Lappeenrannan teknillinen yliopisto
Tuotantotalouden tiedekunta
Tietotekniikan koulutusohjelma

Diplomityö

Turo Ulvinen

REPLIKOITAVAN PALVELINJÄRJESTELMÄN TOTEUTTAMINEN JULKISEN LIIKENTEEN AVOIMEN DATAN JAKELUUN

Työn tarkastajat: Tutkijaopettaja Uolevi Nikula
Professori Jari Porras

Työn ohjaaja: Tutkijaopettaja Uolevi Nikula

TIIVISTELMÄ

Lappeenrannan teknillinen yliopisto
Tuotantotalouden tiedekunta
Tietotekniikan koulutusohjelma

Turo Ulvinen

Replikoitavan palvelinjärjestelmän toteuttaminen julkisen liikenteen avoimen datan jakeluun

Diplomityö

2014

70 sivua, 14 kuvaa, 16 taulukkoa, 5 liitettä

Työn tarkastajat: Tutkijaopettaja Uolevi Nikula
Professori Jari Porras

Hakusanat: avoin data, julkinen liikenne, SIRI, OneBusAway
Keywords: open data, public transport, SIRI, OneBusAway

Tässä diplomityössä tutkittiin, miten toteuttaa replikoitava palvelinjärjestelmä julkisen liikenteen avoimen datan jakeluun. Tutkimuksessa selvitettiin, onko vastaavanlaisia järjestelmiä suunniteltu aiemmin, vai pitääkö järjestelmä suunnitella itse. Projektissa käytettiin avoimen lähdekoodin OneBusAway-ohjelmistokokonaisuutta. Projektin avulla osoitettiin, että kyseinen ohjelmisto toimi yliopiston testikäytössä hyvin. Ohjelmiston avulla pystytään jakelemaan staattista ja reaaliaikaista dataa, ja se on replikoitavissa kunnasta toiseen maailmanlaajuisesti. Tulevaisuudessa olisi kuitenkin hyvä selvittää, miten ohjelmistosta puuttuva reittihakuominaisuus kannattaisi toteuttaa, sekä olisiko REST-rajapinta mahdollista muuttaa sellaiseksi, että se noudattaisi julkisen liikenteen standardeja.

ABSTRACT

Lappeenranta University of Technology
School of Industrial Engineering and Management
Degree Program in Computer Science

Turo Ulvinen

Designing a replicated server system for distributing the open data of the public transport

Master's Thesis

70 pages, 14 figures, 16 tables, 5 appendices

Examiners : Associate Professor Uolevi Nikula
Professor Jari Porras

Keywords: open data, public transport, SIRI, OneBusAway

The central aim of this master's thesis was to study how to design a replicated server system for distributing open data for public transportation. It was examined whether similar systems were developed before or whether it was necessary to design a completely new system. Existing open source software called OneBusAway was found, and was used in the project. It was shown that the software performed well in test use at the university. The software enables distributing static and real time data, and it can be replicated across cities worldwide. However, it is recommended to further examine how the routing feature, which is currently missing from the software, should be designed. Another direction for further research is to examine whether the REST interface can be modified to follow the standards of public transportation.

ALKUSANAT

Tämä diplomityö on tehty Lappeenrannan teknillisen yliopiston tietotekniikan osastolla. Kyseinen tutkimus on julkaistu aiemmin tutkimusraporttina Euroopan aluekehitysrahaston (EAKR) rahoittamassa projektissa ”Avoimilla ratkaisuilla kasvua ja vaikuttavuutta alueelle”.

Haluan kiittää tutkijaopettaja Uolevi Nikulaa ja professori Jari Porrasta arvokkaista ohjeista, sekä kannustavasta suhtautumisesta työtäni kohtaan. Haluan kiittää myös työkaveriani Reko Junttoa, sekä myös kaikkia niitä henkilöitä, jotka ovat tukeneet minua tässä projektissa.

Omistan tämän kirjan vanhemmilleni Seijalle ja Velille sekä avovaimolleni Jennille. Kiitän heitä siitä, että he ovat aina jaksaneet kannustaa ja tukea minua.

Lappeenrannassa, 3.12.2014

Turo Ulvinen

SISÄLLYSLUETTELO

1 JOHDANTO 4

1.1 TAUSTA 4

1.2 TUTKIMUS JA RAJAUKSET 5

1.3 TUTKIMUSKYSYMYKSET 6

1.4 RAPORTIN RAKENNE 6

2 KIRJALLISUUSKATSAUS JA AIEMPI TUTKIMUS 7

2.1 JULKISEN LIIKENTEEN AVOIN DATA 7

2.1.1 Avoimen datan määritelmä 7

2.1.2 Saatavilla oleva julkisen liikenteen avoin data 8

2.1.3 Olemassa olevia jakelujärjestelmiä 9

2.1.4 Hyödyt julkiselle liikenteelle 11

2.2 YLEISIMMIN KÄYTETYT STANDARDIT 12

2.2.1 Transmodel 12

2.2.2 GTFS 13

2.2.3 IFOPT 15

2.2.4 NeTEx 15

2.2.5 GTFS Real-time 16

2.2.6 SIRI 17

2.2.7 Vertailu 21

2.3 YLEISIMMIN KÄYTETYT DATAN SERIALISOINTIMENETELMÄT 24

2.3.1 JSON 24

2.3.2 XML 25

2.3.3 Protocol Buffers 25

2.3.4 Vertailu 26

3 JAKELUJÄRJESTELMÄN TOTEUTUS 27

3.1 JÄRJESTELMÄN VALINTA 27

3.2 JÄRJESTELMÄN KUVAUS 28

3.2.1 Datapalvelin 30

3.2.2 Aluetietopalvelin *31*
3.2.3 Web-ohjelmisto *32*
3.2.4 Mobiili-ohjelmistot *35*
3.3 JÄRJESTELMÄN KÄYTTÖÖNOTTO 38
3.4 JÄRJESTELMÄN LOKALISOINTI 43
3.5 JÄRJESTELMÄN YLLÄPITO 44
3.6 ILMENNEET ONGELMAT 45
3.7 TULOSTEN YHTEENVETO JA TULKINTA 47
4 ARVIOINTIA JA POHDINTAA 51
5 JOHTOPÄÄTÖKSET 53
LÄHTEET 55
LIITTEET

SYMBOLI- JA LYHENNELUETTELO

CEN	Comité Européen de Normalisation
CM	Connection Monitoring
CSV	Comma-separated values
CT	Connection Timetable
ET	Estimated Timetable
GM	General Messaging
GPS	Global Positioning System
GTFS	General Transit Feed Specification
GTFS-R	General Transit Feed Specification Real-time
GWT	Google Web Toolkit
HAFAS	HaCon Fahrplan-Auskunfts-System
HTML	Hypertext Markup Language
IFOPT	Identification of Fixed Objects in Public Transport
JSON	JavaScript Object Notation
JSP	JavaServer Pages
NeTEx	Network and Timetable Exchange
PT	Production Timetable
REST	Representational State Transfer
SIRI	Service Interface for Real Time Information
SM	Stop Monitoring
ST	Stop Timetable
URL	Uniform Resource Locator
VM	Vehicle Monitoring
XML	Extensible Markup Language

1 JOHDANTO

Tässä raportissa esitellään tutkimus ”Replikoitavan palvelinjärjestelmän toteuttaminen julkisen liikenteen avoimen datan jakeluun”. Luvussa 1.1 esitellään tutkimuksen tausta. Luvussa 1.2 kerrotaan mitä tutkimus sisältää ja mitä rajauksia sille asetettiin. Luvussa 1.3 esitellään tutkimuksen tutkimuskysymykset. Luvussa 1.4 kuvataan raportin rakenne.

1.1 Tausta

Avoimen datan suosio on ollut kasvussa viime aikoina ympäri maailmaa. Suosion kasvamisen myötä dataa on avattu yhä enemmän eri organisaatioiden toimesta. Tämä on avannut paljon mahdollisuuksia sovelluskehittäjille, jotka ovat voineet kehittää dataa hyödyntäviä ohjelmia. Isoimmat kunnat ovat toimineet avoimen datan edelläkävijöinä ja pienemmät kunnat ovat seuraamassa niiden jalanjälkiä [1]. Etenkin julkisen liikenteen staattista ja reaaliaikaista aikatauluinformaatiota on avattu useammassa kunnassa avoimena datana [2, 3]. Tämän datan on havaittu tuovan monia hyötyjä sekä julkisen liikenteen toimijoille että niiden asiakkaille [4, 5]. Erityisesti julkisen liikenteen reaaliaikainen aikatauluinformaatio ja siihen liittyvä muu informaatio (esim. häiriötiedotteet) on havaittu olevan todella hyödyllisiä [6].

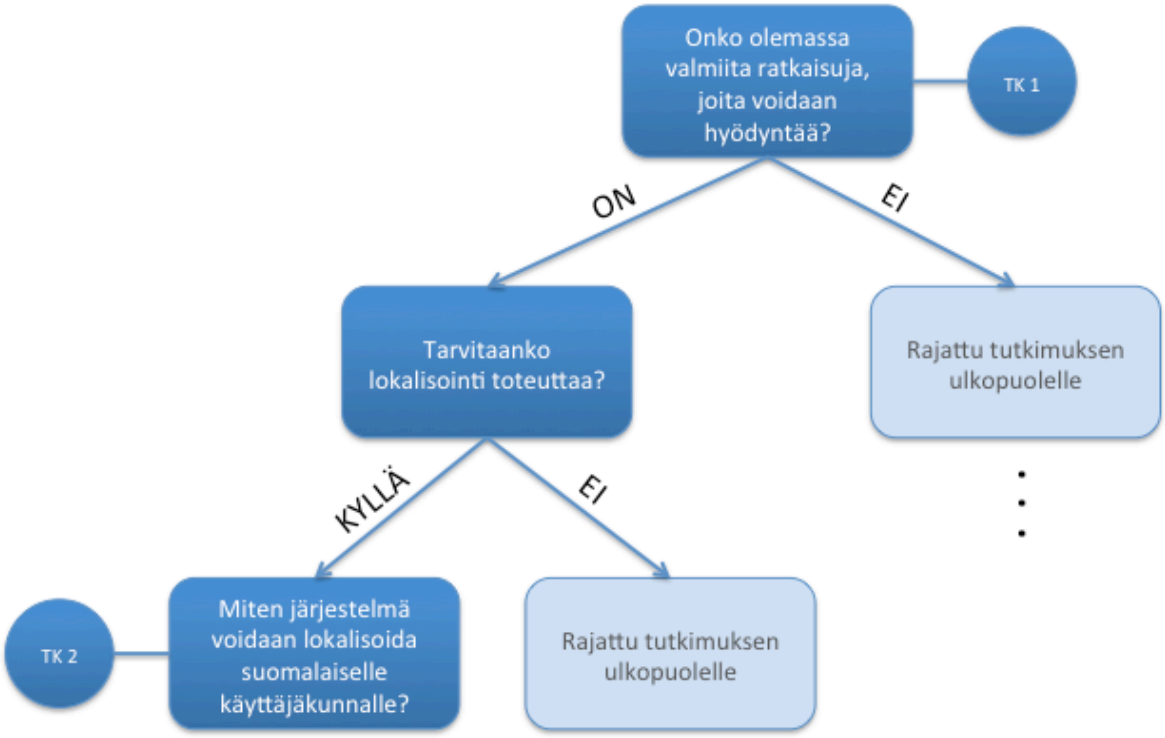
Vaikka dataa on saatavilla monissa kunnissa, niin sen hyödyntäminen ohjelmistoissa ei ole itsestäänselvyys. Datan jakeleminen asiakasohjelmistoille aiheuttaa tällä hetkellä ongelmia, koska eri kunnissa dataa jaellaan erilaisten standardoimattomien rajapintojen välityksellä. Näin ollen asiakasohjelmistot eivät ole suoraan yhteensopivia muiden kuntien dataa jakelevien järjestelmien kanssa. Tästä johtuen asiakasohjelmaan joudutaan toteuttamaan ylimääräistä logiikkaa, jos se halutaan saada toimimaan eri kunnissa.

Tämän tutkimuksen tavoitteena on selvittää, kuinka pystytään toteuttamaan replikoitava palvelinjärjestelmä, jonka avulla pystytään jakelemaan julkisen liikenteen avointa dataa asiakasohjelmistoille. Replikoitavuus mahdollistaa sen, että sama järjestelmä pystytään kopioimaan kunnasta toiseen mahdollisimman pienillä kustannuksilla, ja näin ollen pystytään kehittämään yksi asiakasohjelmisto, joka toimii kaikissa samoja käytäntöjä

noudattavissa kunnissa. Kyseinen järjestelmä toteutettiin pilotti-projektina Lappeenrannan kaupungille.

1.2 Tutkimus ja rajaukset

Kuvassa 1 on havainnollistettu tutkimuksen päämäärät ja rajaukset. Tutkimuksen aluksi selvitettiin, onko vastaavanlaisia järjestelmiä suunniteltu jo aiemmin (tutkimuskysymys 1). Havaittiin, että Yhdysvalloissa Washingtonin yliopistossa on tutkittu julkisen liikenteen avointa dataa ja sen tutkimusprojektin tuloksena on syntynyt OneBusAway-niminen ohjelmistokokonaisuus, joka on sen jälkeen laajentunut avoimen lähdekoodin projektiksi [7]. Ohjelmistokokonaisuus sisältää palvelinohjelmiston lisäksi valmiit web- ja mobiili-ohjelmistot. Tässä pilotti-projektissa ei kehitetä omaa jakelujärjestelmää vaan käytetään OneBusAway-ohjelmistokokonaisuutta. Seuraavaksi pohdittiin, tarvitseeko valittu ohjelmistokokonaisuus lokalisoida suomen kielelle. Lokalisointi nähtiin tarpeelliseksi, koska suurin osa ohjelmiston käyttäjäkunnasta on suomalaisia. Lopuksi pohdittiin, miten järjestelmä voidaan lokalisoida suomalaiselle käyttäjäkunnalle (tutkimuskysymys 2).



Kuva 1: Tutkimuksen päämäärät ja rajaukset.

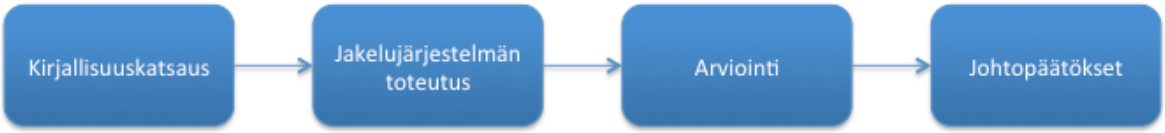
1.3 Tutkimuskysymykset

Tutkimuksen tutkimuskysymykset koostuvat alla olevista kahdesta kysymyksestä. Näistä ensimmäiseen kysymykseen saadaan vastaus luvussa 2 ja toiseen kysymykseen saadaan vastaus luvussa 3.

TK 1: Onko olemassa valmiita ratkaisuja, joita voidaan hyödyntää replikoitavan palvelinjärjestelmän toteuttamiseen?
TK 2: Miten käytettävä OneBusAway-järjestelmä lokalisoidaan suomalaiselle käyttäjäkunnalle?

1.4 Raportin rakenne

Raportti koostuu viidestä luvusta. Raportin rakenne on esitetty kuvassa 2. Luvussa 2 esitellään kirjallisuuskatsaus ja mitä aiempia tutkimuksia aiheeseen liittyen on tehty. Luvussa 3 kerrotaan, miten julkisen liikenteen staattisen ja reaaliaikaisen aikatauluinformaation jakelujärjestelmä toteutettiin, mitä ongelmia siinä esiintyi ja mitä tuloksia saavutettiin. Luvussa 4 arvioidaan tutkimuksen tuloksia. Lopuksi luvussa 5 esitellään tutkimuksen johtopäätökset.



Kuva 2: Raportin rakenne.

2 KIRJALLISUUSKATSAUS JA AIEMPI TUTKIMUS

Tässä luvussa tarkastellaan, mitä julkisen liikenteen avoimesta datasta ja sen jakelusta on kirjoitettu ja mitä aiempaa tutkimusta siitä on tehty. Luvussa 2.1 esitellään julkisen liikenteen avointa dataa: mitä avoin data on, millaista avointa dataa on tällä hetkellä saatavilla, millaisia jakelujärjestelmiä on tällä hetkellä käytössä ja mitä hyötyjä avoimesta datasta on julkiselle liikenteelle. Luvussa 2.2 käsitellään yleisemmin käytettyjä standardeja julkisen liikenteen avoimen datan jakelussa ja vertaillaan niitä keskenään. Luvussa 2.3 esitellään yleisimmin käytettyjä datan serialisointimenetelmiä ja vertaillaan mikä niistä on paras mihinkin tarkoitukseen.

2.1 Julkisen liikenteen avoin data

Avoimen datan suosio on viime aikoina ollut kasvussa ja dataa on avattu eri organisaatioiden toimesta. Tästä johtuen myös julkisen liikenteen informaatiotakin on käyty tarjoamaan monissa kunnissa avoimena datana. Saatavilla on sekä staattista että reaaliaikaista aikatauluinformaatiota. Luvussa 2.1.1 esitellään avoimen datan määritelmä ja luvussa 2.1.2 kerrotaan mitä avointa dataa on saatavilla ja mitä standardeja niiden jakelussa käytetään. Julkisen liikenteen avoimen datan jakeluun tarvitaan siihen tarkoitukseen räätälöity jakelujärjestelmä. Tähän tarkoitukseen soveltuvia jakelujärjestelmiä esitellään luvussa 2.1.3. Avoimen datan on havaittu tuovan myös monia hyötyjä julkisen liikenteen toimijoille ja niiden asiakkaille. Nämä hyödyt esitellään luvussa 2.1.4.

2.1.1 Avoimen datan määritelmä

Avoimella datalla tarkoitetaan sellaista informaatiota, joka on vapaasti kaikkien hyödynnettävissä, uudelleen käytettävissä ja uudelleen jaettavissa [8]. Data pitää lisensoida niin, että sen uudelleenkäytölle ja -jakamiselle ei ole esteitä [8]. Sen yhdistely muiden data-aineistojen kanssa pitää sallia. Datan käytöstä saa periä maksua korkeintaan sen irrotuskustannusten verran, mutta paras ratkaisu olisi tarjota se ilmaiseksi. Sen tulee olla myös kaupallisesti hyödynnettävissä. Data tulee tarjota sellaisessa muodossa, että se on

helposti koneellisesti luettavissa sekä mielellään ladattavissa internetistä. [8]

2.1.2 Saatavilla oleva julkisen liikenteen avoin data

Julkisen liikenteen aikatauluinformaatiota on ollut jo pitkään saatavilla internetin välityksellä. Avoimen datan suosion kasvaessa useammat kaupungin ovat alkaneet jakelemaan tätä informaatiota myös avoimena datana. Kyseistä informaatiota jaellaan rajapintojen välityksellä, jotta se olisi helposti ohjelmallisesti luettavissa. Informaation jakelussa käytetään yleensä julkisen liikenteen standardeja, mutta sitä jaellaan myös standardoimattomien rajapintojen välityksellä. Tarjolla on yleensä vähintään staattista aikatauluinformaatiota, mutta monet kaupungit tarjoavat myös reaaliaikaista aikatauluinformaatiota sekä siihen liittyvää muuta informaatiota kuten esimerkiksi häiriötiedotteet.

Saatavilla olevaa avointa dataa selvitettiin seuraavien 4 kaupungin osalta: Helsinki, Tampere, Berliini ja New York. Kaupunkeja valitessa päätettiin, että kaksi kaupunkia valitaan Suomesta, jotta saadaan hyvä kuva kotimaisesta tilanteesta. Lisäksi valittiin yksi kaupunki Euroopasta ja yksi Amerikasta, jotta saadaan myös kuvaa kansainvälisestä tilanteesta. Kaupungit valittiin sillä kriteerillä, että saadaan mahdollisimman laaja kuva käytetyistä jakeluformaateista. Selvityksessä tarkasteltiin mitä koneellisesti luettavaa avointa dataa kussakin kaupungissa on avattu ja mitä jakelumuotoja niiden jakelussa käytetään. Eri kaupungeissa käytettävät datan jakelumuodot on esitetty taulukossa 1. Selvityksessä ilmeni, että Helsingissä ja Tampereella on saatavilla staattista ja reaaliaikaista aikatauluinformaatiota [2, 3]. Molemmissa kaupungeissa staattisen informaation jakeluun käytetään GTFS (**G**eneral **T**ransit **F**eed **S**pecification) -standardia. Lisäksi molemmissa kaupungeissa dataa jaellaan myös CGI:n toteuttaman Reittiopas-rajapinnan kautta ja Kalkatin.net tietokanta dump-tiedostona. CGI:n toteuttama rajapinta ei perustu mihinkään standardiin. Kalkatin.net on suomalainen Liikenne- ja Viestintäministeriön kehittämä standardi, jonka avulla voidaan kuvata XML:ää (E**x**tensible **M**arkup **L**anguage) käyttäen julkisen liikenteen ja rahtiliikenteen tarvitsemat tiedot, jotta niitä voidaan siirtää eri järjestelmien välillä [9]. Reaaliaikaisen informaation jakeluun Helsingissä käytetään HSL Live -rajapintaa ja Tampereella SIRI-standardia (**S**ervice

Interface for **R**eal Time **I**nformation) [10]. HSL Live on Mattersoft:in toteuttama rajapinta, eikä se perustu mihinkään standardiin. Berliinissä saatavilla on ainoastaan staattista aikatauluinformaatiota, ja sen jakelussa käytetään GTFS-standardia. Lisäksi dataa jaellaan myös HaCon:in toteuttaman standardoimattoman HAFAS (**Ha**Con **F**ahrplan-**A**uskunfts-**S**ystem) -rajapinnan välityksellä [11]. New York:ssa on saatavilla staattista ja reaaliaikaista informaatiota [12]. Siellä staattisen informaation jakeluun käytetään GTFS-standardia. Reaaliaikainen informaatio jaellaan käyttäen SIRI- tai GTFS-R (**G**eneral **T**ransit **F**eed **S**pecification **R**eal-time) -standardia [13]. Bussit käyttävät SIRI-muotoa ja metrot GTFS-R-muotoa. [2, 3, 12]

Taulukko 1: Julkisen liikenteen avoimen datan jakelumuodot eri kaupungeissa.

Jakelumuoto	*Helsinki*	*Tampere*	*Berliini*	*New York*
GTFS	x	x	x	x
Reittiopas*	x	x		
Kalkatin.net	x	x		
HAFAS*			x	
GTFS-R**				x
SIRI**		x		x
HSL Live**	x			
(* standardoimaton) (reaaliaikainen)**				

Tehty selvitys osoittaa, että avointa dataa on saatavilla kaikissa neljässä tutkitussa kaupungissa. Jokaisessa kaupungissa on tarjolla staattista aikatauluinformaatiota. Berliiniä lukuun ottamatta kaikissa kaupungeissa on tarjolla myös reaaliaikaista informaatiota. Yleisin staattisen aikatauluinformaation välitykseen käytetty standardi on GTFS. Reaaliaikaisen informaation välitykseen suosituin on SIRI. Monessa kaupungissa dataa jaellaan myös standardoimattomien rajapintojen välityksellä.

2.1.3 Olemassa olevia jakelujärjestelmiä

Julkisen liikenteen staattisen ja reaaliaikaisen aikatauluinformaation jakeluun on käytössä erilaisia jakelujärjestelmiä, jotka voidaan jakaa kolmeen eri ryhmään: maailmanlaajuiset, kansalliset ja paikalliset. Maailmanlaajuiset jakelujärjestelmät kattavat useamman maan. Kansalliset jakelujärjestelmät kattavat tietyn maan, eli ne jakelevat keskitetysti koko maan

aikatauluinformaation. Paikalliset jakelujärjestelmät kattavat tietyn paikallisen alueen, kuten esimerkiksi tietyn kaupungin.

Olemassa olevia jakelujärjestelmiä tutkittiin pääasiassa maailmanlaajuisella tasolla, koska tutkimuksen tavoitteena oli löytää palvelinjärjestelmä, joka pystytään replikoimaan kunnasta toiseen ja joka toimii myös kansainvälisesti. Suomen osalta tutkittiin myös jakelujärjestelmiä kansallisella ja paikallisella tasolla Helsingissä ja Tampereella. Maailmanlaajuisella tasolla havaittiin olevan kaksi erilaista avoimen lähdekoodin jakelujärjestelmää: OneBusAway ja Navitia. OneBusAway jakelee staattista ja reaaliaikaista aikatauluinformaatiota REST-rajapinnan (**Re**presentational **S**tate **T**ransfer) välityksellä käyttäen GTFS-, GTFS-R- ja SIRI-standardeja [7]. OneBusAway on kattavasti dokumentoitu ja sillä on kehittäjäyhteisö, joka kehittää ohjelmistoa aktiivisesti. Navitia jakelee ainoastaan staattista aikatauluinformaatiota REST-rajapinnan välityksellä käyttäen GTFS-standardia [14]. Navitian REST-rajapinta on dokumentoitu, mutta itse ohjelmiston käyttöönotosta ei ole mitään dokumentaatiota. Informaation jakelu tapahtuu OneBusAway:ssa hajautetusti (jokaisella alueella oma palvelin) ja Navitia:ssa keskitetysti (yksi maailmanlaajuinen palvelin). Kummankaan jakelujärjestelmän REST-rajapinta ei perustu mihinkään julkisen liikenteen standardiin. Kansallisella tasolla Suomessa on käytössä CGI:n toteuttama jakelujärjestelmä, jota käytetään Liikenneviraston Matka.fi-palvelussa [15]. Se jakelee maanlaajuisesti staattista informaatiota REST-rajapinnan välityksellä, joka ei perustu mihinkään julkisen liikenteen standardiin. Samaa jakelujärjestelmää käytetään myös paikallisella tasolla Helsingissä ja Tampereella. Jakelujärjestelmä ei perustu avoimeen lähdekoodiin, joten sitä ei voida hyödyntää maksutta. [7, 14, 16]

Tehdyn selvityksen perusteella varteenotettavia avoimen lähdekoodin jakelujärjestelmiä on vähän. Lisäksi havaittiin, että jokaisella jakelujärjestelmällä on oma REST-rajapinta, joka ei perustu mihinkään julkisen liikenteen standardiin, eivätkä rajapinnat ole keskenään yhteensopivia. Näistä jakelujärjestelmistä OneBusAway vaikuttaa parhaalta, koska se perustuu avoimeen lähdekoodiin ja sillä on kehittäjäyhteisö, joka kehittää järjestelmää aktiivisesti. Lisäksi se on ainoa jakelujärjestelmä, joka jakelee myös reaaliaikaista informaatiota. OneBusAway:n maailmanlaajuinen hajautettu arkkitehtuuri vaikuttaa myös

luotettavammalta kuin Navitia:n suosima yhden palvelimen keskitetty ratkaisu.

2.1.4 Hyödyt julkiselle liikenteelle

Julkisen liikenteen avoimesta datasta on havaittu olevan monia hyötyjä sekä julkisen liikenteen toimijoille että niiden asiakkaille. Etenkin on havaittu, että reaaliaikaisesta informaatiosta on merkittävää hyötyjä. Sen vaikutuksia ihmisiin onkin tutkittu monessa tutkimuksessa.

Eräässä tutkimuksessa tutkittiin reaaliaikaisen informaation vaikutusta julkista liikennettä käyttäviin ihmisiin OneBusAway ohjelmiston avulla [17]. Tutkimuksessa havaittiin, että reaaliaikaisen informaation myötä 92 % tutkimukseen osallistuneista käyttäjistä oli tyytyväisempiä julkiseen liikenteeseen. Lisäksi havaittiin, että 91 % käyttäjistä käytti vähemmän aikaa pysäkillä odottamiseen ja julkisen liikenteen viikoittaiset käyttökerrat lisääntyivät noin 25 %. Toisessa tutkimuksessa tutkittiin reaaliaikaisen informaation vaikutusta ihmisen kokemaan odotusaikaan mobiili-laitteiden avulla [18]. Käytetyissä laitteissa oli asennettuna ohjelma, joka näytti tietylle pysäkille saapuvien kulkuneuvojen saapumisaikaennusteen. Tutkimuksessa havaittiin, että reaaliaikainen informaatio lyhentää ihmisen kokemaa kulkuneuvon odotusaikaa sekä vähentää sen odottamisesta johtuvaa stressiä. Lisäksi havaittiin, että ihmiset saapuvat pysäkille myöhemmin, koska he tietävät tarkalleen milloin kulkuneuvo saapuu kyseiselle pysäkille. Tutkimuksessa havaittiin myös, että reaaliaikaisesta informaatiosta on eniten hyötyä silloin kuin perättäisten kulkuneuvojen saapumisväli on suurempi kuin 10 minuuttia. Kolmannessa tutkimuksessa tutkittiin reaaliaikaisen informaation vaikutusta ihmisiin pysäkeille ja asemille sijoitettujen näyttötaulujen avulla [19]. Todettiin, että ihmisten kokema kulkuneuvon odotusaika lyheni noin 20 % reaaliaikaisen informaation myötä. Lisäksi todettiin, että kulkuneuvojen vuorojen määriä lisäämällä saavutetaan samanlainen vaikutus, mutta siitä koituvat kustannukset ovat viisi kertaa korkeammat kuin näyttötaulujen kustannukset. [17, 18, 19]

Edellä esitetyt tutkimukset käsittelivät reaaliaikaisen informaation vaikutusta julkista liikennettä käyttäviin ihmisiin. Näiden tutkimusten perusteella voidaan todeta, että reaaliaikaisella julkisen liikenteen avoimella datalla saavutetaan seuraavia merkittäviä

hyötyjä:

- Käyttäjät ovat tyytyväisempiä
- Todellinen odotusaika lyhenee, koska pysäkille voi saapua juuri ennen kulkuneuvon saapumista
- Käyttäjien kokema odotusaika lyhenee, koska reaaliaikaisen informaation myötä odotusaika tuntuu todellista lyhyemmältä
- Viikoittaiset käyttäjämäärät kasvavat
- Odottamisesta aiheutuva stressi vähenee

Vaikka tutkimuksissa käsiteltiin vain reaaliaikaista informaatiota, voidaan silti olettaa, että myös staattisesta informaatiosta on joissain määrin samoja hyötyä, jos se esitetään vastaavalla tavalla. Esimerkiksi ensimmäiseksi mainitussa tutkimuksessa [17] käytetty OneBusAway ohjelmisto esittää käyttäjälle staattista informaatiota, mikäli reaaliaikaista informaatiota ei ole saatavilla.

2.2 Yleisimmin käytetyt standardit

Julkisen liikenteen aikatauluinformaation ja muun siihen liittyvän informaation jakeluun on olemassa monia standardeja. Tässä kirjallisuuskatsauksessa rajoitutaan tutkimaan ainoastaan eurooppalaisia ja kansainvälisiä standardeja. Näistä tunnetuimpia standardeja ovat Transmodel, GTFS, IFOPT (**I**dentification of **F**ixed **O**bjects in **P**ublic **T**ransport), NeTEx (**Ne**twork and **T**imetable **Ex**change), GTFS-R ja SIRI. Luvuissa 2.2.1 – 2.2.6 esitellään nämä standardit ja lopuksi luvussa 2.2.7 vertaillaan mitkä standardit näistä ovat tällä hetkellä parhaimpia.

2.2.1 Transmodel

Transmodel on eurooppalainen CEN (**C**omité **E**uropéen de **N**ormalisation) -standardi, joka määrittelee julkisen liikenteen tietomallin abstraktilla tasolla. Sen avulla kuvataan julkisen liikenteen käsitteistöt ja tietorakenteet. Tietomallin avulla voidaan kehittää tietojärjestelmiä, jotka ovat yhteensopivia muiden vastaavien järjestelmien kanssa.

Olemassa olevat eurooppalaiset standardit pohjautuvat Transmodeliin. Se tukee seuraavia kulkuneuvotyyppejä: bussi, johdinauto, raitiovaunu ja metro. Tietomalli voidaan jakaa kuuteen toiminnalliseen osaan: Tactical planning, Personel disposition, Operations monitoring and control, Passanger information, Fare collection ja Management information and statistics. Toiminnalliset osat ja niiden sisältö on esitetty taulukossa 2. [20]

Taulukko 2: Transmodelin toiminnalliset osat.

Toiminnallinen osa	*Sisältö*
Tactical planning	Kuljettajien aikataulutus
Personel disposition	Kuljettajien hallinta ja työajanseuranta
Operations monitoring and control	Operoinnin reaaliaikainen seuranta ja hallinta
Passanger information	Matkustajille välitettävä staattinen ja reaaliaikainen informaatio
Fare collection	Lippujen hinnoittelu
Management information and statistics	Hallinnollinen informaatio ja tilastot

2.2.2 GTFS

GTFS on Google ja TriMet yritysten yhteistyön tuloksena syntynyt formaatti, jolla kuvataan julkisen liikenteen staattisia aikatauluja sekä niihin liittyviä tietoja. Formaatista ei ole tehty virallista standardia, mutta se on vakiintunut käyttöön maailmanlaajuisesti de facto -standardina. [21, s. 126-128]

GTFS koostuu 13:sta CSV-formaatissa (**C**omma-**s**eparated **v**alues) olevasta tekstitiedostoista, joista 6 on pakollisia. Kukin tiedostoista määrittelee tietyn osa-alueen julkisen liikenteen informaatiosta (pysäkit, reitit, matkat, muut aikataulutiedot). Tiedostot on linkitetty toisiinsa id-arvojen avulla sekä pakattu zip-formaattiin. Tiedostot ja niiden sisältämä tieto on esitelty taulukossa 3. [22]

Taulukko 3: Transmodelin toiminnalliset osat.

Tiedosto	*Sisältö*	*Pakollinen*
agency.txt	Liikennöitsijän tiedot (voi olla useampia)	x
stops.txt	Pysäkkien tiedot	x
routes.txt	Reittien tiedot (koostuu useista matkoista)	x
trips.txt	Matkojen tiedot	x
stop_times.txt	Kunkin pysäkin saapumis- ja lähtöajat	x
calendar.txt	Kertoo milloin kukin matka liikennöidään	x
calendar_dates.txt	Poikkeukset normaaliin aikatauluun (calendar.txt). Lisäksi koko aikataulu voidaan myös kuvata tässä tiedostossa, jos aikataulu vaihtelee päivittäin. Tällöin calendar.txt tiedostoa ei käytetä.	
fare_attributes.txt	Lippujen hintatiedot	
fare_rules.txt	Säännöt miten lippujen hinnat määräytyvät tietyllä reitillä	
shapes.txt	Koordinaattitiedot reittien piirtämiseen kartalle	
frequencies.txt	Jos matkalle ei ole pysäkkikohtaisia saapumis- ja lähtöaikoja, niin tässä tiedostossa määritetään kullekin matkalle intervalli milloin kulkuneuvot lähtevät tietyllä aikavälillä.	
transfers.txt	Reittien välisten vaihtoyhteyksien määritys	
feed_info.txt	Tietoja kyseisestä aikatauluinformaatiosta (esim. aikatauluinformaation voimassaoloaika)	

GTFS-formaattia käyttäen julkisen liikenteen toimijat pystyvät toimittamaan aikatauluinformaationsa Googlelle, joka hyödyntää tätä informaatiota sen Google Transit palvelussa. Samalla formaatilla pystytään jakelemaan aikatauluinformaatiota myös kolmansien osapuolten ohjelmille. Jakelu kummassakin yllämainitussa tapauksessa tapahtuu yleensä niin, että julkisen liikenteen toimija muodostaa omasta aikatauluinformaatiosta määrityksen mukaisen zip-tiedoston (sisältää vähintään pakolliset tekstitiedostot), sekä julkaisee sen web-palvelimellaan ja tiedottaa sen sijainnista sitä käyttäviä osapuolia. GTFS tukee seuraavia kulkuneuvoja: raitiovaunu, metro, juna, bussi, lautta, kaapeliauto, gondolihissi, kiskoköysirata. [22]

2.2.3 IFOPT

IFOPT on eurooppalainen CEN –standardi, joka on kehitetty kuvaamaan tärkeimpiä kohteita (esim. pysähdysalueen yksityiskohtainen kuvaus, kulkuneuvoon nousemiskohdat ja siirtyminen näiden kohtien välillä), jotka ovat oleellisia tietoja julkisilla kulkuneuvoilla liikuttaessa. Näiden tietojen avulla julkista kulkuneuvoa käyttävä henkilö pystytään esimerkiksi ohjaamaan oikeaan paikkaan kulkuneuvosta toiseen siirryttäessä. Sitä voidaan käyttää yhdessä muiden eurooppalaisten julkisen liikenteen standardien kanssa. IFOPT pohjautuu Transmodeliin ja koostuu neljästä itsenäisestä moduulista, jotka täydentävät toisiaan: Stop Place Model, Point of Interest Model, Gazetteer Topographical Model ja Administrative Model. Näistä ainoastaan Stop Place Model -moduuli on pakollinen. Moduulit ja niiden sisältö on esitetty taulukossa 4. [23]

Taulukko 4: IFOPT:n sisältämät moduulit.

Moduuli	*Sisältö*	*Pakollinen*
Stop Place Model	Yksityiskohtainen kuvaus pysähdysalueen (asema, bussipysäkki, lentokenttä, yms.) rakenteesta, sisältäen kulkuneuvoon nousemiskohdat ja reitit näiden pisteiden välillä.	x
Point of Interest Model	Yksityiskohtainen kuvaus kiinnostuksen kohteista ja niiden sisäänkäyntikohdista.	
Gazetteer Topographical Model	Topografinen kuvaus asutusalueesta, jossa matkustaminen tapahtuu.	
Administrative Model	Organisaatiomalli vastuualueiden hallintaan, koskien datan luontia ja sen hallintaa yhteistyössä muiden toimijoiden kanssa.	

2.2.4 NeTEx

NeTEx on kehitteillä oleva eurooppalainen CEN-standardi, joka on kehitetty julkisen liikenteen aikataulujen ja niihin liittyvän muun informaation välitykseen. Se on suunniteltu yleiskäyttöiseksi standardiksi, joten sen avulla pystytään välittämään kaikkien yleisempien kulkuneuvomuotojen (juna, bussi, metro, raitiotie, lautta, yms.) aikatauluinformaatiota.

NeTEx perustuu Transmodel-, IFOPT-, ja SIRI-standardeihin ja koostuu kolmesta osasta. Osa 1 määrittelee julkisen liikenteen verkkotopologian (reitit, linjat, reittipisteet, pysäkit, yms.) [24]. Osa 2 määrittelee aikataulut ja niihin liittyvät informaatiot (vaihtomahdollisuudet, kulkuneuvojen eritysominaisuudet, yms.) [25]. Osa 3 määrittelee lippujen hinnoittelun. Näistä ensimmäinen osa on täysin itsenäinen, mutta osat 2 ja 3 vaativat osan 1 toimiakseen. Standardin kaksi ensimmäistä osaa on virallisesti julkaistu [26, 27], mutta kolmas osa on tällä hetkellä luonnosvaiheessa. [24, 25]

2.2.5 GTFS Real-time

GTFS-R on Googlen kehittämä laajennus GTFS-formaattiin. Se on suunniteltu yhteistyössä useamman julkisen liikenteen toimijan kanssa. Laajennuksen avulla mahdollistetaan reaaliaikaisen tiedon välitys. GTFS-R koostuu kolmesta syötetyypistä: Trip Updates, Service Alerts ja Vehicle Positions. Nämä syötetyypit ja niiden sisältö on esitetty taulukossa 5. [13]

Taulukko 5: GTFS-R syötetyypit.

Syötetyyppi	*Sisältö*
Trip Updates	Päivitykset matkoihin (viivästykset, peruuntumiset, reittimuutokset)
Service Alerts	Häiriöilmoitukset
Vehicle Positions	Kulkuneuvon sijainti (koordinaatti, suuntima, kulkuneuvon kulkema matka, nopeus), ruuhkatieto, kulkuneuvon pysäkki status (saapumassa, lähdössä, matkalla), kulkuneuvon tiedot (id, kulkuneuvon nimi, rekisterikilvennumero)

Tiedonvälitys toteutetaan käyttämällä Protocol Buffers -menetelmää. Se on kieli- ja alustariippumaton strukturoidun tiedon serialisointimenetelmä. Välitettävä tieto jäsennellään haluttuun muotoon ja tämän jälkeen se käännetään binääri-tiedostoksi hyödyntäen valmiita kirjastoja. Google tarjoaa nämä kirjastot seuraaville ohjelmointikielille: Java, C++ ja Python. [28]

2.2.6 SIRI

SIRI on eurooppalainen CEN -standardi, jota käytetään julkisen liikenteen aikatauluinformaation välitykseen. Se perustuu luvussa 2.2.1 esitettyyn Transmodel -standardiin. SIRI:llä voidaan välittää staattista ja reaaliaikaista aikatauluinformaatiota, sekä muuta aikatauluihin liittyvää tietoa. Se koostuu kahdeksasta toiminnallisesta palvelusta: PT (**P**roduction **T**imetable), ET (**E**stimated **T**imetable), ST (**S**top **T**imetable), SM (**S**top **M**onitoring), VM (**V**ehicle **M**onitoring), CT (**C**onnection **T**imetable), CM (**C**onnection **M**onitoring), GM (**G**eneral **M**essaging). Palvelut ja niiden sisältö on esitetty taulukossa 6. Ne ovat toisistaan riippumattomia, joten rajapintaa käyttävä voi vapaasti valita mitä palveluita halutaan käyttää. [10]

Taulukko 6: SIRI:n toiminnalliset palvelut.

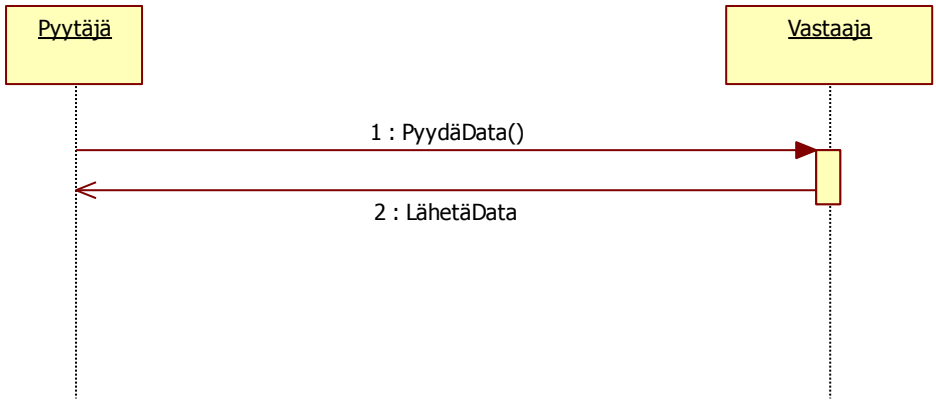
Palvelu	*Sisältö*
Production Timetable	Staattisen aikataulutiedon välitys. Tietoja voidaan rajata tietyillä kriteereillä (liikennöitsijä, reitti, suunta, aikaväli).
Estimated Timetable	Reaaliaikaisen aikataulutiedon välitys. Tietoja voidaan rajata tietyillä kriteereillä (liikennöitsijä, reitti, suunta).
Stop Timetable	Staattisen pysäkkikohtaisen aikataulutiedon välitys (kulkuneuvojen saapumis- ja lähtöajat). Tietoja voidaan rajata tietyillä kriteereillä (pysäkki, liikennöitsijä, reitti, suunta).
Stop Monitoring	Reaaliaikaisen pysäkkikohtaisen aikataulutiedon välitys (kulkuneuvojen saapumis- ja lähtöajat). Tietoja voidaan rajata tietyillä kriteereillä (pysäkki, liikennöitsijä, reitti, suunta).
Vehicle Monitoring	Kulkuneuvojen reaaliaikaisen paikkatiedon ja siihen liittyvän muun tiedon välitys. Tietoja voidaan rajata tietyillä kriteereillä (kulkuneuvotunniste, kulkuneuvoryhmä, reitti, suunta).
Connection Timetable	Staattisen tiedon välitys vaihtomahdollisuuksista eri liikennöintiyritysten välillä tietyssä vaihtopisteessä. Tietoja voidaan rajata tietyillä kriteereillä (vaihtopiste, reitti, suunta, kulkuneuvotunniste).

Connection Monitoring	Reaaliaikaisen tiedon välitys vaihtomahdollisuuksista eri liikennöintiyritysten välillä tietyssä vaihtopisteessä. Tietoja voidaan rajata tietyillä kriteereillä (vaihtopiste, reitti, suunta, kulkuneuvotunniste).
General Messaging	Viestien välitys (esim. häiriötiedot). Viesti jaetaan eri ryhmiin niiden sisällön perusteella (esim. virheet, varoitukset, muut viestit).

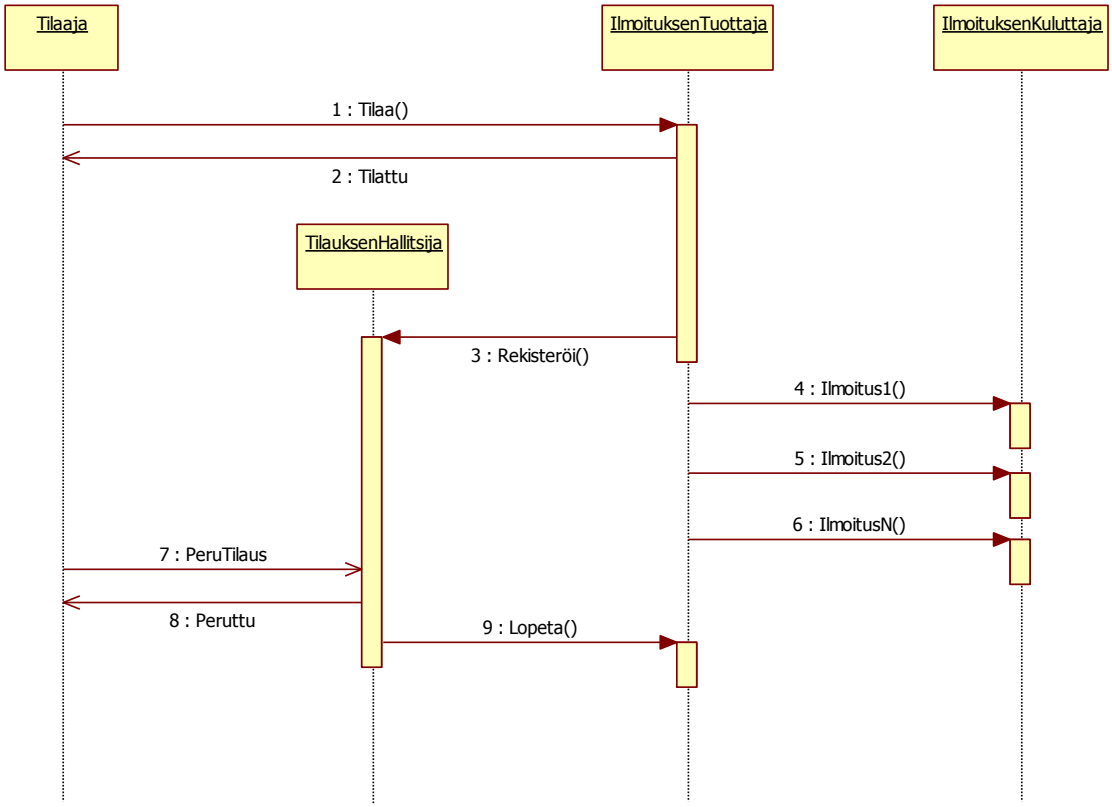
PT-palvelun avulla välitetään staattisia aikataulutietoja. Aikatauluihin voidaan tehdä muutoksia niiden voimassaoloaikana. Vastaanottava osapuoli voi rajata tuloksia seuraavilla parametreilla: liikennöitsijä, reitti, suunta, aikaväli. ET-palvelun avulla välitetään reaaliaikaisia aikataulutietoja. Aikataulutiedot päivittyvät jatkuvasti sen mukaan onko kulkuneuvo edellä vai jäljessä aikataulua. Palvelun avulla pystytään myös välittämään muutoksia aikatauluihin, kun reiteillä tapahtuu peruutuksia, uusia reittejä lisätään tai joudutaan käyttämään poikkeavaa reittiä (esim. tietyö). Vastaanottava osapuoli voi rajata tuloksia seuraavilla parametreilla: liikennöitsijä, reitti, suunta. ST-palvelun avulla välitetään staattista pysäkkikohtaista tietoa saapumis- ja lähtöajoista. Vastaanottava osapuoli voi rajata tuloksia seuraavilla parametreilla: pysäkki, liikennöitsijä, reitti, suunta. SM-palvelun avulla välitetään reaaliaikaisia pysäkkikohtaisia tietoja saapumis- ja lähtöajoista. Vastaanottava osapuoli voi rajata tuloksia seuraavilla parametreilla: pysäkki, liikennöitsijä, reitti, suunta. VM-palvelun avulla välitetään kulkuneuvojen reaaliaikaista paikkatietoa ja siihen liittyvää muuta informaatiota. Vastaanottava osapuoli voi rajata tuloksia seuraavilla parametreilla: kulkuneuvotunniste, kulkuneuvoryhmä, reitti, suunta. CT-palvelun avulla välitetään staattista aikataulutietoa vaihtomahdollisuuksista eri liikennöintiyritysten välillä tietyssä vaihtopisteessä. Vastaanottava osapuoli voi rajata tuloksia seuraavilla parametreilla: vaihtopiste, reitti, suunta, kulkuneuvotunniste. CM-palvelun avulla välitetään reaaliaikaista aikataulutietoa vaihtomahdollisuuksista eri liikennöintiyritysten välillä tietyssä vaihtopisteessä. Mikäli yhteyksissä tapahtuu viivästyksiä, tämän palvelun avulla pystytään informoimaan toista liikennöintiyritystä. Näin ollen kyseinen liikennöintiyritys voi odottaa myöhässä saapuvan kulkuneuvon matkustajia. Vastaanottava osapuoli voi rajata tuloksia seuraavilla parametreilla: vaihtopiste, reitti, suunta, kulkuneuvotunniste. GM-palvelun avulla voidaan välittää informatiivisia viestejä. Voidaan esimerkiksi kertoa reittien peruuntumisesta, muutoksista

tai välittää muita viestejä liittyen matkustamiseen. Viestit jaetaan eri ryhmiin niiden sisällön mukaan (esim. virheet, varoitukset, muut viestit). [10]

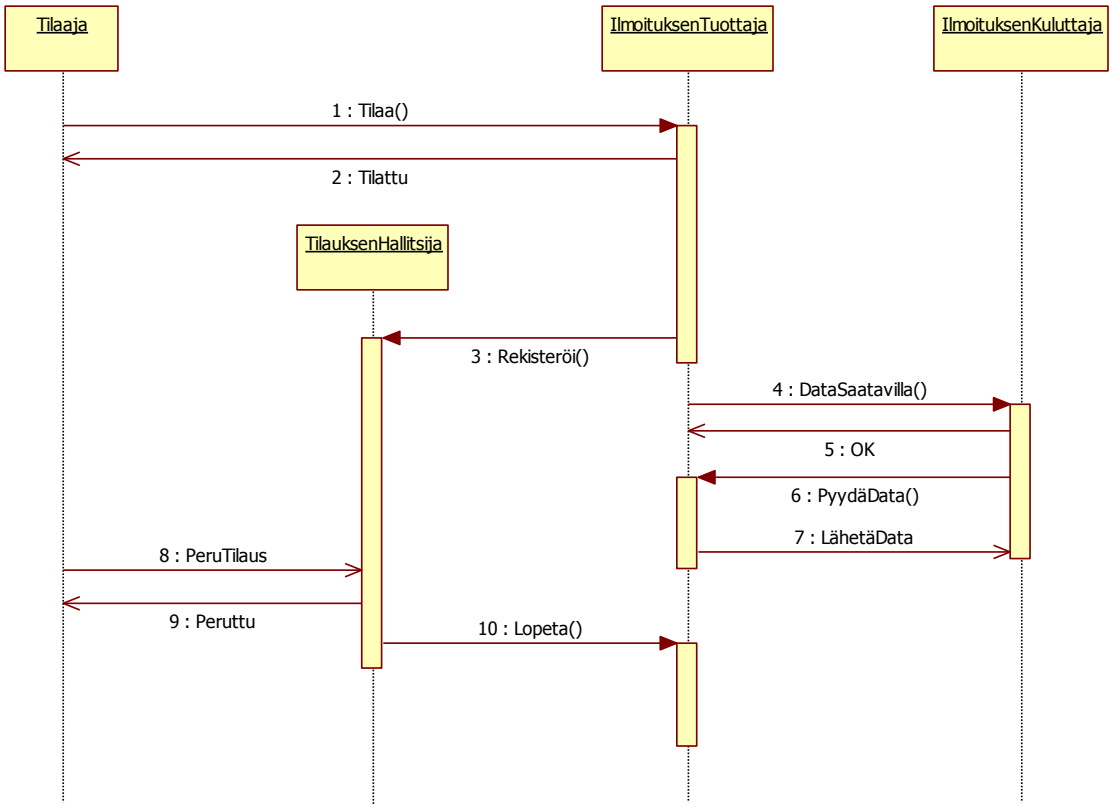
SIRI:n kommunikointiprotokolla mahdollistaa datan välityksen kolmella eri menetelmällä: Request/Response, Publish/Subscribe ja Publish/Subscribe With Fetched Delivery. Viestit kuvataan XML:ää käyttäen. Request/Response -menetelmässä tiedonpyytäjä lähettää pyynnön halutulle palvelulle, jonka jälkeen paluuviestinä saadaan haluttu data. Tämän menetelmän toiminta on esitetty kuvassa 3. Publish/Subscribe -menelmässä tiedonpyytäjä rekisteröityy halutun palvelun tilaajaksi ja tiedontoimittaja toimittaa datan tilaajalle heti, kun uutta tietoa on saatavilla. Tämän menetelmän toiminta on esitetty kuvassa 4. Publish/Subscribe With Fetched Delivery -menetelmä toimii kuin edellä mainittu Publish/Subscribe -menetelmä, mutta tiedontoimittaja ei lähetä dataa heti, kun se on saatavilla. Se ainoastaan informoi tiedonpyytäjää, että data on saatavilla. Tämän jälkeen tiedonpyytäjä voi pyytää kyseisen datan haluamallaan hetkellä. Tämän menetelmän toiminta on esitetty kuvassa 5. [10]



Kuva 3: Request/Response -menetelmä (kaavio suomennettu). [10]



Kuva 4: Publish/Subscribe -menetelmä (kaavio suomennettu). [10]



Kuva 5: Publish/Subscribe With Fetched Delivery -menetelmä (kaavio suomennettu). [10]

2.2.7 Vertailu

Luvuissa 2.2.1 – 2.2.6 esitetyt standardit voidaan jakaa eurooppalaisiin ja kansainvälisiin standardeihin. Eurooppalaisia ovat Transmodel, IFOPT, NeTEx ja SIRI. Kansainvälisiä ovat GTFS ja GTFS-R. Eurooppalaiset standardit ovat CEN-organisaation standardoimia standardeja, joista jokainen pohjautuu abstraktiin Transmodel-standardiin. Ne on suunniteltu keskenään yhteensopiviksi ja riippuvuudet niiden välillä on pyritty pitämään mahdollisimman pieninä. GTFS ja GTFS-R ovat niin kutsuttuja de-facto standardeja, eli niitä ei ole standardoitu minkään standardointiorganisaation toimesta, vaan ne ovat vakiintuneet käyttöön kansainvälisesti ajan kuluessa. Ne eivät perustu Transmodel-standardiin ja tästä johtuen ne eivät ole yhteensopivia CEN-standardien kanssa.

Staattisen informaation kuvaamiseen kilpailevia standardeja ovat NeTEx ja GTFS. Molemmilla voidaan kuvata julkisen liikenteen staattista informaatiota. NeTEx on huomattavasti laajempi kuin GTFS, koska se sisältää myös IFOPT- ja SIRI -standardit.

NeTEx tukee myös useampia kulkuneuvomuotoja. Tästä johtuen se on paljon monimutkaisempi standardi. NeTEx:in ja GTFS:n esitysmuodot eroavat toisistaan. NeTEx käyttää tiedon kuvaamiseen XML:ää, kun taas GTFS käyttää pakattuja CSV-tiedostoja. GTFS:ssä kaikki data joudutaan lähettämään kerralla. NeTEx:ssä ainoastaan tarvittavat tiedot joudutaan lähettämään. Tästä johtuen GTFS soveltuu ainoastaan palvelinohjelmistojen väliseen tiedonsiirtoon. NeTEx sen sijaan soveltuu tiedonsiirtoon palvelinohjelmistojen ja asiakasohjelmien välillä. NeTEx standardi ei ole tällä hetkellä vielä aivan valmis (osa 3 luonnosvaiheessa), joten sen käyttö informaation jakelussa ei ole vielä yleistä. Taulukossa 7 on vertailtu NeTEx:n ja GTFS:n toiminnallisuuksien keskeisimpiä eroja. Valinta NeTEx:in ja GTFS:n välillä ei ole itsestäänselvää. Tehdyn selvityksen perusteella (kts. luku 2.1.2) voidaan todeta, että GTFS on tällä hetkellä käytetyin standardi julkisen liikenteen staattisen informaation jakelussa. Se on myös paljon yksinkertaisempi. NeTEx:n käyttöä rajoittaa se, että sillä ei vielä pysty kuvaamaan lippujen hinnoittelua, koska standardin kehitys on siltä osin vielä kesken. GTFS on tällä hetkellä parempi vaihtoehto staattisen informaation kuvaamiseen, jos halutaan kuvata myös lippujen hinnoittelu. Muussa tapauksessa kannattaa ehdottomasti valita NeTEx, koska se on niin paljon laajempi kokonaisuus ja sen avulla pystytään myös välittämään informaatiota asiakasohjelmistoille.

Taulukko 7: NeTEx:n ja GTFS:n toiminnallisuuksien vertailu.

Toiminnallisuus	*NeTEx*	*GTFS*
Verkkotopologia (reitit, linjat, reittipisteet, pysäkit)	x	x
Aikataulut	x	x
Vaihtomahdollisuudet	x	x
Kulkuneuvojen erityisominaisuudet	x	
Lippujen hinnoittelu	x*	x
Kiinteät kohteet (IFOPT)	x	
(* ei vielä valmis)		

Reaaliaikaisen informaation kuvaamiseen kilpailevia standardeja ovat SIRI ja GTFS-R. Molemmilla voidaan kuvata julkisen liikenteen reaaliaikaista aikatauluinformaatiota ja siihen liittyvää muuta informaatiota. SIRI on hieman laajempi kuin GTFS-R. Se tarjoaa pysäkkikohtaiset aikataulut ja tiedot vaihtomahdollisuuksista, joita ei GTFS-R:stä

löydy. GTFS-R on GTFS-standardin laajennus, jonka takia se ei toimi itsenäisesti vaan vaatii toimiakseen GTFS-informaation. SIRI:n ja GTFS-R:n esitysmuodot eroavat toisistaan. SIRI käyttää tietojen kuvaamiseen XML:ää, kun taas GTFS-R käyttää Protocol Buffers -menetelmää. Molemmat standardit soveltuvat tiedonsiirtoon palvelinohjelmistojen ja asiakasohjelmistojen välillä. GTFS-R on pyritty pitämään yksinkertaisena käyttää. Se tarjoaa myös huomattavasti paremman suorituskyvyn tiedonsiirtoon kuin SIRI, koska Protocol Buffers -menetelmän ansiosta tietojen lukeminen ja kirjoittaminen on paljon nopeampaa kuin XML:ää käyttäen (kts. luku 2.3.4). GTFS-R:ssä on myös omat heikkoutensa. Protocol Buffers vaatii toimiakseen Protocol Buffers -kääntäjän jokaiselle eri ohjelmointikielelle. Lisäksi siirrettävä data ei ole tekstimuotoista, joten sen tarkastelu ongelmatilanteissa on paljon työläämpää kuin tekstimuotoisen informaation. Taulukossa 8 on vertailtu SIRI:n ja GTFS-R:n toiminnallisuuksien keskeisimpiä eroja. Tehdyn selvityksen perusteella (kts. luku 2.1.2) voidaan todeta, että SIRI on tällä hetkellä näistä kahdesta standardista käytetympi. Se on käytössä jopa Euroopan ulkopuolella New York:ssa Yhdysvalloissa. SIRI:n avulla voidaan välittää GTFS-R:ä laajempaa informaatiota, eikä sen toiminta ole riippuvainen muista standardeista. Lisäksi SIRI sisältyy kehitteillä olevaan NeTEx-standardiin. Edellä mainittujen ominaisuuksien perusteella reaaliaikaisen informaation kuvaamiseen kannattaa valita SIRI, mikäli välitettävän informaation laajuus ja sen yhteensopivuus muiden standardien kanssa on etusijalla. Mikäli informaation kuvaaminen halutaan pitää yksinkertaisena yhteensopivuuden kustannuksella, niin silloin kannattaa valita GTFS-R.

Taulukko 8: SIRI:n ja GTFS-R:n toiminnallisuuksien vertailu.

Toiminnallisuus	*SIRI*	*GTFS-R*
Päivitykset aikatauluun	x	x*
Pysäkkikohtainen aikataulu	x	
Kulkunevojen paikkatieto	x	x
Vaihtomahdollisuudet	x	
Häiriötiedotteet	x	x
(* vaatii tiedot GTFS:stä)		

Voidaan todeta, että olemassa olevat julkisen liikenteen staattisen ja reaaliaikaisen informaation kuvaamiseen tarkoitetut standardit ovat tällä hetkellä niin kutsutussa murrosvaiheessa. Uusia standardeja kehitellään ja vanhempien standardien hyviä puolia

otetaan niihin mukaan. Haasteena ovat eri liikennemuotojen erityispiirteiden asettamat vaatimukset. Joudutaan myös tasapainoilemaan standardin laajuuden, käytettävyyden ja nopeuden välillä. Myös suorituskykyasiat pitää huomioida standardeja kehitettäessä, erityisesti kun on kyse reaaliaikaisesta informaatiosta. Sovelluksia kehitettäessä kannattaa varautua siihen, että olemassa olevat standardit saattavat muuttua tai ne saatetaan korvata kokonaan uusilla standardeilla. Tämä on hyvin todennäköistä, koska tällä hetkellä ei ole olemassa kaiken kattavia maailmanlaajuisia standardeja julkisen liikenteen staattisen ja reaaliaikaisen informaation kuvaamiseen.

2.3 Yleisimmin käytetyt datan serialisointimenetelmät

Datan serialisointi on menetelmä, jossa haluttu rakenteellinen informaatio muutetaan sellaiseen muotoon, että eri ohjelmat pystyvät kommunikoimaan keskenään riippumatta millä ohjelmointikielellä ne on toteutettu ja millä alustalla niitä ajetaan. Kolme yleisimmin käytettyä julkisen liikenteen avoimen datan serialisointimenetelmää ovat JSON (**J**ava**S**cript **O**bject **N**otation), XML ja Protocol Buffers. Luvuissa 2.3.1 – 2.3.3 esitellään nämä serialisointimenetelmät ja lopuksi luvussa 2.3.4 vertaillaan niiden vahvuuksia ja heikkouksia.

2.3.1 JSON

JSON on kevyt tekstimuotoinen tiedonsiirtoformaatti, joka on koneellisesti helppo lukea ja muodostaa. Sen formaatti on myös helposti ihmisluettavissa. JSON on suoraan tuettuna JavaScript-ohjelmointikielessä, koska se sisältyy kieleen. Se ei kuitenkaan ole JavaScript:stä riippuvainen. JSON käyttää konventiota, joka on tuttu C-kieleen pohjautuvista ohjelmointikielistä, mutta se on silti ohjelmointikieliriippumaton. JSON soveltuu hyvin tiedonsiirtoon eri järjestelmien välillä. Alla oleva esimerkki esittää JSON:lla määriteltyä dataa. [29]

```
{"employees":[
   {"firstName":"Matti", "lastName":"Meikäläinen"}
]}
```

2.3.2 XML

XML on rakenteellinen merkkauskieli, jonka avulla laajoja tietomassoja pystytään jäsentämään selkeästi. Se on tekstimuotoista ja muistuttaa HTML-merkkauskieltä (**H**yper**t**ext **M**arkup **L**anguage). Sitä ei ole kuitenkaan suunniteltu web-sivustojen merkkauskieleksi vaan tiedonsiirtoon eri järjestelmien välillä. XML mahdollistaa sen, että eri ohjelmointikielillä toteutetut ohjelmat voivat helposti kommunikoida keskenään. XML on suunniteltu niin, että se on sekä koneluettavassa että ihmisluettavassa muodossa. Se ei sisällä ennalta määritettyjä elementtejä, vaan sitä käyttävä voi määritellä ne vapaasti itse. XML standardi tosin määrittelee dokumentille muutamia vaatimuksia, kuten esimerkiksi dokumentti saa sisältää ainoastaan yhden juurielementin ja jokaisella elementillä pitää olla aloitus- ja lopetusmerkki. Alla oleva esimerkki esittää XML:llä määriteltyä dataa. [30]

```
<employees>
   <employee>
      <firstName>Matti</firstName>
         <lastName>Meikäläinen</lastName>
   </employee>
</employees>
```

2.3.3 Protocol Buffers

Protocol Buffers on Googlen kehittämä rakenteisen datan serialisointimenetelmä, joka on kieli- ja alustariippumaton. Sen toiminta perustuu binääritiedoston kirjoittamiseen ja lukemiseen. Protocol Buffers:in avulla eri ohjelmat voivat kommunikoida keskenään riippumatta siitä millä alustalla niitä ajetaan ja millä ohjelmointikielellä ne on toteutettu. Sitä voidaan käyttää myös tiedon tallentamiseen paikallisesti. Protocol Buffer toimii niin, että ensimmäiseksi serialisoitavan datan rakenne pitää määritellä Protocol Buffers -rajapintamäärityskielen avulla proto-päätteiseen tiedostoon. Tämän jälkeen tiedosto käännetään Protocol Buffer -kääntäjällä, joka muodostaa tarvittavat luokat binääritiedoston kirjoittamiseen ja sen lukemiseen. Jokaiselle kielelle tarvitaan oma kääntäjä. Google tarjoaa Protocol Buffers -kääntäjät tällä hetkellä ainoastaan seuraaville ohjelmointikielille:

Java, C++ ja Python. Kääntäjät on julkaistu avoimen lähdekoodin lisenssillä, joten niitä on tehty myös muillekin ohjelmointikielille, kuten esimerkiksi Objective-C- ja C-kielelle [31]. Protocol Buffers mahdollistaa myös datarakenteen muokkaamisen ilman yhteensopivuuden rikkoutumista vanhempien versioiden kanssa. [28]

2.3.4 Vertailu

Luvuissa 2.3.1 – 2.3.3 esiteltiin kolme rakenteisen datan serialisointimenetelmää: JSON, XML ja Protocol Buffers. Näistä kahta ensimmäistä käytetään sekä staattisen että reaaliaikaisen informaation serialisointiin ja viimeistä ainoastaan reaaliaikaisen tiedon serialisointiin.

JSON:in ja XML:n suorituskykyä on tutkittu aiemmassa tutkimuksessa [32], jonka lisäksi toisessa aiemmassa tutkimuksessa on vertailtu näiden lisäksi myös Protocol Buffers -menetelmää [33]. Nämä tutkimukset osoittavat, että tekstipohjaisista serialisointimenetelmistä JSON on huomattavasti parempi kuin XML, koska sillä serialisoitu data mahtuu pienempään tilaan, se on nopeampi lukea ja kirjoittaa, ja se käyttää vähemmän resursseja. Protocol Buffers -menetelmällä data saadaan mahtumaan hieman pienempään tilaan, mutta suurin hyöty saavutetaan luku- ja kirjoitusnopeudessa. Kirjoitusnopeuden ero on noin kaksinkertainen JSON:in verrattuna ja lukunopeuden ero noin nelinkertainen. Protocol Buffers -menetelmällä on kuitenkin myös omat heikkoutensa: se ei ole ihmisluettavassa muodossa, ja sen käyttäminen vaatii kääntäjän jokaiselle eri kielelle. Ensiksi mainitusta syystä johtuen esimerkiksi vastaanotetun datan debugaaminen järjestelmien välillä on huomattavasti hankalampaa kuin tekstipohjaisia menetelmiä käyttäen. [32, 33]

Edellä mainittujen tutkimusten perusteella serialisointimenetelmäksi kannattaa valita JSON, mikäli halutaan varmistua siitä, että data on käyttökelpoista kaikilla ohjelmointikielillä alustasta riippumatta. Mikäli järjestelmältä vaaditaan parasta mahdollista suorituskykyä yhteensopivuuden kustannuksella, niin silloin menetelmäksi kannattaa valita Protocol Buffers.

3 JAKELUJÄRJESTELMÄN TOTEUTUS

Tässä luvussa kuvataan, miten pilotti-projektia varten käyttöönotettu jakelujärjestelmä toteutettiin ja miten projekti eteni. Luvussa 3.1 esitellään mitä ratkaisua ja valmiita ohjelmia jakelujärjestelmän toteutuksessa käytettiin ja miksi näihin päädyttiin. Luvussa 3.2 kuvataan käyttöönotettu jakelujärjestelmä. Luvussa 3.3 kerrotaan miten järjestelmän käyttöönotto toteutettiin. Luvussa 3.4 kerrotaan miten järjestelmän lokalisointi suomalaiselle käyttäjäkunnalle toteutettiin. Luvussa 3.5 esitellään miten käyttöönotetun jakelujärjestelmän ylläpito hoidetaan. Luvussa 3.6 esitellään järjestelmän käyttöönoton aikana ilmenneet ongelmat ja miten ne saatiin ratkaistua. Lopuksi luvussa 3.7 pohditaan mitä tuloksia pilotti-projektilla saavutettiin.

3.1 Järjestelmän valinta

Järjestelmän toteutus aloitettiin päättämällä miten järjestelmä tullaan toteuttamaan, pohjautuen kirjallisuuskatsauksessa ilmenneisiin asioihin. Todettiin, että jakelujärjestelmän kehittäminen alusta asti veisi niin paljon aikaa, että siihen ei kannata ryhtyä tämän projektin puitteissa. Sen sijaan havaittiin, että saatavilla on kaksi avoimeen lähdekoodiin perustuvaa jakelujärjestelmää: OneBusAway ja Navitia. Näistä OneBusAway osoittautui soveltuvimmaksi järjestelmäksi tähän tarkoitukseen, koska se sisältää palvelinohjelmiston julkisen liikenteen staattisen ja reaaliaikaisen informaation jakeluun sekä web- ja mobiili-asiakasohjelmistot. Navitia:lla on ainoastaan mahdollista jaella julkisen liikenteen staattista aikatauluinformaatiota, eikä se sisällä valmiita web- ja mobiili-ohjelmistoja.

OneBusAway:ssä on monia hyviä puolia. Se käyttää staattisen aikatauluinformaation tuontiin GTFS-standardia ja reaaliaikaisen informaation tuontiin GTFS-R- tai SIRI-standardia. Kirjallisuuskatsauksessa todettiin, että GTFS ja SIRI ovat tällä hetkellä parhaat mahdolliset standardit. GTFS-informaatiota käytetään myös Google Transit -palvelussa. Tästä johtuen datan toimittaja voi antaa saman informaation myös Googlen käyttöön, jolloin kyseisen kaupungin aikataulut löytyvät myös Google Transit -palvelusta. OneBusAway-projektissa on myös maailmanlaajuinen aluetietopalvelin, jonka avulla OneBusAway:n asiakasohjelmistot toimivat maailmanlaajuisesti, eli sama asiakasohjelma

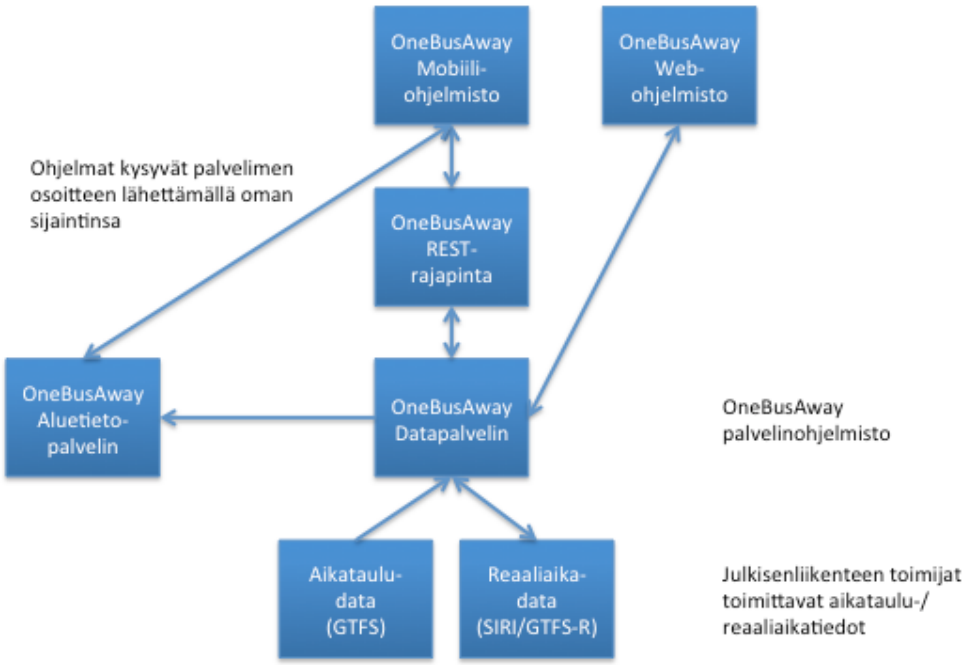
toimii jokaisella alueella, missä on käytössä OneBusAway:n palvelinohjelmisto. Valmiit asiakasohjelmistot madaltavat järjestelmän käyttöönoton kynnystä muissakin kunnissa kuin Lappeenrannassa. OneBusAway on käytössä Yhdysvalloissa muun muassa Tampassa, jossa se toimii hyvin noin 350 000 asukkaan kaupungissa [34]. Tästä voidaan päätellä, että se tulee toimimaan myös Lappeenrannassa ilman ongelmia, jossa asukkaita on noin 73 000 [35]. OneBusAway:ssä on eräs rajoite, josta johtuen sen asiakasohjelmistot eivät toimi muiden jakelujärjestelmien kanssa. Tämä johtuu siitä, että OneBusAway:n REST-rajapinta ei noudata mitään yleistä standardia.

OneBusAway projektiin perehtymällä havaittiin, että sillä on aktiivinen yhteisö, joka kehittää järjestelmää ja joilta saa tarvittaessa apua ongelmiin keskustelupalstan kautta noin 24 tunnin sisällä. OneBusAway on myös kustannustehokas ratkaisu, koska se on valmis palvelinjärjestelmä, joka voidaan replikoida kunnasta toiseen. Tällöin kustannuksia aiheuttaa vain järjestelmän käyttöönotto ja ylläpito, sekä sen lokalisointi. Avoimen lähdekoodin ansiosta ohjelmistosta ei myöskään peritä lisenssimaksuja.

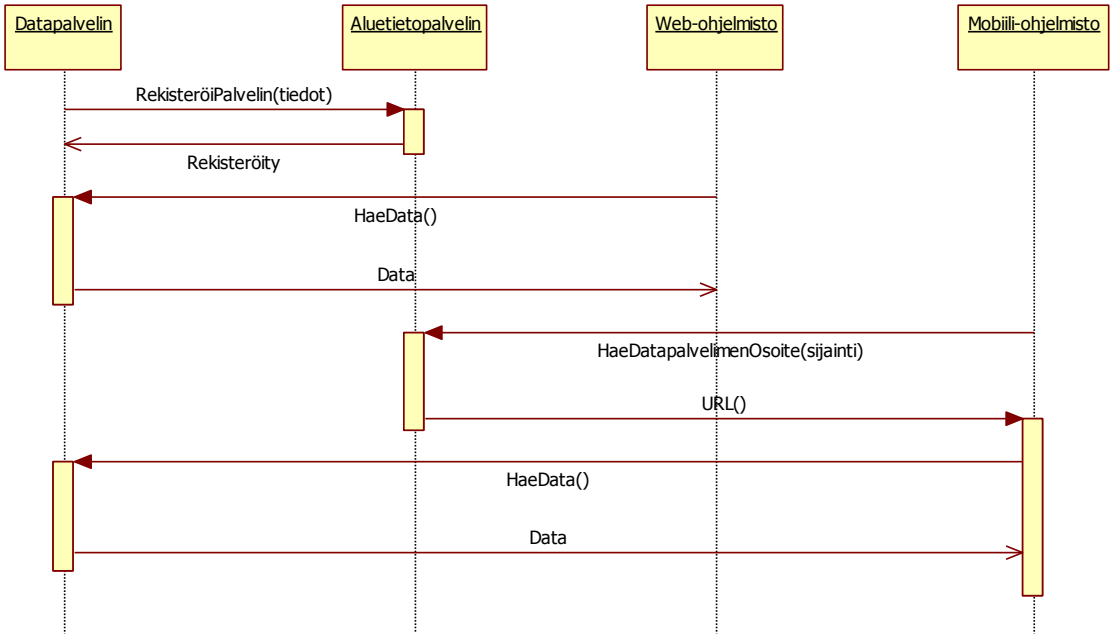
Edellä mainittujen ominaisuuksien perusteella voidaan todeta, että OneBusAway on tällä hetkellä monipuolisin ja kustannustehokkain avoimeen lähdekoodiin perustuva jakelujärjestelmä. Se valittiin näillä perusteilla projektissa käytettäväksi jakelujärjestelmäksi.

3.2 Järjestelmän kuvaus

Järjestelmänä projektissa käytetään OneBusAway-ohjelmistokokonaisuutta. Se koostuu datapalvelimesta, web- ja mobiili-ohjelmistoista. Palvelinohjelmisto varastoi ja jakelee julkisen liikenteen staattisen ja reaaliaikaisen informaation OneBusAway:n web- ja mobiili-ohjelmistoille sekä kolmannen osapuolten asiakasohjelmistoille. OneBusAway-projektissa on myös maailmanlaajuinen aluetietopalvelin, jonka avulla OneBusAway:n mobiiliohjelmat löytävät lähimmän datapalvelimen. Ohjelmistokokonaisuuden arkkitehtuuri on kuvattu kuvassa 6 ja ohjelmien välinen kommunikointi kuvassa 7. Luvuissa 3.2.1 – 3.2.4 esitellään tarkemmin järjestelmään kuuluvat komponentit.



Kuva 6: OneBusAway-ohjelmistokokonaisuuden arkkitehtuuri.



Kuva 7: OneBusAway-ohjelmistokokonaisuuden ohjelmien välinen kommunikointi.

3.2.1 Datapalvelin

Datapalvelin sisältää palvelinohjelmiston, joka varastoi tarvittavan julkisen liikenteen informaation ja jakelee sen web-ohjelmistolle ja asiakasohjelmistoille. Se on toteutettu Java-ohjelmointikielellä hyödyntäen J2EE-alustaa. Datapalvelin kattaa tietyn maantieteellisen alueen, ja tämä alue voidaan vapaasti määrittää GPS (**G**lobal **P**ositioning **S**ystem) -koordinaattien avulla. Palvelinohjelmiston toiminnot on jaettu erillisiin moduuleihin, jotta riippuvuudet niiden välillä pystytään pitämään mahdollisimman pieninä. Näin ollen ohjelmistoon pystytään helposti tekemään muutoksia ja laajennuksia sekä integroimaan muita järjestelmiä. Ohjelmistoa käyttävä voi vapaasti valita mitä ominaisuuksia halutaan käyttää. Yleisimmät ominaisuudet on esitetty taulukossa 9.

Taulukko 9: OneBusAway palvelinohjelmiston ominaisuudet.

Ominaisuus	*Kuvaus*	*Pakollisuus*
onebusaway-transit-data-federation-webapp	Aikatauludatan käsittely, matkan suunnittelu.	x
onebusaway-transit-data-federation-builder	Bundle-paketin luonti.	x
onebusaway-webapp	Web-sovellus	
onebusaway-api-webapp	JSON/XML-rajapinta informaation jakeluun mobiili-ohjelmistoille.	
onebusaway-sms-webapp	Aikataulutietojen kysely tekstiviestillä.	
onebusaway-phone-webapp	Aikataulutietojen kysely soittamalla.	

Tietokantana palvelinohjelmisto käyttää oletuksena sisäänrakennettua HSQLDB-tietokantaa. Palvelinohjelmisto tarjoaa myös mahdollisuuden käyttää jotakin ulkoista tietokantaa (esim. MySQL), mutta se vaatii erillisen konfiguroinnin. Julkisen liikenteen staattinen informaatio tuodaan järjestelmään käyttämällä GTFS-standardia. Tästä informaatiosta pitää muodostaa bundle-paketti ennen kuin informaatiota pystytään hyödyntämään. Bundle-paketti luodaan käyttäen onebusaway-transit-data-federation-builder -moduulia. Muodostettu bundle-paketti sisältää staattisen informaation muutettuna optimaaliseen tietorakenteeseen, jotta pystytään tarjoamaan paras mahdollinen suorituskyky. Reaaliaikainen informaatio tuodaan järjestelmään käyttäen GTFS-R- tai SIRI-standardia.

Informaation jakelu asiakasohjelmistoille tapahtuu REST-rajapinnan (OneBusAway REST API) kautta. Tiedonsiirtomuotona käytetään JSON- tai XML-formaattia, joka on käyttäjän valittavissa. Rajapintaa hyödyntävät OneBusAway:n viralliset mobiili-ohjelmat, mutta on myös mahdollista kehittää omia asiakasohjelmia, jotka käyttävät kyseistä rajapintaa. Rajapinnan tärkeimmät ominaisuudet on kuvattu taulukossa 10. Ominaisuuksien kutsutavat vaihtelevat, ja tarvittaessa jokaisen ominaisuuden kutsutapa ja käyttöesimerkit ovat löydettävissä OneBusAway:n kehittäjäsivustolta [36]. Esimerkiksi route-ominaisuutta voidaan kutsua liittämällä ominaisuuden nimi ja parametrit rajapinnan URL:n (**U**niform **R**esource **L**ocator) perään (esim. http://localhost:8080/api/where/route/5.json). Pisteen jälkeen määritetään palauttaako rajapinta vastauksen JSON- vai XML-muodossa.

Taulukko 10: OneBusAway REST-rajapinnan tärkeimmät ominaisuudet. [36]

Ominaisuus	***Kuvaus***
arrival-and-departure-for-stop	Pysäkille saapuvan tai lähtevän kulkuneuvon tiedot
arrivals-and-departures-for-stop	Pysäkin saapuvat ja lähtevät kulkuneuvot
plan-trip	Matkan suunnittelu pisteestä A pisteeseen B
route	Reitin tiedot
routes-for-location	Reittivaihtoehdot lähellä tiettyä sijaintia
schedule-for-stop	Pysäkin aikataulu
shape	Reitin koordinaatit
stop	Pysäkin tiedot
stops-for-location	Pysäkit lähellä tiettyä sijaintia.
stops-for-route	Reitin sisältämät pysäkit
trip-details	Laajennetut matkan tiedot
trip-for-vehicle	Kulkuneuvon tämänhetkisen matkan laajennetut tiedot
trip	Matkan tiedot
trips-for-location	Aktiiviset matkat lähellä tiettyä sijaintia
trips-for-route	Aktiiviset matkat tietyllä reitillä

3.2.2 Aluetietopalvelin

Aluetietopalvelin sisältää tietokannan, jonne tallennetaan kunkin datapalvelimen osoite ja sen kattaman maantieteellisen alueen koordinaatit. Sovellukset kyselevät tietyn datapalvelimen sijaintia REST-rajapinnan kautta lähettämällä päätelaitteensa paikkatiedon. Aluetietopalvelin analysoi paikkatiedon ja vertaa sitä tietokannassa oleviin arvoihin, ja mikäli kyseinen koordinaatti sijaitsee tietyn datapalvelimen alueella palauttaa

aluetietopalvelin tämän palvelimen osoitteen. Sovellukselle palautetaan virheilmoitus, jos kyseisellä alueella ei ole datapalvelinta. Aluetietopalvelimelle voidaan lisätä uusia datapalvelimia tekemällä pyyntö OneBusAway:n postituslistalle (onebusaway-developers) [37]. Datapalvelimen lisääminen on kaksivaiheinen prosessi. Ensimmäisessä vaiheessa palvelin pitää hyväksyä kokeelliseksi palvelimeksi OneBusAway:n kehittäjäkunnan toimesta. Seuraavassa vaiheessa palvelin pitää täyttää alla olevassa listassa kuvatut vaatimukset ja saada OneBusAway:n hallituksen hyväksyntä, jotta se voidaan hyväksyä OneBusAway:n tuotantopalvelimeksi. Tarkastuslista vaadituista toimenpiteistä ja vaatimuksista on kuvattu liitteessä 1. [37, 38]

- Pitää olla ollut käytössä vähintään 1kk, toimintavarmuudella 99.5 %.
- Pitää tarjota riittävän kattavasti reaaliaikaista dataa kyseiseltä alueelta.
- Reaaliaikatiedon paikkansapitävyys pitää olla varmistettu.
- Alkuperäisten OneBusAway mobiili-ohjelmien (Android, iOS, Windows Phone) pitää toimia palvelimen kanssa.
- Pitää tukea OneBusAway:n alkuperäistä rajapintaa (onebusaway-api-webapp).
- Palvelimen ylläpitäjän pitää taata, että aikatauluinformaatio pysyy ajan tasalla.

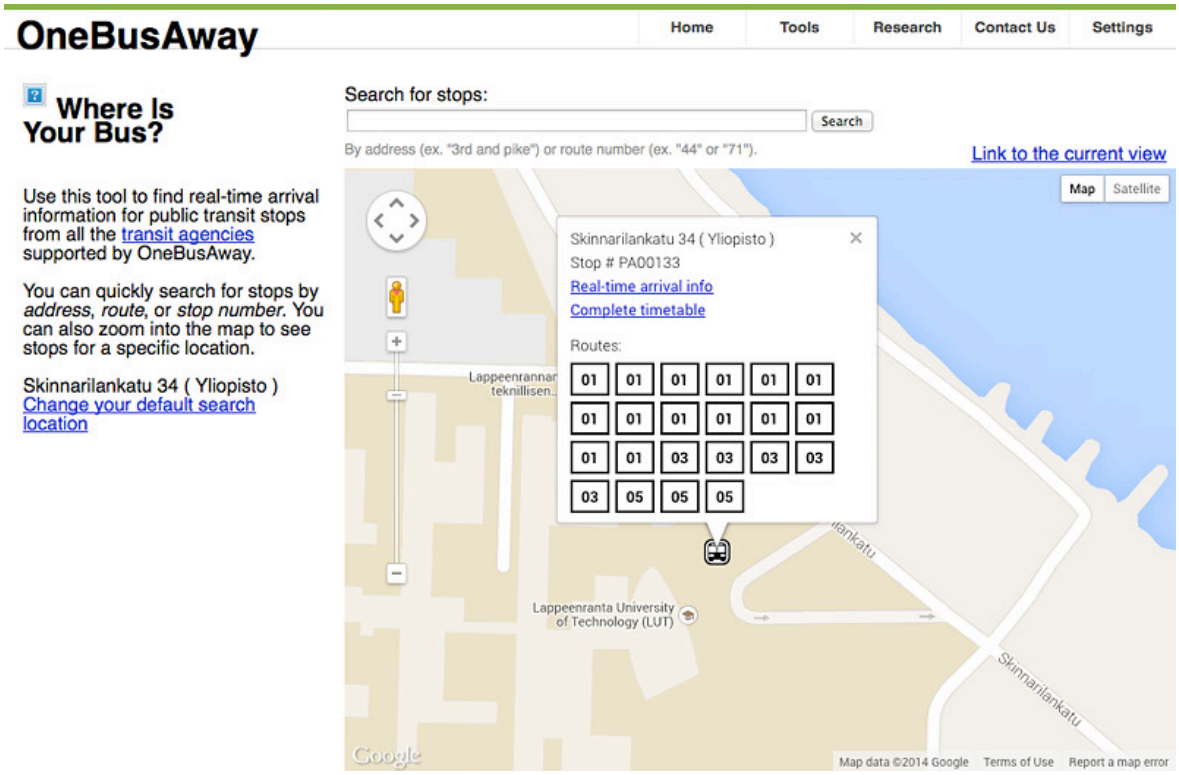
OneBusAway:n hallituksella on oikeus poistaa datapalvelimen tiedot aluetietopalvelimelta, jos sen toiminnassa ilmenee ongelmia ja niitä ei ole saatu korjattua viikon sisällä siitä, kun ongelma raportoitiin. [37]

3.2.3 Web-ohjelmisto

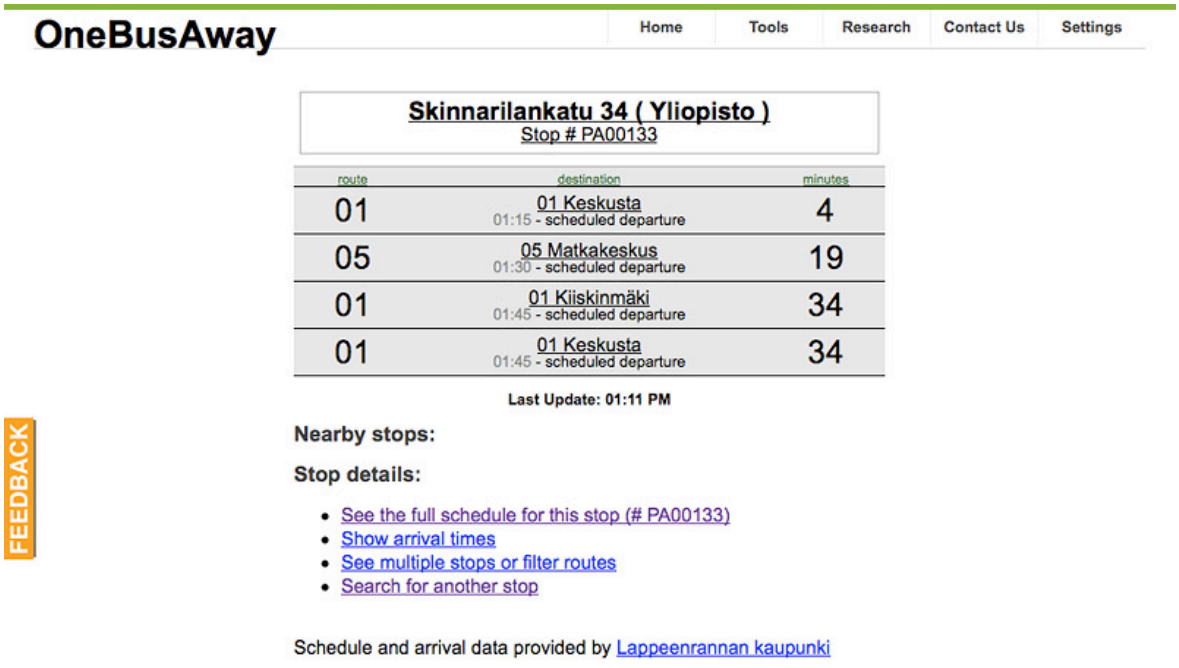
Web-ohjelmisto on toteutettu käyttämällä Apache Struts -framework:iä, JSP:tä (**J**ava**S**erver **P**ages), GWT:tä (**G**oogle **W**eb **T**oolkit) ja Google Maps API:a. Apache Struts:in avulla saapuvat web-pyynnöt ohjataan Javan tapahtumakäsittelijöille ja tämän jälkeen oikealle JSP-sivulle, joka muodostaa käyttäjän näkemän sivun. GWT:n avulla Java-koodi käännetään optimoiduksi JavaScript-koodiksi, jotta pystytään tarjoamaan Javaa parempi suorituskyky. Google Maps API:a hyödynnetään web-ohjelmiston karttanäkymässä. [39]

Web-ohjelmisto sisältää karttanäkymän sekä pysäkkikohtaisen kulkuneuvojen

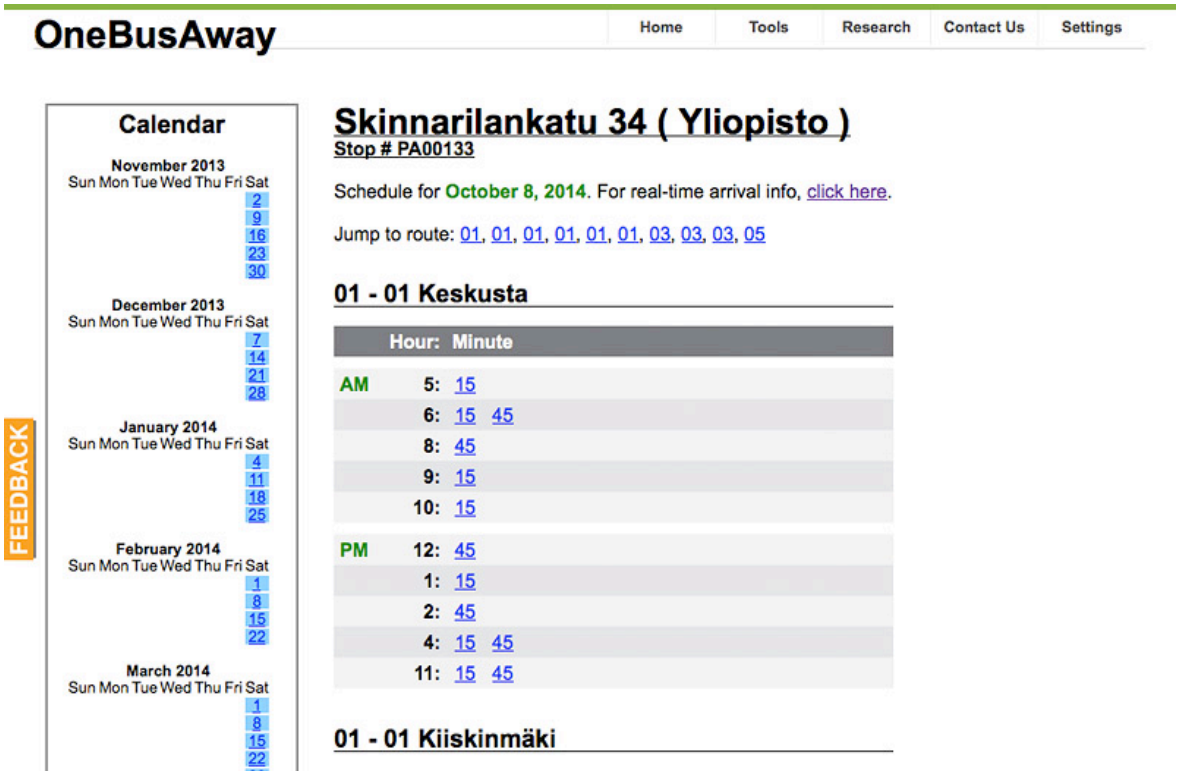
saapumisaika- ja aikataulunäkymät. Ne on esitetty kuvissa 8, 9 ja 10. Karttanäkymässä käyttäjä voi tarkastella tietyllä alueella olevia pysäkkejä sekä tietyn reitin kulkua kartalla ja sen sisältämiä pysäkkejä. Tiettyä pysäkkiä klikkaamalla aukeaa puhekuplan muotoinen ikkuna, jossa esitetään pysäkin tiedot ja sitä käyttävien linjojen numerot. Ikkunassa näkyy saman linjan numero useampaan kertaan, jos kyseinen linja liikennöi samalla numerolla eri reittejä. Ikkuna sisältää myös linkit pysäkin reaaliaikaiseen saapumisaikanäkymään ja aikataulunäkymään. Saapumisaikanäkymä sisältää tiedot pysäkille saapuvista kulkuneuvoista aikajärjestyksessä. Näkymässä esitetään kunkin saapuvan kulkuneuvon numero, määränpää, saapumisaikaennuste minuuteissa sekä onko kulkuneuvo saapumassa pysäkille aikataulun mukaisesti vai edellä tai myöhässä aikataulua. Aikataulunäkymä sisältää kaikkien pysäkkiä käyttävien kulkuneuvojen täydelliset aikataulut niiden saapumisajasta pysäkille. Näkymän ylälaidasta voidaan valita haluttu reitti ja vasemmassa laidassa olevasta kalenterista haluttu päivä. Kalenteri on oletuksena sen päivämäärän kohdalla, jolloin aikatauluinformaation voimassaolo alkaa. Web-ohjelmisto sisältää myös iPhone:lle optimoidun käyttöliittymän sekä tekstimuotoisen käyttöliittymän muille mobiili-laitteille. Lisäksi ohjelmisto sisältää myös tekstiviesti- ja puhelinrajapinnat saapumisaikojen kyselyyn. Nämä ominaisuudet ovat konfiguroitavissa, mutta se vaatii yhteistyötä puhelinoperaattoreiden kanssa. [40]



Kuva 8: OneBusAway web-ohjelmiston karttanäkymä.



Kuva 9: OneBusAway web-ohjelmiston pysäkkikohtainen saapumisaikanäkymä.



Kuva 10: OneBusAway web-ohjelmiston aikataulunäkymä.

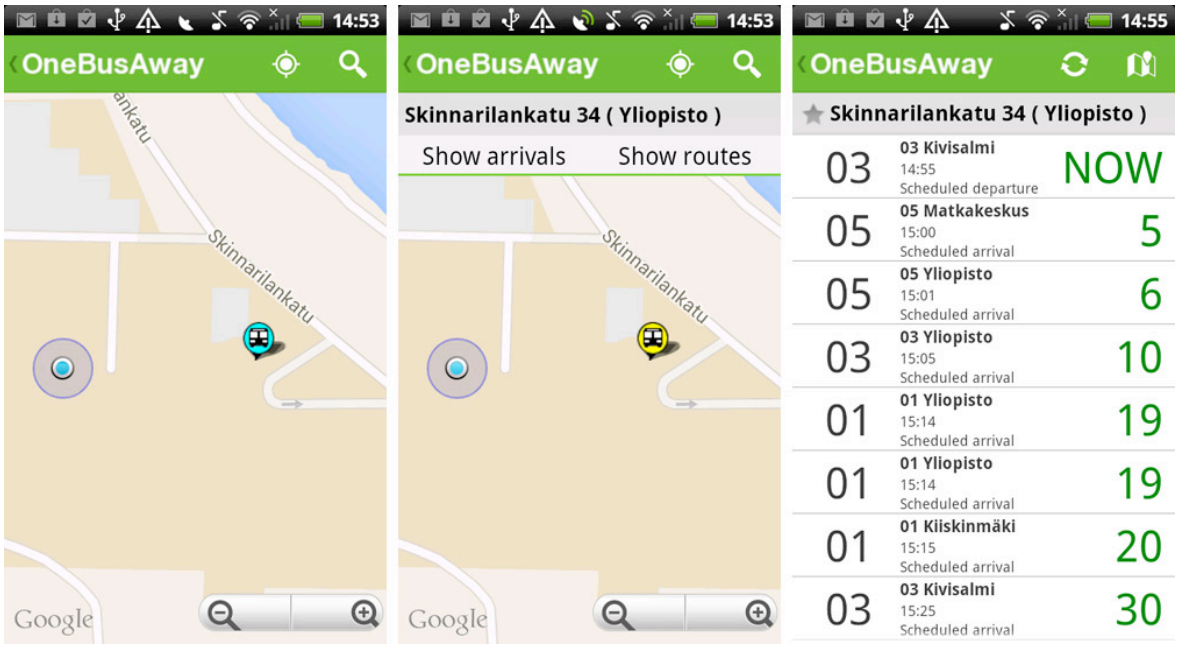
3.2.4 Mobiili-ohjelmistot

Mobiili-ohjelmistot on toteutettu Android-, iOS- ja Windows Phone -alustoille. Ne hyödyntävät OneBusAway:n aluetietopalvelinta, jonka avulla ne osaavat ottaa yhteyttä oikeaan datapalvelimeen. Windows Phone -versio toimii vain OneBusAway:n tuotantokäytössä olevilla datapalvelimilla. Android- ja iOS-versiot toimivat myös OneBusAway:n kokeellisilla palvelimilla. Ne pystyvät myös hyödyntämään sellaisia palvelimia, jotka eivät ole rekisteröity OneBusAway:n aluetietopalvelimelle. Eri alustojen mobiili-ohjelmistot eroavat ominaisuuksiltaan hieman toisistaan. Nämä ominaisuudet ja niiden erot eri alustojen välillä on esitetty taulukossa 11.

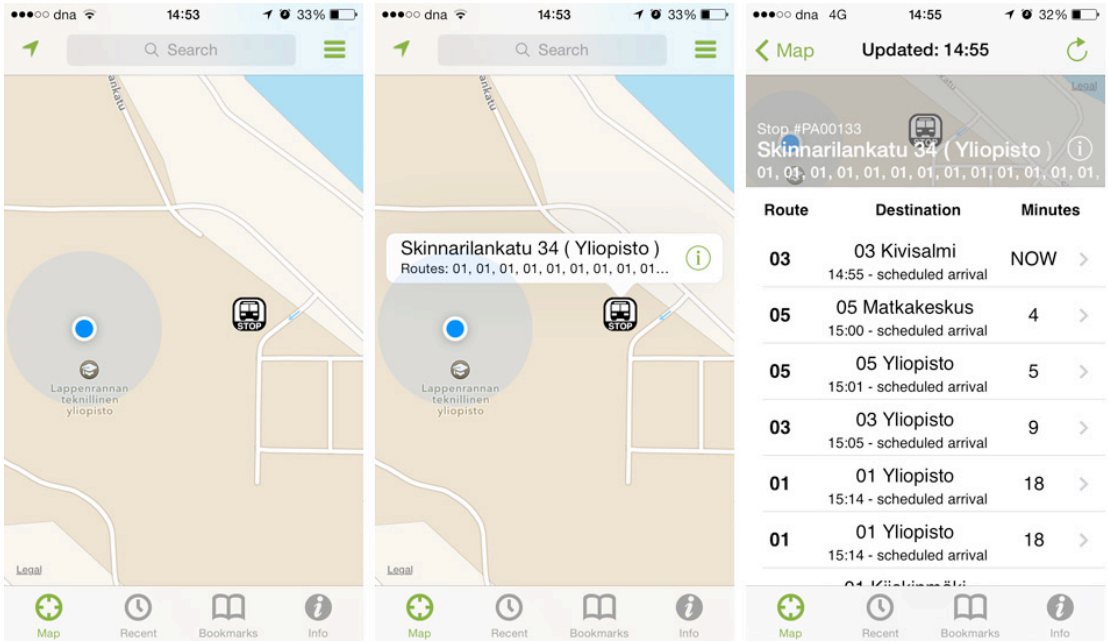
Taulukko 11: OneBusAway mobiili-ohjelmien ominaisuudet ja niiden erot eri alustojen välillä.

Ominaisuus	*Android*	*iOS*	*Windows Phone*
Reaaliaikaiset saapumisaikaennusteet	x	x	x
Reittien ja pysäkkien esittäminen kartalla	x	x	x
Lähimpien pysäkkien näyttäminen laitteen sijainnin mukaan	x	x	x
Häiriöilmoitukset	x	x	
Matkaan tai pysäkkiin liittyvän ongelman raportointi	x	x	
Pysäkkien kirjanmerkit	x	x	x
Pysäkkihaku reitin, osoitteen tai pysäkkinumeron perusteella	x	x	x
Manuaalinen palvelimen valinta ohjelman käyttöliittymästä	x	x	
Saapuvan kulkuneuvon muistutus	x		

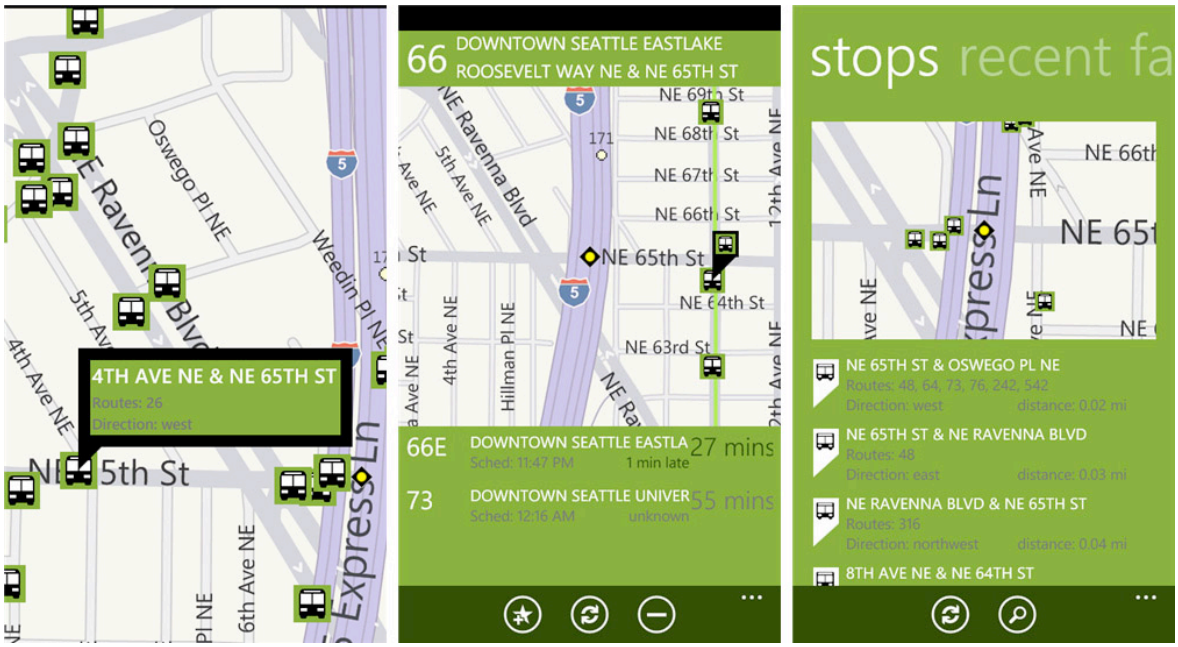
Ohjelmistojen käyttöliittymien välillä on eroja, mutta pääpiirteittäin ne näyttävät samanlaisilta. Isoimmat erot aiheutuvat kunkin alustan ominaispiirteistä (napit, tekstikentät, yms.). Eri alustojen käyttöliittymät on esitetty kuvissa 11, 12 ja 13. Ohjelmistot sisältävät karttanäkymän ja pysäkkikohtaisen kulkuneuvojen saapumisaikanäkymän. Näkymät toimivat samalla tavalla kuin web-ohjelmassa ja sisältävät samat ominaisuudet. Tosin pysäkkikohtaiseen aikatauluun ei pääse käsiksi suoraan mobiili-ohjelmasta, mutta ohjelmassa olevaa linkkiä klikkaamalla aukeaa web-selain, jonka kautta aikataulutietoja pääsee tarkastelemaan. Mobiili-ohjelmistoissa on lisäksi mahdollisuus tarkastella voimassa olevia häiriöilmoituksia sekä ilmoittaa tiettyyn matkaan tai pysäkkiin liittyvästä ongelmasta. Ohjelmistot tallentavat historiatietoa viimeksi selatuista pysäkeistä. Halutut pysäkit on mahdollista tallentaa kirjanmerkkeihin. Historiatiedon ja kirjanmerkkien avulla pääsee nopeasti käsiksi useasti käytettyjen pysäkkien tietoihin. Ohjelmistot sisältävät myös hakutoiminnon, jonka avulla pysäkkejä pystyy hakemaan reittinumeron, osoitteen tai pysäkin numeron perusteella. Lisäksi tietylle saapuvalle kulkuneuvolle on mahdollista asettaa muistutus, mutta tämä ominaisuus on toteutettu ainoastaan ohjelmiston Android-versioon.



Kuva 11: OneBusAway mobiili-ohjelmiston Android-version käyttöliittymä.



Kuva 12: OneBusAway mobiili-ohjelmiston iOS-version käyttöliittymä.



Kuva 13: OneBusAway mobiili-ohjelmiston Windows Phone -version käyttöliittymä. [41]

3.3 Järjestelmän käyttöönotto

Järjestelmän käyttöönotto aloitettiin asentamalla OneBusAway-palvelinohjelmisto yliopistolle projektia varten käyttöönotettuun Linux-palvelimeen. Ohjelmisto oli mahdollista asentaa käyttäen Quick-Start-ohjelmistopakettia tai asentaa se manuaalisesti. Ensimmäiseksi ohjelmiston asentamista testattiin käyttäen Quick-Start-ohjelmistopakettia, koska se oli helpoin ja nopein tapa ottaa ohjelmisto käyttöön. Näin ollen pystyttiin nopeasti varmistumaan siitä, että ohjelmisto toimii ja täyttää projektin vaatimukset. OneBusAway:n kehittäjäsivustolta löytyi kattavat asennusohjeet, jonka avulla Quick-Start-ohjelmistopaketin asennus sujuin ilman ongelmia noin 10 minuutissa. Asennusohjeet on kuvattu liitteessä 2. Quick-Start-ohjelmistopaketti oli suunniteltu niin, että asennusprosessi olisi mahdollisimman yksinkertainen. Ainoa esivaatimus oli Java, josta piti asentaa JDK 1.6 tai uudempi versio. Tässä projektissa käytettiin JDK 1.6:sta, koska se toimi tällä hetkellä parhaiten ohjelmiston kanssa. Quick-Start-ohjelmistopaketista käytettiin versiota 1.1.11. Ohjelmistopaketista oli saatavilla kaksi erilaista war-pakettia. Nämä paketit ja niiden sisältö on kuvattu taulukossa 12. Näistä paketeista valittiin onebusaway-quickstart-assembly-webapp.war, koska se sisälsi kaikki tarvittavat komponentit (palvelinohjelmisto, web-ohjelmisto, REST-rajapinta). GTFS-aikatauludatasta piti luoda bundle-paketti, jotta

aikatauludataa pystyttiin hyödyntämään ohjelmistossa. Paketti luotiin komentoriviltä kutsumalla Quick-Start-ohjelmistopakettia ja antamalla sille tarvittavat parametrit. Paketin luonnin jälkeen ohjelmisto käynnistettiin antamalla parametrina luodun bundle-paketin polku. Havaittiin, että palvelinohjelmisto, web-ohjelmisto ja REST-rajapinta toimivat moitteettomasti.

Taulukko 12: OneBusAway Quick-Start-asennuksen ohjelmistopaketit.

Paketti	*Sisältö*
onebusaway-quickstart-assembly-webapp.war	Palvelinohjelmisto, web-ohjelmisto, REST-rajapinta
onebusaway-quickstart-assembly-api-webapp.war	Palvelinohjelmisto, REST-rajapinta

Seuraavaksi ohjelmiston asennusta testattiin manuaalisesti. Tähän vaihtoehtoon ei kehittäjäsivustolta löytynyt yksiselitteistä asennusohjetta, jonka avulla ohjelmiston asennus olisi onnistunut ilman ongelmia. Tämän takia ohjeita jouduttiin kyselemään OneBusAway:n kehittäjiltä postituslistan kautta. Lopulta ohjelmisto saatiin asennettua kehittäjien antamien neuvojen avulla. Kehittäjät lupasivat parantaa ohjeita tulevaisuudessa. Ongelmana oli se, että ei ollut olemassa asennusohjetta, jossa olisi opastettu asennus alusta loppuun asti askel askeleelta, vaan eri moduulien asennusohjeet oli kuvattu eri paikoissa ja osa niistä oli vanhentunut. Lisäksi asennusohjeissa oletettiin, että GitHub:in toimintatapa ja avoimen lähdekoodin ohjelmistojen kehitys on entuudestaan tuttua. Ohjelmiston asennukseen ja konfigurointiin meni aikaa noin 2 tuntia, kun oli selvillä miten se suoritetaan. Asennusohjeet on kuvattu liitteessä 3. Manuaalinen asennus oli huomattavasti monimutkaisempi kuin Quick-Start-asennus, koska kaikki jouduttiin tekemään itse. Lisäksi manuaalinen asennus asetti myös paljon enemmän esivaatimuksia. Vaatimuksena oli Java, Tomcat, Git, Maven ja XWiki ohjelmistot. Tomcat on web-palvelinohjelmisto, jonka avulla pystytään ajamaan Javalla toteutettuja web-sovelluksia. Tomcat:stä käytettiin versiota 6. Git on versionhallintaohjelma, jonka avulla pystytään hakemaan lähdekoodit esimerkiksi GitHub:sta. Git:stä käytettiin versiota 1.7.9.5. Maven on automaatiotyökalu, jonka avulla pystytään automatisoimaan tehtävät (kääntäminen, testien suorittaminen, ohjelmistopaketin luominen, yms.), joita ohjelmiston kehitysprosessissa tarvitaan. Maven:sta käytettiin versiota 3.0.4. XWiki on sisällönhallintaohjelma, jota OneBusAway:n web-ohjelmisto käyttää. Sen avulla pystytään päivittämään web-ohjelman sisältöä ja

lokalisoimaan se halutulle kielelle. XWiki:stä käytettiin versiota 5.4.4. Manuaalisessa asennuksessa OneBusAway-ohjelmiston lähdekoodi jouduttiin lataamaan GitHub:sta, jonka jälkeen lähdekoodi käännettiin Maven:in avulla. Versioksi valittiin 1.1.11, koska se oli asennushetkellä uusin vakaa versio. Kääntämisen lopputuloksena syntyi kolme erilaista war-pakettia. Nämä paketit ja niiden sisältö on kuvattu taulukossa 13. Paketeista valittiin onebusaway-combined-webapp-1.1.11-full.war, koska se sisälsi kaikki tarvittavat komponentit (palvelinohjelmisto, web-ohjelmisto, REST-rajapinta). Valittu war-paketti asennettiin Tomcat:iin, ja ohjelman konfigurointi-tiedostoon (data-sources.xml) tehtiin tarvittavat määritykset (käytettävä portti, bundle-paketin sijainti, yms.). Bundle-paketin luontiin käytettiin lähdekoodista löytyvää onebusaway-transit-data-federation-builder -moduulia. Seuraavaksi XWiki asennettiin ja sen konfigurointi-tiedostoon tehtiin tarvittavat muutokset, jotta OneBusAway:n kehittäjäsivustolta ladattu web-ohjelmiston esimerkkisisältö saatiin tuotua järjestelmään. Tämän jälkeen esimerkkisisältö editoitiin halutun mukaiseksi XWiki:n avulla, ja lopuksi web-ohjelmiston konfigurointi-tiedostoon tehtiin tarvittavat muutokset, jotta XWiki saatiin integroitua ohjelmistoon. Asennuksen jälkeen havaittiin, että palvelinohjelmisto, XWiki ja REST-rajapinta toimivat moitteettomasti, mutta web-ohjelmistossa esiintyi muutamia ongelmia. Nämä ongelmat on esitetty luvussa 3.6.

Taulukko 13: OneBusAway manuaalisen asennuksen ohjelmistopaketit.

Paketti	*Sisältö*
onebusaway-combined-webapp-1.1.11-api-only.war	Palvelinohjelmisto, REST-rajapinta
onebusaway-combined-webapp-1.1.11-full.war	Palvelinohjelmisto, web-ohjelmisto, REST-rajapinta
onebusaway-combined-webapp-1.1.11-ui-only.war	Palvelinohjelmisto, web-ohjelmisto

Quick-Start- ja manuaalisen asennuksen välillä oli paljon eroja. Quick-Start-asennuspaketti oli suunniteltu niin, että palvelinohjelmiston käyttöönotto olisi mahdollisimman helppoa ja nopeaa. Tästä johtuen Quick-Start-ohjelmistopaketti sisälsi valmiiksi käännetyn ohjelmiston, joka oli myös konfiguroitu valmiiksi. Manuaalisessa asennuksessa jouduttiin asentamaan kaikki järjestelmän vaatimat ohjelmat, kääntämään lähdekoodi itse, sekä konfiguroimaan järjestelmä. Quick-Start-asennus sujuikin huomattavasti manuaalista asennusta nopeammin. Quick-Start-ohjelmistopaketti asetti kuitenkin seuraavat rajoitukset

ohjelmiston toiminnalle: ohjelmiston asetuksia ei pysty muuttamaan, web-ohjelmassa ei ole admin-tunnuksia ja web-ohjelman sisältöä ei pysty muuttamaan. Näistä rajoituksista johtuen Quick-Start-ohjelmistopaketti ei sovellu tuotantokäyttöön, koska ohjelmiston asetuksia ja sen sisältöä ei pysty muuttamaan. Mikäli ohjelmisto halutaan ottaa tuotantokäyttöön, joudutaan se asentamaan käyttäen manuaalista asennusta. Näin ollen tässä projektissa päädyttiin käyttämään manuaalisesti asennettua versiota ohjelmistosta.

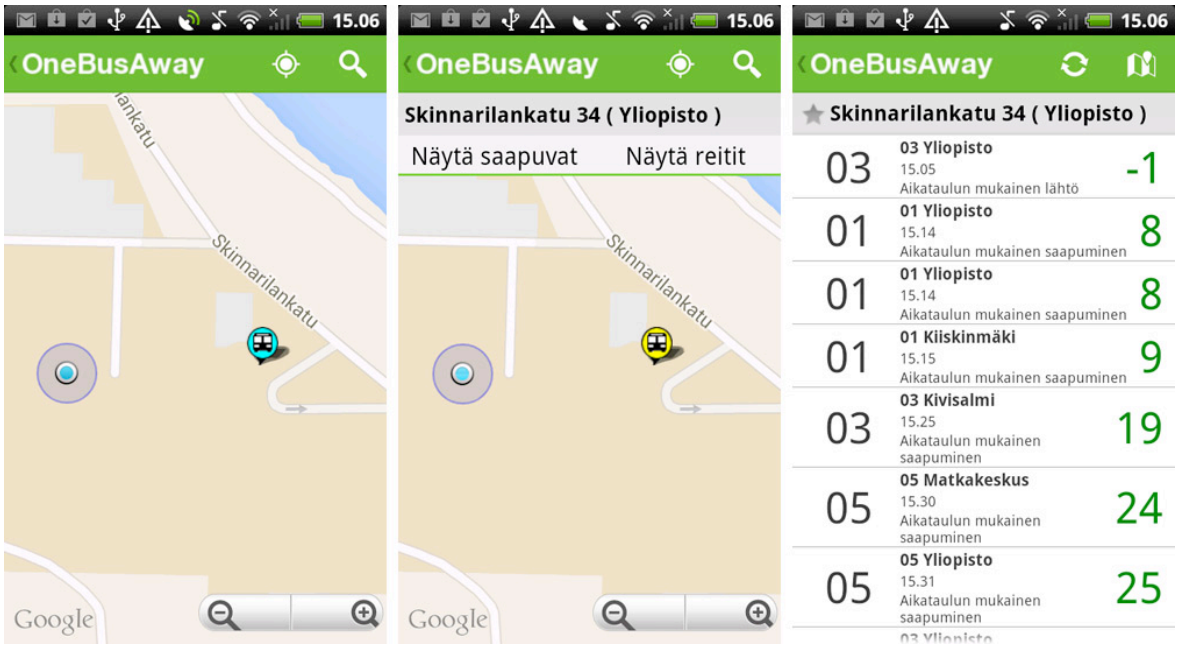
Seuraavaksi tutustuttiin OneBusAway:n mobiili-ohjelmistoihin, jotka ovat toteutettu Android-, iOS- ja Windows Phone -alustoille. Näistä jokaisen alustan ohjelmisto toimi moitteettomasti OneBusAway:n virallisilla tuotantopalvelimilla. Android- ja iOS-versioissa oli mahdollista myös käyttää OneBusAway:n kokeellisia palvelimia sekä palvelimia, jotka ei ole rekisteröity OneBusAway:n aluetietopalvelimelle. Rekisteröimättömiin palvelimiin saatiin yhteys määrittämällä palvelimen URL-osoite manuaalisesti mobiili-ohjelman asetuksista. Manuaalista osoitteen syöttöä tarvittiin, koska yliopiston testipalvelinta ei ole rekisteröity OneBusAway:n aluetietopalvelimelle. Tästä johtuen palvelinohjelmistoa pystyttiin testaamaan ainoastaan Android- ja iOS-alustoilla. Ensimmäiseksi kokeiltiin Android-ohjelmiston toimivuutta. Havaittiin, että ohjelmistossa on bugi, koska manuaalista URL-osoitteen syöttämistä käyttäen ohjelmisto pystyy ottamaan yhteyttä ainoastaan palvelimeen, joka vastaa portista 80. Tämän seurauksena ohjelmistoa ei saatu toimimaan yliopiston testipalvelimen kanssa, koska se käytti porttia 8080. Bugi raportoitiin OneBusAway:n kehittäjille, jotka korjasivat sen noin kahdessa viikossa. Ohjelma saatiin toimimaan yliopiston testipalvelimen kanssa, kun tehty korjaus bugiin ladattiin GitHub:sta. Tällä hetkellä ohjelmistoon on saatavilla päivitys Android Market:sta, joka korjaa kyseisen ongelman. Seuraavaksi kokeiltiin iOS-version toimivuutta. Havaittiin, että manuaalinen URL-osoitteen syöttäminen kaataa ohjelman. Bugi raportoitiin OneBusAway:n kehittäjille, jotka korjasivat sen alle kahdessa viikossa. Muutoksien toimivuutta pystyttiin testaamaan ohjelmiston beta-versiolla. Beta-versio ladattiin TestFlight-palvelun avulla. Palvelun käyttö kuitenkin vaati käyttöoikeuksien anomista OneBusAway:n kehittäjäkunnalta. Käyttöoikeudet palveluun anottiin ja ne aktivoituivat muutaman tunnin kuluessa anomuksesta. Tämän jälkeen ohjelman beta-versio ladattiin TestFlight:sta. Havaittiin, että tehdyt muutokset korjaavat ongelman ja ohjelma saatiin toimimaan yliopiston testipalvelimen kanssa. Tällä hetkellä ohjelmistoon on

saatavilla päivitys App Store:sta, joka korjaa kyseisen ongelman.

OneBusAway-järjestelmän testaamisessa käytettiin staattisena ja reaaliaikaisena aikatauluinformaationa aluksi Tampereen tarjoamaa GTFS- ja SIRI-informaatiota, koska Lappeenrannassa näitä informaatioita ei ollut projektin alkaessa vielä saatavilla. Projektin aikana Lappeenrannan kaupungin kanssa käytiin neuvotteluja GTFS-informaation toimittamisesta. Datan toimittaja lupautui muodostamaan kyseisen informaation ja se saatiin käyttöön muutaman kuukauden päästä. Lisäksi neuvoteltiin myös reaaliaikaisen informaation toimittamisesta. Selvisi, että Lappeenrannan paikallisliikenteen busseissa ei ole GPS-paikantimia. Tästä johtuen reaaliaikaista informaatiota ei ollut projektin aikana Lappeenrannassa saatavilla, joten testasimme järjestelmää reaaliaikaisen informaation osalta ainoastaan Tampereen SIRI-informaatiota hyödyntäen. OneBusAway-ohjelmisto saatiin toimimaan staattisen informaation osalta sekä Tampereen että Lappeenrannan GTFS-informaatioilla. Lappeenrannan GTFS-informaatiossa oli aluksi virheitä. Virheellisiä tietoja esiintyi liikennöitsijä- ja aikavyöhyketiedoissa, pysäkkikohtaisissa saapumis- ja lähtöajoissa sekä puolenyön ylittävissä matkoissa. Virheitä havaittiin olevan 398 kappaletta 58657 rivin joukossa. Ne raportoitiin Lappeenrannan kaupungille, ja virheet tullaan korjaamaan, mutta tarkempaa aikataulua korjauksien valmistumisesta ei ole tällä hetkellä tiedossa. Virheet korjattiin itse, jotta järjestelmää päästiin testaamaan nopeammin. Liikennöitsijä- ja aikavyöhyketiedot korjattiin käsin. Lähtö- ja saapumisaikojen korjaamiseen kehitettiin pieni apuohjelma, joka korvasi virheelliset ajat viimeisellä paikkansa pitävällä ajalla. Tästä johtuen joidenkin pysäkkien lähtö- ja saapumisajat eivät ole oikeita, mutta järjestelmää pystytään testaamaan näillä ajoilla, koska ne eivät aiheuta enää ongelmia. OneBusAway-ohjelmistoa ei saatu toimimaan Tampereen reaaliaikaisella SIRI-informaatiolla, koska GTFS- ja SIRI-informaatioiden väliltä puuttui yksiselitteinen linkitys. Lisäksi OneBusAway käyttää tiedonsiirtomenetelmänä SIRI:n Publish/Subscribe-menetelmää, joka ei ole Tampereella tuettuna. OneBusAway:n käyttämä menetelmä voidaan kyllä vaihtaa, mutta tätä ei pidetty aiheellisena tällä erää. Liitteessä 4 on kuvattu SIRI:n käyttöönotto ja sen asettamat vaatimukset.

3.4 Järjestelmän lokalisointi

Järjestelmän lokalisointi suomalaiselle käyttäjäkunnalle aloitettiin OneBusAway:n web-ohjelmasta. Asiaa tiedusteltiin OneBusAway:n postituslistalta ja tultiin siihen tulokseen, että web-ohjelman lokalisointi on aikaa vievä operaatio ja se joudutaan tekemään kahteen eri paikkaan. XWiki:ssä oleva sisältö voidaan lokalisoida helposti, mutta osa web-ohjelman lokalisoinnista joudutaan tekemään Javan J2EE-tasolle. OneBusAway:ssä ei ole tällä tasolla toteutettu lokalisointimekanismia, joten sellainen jouduttaisiin implementoimaan ja se olisi aikaa vievä operaatio. Web-ohjelman J2EE-tasolla on lähdekoodia noin 30 000 riviä, jonka lokalisoiminen kestää arviolta noin kolme henkilötyöviikkoa, mikäli J2EE-alusta on entuudestaan tuttu. Näin ollen web-ohjelman lokalisointia ei tämän projektin aikana tehty ajanpuutteen vuoksi, vaan siirryttiin mobiili-ohjelmistojen lokalisointiin. Selvisi, että Android- ja iOS-alustoille toteutettujen OneBusAway-mobiiliohjelmien lokalisointi on nopea toteuttaa, koska näille alustoille lokalisointi oli jo aiemmin toteutettu muutamalle eri kielille. Windows Phone -alustalle lokalisointia ei ollut aikaisemmin toteutettu, joten sen implementoiminen kestää huomattavasti pitempään kuin Android- ja iOS-alustoille, koska Windows Phone -alustan tarjoama lokalisointi menetelmä joudutaan ottamaan käyttöön, ja se vaatii paljon muutoksia ohjelmiston lähdekoodiin. Projektin aikana lokalisointi toteutettiin Android-alustalle. Lokalisoidun Android-version käyttöliittymä on esitetty kuvassa 14. Lokalisointi suoritettiin luomalla ohjelman res-kansioon values-fi-niminen kansio. Luotuun kansioon kopioitiin values-kansion sisältö ja tämän jälkeen strings.xml ja arrays.xml tiedostojen sisältämät tekstit käännettiin englannista suomeksi. Näiden muutoksien lisäksi lokalisointia varten lähdekoodiin jouduttiin tekemään yksi muutos, joka mahdollistaa käytettävän mittayksikön valinnan. OneBusAway:n kehittäjät lupasivat tehdä tämän muutoksen lähiaikoina. Suomalaiselle käyttäjäkunnalle toteutettu lokalisointi otetaan käyttöön seuraavan päivityksen myötä.



Kuva 14: OneBusAway mobiili-ohjelmiston Android-version käyttöliittymä suomeksi lokalisoituna (vrt. kuva 11).

3.5 Järjestelmän ylläpito

Järjestelmän ylläpito vaatii perehtymistä OneBusAway-projektiin sekä Java J2EE-alustan tuntemusta. Ylläpidon vaatimukset on kuvattu liitteessä 5. OneBusAway-projektissa on kolme postituslistaa, joiden avulla ongelmiin saa nopeasti apua. Postituslistoilla esitettyihin kysymyksiin vastataan noin 24 tunnin sisällä. Postituslistat on jaettu eri aihealueisiin ja ne on kuvattu taulukossa 14. Ennen kysymyksien lähettämistä kannattaa kuitenkin varmistua, että kysymykset lähetetään oikealle listalle. Muuten voi käydä niin, että kysymyksiin ei saada vastauksia. Postituslistoja on hyvä seurata säännöllisesti, jotta pysyy tietoisena mitä projektissa tapahtuu ja millaisia muutoksia seuraaviin versioihin on tulossa. OneBusAway:n kehittäjäsivustolla on tarkistuslista, jonka avulla voidaan varmistua, että ohjelmisto toimii oikein. Tämä tarkistuslista kannattaa käydä läpi aina kuin uusi versio ohjelmistosta otetaan käyttöön. Kehittäjäsivustolta pystyy myös tarkastamaan ohjelman uusimman vakaan version. Sovelluspalvelimen lokitiedostot on syytä tarkistaa säännöllisin väliajoin sekä joka kerta ohjelmiston käynnistymisen jälkeen. Lisäksi pitää varmistua, että järjestelmään ladattu GTFS-informaatio on ajan tasalla, eikä sen voimassaoloaika ole umpeutunut. Vaikka aikatauluihin ei olisi tehty mitään muutoksia, niin silti GTFS-aikatauluinformaatio pitää päivittää, jos voimassaoloaika on umpeutunut, koska muuten

järjestelmä ei näytä kyseistä aikatauluinformaatiota.

Taulukko 14: OneBusAway:n postituslistat.

Nimi	*Sisältö*
onebusaway-users	Käyttöön ja käyttöönottoon liittyvät kysymykset.
onebusaway-developers	Lähdekoodiin liittyvät kysymykset.
onebusaway-api	REST-rajapintaan liittyvät kysymykset.

3.6 Ilmenneet ongelmat

Projektin aikana kohdattiin kuusi ongelmaa. Ongelmia aiheuttivat web-ohjelma, Android- ja iOS-mobiiliohjelmat sekä Lappeenrannan GTFS-data. Nämä ongelmat on esitetty taulukossa 15 ja tarkemmat kuvaukset niistä on esitetty taulukon jälkeen.

Taulukko 15: Ilmenneet ongelmat.

Ongelma	*Syy*	*Ratkaistu*
1. Web-ohjelmisto	Esimerkkisisällössä on linkkejä, jotka osoittavat väärään paikkaan.	x
2. Web-ohjelmisto	Käyttäjien poistaminen, salasanan muuttaminen ja käyttöoikeuksien muuttaminen ei mahdollista käyttöliittymän kautta.	
3. Mobiili-ohjelmisto (Android)	Manuaalinen datapalvelimen osoitteen syöttö ei toimi, jos osoitteen perässä on porttimääritys.	x
4. Mobiili-ohjelmisto (iOS)	Manuaalinen datapalvelimen osoitteen syöttö kaataa ohjelman.	x
5. Lappeenrannan GTFS-aikatauluinformaatio	GTFS-datassa virheitä: operaattorin tiedot ja aikavyöhyke väärin, pysähtymisajoissa virheellisiä aikoja.	x
6. Tampereen SIRI-aikatauluinformaatio	GTFS- ja SIRI-informaatioiden väliltä puuttui yksiselitteinen linkitys, informaation jakelussa käytetty SIRI:n Request/Response-menetelmä ei ole tuettuna OneBusAway:ssä.	

OneBusAway:n käyttöönotossa esiintyi seuraavat ongelmat:

1. Web-ohjelmiston esimerkkisisältö sisälsi linkkejä, jotka osoittivat väärään paikkaan. Tästä johtuen navigointi ohjelmassa oli hieman hankalaa. Ongelma ratkaistiin päivittämällä rikkinäiset linkit.
2. Web-ohjelmistossa käyttäjien poistaminen sekä salasanan ja käyttöoikeuksien muuttaminen ei ollut mahdollista ohjelmiston käyttöliittymän kautta. J2EE-tasolla näille ominaisuuksille oli toteutettu toiminnallisuudet, mutta niitä ei kutsuttu käyttöliittymästä. Ominaisuudet saadaan toimimaan, kun tarvittavat muutokset toteutetaan ohjelmiston käyttöliittymään. Näitä muutoksia ei tehty tämän projektin aikana.
3. Mobiili-ohjelmiston Android-versiossa ongelmia aiheutti manuaalinen datapalvelimen URL-osoitteen syöttö. Ohjelmisto hyväksyi ainoastaan sellaisen osoitteen, jossa ei ollut porttimääritystä. Eli käytännössä ohjelma toimi vain sellaisten datapalvelimien kanssa, jotka vastasivat portista 80. Ongelma raportoitiin OneBusAway:n kehittäjille, jotka korjasivat sen.
4. Mobiili-ohjelmiston iOS-versiossa oli myös ongelma, joka liittyi manuaaliseen datapalvelimen URL-osoitteen syöttöön. Se ei kuitenkaan liittynyt porttimääritykseen vaan ohjelmisto kaatui aina, kun jokin osoite oli syötetty osoitekenttään. Ongelma raportoitiin OneBusAway:n kehittäjille, jotka korjasivat sen.
5. Lappeenrannan kaupungin GTFS-aikatauluinformaatio aiheutti myös ongelmia. Siinä esiintyi monia virheitä. Liikennöitsijätiedoissa ja aikavyöhyketiedoissa oli virheitä (agency.txt). Virheitä esiintyi myös pysäkkien lähtö- ja saapumisajoissa (stop_times.txt). Esimerkiksi, pysäkin saapumisaika oli aikaisempi kuin edellisen pysäkin lähtöaika. Lisäksi ongelmia aiheuttivat puolenyön ylittävät matkat. GTFS-standardin mukaan puolenyön ylittäviin pysäkin saapumis- ja lähtöaikoihin pitää lisätä 24 tuntia. Esimerkiksi, jos matka alkaa klo 23:00 ja puolenyön jälkeen saapumisaika pysäkille on 00:15, niin kyseinen saapumisaika pitää ilmoittaa muodossa 24:15. Virheet raportoitiin datan toimittajalle ja pyydettiin korjaamaan ne. Virheet korjattiin omiin tiedostoihin itse, jotta järjestelmää päästiin testaamaan.
6. Tampereen reaaliaikaisen SIRI-informaation kanssa ilmeni myös ongelmia. Ongelma johtui siitä, että Tampereen GTFS-informaation trip id ei vastaa SIRI:n

VechicleActivity-tietueen id:tä. Näin ollen OneBusAway ei osaa linkittää matkan reaaliaikaista informaatiota kyseisen matkan staattiseen informaatioon. Lisäksi havaittiin, että Tampereella informaation jakelussa käytetään SIRI:n Request/Response-menetelmää (kts. luku 2.2.6), jota OneBusAway ei tue. Näistä ongelmista johtuen Tampereen SIRI-informaatiota ei pystytty hyödyntämään järjestelmässä.

Ongelmat 1, 3, 4 ja 5 saatiin ratkaistua projektin aikana, mutta ongelmat 2 ja 6 jäi ratkaisematta. Ongelma 2 pystytään ratkaisemaan, mutta se vaatii tarkempaa perehtymistä OneBusAway:n web-ohjelmistoon. Ongelma 6 ratkaistaan muodostamalla yksiselitteinen linkitys GTFS- ja SIRI-informaation välille. Lisäksi vaaditaan, että SIRI-informaation jakelua muutetaan niin, että siinä käytetään SIRI:n Publish/Subscribe-menetelmää. Mahdollista on myös tehdä muutos OneBusAway-palvelinohjelmistoon, jotta se pystyy hakemaan SIRI-informaation käyttämällä Request/Response-menetelmää.

3.7 Tulosten yhteenveto ja tulkinta

Pilotti-projektin aikana käyttöönotettu OneBusAway-ohjelmistokokonaisuus osoittautui toimivaksi järjestelmäksi. Projekti jakautui viiteen vaiheeseen: palvelinohjelmiston käyttöönotto, web-ohjelmiston käyttöönotto, mobiili-ohjelmistoihin tutustuminen, järjestelmän lokalisointi ja järjestelmän ylläpidon selvittäminen. Kaikkia asioita ei ehditty toteuttamaan täydellisesti projektin puitteissa. Taulukossa 16 on esitetty asiat, jotka ehdittiin tekemään, sekä asiat joita ei ehditty tekemään.

Taulukko 16: Projektin aikana tehdyt ja tekemättä jääneet asiat.

Tehtävä	*Tehty*	*Tekemättä*
Palvelinohjelman käyttöönotto	x	
Web-ohjelman käyttöönotto	x	
Web-ohjelman ongelmien selvitys		x
Järjestelmän ylläpidon selvitys	x	
Järjestelmän testaus staattisella aikatauludatalla (GTFS)	x	
Reaaliaikaisen aikatauluinformaation käyttöönotto (SIRI)		x
Android-ohjelman testaus	x	
iPhone-ohjelman testaus	x	
Windows Phone -ohjelman testaus	x	
Web-ohjelman lokalisointi		x
Android-ohjelman lokalisointi	x	
iPhone-ohjelman lokalisointi		x
Windows Phone -ohjelman lokalisointi		x

Palvelinohjelmiston käyttöönotto sujui ilman ongelmia. Ohjelmisto osoittautui hyväksi valinnaksi, koska se toimi hyvin ja sisälsi lähes kaikki tarvittavat ominaisuudet julkisen liikenteen staattisen ja reaaliaikaisen informaation jakeluun. Palvelinohjelmistossa esiintyi kuitenkin kolme puutetta: GTFS-informaatiota ei pystynyt päivittämään jakelujärjestelmän ollessa käynnissä, REST-rajapinta ei noudattanut mitään julkisen liikenteen standardia informaation jakelussa ja järjestelmällä ei ollut mahdollista suorittaa reittihakua pisteestä A pisteeseen B. Kyseiset ongelmat raportoitiin OneBusAway:n kehittäjäkunnalle. GTFS-informaation päivitysongelmaan ei ole tällä hetkellä ratkaisua. REST-rajapinnan puutteeseen järkevin ratkaisu voisi olla rajapinnan muuttaminen sellaiseksi, että se noudattaa NeTEx-standardia, kun standardi on kokonaisuudessaan valmistunut. NeTEx-standardi tosin on vain Euroopan laajuinen standardi, joten ensin pitäisi tutkia olisiko sitä mahdollista käyttää myös maailmanlaajuisesti. Mikäli rajapinta muutettaisiin käyttämään NeTEx-standardia, niin kaikki tätä standardia käyttävät asiakasohjelmistot toimisivat myös OneBusAway:n kanssa. Reittihakuominaisuus oli aiemmin implementoitu OneBusAway:n REST-rajapintaan, mutta se poistettiin käytöstä, koska OneBusAway:n kehittäjät olivat tulleet siihen tulokseen, että avoimen lähdekoodin ohjelmisto OpenTripPlanner tarjoaa reittihaulle paremman valmiin ratkaisun. Tulevaisuudessa OpenTripPlanner:in reittihaku on tarkoitus integroida OneBusAway:hin, mutta tarkemmat suunnitelmat ja rahoitus on vielä avoinna. OneBusAway:n kehittäjäkunnan on syytä panostaa havaittujen

puutteiden ratkaisuun järjestelmän jatkokehityksessä, jotta järjestelmästä saataisiin kehitettyä vieläkin parempi ja sen maailmanlaajuinen levittäminen kaupungista toiseen olisi entistä helpompaa.

Web-ohjelmiston käyttöönotossa esiintyi ongelmia. Ohjelmisto saatiin kuitenkin otettua käyttöön, mutta kaikkia ongelmia ei ehditty ratkaisemaan projektin aikana. Ohjelmisto saatiin toimimaan peruskäyttäjän näkökulmasta, mutta admin-tunnuksia käyttäen ongelmia aiheutti admin-paneelin käyttöliittymä. Tätä ongelmaa ei ehditty ratkaisemaan projektin aikana. Admin-paneelin käyttöliittymäongelma pitää saada ratkaistua ennen kuin järjestelmä otetaan tuotantokäyttöön.

Mobiili-ohjelmistojen testaamisessa havaittiin, että ainoastaan Android- ja iOS-versiot toimivat käyttöönotetun palvelinohjelmiston kanssa yliopiston testipalvelimella, koska niissä oli mahdollista määrittää datapalvelimen URL-osoite manuaalisesti. Windows Phone -versiossa tätä ominaisuutta ei ollut, joten sitä ei pystytty testaamaan. Android- ja iOS-versioissa havaittiin muutama bugi, jotka korjattiin OneBusAway:n kehittäjien toimesta. Mobiili-ohjelmistot sisälsivät hyödyllisiä ominaisuuksia ja niiden käytettävyys oli huomattavasti parempi kuin web-ohjelmiston. Android-versiossa oli lisäksi ominaisuus, jonka avulla tietylle saapuvalle kulkuneuvolle pystyttiin asettamaan muistutus. Tämä osoittautui hyödylliseksi ominaisuudeksi, joten se kannattaisi implementoida myös muihinkin versioihin. Mobiili-ohjelmiston pysäkkikohtainen saapumisaikanäkymä osoittautui hieman sekavaksi. Esimerkiksi, jos kyseessä on pysäkki, joka toimii reitin lähtö- ja päätepysäkkinä, saapumisaikanäkymä esittää samasta kulkuneuvosta kaksi erillistä merkintään. Ensimmäinen merkintä ilmoittaa kulkuneuvon saapumisen pysäkille ja seuraava kulkuneuvon lähtemisen pysäkiltä. Muilla pysäkeillä esitetään pelkästään kulkuneuvon saapuminen pysäkille. Käyttäjän kannalta voisi olla helpompaa, jos molemmat ajat (saapumis- ja lähtöaika) olisi esitetty ja kulkuneuvosta olisi vain yksi merkintä. Lisäksi ilmeni toinenkin sekavuutta lisäävä asia. Mikäli kulkuneuvo on aikataulun mukaan jo lähtenyt pysäkiltä, saapumisaikanäkymä näyttää edelleen kyseisen kulkuneuvon listassa. Saapumisaika tosin esitetään tässä tapauksessa negatiivisena arvona. Käyttäjän saattaisi olla helpompi lukea aikataulua, jos lähteneet kulkuneuvot jätettäisiin näyttämättä. Toinen mahdollisuus voisi olla, että lähteneet kulkuneuvot saataisiin halutessa

piilotettua listasta tai ne esitettäisiin haaleampana kuin muut kulkuneuvot. Näin ollen käyttäjän olisi nopeampi katsoa, mikä kulkuneuvoista on seuraavaksi saapumassa pysäkille.

OneBusAway-ohjelmistokokonaisuuden lokalisointia tutkittaessa havaittiin, että web-ohjelmiston lokalisointi on huomattavasti työläämpi kuin mobiili-ohjelmistojen lokalisointi, koska web-ohjelmiston lokalisointi vaatii lokalisoinnin toteuttamisen kahdelle eri tasolle. Web-ohjelman lokalisointia ei suoritettu tämän projektin aikana, koska se olisi vienyt liian paljon aikaa. Ennen web-ohjelman lokalisoinnin toteuttamista on hyvä selvittää olisiko lokalisoinnin keskittäminen yhteen paikkaan järkevämpi vaihtoehto. Mobiili-ohjelmistojen lokalisointi toteutettiin ainoastaan Android-versiolle. iOS-version lokalisointi pystytään toteuttamaan yhtä helposti, mutta Windows Phone -version lokalisointiin menee paljon enemmän aikaa, koska kaikki käyttöliittymän tekstit ovat kovakoodattuna lähdekoodin sekaan.

OneBusAway-palvelinohjelmiston ylläpitoa tutkittaessa havaittiin, että ohjelman ylläpitäminen vaatii perehtymistä OneBusAway-projektiin sekä Java J2EE-alustan tuntemusta. OneBusAway-projektin postituslistoja pitää seurata säännöllisesti, jotta pysyy tietoisena mitä projektissa tapahtuu ja millaisia muutoksia seuraaviin versioihin on tulossa. Postituslistojen kautta saa myös apua ongelmatilanteissa. Ohjelmiston ylläpidolle pitää varata sellainen miehitys, että ongelmatilanteissa ongelma saadaan ratkaistua viimeistään yhden viikon kuluessa. Tästä luonnollisesti aiheutuu kuluja.

4 ARVIOINTIA JA POHDINTAA

Tutkimuksen tavoitteena ollut replikoitavan palvelinjärjestelmän toteuttaminen julkisen liikenteen avoimen datan jakeluun toteutui hyvin. Projektin alussa tehdyssä kirjallisuuskatsauksessa selvisi monia tärkeitä asioita, joista ilmeni millaista julkisen liikenteen avointa dataa ja millaisia valmiita jakelujärjestelmiä on saatavilla sekä mitä hyötyä avoimesta datasta on julkisen liikenteen toimijoille ja niiden käyttäjille. Lisäksi teknisellä tasolla selvisi, mitkä ovat tällä hetkellä parhaat julkisen liikenteen standardit ja datan serialisointimenetelmät.

Projektin aikana havaittiin, että avoimen lähdekoodin ohjelmistot ovat varteenotettavia ratkaisuja julkisen liikenteen avoimen datan jakeluun, mutta ne eivät ole välttämättä ihan niin hyvin viimeisteltyjä kuin kaupalliset järjestelmät. Esimerkiksi ohjelmistojen käyttöliittymiä joudutaan yleensä parantelemaan, jotta niistä saataisiin helppokäyttöisempiä ja houkuttelevampia. Lisäksi ohjelmistojen käyttöönotto ja ylläpito vaatii tuntemusta kyseisestä avoimen lähdekoodin projektista. Monissa kunnissa tähän tarvitaan ulkopuolista yritystä, joka hoitaa järjestelmän käyttöönoton ja ylläpidon. Avoimen lähdekoodin ohjelmistoilla on kuitenkin hyviä puolia. Avoimen lähdekoodin ohjelmistot ovat kustannustehokkaita, koska ne ovat valmiita järjestelmiä. Tällöin kustannuksia aiheuttaa vain järjestelmän käyttöönotto ja ylläpito, sekä haluttujen lisäominaisuuksien toteuttaminen (esim. lokalisointi, käyttöliittymän parantaminen). Lisäksi niiden käytöstä ei peritä lisenssimaksuja. Avoimen lähdekoodin ohjelmiston kehittäjäyhteisö huolehtii ohjelmiston kehittämisestä ja bugien korjaamisesta. Mikäli avoimen lähdekoodin ohjelmistolla on suuri kehittäjäyhteisö, on mahdollista, että muutoksia ja parannuksia ohjelmistoon saadaan nopeallakin aikataululla, kuten OneBusAway-ohjelmistokokonaisuuden kanssa havaittiin. Lisäksi tehdyt muutokset ja parannukset ovat ilmaiseksi kaikkien käytettävissä. Avoimen lähdekoodin projekteissa jokaisella on myös mahdollisuus kehittää ohjelmistosta oma versio, johon voi toteuttaa omia lisäominaisuuksia ja parannuksia. Nämä muutokset voidaan yhdistää ohjelmiston pääversioon, jonka seurauksena tehdyt muutokset ovat automaattisesti muidenkin käytettävissä. Näin ollen jokaisen ei tarvitse toteuttaa samoja ominaisuuksia erikseen, vaan jokainen hyötyy tehdyistä muutoksista. Tämän seurauksena ohjelmistoa

voidaan kehittää nopeasti, jos kehittäjäyhteisö on tarpeeksi suuri.

Projektissa jakelujärjestelmäksi valittu avoimen lähdekoodin ohjelmisto OneBusAway vaikuttaa toimivalta ratkaisulta ja se kannattaa ottaa käyttöön Lappeenrannassa, koska se sisältää lähes kaikki tarvittavat ominaisuudet julkisen liikenteen staattisen ja reaaliaikaisen informaation jakeluun. Tutkimuksen aikana havaittiin, että koneellisesti luettavaa avointa dataa on jo saatavilla Lappeenrannassa Liikenneviraston Matka.fi-palvelusta (kts. luku 2.1.3). Matka.fi-palvelu sisältää reittihakuominaisuuden, joka OneBusAway:stä tällä hetkellä puuttuu. Tästä huolimatta OneBusAway tarjoaa kuitenkin paremman kokonaisratkaisun datan jakeluun. OneBusAway:ssa on monia hyviä puolia verrattuna Matka.fi-palveluun. OneBusAway on avoimen lähdekoodin ohjelmistokokonaisuus, joten sen replikointi kunnasta toiseen onnistuu huomattavasti pienemmillä kustannuksilla. Arkkitehtuuri on suunniteltu niin, että replikointi on yksinkertaista. Avoimen lähdekoodin ansiosta kaikki siihen tehdyt parannukset ovat ilmaiseksi kaikkien käytettävissä. Lisäksi OneBusAway:n ympärille on muodostunut kehittäjäyhteisö, joka kehittää sitä aktiivisesti. OneBusAway sisältää myös valmiit asiakasohjelmistot, jotka osaavat automaattisesti yhdistää lähimmälle datapalvelimelle sijaintinsa perusteella hyödyntämällä OneBusAway:n maailmanlaajuista aluetietopalvelinta.

5 JOHTOPÄÄTÖKSET

Tutkimuksen tarkoituksena oli selvittää, miten saadaan toteutettua replikoitava palvelinjärjestelmä julkisen liikenteen avoimen datan jakeluun. Projektin aikana tultiin siihen tulokseen, että palvelinjärjestelmää ei kannata käydä kehittämään itse, koska sitä ei olisi ehditty saada valmiiksi tämän projektin aikana. Sen sijaan palvelinjärjestelmäksi valittiin avoimen lähdekoodin ohjelmisto OneBusAway, koska se on tällä hetkellä olemassa olevista palvelinohjelmistoista parempi vaihtoehto julkisen liikenteen avoimen datan jakeluun. OneBusAway on monipuolisin, koska se pystyy jakelemaan sekä staattista että reaaliaikaista aikatauluinformaatiota ja se sisältää palvelinohjelmiston lisäksi myös valmiit web- ja mobiili-ohjelmistot. Lisäksi OneBusAway on kustannustehokas ratkaisu, koska se voidaan replikoida kunnasta toiseen ilman lisenssimaksuja, ja valmiiden ohjelmistojen ansiosta ainoat kustannukset muodostuvat järjestelmän käyttöönotosta, ylläpidosta ja lokalisoinnista.

Palvelinjärjestelmän käyttöönottoa varten käynnistettiin pilotti-projekti Lappeenrannan kaupungin kanssa. Palvelinjärjestelmä otettiin testikäyttöön Lappeenrannan teknillisessä yliopistossa ja sitä testattiin Lappeenrannan staattisella GTFS-informaatiolla. Järjestelmää ei testattu Lappeenrannan reaaliaikaisella informaatiolla, koska sitä ei ollut vielä saatavilla. OneBusAway:n web- ja mobiili-ohjelmistot otettiin käyttöön ja niiden lokalisointia suomalaiselle käyttäjäkunnalle selvitettiin. Web-ohjelmiston lokalisointi osoittautui laajaksi operaatioksi, joten sitä ei tehty tämän projektin aikana. Mobiili-ohjelmistojen lokalisointia selvitettiin myös ja todettiin, että Android- ja iOS-versioiden lokalisointi onnistuu helposti, mutta Windows Phone -version lokalisointi vie huomattavasti enemmän aikaa. Mobiili-ohjelmistoista Android-versio lokalisoitiin suomalaiselle käyttäjäkunnalle projektin aikana.

Projektin avulla osoitettiin, että Lappeenrannan teknillisessä yliopistossa käyttöönotettu OneBusAway-palvelinohjelmisto toimi yliopiston testikäytössä hyvin. Ohjelmiston avulla pystytään jakelemaan julkisen liikenteen avointa dataa ja ohjelmisto on replikoitavissa kunnasta toiseen maailmanlaajuisesti. OneBusAway:n valmiit web- ja mobiili-ohjelmistot ovat käyttökelpoisia, mutta ne pitää lokalisoida suomalaiselle käyttäjäkunnalle.

Ohjelmistojen käytettävyyttä ja niiden graafista ilmettä olisi syytä parantaa, jotta niistä saataisiin houkuttelevampia ja helppokäyttöisempiä. Tarpeellista on myös tutkia antaisiko kulkuneuvojen reaaliaikainen sijainti esitettynä kartan päällä lisäarvoa käyttäjille, ja mitä muutoksia palvelinohjelmistoon tarvittaisiin tämän takia. Tarpeellista on myös tutkia onko OpenTripPlanner:in integrointi OneBusAway:hin paras mahdollinen ratkaisu, jotta reittihakuominaisuus saataisiin otettua käyttöön, ja miten reittihaku eri alueiden välillä pystytään suorittamaan. Lisäksi on tarpeellista tutkia olisiko OneBusAway:n REST-rajapinta mahdollista muuttaa sellaiseksi, että se noudattaisi julkisen liikenteen standardeja.

LÄHTEET

1. Sitra (2013). Helsinki Region Infoshare sai Euroopan komission innovaatiopalkinnon [verkkodokumentti]. Viitattu [11.10.2014]. Saatavilla http://www.sitra.fi/uutiset/avoin-data/helsinki-region-infoshare-sai-euroopan-komission-innovaatiopalkinnon.
2. Helsingin seudun liikenne (2014). Reittiopas API rajapinnan ohjeet [verkkodokumentti]. Viitattu [12.6.2014]. Saatavilla http://developer.reittiopas.fi/pages/fi/reittiopas-api.php?lang=FI.
3. Tampereen joukkoliikenne (2014). TRE API Developer's Guide [verkkodokumentti]. Viitattu [12.6.2014]. Saatavilla http://developer.publictransport.tampere.fi/pages/en/home.php.
4. Helsinki Region Infoshare (2014). Läpinäkyvä kaupunki [verkkodokumentti]. Viitattu [11.10.2014]. Saatavilla http://www.hri.fi/fi/ajankohtaista/lapinakyva-kaupunki.
5. Ramboll (2013). Avoin data edesauttaa älyliikenteen palvelujen kehittämistä [verkkodokumentti]. Viitattu [11.10.2014]. Saatavilla http://www.ramboll.fi/media/rfi/avoin-data-edesauttaa-alyliikenteen-palvelujen-kehittamista.
6. Editori (2014). Älyliikennepalveluiden kehitystrendit - lähitulevaisuus ja seuraavan sukupolven ratkaisut [verkkodokumentti]. Viitattu [11.10.2014]. Saatavilla http://www.editori.fi/yhteiskunta/alyliikennepalveluiden-kehitystrendit-lahitulevaisuus-ja-seuraavan-sukupolven-ratkaisut/#.VDmC_Evw4ds.
7. Ferris, B., Watkins, K., & Borning, A. (2010). OneBusAway: A Transit Traveler Information System. In Mobile Computing, Applications, and Services (pp. 92-106). Springer Berlin Heidelberg.
8. Open Knowledge Foundation (2012). Open Data Handbook Documentation [verkkodokumentti]. Viitattu [4.6.2014]. Saatavilla http://opendatahandbook.org/pdf/OpenDataHandbook.pdf.

9. Liikennevirasto (2014). Matka.fi API Developer's Guide - Kalkati.net, XML database dump [verkkodokumentti]. Viitattu [11.7.2014]. Saatavilla http://developer.matka.fi/pages/en/kalkati.net-xml-database-dump.php.
10. Kizoom (2008). SIRI Handbook & Functional Service Diagrams [verkkodokumentti]. Viitattu [6.3.2014]. Saatavilla http://user47094.vs.easily.co.uk/siri/schema/1.3/doc/Handbook/Handbookv15.pdf.
11. Verkehrsverbund Berlin-Brandenburg (2014). API-Schnittstelle für Webentwickler [verkkodokumentti]. Viitattu [12.6.2014]. Saatavilla http://www.vbb.de/de/article/webservices/schnittstellen-fuer-webentwickler/5070.html.
12. Metropolitan Transportation Authority (2014). Developer Resources [verkkodokumentti]. Viitattu [12.6.2014]. Saatavilla http://web.mta.info/developers/.
13. Google Inc. (2012). What is GTFS-realtime? [verkkodokumentti]. Viitattu [19.2.2014]. Saatavilla https://developers.google.com/transit/gtfs-realtime.
14. CanalTP (2014). Navitia [verkkodokumentti]. Viitattu [13.8.2014]. Saatavilla https://github.com/CanalTP/navitia/wiki.
15. Liikennevirasto (2014). Matka.fi [verkkodokumentti]. Viitattu [13.8.2014]. Saatavilla http://www.matka.fi.
16. CanalTP (2014). Navitia Interface V1's documentation [verkkodokumentti]. Viitattu [13.8.2014]. Saatavilla https://github.com/CanalTP/navitia/blob/dev/documentation/navitia.io/source/integration.rst.
17. Ferris, B., Watkins, K., & Borning, A. (2010, April). OneBusAway: results from providing real-time arrival information for public transit. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 1807-1816). ACM.
18. Watkins, K. E., Ferris, B., Borning, A., Rutherford, G. S., & Layton, D. (2011). Where Is My Bus? Impact of mobile real-time information on the perceived and actual wait time of transit riders. Transportation Research Part A: Policy and Practice, 45(8), 839-848.

19. Dziekan, K., & Kottenhoff, K. (2007). Dynamic at-stop real-time information displays for public transport: effects on customers. Transportation Research Part A: Policy and Practice, 41(6), 489-501.
20. CEN (2001). Reference Data Model for Public Transport [verkkodokumentti]. Viitattu [1.7.2014]. Saatavilla http://www.transmodel.org/en/cadre1.html.
21. Goldstein B., Dyson L., Beyond Transparency: Open Data and the Future of Civic Innovation, 1st edition, Code for America Press, USA 2013.
22. Google Inc. (2012). What is GTFS? [verkkodokumentti]. Viitattu [18.2.2014]. Saatavilla https://developers.google.com/transit/gtfs.
23. CEN (2008). Road traffic and transport telematics - Public transport - Identification of fixed objects in public transport [verkkodokumentti]. Viitattu [3.7.2014]. Saatavilla http://www.dft.gov.uk/naptan/ifopt/ifoptV1.0-36/CENTC278WG3SG6_IFOPT_20081110_36.pdf.
24. CEN (2009). NeTEx - Network and Timetable Exchange - Part 1: Network Topology [verkkodokumentti]. Viitattu [4.7.2014]. Saatavilla http://user47094.vs.easily.co.uk/netex/schema/doc/CEN%20TS_278307-NeTEX-Part-1_-%28E-%29-V29.pdf.
25. CEN (2011). NeTEx - Network and Timetable Exchange - Part 2: Network Timing Information [verkkodokumentti]. Viitattu [4.7.2014]. Saatavilla http://user47094.vs.easily.co.uk/netex/schema/doc/CEN%20TS_278308-NeTEX-Part-2_%28E%29-V13.pdf.
26. CEN (2014). CEN/TC 278 - Intelligent transport systems, CEN/TS 16614-1:2014 [verkkodokumentti]. Viitattu [5.7.2014]. Saatavilla http://standards.cen.eu/dyn/www/f?p=204:110:0::::FSP_PROJECT,FSP_ORG_ID:36786,6259&cs=1992D15FE920C7CD7940798D0E37940CB.
27. CEN (2014). CEN/TC 278 - Intelligent transport systems, CEN/TS 16614-2:2014 [verkkodokumentti]. Viitattu [5.7.2014]. Saatavilla http://standards.cen.eu/dyn/www/f?p=204:110:0::::FSP_PROJECT,FSP_ORG_ID:36787,6259&cs=167E6E8F0FE2D0BEDFBCCF48AAFE95062.
28. Google Inc. (2012). What Are Protocol Buffers? [verkkodokumentti]. Viitattu [19.2.2014]. Saatavilla https://developers.google.com/protocol-buffers.

29. Ecma International (2013). ECMA-404 The JSON Data Interchange Standard [verkkodokumentti]. Viitattu [14.6.2014]. Saatavilla http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf.
30. World Wide Web Consortium (2006). Extensible Markup Language (XML) 1.0 (Fourth Edition) [verkkodokumentti]. Viitattu [14.6.2014]. Saatavilla http://www.w3.org/TR/2006/REC-xml-20060816/.
31. Google code (2014). Third-Party Add-ons for Protocol Buffers [verkkodokumentti]. Viitattu [15.6.2014]. Saatavilla https://code.google.com/p/protobuf/wiki/ThirdPartyAddOns.
32. Nurseitov, N., Paulson, M., Reynolds, R., & Izurieta, C. (2009). Comparison of JSON and XML Data Interchange Formats: A Case Study. Caine, 9, 157-162.
33. Sumaray, A., & Makki, S. K. (2012). A comparison of data serialization formats for optimal efficiency on a mobile platform. In Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication (p. 48). ACM.
34. U.S. Census Bureau (2014). State & County QuickFacts: Tampa, Florida [verkkodokumentti]. Viitattu [11.10.2014]. Saatavilla http://quickfacts.census.gov/qfd/states/12/1271000.html.
35. Tilastokeskus (2013). Kuntien avainluvut: Lappeenranta [verkkodokumentti]. Viitattu [11.10.2014]. Saatavilla http://www.tilastokeskus.fi/tup/kunnat/kuntatiedot/405.html.
36. OneBusAway (2013). The OneBusAway RESTful API [verkkodokumentti]. Viitattu [28.3.2014]. Saatavilla http://developer.onebusaway.org/modules/onebusaway-application-modules/current/api/where/index.html.
37. OneBusAway (2014). Adding Regions to the OneBusAway Multi-Region Scheme [verkkodokumentti]. Viitattu [11.6.2014]. Saatavilla http://onebusaway.org/transit-agencies/adding-regions.php.
38. OneBusAway (2013). OneBusAway multi region architecture [verkkodokumentti]. Viitattu [6.3.2014]. Saatavilla https://github.com/OneBusAway/onebusaway/wiki/Multi-Region.

39. OneBusAway (2012). Webapp Configuration Guide [verkkodokumentti]. Viitattu [25.7.2014]. Saatavilla https://github.com/OneBusAway/onebusaway-application-modules/wiki/Webapp-Configuration-Guide.
40. OneBusAway (2012). OneBusAway Web [verkkodokumentti]. Viitattu [25.7.2014]. Saatavilla https://github.com/OneBusAway/onebusaway-application-modules/wiki/OneBusAway-Web.
41. Windows Phone Store (2013). OneBusAway [verkkodokumentti]. Viitattu [11.10.2014]. Saatavilla http://www.windowsphone.com/en-us/store/app/onebusaway/30dcbcc4-e3d0-df11-9eae-00237de2db9e?type=phoneapp&id=30dcbcc4-e3d0-df11-9eae-00237de2db9e&source=onebusawaysite.

LIITE 1. Tarkastuslista: Palvelimen hyväksyminen OneBusAway:n tuotantopalvelimeksi

- Anomus OneBusAway:n postituslistalle
- Vaatimukset
    - Pitää olla ollut käytössä vähintään 1kk, toimintavarmuudella 99.5 %.
    - Pitää tarjota riittävän kattavasti reaaliaikaista dataa kyseiseltä alueelta.
    - Reaaliaikatiedon paikkansapitävyys pitää olla varmistettu.
    - Alkuperäisten OneBusAway mobiili-ohjelmien (Android, iOS, Windows Phone) pitää toimia palvelimen kanssa.
    - Pitää tukea OneBusAway:n alkuperäistä rajapintaa (onebusaway-api-webapp).
    - Palvelimen ylläpitäjän pitää taata, että aikatauluinformaatio pysyy ajan tasalla.
    - Pitää saada OneBusAway:n hallituksen hyväksyntä

Mahdolliset ongelmat tulee pystyä korjaamaan viikon sisällä siitä, kun ongelmat havaittiin, muuten OneBusAway:lla on oikeus poistaa palvelimen tiedot aluetietopalvelimelta.

LIITE 2. Quick-Start-asennuksen asennusohje (Linux Ubuntu 12.04 LTS)

Palvelimen asetukset:

- Muisti: 2 Gt
- Kiintolevytila: 10Gt
- Ohjelmistot: Java 1.6

Asennus:

1. Asenna Java, JDK 1.6
    - apt-get install openjdk-6-jdk
2. Lataa OneBusAway-ohjelmistopaketti (kaksi vaihtoehtoa)
    - onebusaway-quickstart-assembly-webapp.war (web + rest api)
    - onebusaway-quickstart-assembly-api-webapp.war (rest api)
3. Luo bundle-paketti
    - java -Xmx1G -server -jar [war_paketin_nimi] -build [gtfs_tiedoston_polku] [bundle_paketin_polku]
4. Käynnistä OneBusAway
    - java -jar [war_paketin_nimi] -webapp [bundle_paketin_polku]

Quick-Start-asennus ei sovellu tuotantokäyttöön, joten kyseiseen käyttöön ohjelmisto tulee asentaa käyttäen manuaalista asennusta (kts. liite 3).

LIITE 3. Manuaalisen asennuksen asennusohje (Linux Ubuntu 12.04 LTS)

Palvelimen asetukset:

- Muisti: 2 Gt
- Kiintolevytila: 10Gt
- Virtuaalikone: VMWare
- Ohjelmistot: Java 1.6, Tomcat 6, Git 1.7.9.5, Maven 3.0.4, XWiki 5.4.4

Asennus:

1. Asenna Java, JDK 1.6
    - apt-get install openjdk-6-jdk
2. Asenna Tomcat 6
    - apt-get install tomcat6
3. Asenna Git
    - apt-get install git
4. Asenna Maven
    - apt-get install maven
5. Hae koodit GitHub:sta
    - git clone https://github.com/OneBusAway/onebusaway-application-modules.git
6. Tarkista mikä on tämän hetkinen stable-versio
    - http://developer.onebusaway.org/modules/onebusaway-application-modules/current/
7. Tarkasta millä tag:lla kyseinen versio löytyy
    - git tag –l
8. Valitse tämän hetkinen stable-versio
    - git checkout tags/[tag_nimi] (esim. git checkout tags/onebusaway-application-modules-1.1.11)

(jatkuu)

LIITE 3. (jatkoa)

9. Mene kansioon onebusaway-application-modules ja käännä moduulit
    - mvn clean install
10. Luo bundle-paketti
    - cd onebusaway-transit-data-federation-builder/target/
    - java -jar onebusaway-transit-data-federation-builder-x.x.xx-withAllDependencies.jar [GTFS_tiedoston_polku] [bundle_paketin_polku]
11. Anna oikeudet luodulle bundle-paketille
    - chown -R tomcat6:tomcat6 [bundle_paketin_polku]
    - chmod 744 [bundle_paketin_polku]/ServiceAlerts.xml
12. Kopio onebusaway-combined-webapp-x.x.xx-full.war asennuspaketti Tomcat:in webapps-kansioon
    - cp onebusaway-combined-webapp/target/onebusaway-combined-webapp-x.x.xx-full.war /var/lib/tomcat6/webapps/ROOT.war
13. Pura kopioitu paketti ROOT-kansioon (jos kansio ei ole tyhjä -> poista sisältö, jos kansiota ei ole olemassa -> luo se)
    - jar xvf ../ROOT.war
14. Luo data-sources.xml ja kopioi se classes-kansioon
    - cp data-sources.xml /var/lib/tomcat6/webapps/ROOT/WEB-INF/classes/data-sources.xml

(jatkuu)

LIITE 3. (jatkoa)

15. Asenna XWiki
    - Lataa asennuspaketti (http://enterprise.xwiki.org/xwiki/bin/view/Main/Download) ja asenna ohjelma
    - Enabloi superadminuser editoimalla XWiki:n konfiguraatio-tiedostoa (/webapps/xwiki/WEB-INF/xwiki.cfg) ja ota kommenttimerkki pois rivin ”xwiki.superadminpassword=system” edestä
    - Käynnistä XWiki
    - Lataa esimerkkisisältö osoitteesta https://docs.google.com/file/d/0B8oU647elPShczdQWXJpT3F2cEk/edit (nbackup.xar)
    - Tuo sisältö (nbackup.xar) XWiki:in backup-paketin palautustyökalulla
    - Editoi sisältö halutun mukaiseksi. Muista kopioida kuvat Logo.png (http://wiki.onebusaway.org/bin/download/Main/Index/Logo.png) ja Footer.png (http://wiki.onebusaway.org/bin/download/Main/Index/Footer.png) omalle palvelimelle ja päivitä niihin viittaavat linkit.
16. Käynnistä Tomcat
    - /etc/init.d/tomcat6 start
17. Varmista, että käynnistys onnistui
    - less /var/log/tomcat6/catalina.out

LIITE 4. SIRI:n käyttöönotto OneBusAway:ssä ja sen vaatimukset

- Vaatimukset
    - GTFS-informaation trip id pitää olla sama kuin SIRI:n VechicleActivity-tietueen id
    - SIRI:n Publish/Subscribe-menetelmää pitää käyttää tiedonsiirrossa
- Käyttöönotto
    - Lisää data-sources.xml tiedostoon alla oleva SiriController määritys
    - Määritä riville 4, mistä URL-osoitteesta SIRI-informaatio on saatavilla (endpoint)
    - Määritä riville 6 OneBusAway palvelimen URL-osoite, johon SIRI-informaatio lähetetään (clientUrl)

```
<!-- The "name" parameter controls which URL the SIRI client listens for pub-sub data.  See "clientUrl" below. -->
<bean name="/siri.action" class="org.onebusaway.transit_data_federation_webapp.siri.SiriController">
  <!-- Specify the SIRI endpoint -->
  <property name="endpoint" value="Url=http://host/siri-endpoing.xml,ModuleType=VEHICLE_MONITORING" />
<!-- Control the URL your SIRI client publically broadcasts to endpoints for pub-sub data exchange -->
  <property name="clientUrl" value="http://localhost:8080/onebusaway-transit-data-federation-webapp/remoting/siri.action" />
</bean>
```

LIITE 5. Tarkastuslista: OneBusAway:n ylläpidon vaatimukset

- Osaaminen
    - Perehtyminen OneBusAway-projektiin (noin 1 htpv)
    - J2EE-alustan tuntemus
- Toimenpiteet
    - OneBusAway:n postituslistojen säännöllinen seuraaminen, jotta tiedetään milloin palvelinohjelmisto kannattaa päivittää uudempaan versioon
    - Sovelluspalvelimen lokitiedostojen tarkastaminen säännöllisin väliajoin sekä jokaisen käynnistyksen jälkeen
    - Ohjelmiston toiminnan tarkastaminen kehittäjäsivustolta löytyvän tarkastuslistan (https://github.com/OneBusAway/onebusaway/wiki/Troubleshooting) avulla aina, kun palvelinohjelmisto on käynnistetty
    - GTFS-aikatauluinformaation päivittäminen ennen sen umpeutumista