Quuppa. Viitattu 8.6.2015 <http://www.cse.tkk.fi/fi/opinnot/T-110.6000/2013/luennot-files/Locating%20Anyt-hing%20Anywhere%20-%20Internet%20and%20Computing%20Forum%20Kalliola%202013-03-26.pdf>
- Quuppa 2015a. Technology. Viitattu 30.4.2015 <http://quuppa.com/technology/>
- Quuppa 2015b. Technology. Viitattu 30.4.2015 <http://quuppa.com/technology/solutions>
- Quuppa 2015c. Quuppa Positioning Engine APIs (format version 2.0). 4-5. Quuppa. Viitattu 5.5.2015 V4_Quuppa_AssetTrackingAPIDocumentation.pdf
- Real-Time Locating System 2015. Wikipedia. Viitattu 9.6.2015 http://en.wikipedia.org/wiki/Real-time_locating_system

Rose, M. & Douet, M. 2012. France's 20th century radium craze still haunts Paris. Viitattu 8.6.2015 <http://www.reuters.com/article/2012/07/19/us-france-radium-decontamination-idUSBRE86I0AH20120719>

Severance, C. 2012. Discovering JavaScript Object Notation. Computer. Vol. 45, Issue 4/2012. 6-8. IEEE Computer Society. Viitattu 29.4.2015 <http://ieeexplore.ieee.org.ezproxy.turkuamk.fi/stamp/stamp.jsp?tp=&arnumber=6178118>

STUK 2014. Ionisoimaton säteily. Viitattu 7.4.2015 http://www.stuk.fi/ihminen-ja-sateily/mitaon-sateily/fi_FI/ionisoimaton/

STUK 2014. Ionisoiva säteily. Viitattu 7.4.2015 http://www.stuk.fi/ihminen-ja-sateily/mitaon-sateily/fi_FI/ionisoiva/

Townsend, K. 2014. Introduction to bluetooth low energy. Päivitetty 2015. Viitattu 27.4.2015 <https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt>

Tagin JSON-datan esimerkki

```
{  
  
  "responseTS": 1409746066235,  
  
  "tags": [  
  
    {  
  
      "areaId": "TrackingArea1",  
  
      "areaName": "KCM",  
  
      "color": "#0000FF",  
  
      "coordinateSystemId": "CoordinateSystem1",  
  
      "covarianceMatrix": [0.24,0.12,0.12,0.16],  
  
      "id": "001830ecec4",  
  
      "name": "Basket_010",  
  
      "position": [-8.11,25.06,0.8],  
  
      "positionAccuracy": 0.57,  
  
      "positionTS": 1409746065430,  
  
      "smoothedPosition": [-7.25,25.42,0.8],  
  
      "zones": [{  
  
        "id": "Zone005",  
  
        "name": "cashier"  
  
      }]  
  
    },  
  
    {  
  
      "areaId": "TrackingArea1",  
  
      "areaName": "KCM",
```

```
"color": "#0000FF",  
"coordinateSystemId": "CoordinateSystem1",  
"covarianceMatrix": [0.27,-0.05,-0.05,0.76],  
"id": "001830ecf762",  
"name": "Basket_042",  
"position": [26.5,-12.83,0.8],  
"positionAccuracy": 0.88,  
"positionTS": 1409746057001,  
"smoothedPosition": [26.5,-12.83,0.8],  
"zones": [{  
    "id": "Zone001",  
    "name": "Bread"  
}]  
}  
]  
"version": "2.0"  
  
}
```

Ohjelman toimintaa kuvaavat UML-kaaviot

