YRKESHÖGSKOLAN
NOVIA

Övervaking av klimatet i fraktutrymmen med sensorenhet och mobilapplikation

Alexander Enlund

Examensarbete för ingenjör (YH)-examen

Utbildningsprogrammet för el- och automationsteknik

Vasa 2019

EXAMENSARBETE

Författare:	Alexander Enlund
Utbildning och ort:	El- och automationsteknik, Vasa
Inriktningsalternativ:	Informationsteknik
Handledare:	Susanne Österholm

Titel: Övervakning av klimatet i fraktutrymmen med sensorenhet och mobilapplikation

---

Datum: 4.6.2019 Sidantal: 30

Abstrakt

Examensarbetet utfördes på uppdrag av Nyholm Solutions Ab Oy. Examensarbetet täcker Machine to Machine (M2M), med lagring av data på en sensorenhet och kommunikation mellan en sensorenhet och en mobil enhet med Bluetooth Low Energy (BLE). Målet för examensarbetet var att utveckla en mobilapplikation som möjliggör övervakning av klimatet i bilars fraktutrymmen, detta genom visning av temperaturen, lufttrycket och luftfuktigheten på en mobil enhet.

Resultatet för examensarbetet blev en mobilapplikation som utvecklats med Apache Cordova. Mobilapplikationen visar omgivningsdata som blivit överfört från en sensorenhet till en mobil enhet via Bluetooth Low Energy (BLE).

Språk: svenska Nyckelord: Apache Cordova, BLE, M2M, mobilapplikation, sensorenhet, omgivningsdata

---

OPINNÄYTETYÖ

Tekijä:	Alexander Enlund
Koulutus ja paikkakunta:	Sähkö- ja automaatiotekniikka, Vaasa
Suuntautumisvaihtoehto:	Tietotekniikka
Ohjaaja:	Susanne Österholm

Nimike: Kuormaustilan seuranta sensorilaitteella ja mobiilisovelluksella

---

Päivämäärä: 4.6.2019 Sivumäärä: 30

Tiivistelmä

Opinnäytetyö on tehty Nyholm Solutions Ab Oy:n puolesta. Opinnäytetyö kattaa Machine to Machine (M2M) -datan tallentamista sensorilaitteella sekä kommunikaatiota sensorilaitteen ja mobiililaitteen välillä Bluetooth Low Energyn (BLE) avulla. Opinnäytetyön tavoite oli kehittää mobiilisovellus, joka mahdollistaa ilmalaadun seurantaa kuljetusauton kuormatilassa. Ilmalaatua seurataan näyttämällä lämpötilaa, ilmapainetta ja ilmankosteutta mobiililaitteessa.

Opinnäytetyön tulos on mobiilisovellus, joka on suunniteltu Apache Cordova-ohjelmistokehyksen avulla. Mobiilisovellus näyttää ympäristödatan, joka on tuotu sensorilaitteesta mobiililaitteeseen Bluetooth Low Energyn (BLE) kautta.

Kieli: ruotsi Avainsanat: Apache Cordova, BLE, M2M, mobiilisovellus, sensorilaite, ilmaston data

---

BACHELOR'S THESIS

Author:	Alexander Enlund
Degree Programme:	Electrical- and automation engineering
Specialization:	Information technology
Supervisor(s):	Susanne Österholm

Title: Monitoring of Climate in Freight Space with Sensor Device and Mobile Application

---

Date: June 6, 2019 Number of pages: 30

Abstract

The thesis was made for Nyholm Solutions Ab Oy. The thesis includes Machine to Machine (M2M), this by storing data onto the sensor unit and communication between a sensor unit and a mobile over device Bluetooth Low Energy (BLE). The goal for the thesis was to develop a mobile application, which allows for monitoring of the climate in freight spaces. This is done by showing the temperature, air pressure and air humidity on the mobile device.

The result of the thesis is a mobile application which has been developed with Apache Cordova. The mobile application shows the environmental data, which has been transferred from a sensor unit to a mobile device over Bluetooth Low Energy (BLE).

Language: Swedish Key words: Apache Cordova, BLE, M2M, mobile application, sensor unit, environmental data

---

Innehållsförteckning

Ordförklaringar och förkortningar

ANT	Apache Ant (Ett mjukvaruverktyg för automatisering för byggande av program i programmeringsspråket Java)
API	Application Programming Interface (Applikationsprogrammerings-gränssnitt)
ARM	En 32- och 64-bitars processorarkitektur som utvecklats av ARM Holdings
BLE	Bluetooth Low Energy
Bluetooth mesh	(Many-to-many) Många-till-många Bluetooth kommunikation
Bootstrap	Ett verktyg för utveckling med HTML, CSS och JavaScript, för utveckling av responsiva applikationer
Call-id	Nummervisare (visning av telefonnummer på den som ringer på mottagarsidan)
DIY	Do-It-Yourself (Gör-det-själv)
GATT	Generic Attribute (Generisk Egenskap). Definierar hierarkiska datastrukturen vilken exponeras för anslutna bluetooth-enheter
Gräsrotsfinansiering	Förskottsfinansiering för projekt och/eller idéer av intressenter
HEX-fil	En hexadecimal källfil, som ofta används till programmerbara logiska enheter (t.ex. mikrokontroller)
I^2S	Inter-IC Sound (standard elektrisk seriell ingång/utgång interface som används för att ansluta digitala ljudenheter)
IDE	Integrated Development Environment (Integrerad utvecklingsmiljö)
IoT	Internet of Things (Sakernas Internet)
Kickstarter	En gräsrotsfinansieringssida (Crowdfunding website) på internet

LPDDR SDRAM	Low-Power Double Data Rate Synchronous Dynamic Access Memory. Ett primärminne utvecklat för mobila datorenheter. Primärminnet konsumerar mindre ström än ett konventionellt primärminne. Primärminnet är även känt som mobil DDR.
M2M	Machine to Machine (Maskin till Maskin)
PDM	Pulse Density Modulation
Plugin	Ett mjukvarutillägg, som ökar användningen för en specifik egenskap. Till exempel uppspelning av en video på en webbsida kan behöva ett plugin eftersom webbläsaren inte har det nödvändiga verktyget för detta ändamål
PWM	Pulse Width Modulation
RSSI	Received signal strength indication. Indikering av mottagen signalstyrka
RX	Read/Receive Characteristic. (Läs-/Mottagar-datakanal)
SCADA	Supervisory Control And Data Acquisition. System som används för övervakning samt styrning av processer
SIG	Special Interest Group (Bluetooth). Ett sällskap över hela världen som utvecklar Bluetooth teknologi
SoC	System-on-a-Chip. System-på-ett-Chip
TX	Transmit/Write Characteristic. (Läs-/Skriv-datakanal)
UART	Universal Asynchronous Receiver/Transmitter (Universell Asynkron Mottagare/Sändare)
UUID	Universally Unique Identifier (Universal Unik Identifierare), 128-bitars nummer, detta Används för att identifiera information i ett datorsystem
WebView	Speciellt webbläsarfönster, som kan komma åt API-enheter på enhetsnivå

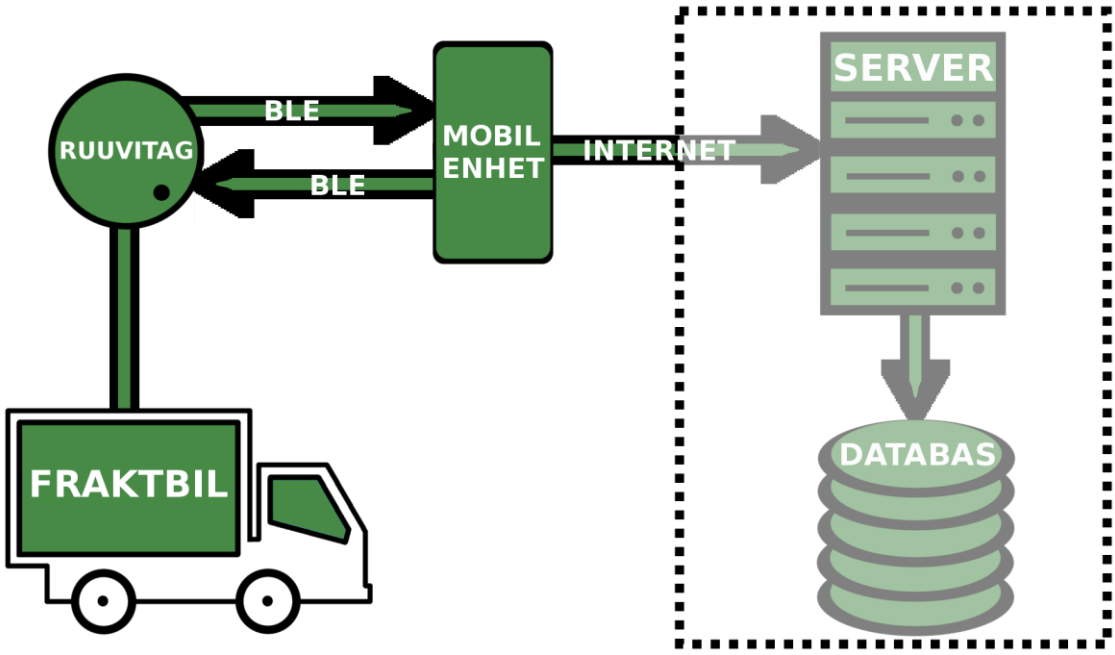
ZIP-fil	Komprimerad arkivfil

1 Inledning

Detta examensarbete beskriver utvecklingen av en prototyp för uppföljning samt dokumentering av klimatet i fraktutrymmen. Lösningen skall vara fullt fungerande såsom den är, men prototypen skall kunna utvecklas vidare efter att arbetet har slutförts.

1.1 Mål och syfte

Målet med detta arbete var att kunna följa upp, analysera och lagra data om klimatet i fraktbilars fraktutrymmen. I de bilar som fraktar speciella varor är det viktigt att en viss temperatur, luftfuktighet och/eller lufttryck hålls för att garantera bästa möjliga klimat för varor med strikta krav på fraktförhållanden. Detta görs med sensorer inne i fraktutrymmet. Sensorerna skall kunna lagra data offline, det vill säga utan att denna sensor är uppkopplad till någon annan kringutrustning. Sensorn skall sedan överföra lagrade data om klimatet i fraktutrymmet trådlöst över Bluetooth Low Energy till en mobil enhet, såsom en smarttelefon eller någon annan enhet som har stöd för BLE. Den mobila enheten skall vara kopplad till internet. Syftet med detta är att den mobila enheten har kontakt med en server som i sin tur lägger in data i en tabell i en databas på servern. Detta underlättar granskningen av fraktutrymmets klimat och uppföljningen och dokumenteringen av temperaturen, luftfuktigheten samt lufttrycket i fraktutrymmet. Detta i sin tur hjälper till att fraktutrymmets klimat hålls på en acceptabel nivå, samt inom ramen för rekommenderade värden för de varor som fraktas. (Se figur 1).



Figur 1: Enkel illustration av examensarbetet.

1.2 Uppdragsgivare

Uppdragsgivaren för detta examensarbete är Nyholm Solutions Oy Ab. Företaget är grundat i Jakobstad år 2004. Företaget utvecklar skräddarsydda webbapplikationer, IoT-, M2M- och industriella lösningar. Företagets kunder finns utspridda över hela landet men verkar huvudsakligen i Österbotten och i huvudstadsregionen. [1]

2 Teori

Detta kapitel behandlar olika tekniker bakom själva projektet och vilka tekniker som används, men också tekniker och lösningar som skulle ha varit möjliga att använda sig av. Eftersom arbetet innehåller många små delar berättas ganska allmänt om varje område.

2.1 Machine to machine (M2M)

Machine to Machine (M2M) eller Maskin till maskin kan förklaras som en teknologi där en eller flera enheter är uppkopplade till ett nätverk där enheter utbyter information sinsemellan och kan utföra åtgärder utan en människas handling. M2M-teknologi hittas inom industrin, hälsovården, affärslivet och inom många andra områden.

M2M's grund är djupt förankrad i produktionsindustrin, där andra teknologier, såsom SCADA och fjärrövervakning möjliggör fjärrstyrning samt kontroll av data från olika utrustningar och enheter. Ursprunget till akronymen M2M är inte verifierat, men användes först av Theodore Paraskevakos, som uppfann och patenterade teknologi relaterad till överföring av data över telefonlinjer. Denna teknologi motsvarar vår tids nummerpresentatör eller call-id. M2M kan kommunicera över icke-IP baserade kanaler, såsom serieportar eller anpassade protokoll. [2]

Nokia var ett av de första företagen som använde akronymen M2M i slutet av 1990-talet. Nokia startade ett affärssamarbete med Opto 22 (ett företag som tillverkar och designar solid state reläer). Samarbetet förenklade Opto 22:s montering av M2M-komponenter, vilket gjorde det möjligt för Opto 22 att erbjuda sina kunder M2M trådlös kommunikation. [3] [4] [5]

2.2 Sensorer och användningsområden

En sensor är en enhet som detekterar och läser fysiska data från omgivningen. Denna specifika data kan vara värme, ljus, tryck, fuktighet, rörelse, eller något annat fysiskt fenomen i omgivningen. Utdatat från sensorn är oftast konverterad till mänskligt läsbart värde. En sensor kan vara mer eller mindre avancerad, beroende på var och hur den används. Ett simpelt exempel är en kvicksilverbaserad glastemperaturmätare, där inmatningen är temperaturen och utmatningen kan läsas av från hur högt eller lågt kvicksilvret ligger inne i glasbehållaren. [6]

2.2.1 Användning

Den mest allmänt förekommande sensorn är temperatursensorn. Denna sensor hittas i allt från smarta termostater till IoT temperaturkontrollerade fraktutrymmen, och från kylar till industriella maskiner. Orsakerna till en växande ökning av IoT- samt M2M-system är att sensorsystem har minskat i storlek samt att priset har sjunkit märkbart, men också framsteg inom halvledarproduktion och mikrobearbetning är bidragande orsaker. [7]

2.2.2 Ruuvis historia

Ruuvi Innovations Oy är ett finländskt uppstartsföretag. Ruuvis team har en lång historia tillsammans, som startade för ungefär 17 år sedan. År 2009 grundades Finlands första sällskap för inbyggd elektronik, detta var första bedriften som gjordes under märket Ruuvi. År

2015 blev Ruuvi Innovations (Ltd) Oy grundat. Sensorenheten RuuviTag som är Ruuvis huvudsakliga produkt, blev gräsrotsfinansierad på webbsidan Kickstarter år 2016. [8]

Figur 3: Ruuvi Innovations Ltd. logo. [9]

2.2.3 RuuviTag

Sensorenheten RuuviTag består av chippet Nordic Semiconductor nRF52832 System-on-chip Software Radio. Detta chip innehåller många funktioner. Det har Bluetooth 5 och stöder många protokoll. Chippet har också lagringsutrymme på 512 kB samt RAM på 64 kB. Systemet stöder protokoll såsom Bluetooth 5, Bluetooth mesh och ANT. [10] [11]

Figur 4: RuuviTag, utan skal. [9]

2.2.4 Specifikationer på RuuviTagen

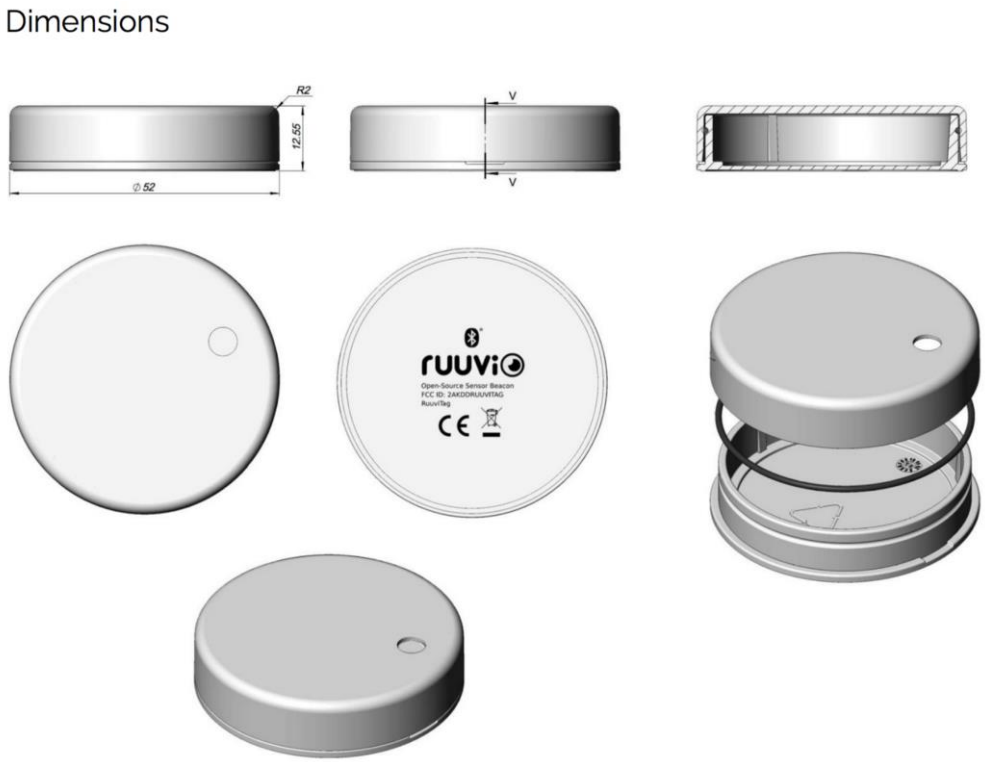
nRF52832-chippet är byggt kring en ARM Cortex-M4 processor som körs med en klockhastighet på 64 MHz. Det har en NFC-A tagg för användning av simpla anslutnings- och kontaktlösa betalningslösningar. Det har ett antal digitala kringutrustningar och gränssnitt såsom PDM och I^2S för användning av mikrofoner och ljud. För att förlänga batteriets livslängd används ett sofistikerat adaptivt strömhanteringssystem. [10] [11]

RuuviTagen består också av en STMicroelectronics LISsDH12 accelerometer. Denna accelerometer är en tre-axlars lineär accelerometer med digitalt seriellt standardgränssnitt. Denna sensor används dock inte i examensarbetet. [12]

RuuviTagen använder Bosch BME280 omgivningssensor för mätning av temperatur, luftfuktighet samt lufttryck. Denna omgivningssensor lämpar sig speciellt bra där energiförbrukningen och storleken skall vara så liten som möjligt, till exempel mobila enheter eller andra små enheter. Luftfuktighetssensorn ger noggranna data i varierande temperaturer. Lufttrycksensorn är en barometersensor. [13]

RuuviTagen drivs med ett 1000 mAh batteri, vilket räcker upp till flera månader, ända upp till år beroende på användningsområde samt programmet som körs på RuuviTagen. Den minimala samt maximala temperaturen för användningen av RuuviTagen är -40°C respektive +85°C, för både elektroniken samt skalet. Batteriets livslängd påverkas av temperaturen. RuuviTagen har också två integrerade knappar, vilka möjliggör uppdatering av RuuviTagen via mobila enheter, som till exempel en smarttelefon. RuuviTagen har också två integrerade LED-lampor. [14]

Själva skalet på RuuviTagen är gjort av polykarbonat. Skalet har ett IP67 certifierat ventilationsmembran upptill som möjliggör mätning av luftfuktighet, och en industriell O-ring som hindrar vätska att tränga mellan skalet, för att skydda elektroniken. [14]



Figur 5: Dimensionerna på RuuviTag. [9]

2.3 Alternativa sensorenheter och kringutrustning

Eftersom sensorer har blivit mera lättillgängliga, priset har sjunkit samt sensorer har blivit allt mindre och kompakta, finns det väldigt många olika alternativ för lösning av examensarbetet. I detta kapitel presenteras alternativa enheter och kringutrustningar som kunde fungera som lösningar till examensarbetet.

2.3.1 Arduino Uno R3 och Raspberry PI 3 B+

Arduino Uno R3 är en mikrokontroll, som består av:

- ATmega328P högt presterande mikrochip.
- 14 digitala ingångar/utgångar, av vilka 6 kan användas som PWM.
- 6 stycken analoga ingångar.
- En 16MHz kvartskristall.
- En USB-port.
- En strömingång.
- En ICSP (dessa pins är oftast förklarade som SPI (Serial Peripheral Interface/seriellt kringutrustningsgränssnitt) vilket i sin tur kan ses som anslutning av kringutrustning och expandering av själva Arduinon).
- En återställningsknapp. [15] [16]

Raspberry PI 3 B+ är en enkortsdator som består av:

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit Soc 1.4 GHz processor.
- 1 GB LPDDR2 SDRAM.
- 2.4 GHz och 5 GHz IEEE 802.11.b/g/b/ac trådlöst LAN (Local Area Network).
- Bluetooth 4.2 (BLE).
- Gb Ethernet över USB 2.0 (med maximal hastighet på 300 Mbps).
- 40 stycken pins (GPIO (General-Purpose Input/Output eller Generell användning ingång/utgång)).
- Full storlek HDMI-port.
- 4 stycken USB 2.0 portar.
- 4-pol stereo-utgång och komposit-video port.
- Micro SD port för laddning av operativsystem och lagring av data.
- 5 V/2.5 A DC strömingång.
- PoE (Power over Ethernet). [17] [18]

2.3.2 Teoretisk lösning med Arduino Uno R3 och Raspberry PI 3 B+

En sensor av typen GROVE – TEMP & HUMI & BAROMETER SENSOR (BME280) skulle anslutas till Arduinon. Denna sensorenhet är likadan som den RuuviTagen använder sig av det vill säga tillverkad av Bosch (kapitel 2.3.4). Denna sensorenhet skulle anslutas till en Arduino Uno R3, och med hjälp av ett program på Arduinon skulle data avläsas från sensorenheten (BME280). Eftersom Arduino har ett lagringsutrymme på 32 kB, skulle bara ett specifikt antal värden lagras på Arduinon. Arduinon skulle mäta omgivningens mätvärden med ett visst intervall. De uppmätta värdena på Arduino skulle sedan överföras via serieport (USB-port) till Raspberry PI 3 B+. [19]

Raspberryn skulle då lagra de värden som den fått från Arduinon lokalt. Eftersom Raspberryn har tillgång till Bluetooth skulle Bluetooth kunna användas för att överföra mätvärdena till en mobil enhet, till exempel en smarttelefon eller en surfplatta över BLE.

Ett problem som medföljer denna lösning är att Arduinon samt Raspberryn behöver externa strömkällor, vilket skulle öka prisnivån eftersom en strömkälla i stil med en ackumulator eller dylikt skulle vara nödvändig. Både Arduinon och Raspberryn använder sig av 5 VDC som strömkälla, men eftersom Arduinon skulle vara ansluten via serieport (USB-port) till Raspberryn, både för överförings- och strömförsörjningsändamål, skulle endast Raspberryn behöva vara ansluten till en extern strömkälla.

2.3.3 Teoretisk lösning med RuuviTag och Raspberry Pi 3 B+

En annan lösning kunde vara användning av sensorenhet RuuviTag och Raspberry Pi 3 B+. Detta skulle lösas på liknande sätt som i kapitel 2.4.2, men nu med sensorenheten RuuviTag, som ersätter en skild sensorenhet och en mikrokontroll. RuuviTagen skulle fungera både som sensorenhet och mikrokontroll. RuuviTagen skulle samla in data och spara ett specifikt antal mätvärden. Dessa skulle sedan skickas över BLE till Raspberryn. Resten skulle lösas på samma sätt som i kapitel 2.4.2.

Ett mindre problem med denna lösning är, eftersom RuuviTagen i princip skulle vara ansluten till Raspberryn hela tiden med BLE, skulle batteriets livslängd förkortas markant. Detta skulle i sin tur kunna lösas med att programmet optimeras på RuuviTagen, genom att Bluetooth aktiveras samt ansluter till Raspberryn när det verkligen behövs, det vill säga när data skall överföras till Raspberryn. Detta skulle ske med ett visst intervall. Denna lösning är fullt möjlig och skulle i sin tur hjälpa till att spara på strömmen i batteriet som driver RuuviTagen. Men problemet med en extern strömkälla vilket Rasberryn är i behov av, skulle ändå behöva lösas.

2.4 Applikationer

När man talar om mobilapplikationer, kanske tankarna går till Facebook eller Instagram, vilket inte är så konstigt, eftersom Facebook var den mest använda applikationen under år 2017 på alla plattformar. Många applikationstillverkare har både mobila webbplatser och mobilapplikationer, för att nå ut till så många användare som möjligt. [20]

2.4.1 Mobilapplikationer

Eftersom det finns olika operativsystem för mobila enheter, såsom smarttelefoner, surfplattor med mera, måste applikationen anpassas till det specifika operativsystemet som applikationen skall köras på. En mobilapplikation som är tillverkad för Apples iOS, fungerar bara på Apples iOS mobila enheter. Och en mobilapplikation tillverkad för en Android mobil enhet fungerar bara på en Android mobil enhet. Men tillverkaren av den mobila applikationen tillverkar oftast motsvarande applikationer för både Apples iOS och Android. [20]

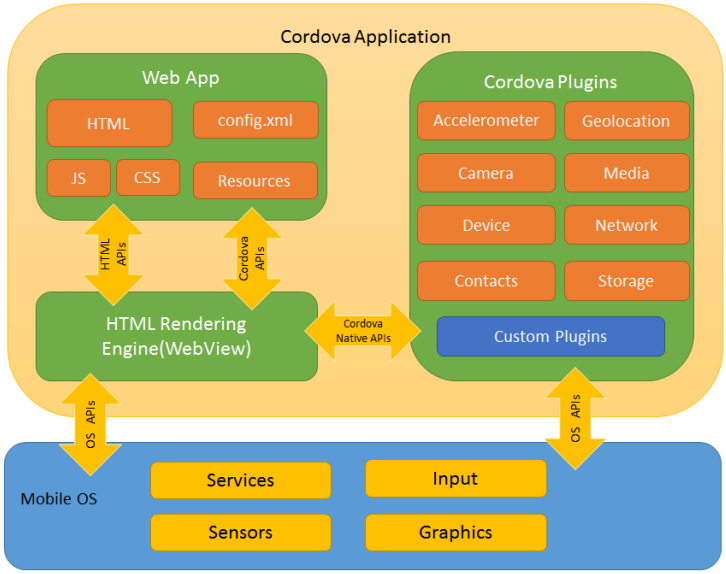
Mobila enheter finns i en mängd olika skärmstorlekar, minneskapaciteter, processoregenskaper, grafiska gränssnitt, knappar och tryckfunktioner. Allt detta måste tillverkaren ta i beaktande vid tillverkningen av en mobilapplikation. Den som använder en mobil enhet för att besöka webbplatser vill knappast scrolla i sidled för att läsa text, se på bilder, eller zooma

in på texten eller bilden eftersom den är så liten. Ett tillägg som måste tas i beaktande för mobilapplikationer är också det tryckkänsliga gränssnittet som i princip är omöjligt att frångå för nutidens mobila enheter. [20]

Förut när mobila enheter inte var vanliga som nu, tillverkades applikationer först till datorer och sedan kanske till mobila enheter, men nu har det blivit alltmer vanligt att tillverkaren tillverkar applikationer för mobila enheter först och sedan till datorer, det vill säga datorapplikationer. Detta på grund av att användningen av surfplattor och smarttelefoner håller på att bli större än användningen av datorer. Antalet nedladdade mobilapplikationer år 2017 var omkring 178 miljarder. År 2018 laddades omkring 205 miljarder applikationer ner, men trenden verkar inte jämnas ut, utan den väntas stiga. Det är förutspått att omkring 258 miljarder applikationer kommer att laddas ner över hela världen år 2022. Denna statistik har gjort att tillverkare har börjar tillverka mobilapplikationer före datoranpassade applikationer. För de flesta applikationstillverkare är mobilapplikationer en standard, medan datorapplikationer anpassas för allt större skärmar samt kraftfullare datorer. [20] [21]

2.4.2 Apache Cordova

Apache Cordova är ett ramverk med öppen källkod för mobilapplikationsutveckling. Med Apache Cordova kan tillverkaren använda sig av standard webbutvecklingsverktyg, såsom HTML5, CSS3 och JavaScript för multiplattformsutveckling av applikationer. Varje applikation exekveras inom den specifika plattformen och förlitar sig på standarder som har åtkomst till varje funktion i enheten, som till exempel sensorer, data, nätverksstatus med mera. I figur 6 ses Apache Cordovas applikationsarkitektur. [22]



Figur 6: Apache Cordova applikationsarkitektur. [23]

Orsakerna till att använda Apache Cordova är:

- Man vill utveckla applikationer till flera än en plattform, utan att behöva implementera applikationen på nytt för varje plattformsspråk och verktyg.
- Man vill distribuera en webbapplikation som är förpackad, för distribution i olika applikationsportaler.
- Man är intresserad av att blanda inbyggda programkomponenter med en WebView, eller tillverkaren vill utveckla ett plugin-gränssnitt mellan inbyggda och WebView-komponenter. [22] [24]

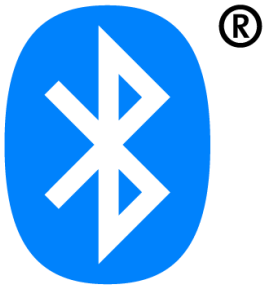
2.5 Bluetooth

Bluetooth används för att trådlöst utbyta data mellan olika enheter som har stöd för Bluetooth, över korta avstånd. Bluetooth använder sig av korta vågor, närmare bestämt UHF (Ultra High Frequency) radiovågor med frekvenser från 2.400 till 2.485 GHz. Bluetooth kan också användas för att bygga upp PANs (Personal Area Networks). [25]

2.5.1 Historia

Namnet Bluetooth är välkänt över hela världen men alla kanske inte vet var namnet härstammar från. Bluetooth kommer från en dansk kung som hade namnet Kung Harald ”Blåtand” Gormsson. Smeknamnet, ”Blåtand” grundar sig på att Kung Harald hade en död tand, som var mörkblå/grå. Kung Harald är också känd för att ha förenat Danmark och Norge år 958.

Bluetooth logon binder samman runorna (Hagall (ᚼ)) och (Bjarkan (ᛒ)), det vill säga Haralds initialer som figur 7 visar. [26]



Figur 7: I Bluetooths logo kan de sammanbundna runorna ᚼ och ᛒ ses. [27]

År 1996 planerade de största industriledarna, Intel, Ericsson och Nokia att standardisera denna kortdistans radioteknologi för att öka och stöda kontakten mellan olika produkter och industrier. Under detta möte föreslog Jim Kardash från Intel att kodnamnet för denna teknologi skulle vara Bluetooth. Kardash blev sedan citerad att ha sagt: *"King Harald Bluetooth…was famous for uniting Scandinavia just as we intended to unite the PC and cellular industries with a short-range wireless link"* (*"Kung Harald Blåtand… var känd för att förena Skandinavien precis som vi hade för avsikt att förena PC- och mobilindustrin med en kortdistans trådlös länk"*). [26]

År 1998

- 5 företag grundade SIG (Special Interest Group). I slutet av 1998 hade SIG 4000 medlemsföretag. Detta år blev Bluetooth det officiella namnet på teknologin. Ett år efter utgavs Bluetooth 1.0. Detta utgjorde grunden för Bluetooth teknologin. Två år efter att SIG hade blivit format tillverkades den första Bluetooth-telefonen, samt det första Bluetoothkortet till datorer. År 2000 demonstrerades också prototyper för Bluetooth datormus.

År 2002

- Utgavs den första Bluetooth tangentbord- och mus-kombinationen, samt GPS-mottagare. Trådlös Bluetooth är integrerat i 500 kvalificerade produkter.

År 2004

- SIG släpper Bluetooth 2.0 EDR (Enhanced Data Rate). Bluetooth är nu installerat på över 250 miljoner enheter. Året därefter öppnas SIG huvudkontor i Washington och regionala kontor öppnas i Sverige och Hong Kong.

År 2007

- Antalet medlemsföretag i SIG överstiger 8000. Följande år fyller Bluetooth 10 år med omkring 2 miljarder Bluetoothaktiverade produkter fraktade över hela världen. Medlemmarna är nu över 10 000 företag. PTS (Profile Tuning Suite) version 3.0 utges. Detta inkluderar automatiska uppdateringar till denna generations egenskaper.

År 2009

- SIG anpassar BLE trådlös teknologi. Denna teknologi är kännetecknande för Bluetooth version 4.0. (Denna teknologi/egenskap används i examensarbetet). Året därefter släpps en formell anpassning av Bluetooth med BLE teknologi för version 4.0.

År 2011

- SIG släpper 29 nya Bluetooth 4.0 profiler, tjänster, protokoll och prototypspecifikationer. Detta utgör infrastrukturen för Bluetooth-enheter. Apple och Nordic Semiconductor ansluter sig till SIG-styrelsen.

År 2016

- Bluetooth version 5 presenteras, vilket innebär nya specifikationer/egenskaper, såsom fyrdubblat avstånd, dubbel hastighet, ökad dataöverföring med en kapacitet på 800 procent. Bluetooth introducerar nya kringutrustningar för utveckling av IoT.

År 2017

- Google och Philips Lighting ansluter sig till SIG styrelsen. Apple, Samung och andra stora smarttelefontillverkare tillägger stöd för Bluetooth 5. Årliga Bluetooth produktförsändelser överskrider 3,6 miljarder.

År 2018

- Bluetooth fyller 20 år. Detta år har SIG 35,000 medlemsföretag. [28]

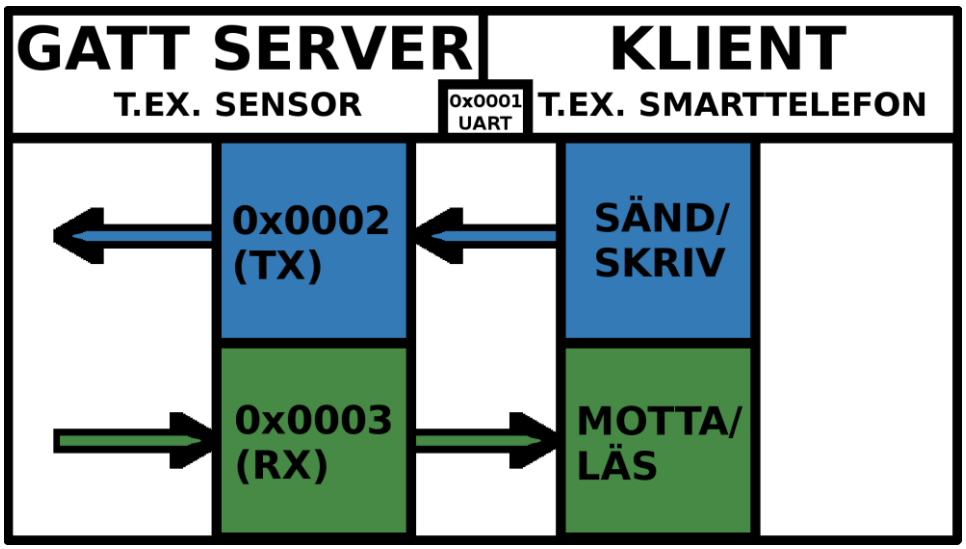
2.5.2 BLE UART

UART är det mest använda protokollet för dataöverföring mellan datorenheter över serieport. Men protokollet UART använder ingen traditionell serieportskommunikation, utan en emulering av detta över BLE. UART-protokollet är en skräddarsydd tjänst (engelskans Service). Denna tjänst har kommit att bli en standard. Så gott som alla Nordic-, Adafruit-, mbed-BLE-hårdvarutillverkare stöder UART-protokollet. [29]

UART-tjänsten använder sig av endera en 128- eller 16-bitars leverantörspecifik UUID på ”6e400001-b5a3-f393-e0a9-e50e24dcca9e” eller ”0x0001”, Dessa är identiska. Denna UUID används nu av varje leverantör som använder sig av UART-tjänsten. UART har två datakanaler, TX och RX. Dessa två datakanaler har såsom UART-tjänsten både en 128- samt 16-bitars UUID. Dessa är för TX ” 6e400002-b5a3-f393-e0a9-e50e24dcca9e” eller ”0x0002” respektive för RX ” 6e400003-b5a3-f393-e0a9-e50e24dcca9e” eller ”0x0003”. UART-tjänsten samt RX- och TX-datakanalen ses i figur 8. UART-tjänsten och datakanalerna implementeras på sensorenheter, till exempel en RuuviTag. Denna sensorenhet ses som en GATT server som figur 9 visar. På GATT servern sätts tillstånd för en utomstående enhet att sända till och/eller ta emot data från GATT servern. [29]

	128-BITAR	16-BITAR
UART-TJÄNST UUID	6e400001-b5a3-f393-e0a9-e50e24dcca9e	0x0001
TX-DATAKANAL UUID	6e400002-b5a3-f393-e0a9-e50e24dcca9e	0x0002
RX-DATAKANAL UUID	6e400003-b5a3-f393-e0a9-e50e24dcca9e	0x0003

Figur 8: 128-bitars UUID respektive 16-bitars UUID.



Figur 9: Enkel illustration över GATT server och klient, samt UART-protokollet med datakanalerna TX och RX.

2.5.3 Tillägget ble-central för BLE Apache Cordova

För att överföringen mellan GATT-servern och klienten skulle vara möjlig, användes ett tillägg för Apache Cordova, nämligen ble-central. Med detta tillägg gjordes kommunikationen möjlig mellan en BLE-kringutrustning (till exempel en omgivningssensor) och en mobil enhet (till exempel en smarttelefon). Detta tillägg är en simpel JavaScript API för IOS och Android. [30] Detta tillägg möjliggör:

- Skanning av kringutrustning
- Anslutning till kringutrustning
- Läsning av värden ur datakanalerna
- Skrivning av värden till datakanalerna
- Uppdatering när en datakanals värde ändrar

Datakanalens annonseringsinformation (eller engelskans Advertising information) returneras när BLE-kringutrustning skannas av en mobil enhet. Tjänst-, datakanal- och egenskapsinformationen returneras då BLE-kringutrustning ansluts till en mobil enhet. All tillgång till BLE-kringutrustningen fås via en BLE-kringutrustnings tjänsts och datakanals UUID. [30]

2.6 Espruino JavaScript Interpreter

Espruino-programvaran är en JavaScript-tolkare av typen öppen källkod som är designad för mikrokontroller som har minst 128 kB lagringsminne och 8 kB RAM-minne. Med denna JavaScripttolkare menas att man kan skriva kod i JavaScript, som sedan översätts till ett lägre programmeringsspråk, vilket i detta fall är programmeringsspråket C. Detta underlättar

programmeringen av mikrokontroller för de som inte kan eller vill programmera i programmeringsspråket C. I och med detta minskar tröskeln för programmering av mikrokontroller, eftersom man inte behöver lära sig lågnivåspråk för programmering av mikrokontroller samt sensorenheter.

Espruino utvecklades av Gordon Williams år 2012. Han ville göra utvecklingen av mikrokontroller möjlig på flera plattformar. Espruino var till en början inte av öppen källkod, men Espruino-programvaran kunde laddas ner gratis till STM32-mikrokontroller. Espruino gjordes till öppen källkod efter en lyckad gräsrotsfinansiering (engelskans kickstarter/crowdfunding) 2013 av en utvecklingskontroll som drevs av Espruino-programvaran. Efter att denna mikrokontroll har blivit släppt, har det kommit många officiella utvecklingskontroller, såsom Espruino Pico, Espruino WiFi, Puck.js med inbyggt Bluetooth och Pixel.js med en inbyggd LC-display och Arduino shield kompatibilitet. Förutom de officiella kontrollerna, drivs omkring 40 olika utvecklingskontroller med Espruino. [31] [32]

2.7 nRF Connect

Applikationen nRF Connect för mobiltelefoner skannar, hittar och ansluter enheter som stöder BLE. nRF Connect för mobiltelefoner stöder ett antal av Bluetooth SIG’s anpassade profiler, men också DFU-profil (Device Firmware Update) av Nordic Semiconductor eller Eddystone av Google (Eddystone används för BLE kommunikation). DFU används för uppdatering av programvaran i till exempel en sensorenhet. [33]

Med nRF Connect kan följande utföras:

- Skanning av BLE-enheter.
- Tolkning av annonserade data.
- Visning av RSSI graf.
- Anslutning till vilken som helst BLE-enhet.
- Hittande och tolkning av tjänster och datakanaler.
- Givande av rättigheter för läsning/skrivning av en datakanal.
- Loggning av händelser och metodanrop.
- Stöder DFU-profil. Som möjliggör användare att ladda upp ny applikation trådlöstfrån en fil (HEX- eller Zip-fil). [33]

Tilläggsfunktioner för nRF applikationen på Android mobiltelefoner:

- Tolkning av värden för de mest kända datakanalerna.
- Konfigurering av GATT server (2.6.2).
- Listning av ihopkopplade enheter.
- BLE annonsering (kringutrustningsroll).
- Skanning, annonsering och upprätthållande av flera anslutningar samtidigt.
- Stöd för DFU-profil. [33]

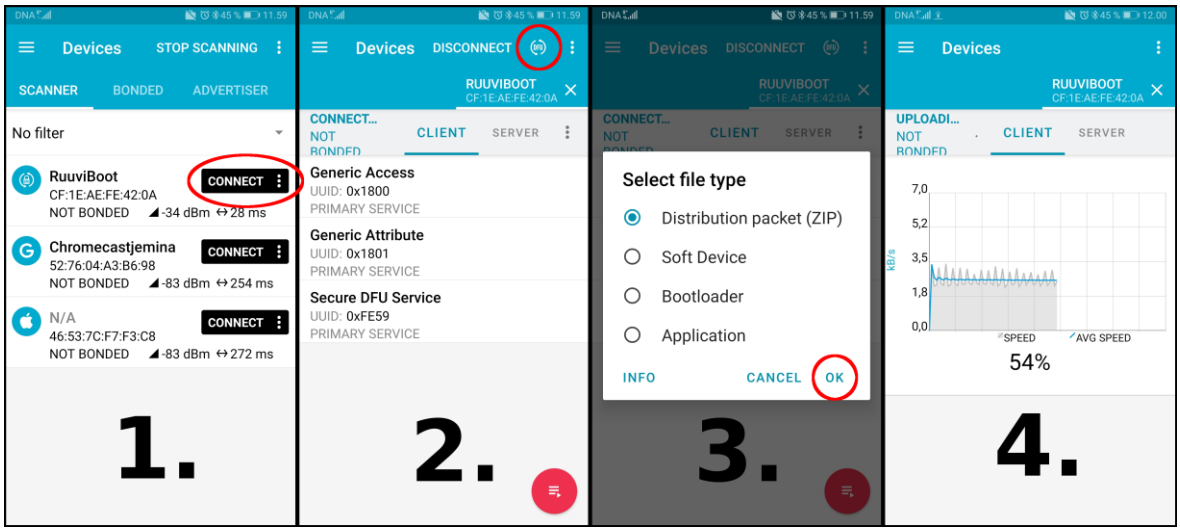
3 Praktiskt utförande

Detta kapitel beskriver det praktiska arbete som gjorts för att möjliggöra detta examensarbete. Här beskrivs också tillvägagångssätt och utförande för att binda samman alla tekniker som använts i examensarbetet till en enda lösning.

3.1 Planering och val av teknik och programmeringsspråk

Planering samt förberedande var en stor del i examensarbetet, eftersom teknologierna som användes i examensarbetet kräver förståelse för BLE och utveckling av mobilapplikationer. Planeringen för hur dessa teknologier skulle sammankopplas samt användas korrekt var nödvändig för att teknologierna skulle bilda en fungerande lösning.

I examensarbetet användes Espruino JavaScript Interpreter (2.6). Programmeringsspråket för programmering av RuuviTagen är standard C, men eftersom tidsschemat var kort, valdes en enklare och snabbare variant, det vill säga Espruino JavaScript Interpreter för RuuviTagen. Detta gjorde programmeringen av RuuviTagen enklare. Installeringen av Espruino på RuuviTgen gjordes med hjälp av en smarttelefon samt en mobilapplikation vid namnet nRF Connect (2.7). nRF Connect användes vid uppdatering av programvaran, för testning av annonsering av data från RuuviTagen samt för att se vilka tjänster och datakanaler RuuviTagen använder sig av. Figur 10 visar överföring och installering av distributionspaketet Espruino JavaScript Interpreter från smarttelefonen till RuuviTagen.

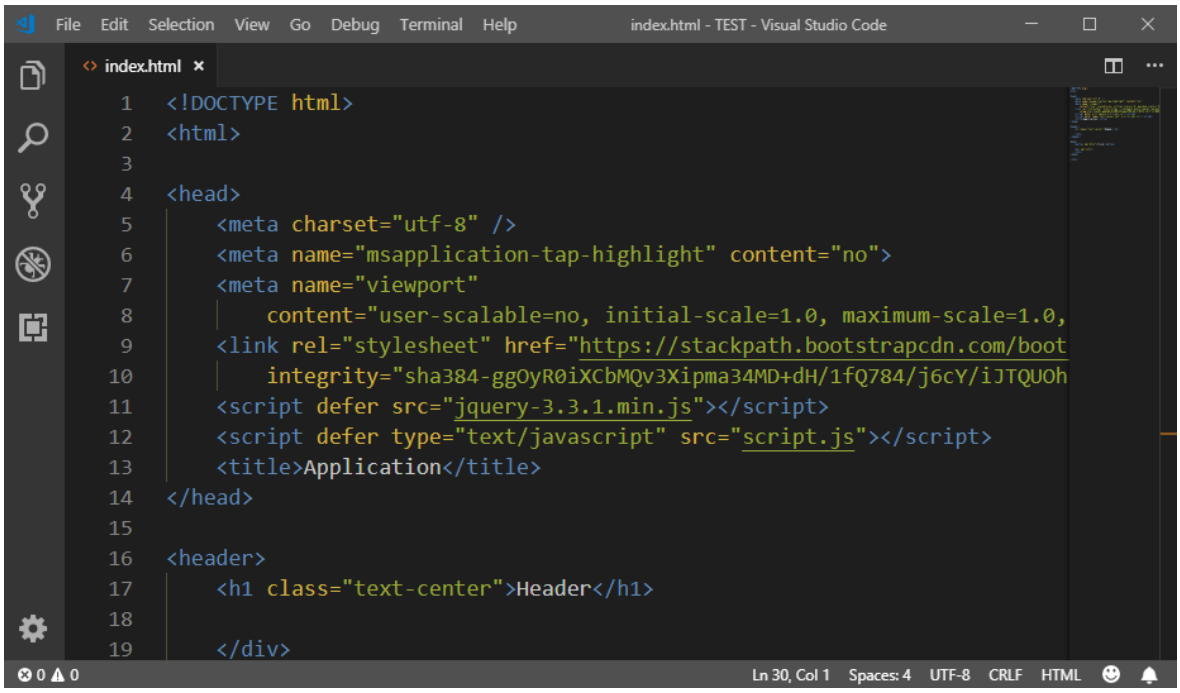


Figur 10: Installering av Espruino JavaScript Interpreter på RuuviTag. 1. Skanning samt val av RuuviTagen (RuuviBoot) över Bluetooth. 2. DFU-val. 3. Val av filtyp för uppdatering. 4. Status för överföring/uppdatering av RuuviTagen från smarttelefonen.

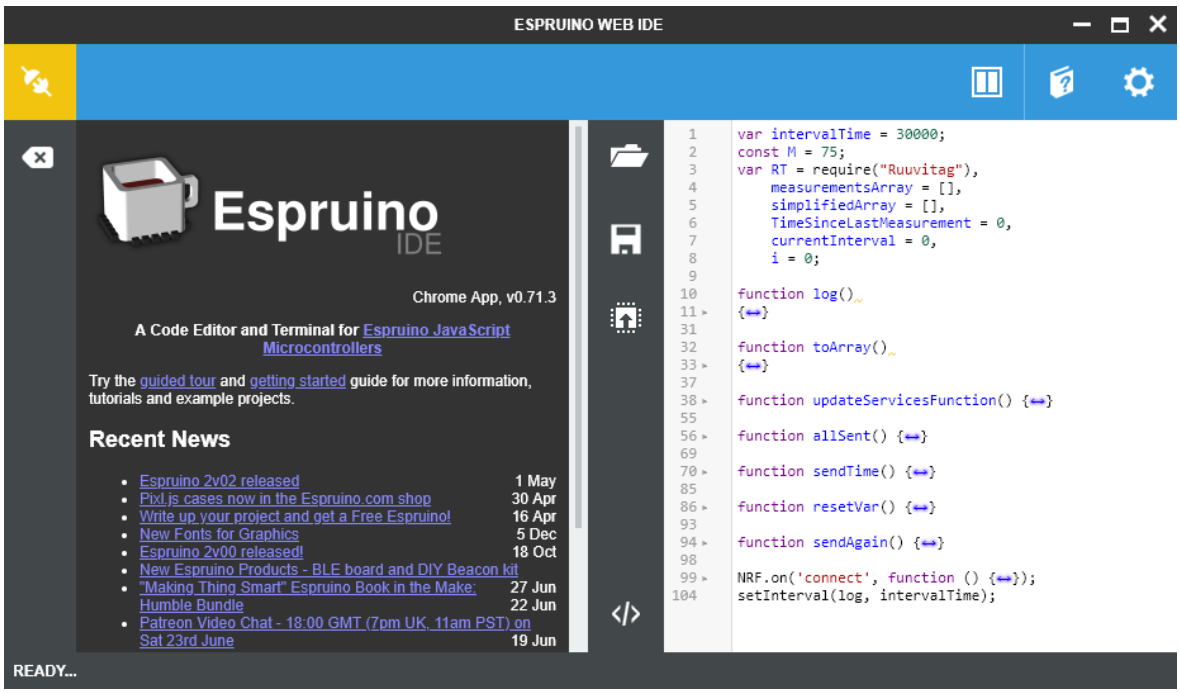
I utvecklingen av mobilapplikationen användes Apache Cordova (2.5.2). Detta för att enkelt kunna utveckla och köra applikationen på olika mobila operativsystem. Apache Cordova sågs som det enda rätta alternativet i examensarbetet, eftersom företaget som examensarbetet utfördes åt, använder sig av detta ramverk för mobilapplikationsutveckling.

3.2 Utvecklingsmiljö

Som utvecklingsmiljö i examensarbetet användes Microsoft Visual Studio Code för utveckling av mobilapplikationen (se figur 11) och för utvecklingen av RuuviTagens program användes utvecklingsmiljön Espruino Web IDE (se figur 12).



Figur 11: Visual Studio Code.



Figur 12: Espruino Web IDE.

3.3 Utveckling av programmet för RuuviTagen

Eftersom Ruuvitagen skulle överföra data till mobilapplikationen över BLE måste programmet för RuuviTagen anpassas för att den skulle kunna överföra data på ett sätt som möjliggör mottagning av data på mottagarsidan. Genom att använda klassbiblioteket NRF, bestämdes vilken data som RuuviTagen annonserar (engelskans advertise). Detta klassbibliotek är gjort specifikt för enheter som använder sig av chiptypen nRF51/nRF52, vilket Ruuvitagen gör.

3.3.1 Insamling av omgivningsdata

För att möjliggöra mätning av omgivningsdata användes klassbiblioteket Ruuvitag. I kodexempel 1 möjliggörs användningen av detta klassbibliotek. Klassbiblioteket innehåller funktioner som möjliggör bland annat aktivering av sensorerna i RuuviTagen. I kodexempel 1 mäts omgivningstemperaturen. Intervallet för hur ofta funktionen skall utföras anges med användningen av setTimeout-funktionen i JavaScript. i kodexempel 1 utförs funktionen ”measureFunction” var 1000:de millisekund, det vill säga varje sekund.

Kodexempel 1: Möjliggör användningen av klassbiblioteket Ruuvitag, aktivering av omgivningssensorer på RuuviTagen och JavaScript setTimeout-funktion.

```
var RT = require("Ruuvitag");

function measureFunction()
 {
     RT.setEnvOn(true);

     RT.getEnvData().temp;

     RT.setEnvOn(false);

     intervalFunction();
 }

 function intervalFunction()
 {
    setTimeout(measureFunction, 1000);
 }
 measureFunction();
```

3.3.2 Lagring av data

Eftersom RuuviTagen inte skulle vara ansluten till en mobil enhet konstant, var ett krav att RuuviTagen skulle lagra data/mätvärden i dess internminne tills någon mobil enhet anslutits samt mottagit data från RuuviTagen. Detta löstes genom att omgivningsdata som RuuviTagen samlade in lagrades i en array/lista. Detta underlättade också överföringen av mycket data, eftersom arrayn då kunde itereras igenom vid annonseringstillfället.

Eftersom RuuviTagen har ett internminne på 512kB behövdes den äldsta uppmätta datan uppdateras mot nyare, mera relevant datan då minnet blev fullt. Detta möjliggjordes med en JavaScript-funktion ”pop”, denna funktion ersätter äldre data med nyare data i en array/lista. Arrayn tömdes också efter att den mobila enheten fått all data av RuuviTagen. Mätningarnas Intervall var en halv minut så länge minnet inte var fullt. När minnet blev fullt skulle intervallet vara 15 minuter.

I RuuviTagen rymdes lagring av omkring 70 positioner i en array. Varje position innehåller tid mellan de uppmätta värdena, temperatur, lufttryck och luftfuktighet. På RuuviTagen finns också lagrat Espruino JavaScript Interpreter samt koden som körs konstant på RuuviTagen.

3.3.3 Överföring av data med BLE UART från RuuviTagen

För att möjliggöra överföring av data från RuuviTagen, användes klassbiblioteket NRF. Med detta klassbibliotek är det möjligt att bestämma vad och hur RuuviTagen annonserar data (engelskans advertise). Data sänds/annonseras med hjälp av protokollet UART för BLE (2.6.2). Datan som skulle överföras från RuuviTagen till en ansluten mobil enhet var följande:

- Temperatur i grader Celsius (°C)
- Lufttryck i hekto Pascal (hPa)
- Luftfuktighet i procent (%)
- Tid sedan senaste uppmätta värdet i sekunder vid anslutningstillfället i sekunder (s)

För att annonsera denna data användes klassbiblioteket NRF's funktion ”updateServices”, som fungerar enligt följande. I koden bestämdes vilken tjänst och datakanal som skulle användas, i detta fall UART-tjänsten och RX-datakanalen. RX-datakanalen skulle uppdateras med relevanta mätvärden. Detta åstadkoms med inställning av vad och hur data annonserades, såsom vilket/vilka värden som skulle annonseras, längden på anonserade datan, samt andra alternativa inställningar. Data som överförs med UART över BLE är av typen ”ArrayBuffer”. För att underlätta läsandet samt användandet av data som publicerats på RuuviTagen lades ett kommatecken (,) mellan varje mätvärde, detta för att kunna skilja på de olika mätvärdena. För att möjliggöra annonseringen av data från RuuviTagen, krävdes att RuuviTagen skulle vara ansluten via BLE till en mobil enhet (se kodexempel 2). Om detta inte uppfylldes, publicerades ingenting, men omgivningsdata samlades likväl in på RuuviTagen.

Kodexempel 2: Innehållet i ”NRF.on” utförs här endast då RuuviTagen anslutits.

```
NRF.on('connect', function ()
{
    i = 0;
    setTimeout(sendTime, 500);
});
```

3.4 Mottagning av data från RuuviTagen på mottagarsidan

Mottagning av data med BLE UART kräver formatering. Eftersom datan annonserades av RuuviTagen och mottogs på mottagarsidan av typen ArrayBuffer, krävdes formatering till en konventionell array, detta för att kunna läsa samt använda datan på mottagarsidan. Med användning menas uträkning av tidpunkt för mätningen, lagring av data i applikationen, med mera.

3.4.1 Formatering samt hantering av inkommande data på mottagarsidan

Datan som mottagarsidan mottog formaterades till en 8-bitars ArrayBuffer, detta eftersom en 8-bitars array var det mest användbara i examensarbetet. Eftersom datan skulle presenteras för användaren, var det mest användbart och passande i denna situation att formatera datan till en sträng. För att uppnå detta måste 8-bitars arrayn itereras igenom, för att möjliggöra formateringen mellan en 8-bitars array till en sträng. Formateringen kunde se ut som i kodexempel 3. Eftersom strängen med data var kommaseparerad kunde delar från strängen plockas ut för att senare användas enligt behov. Ett sådant exempel kan ses i kodexempel 4.

Kodexempel 3: Formatering från ArrayBuffer till en sträng.

```
var RTDataUint8Array = new Uint8Array(RTdata);

var RTDataString = "";

for (var i = 0; i < RTDataUint8Array.length; i++)
{
    RTDataString += String.fromCharCode(RTDataUint8Array[i]);
}
```

Kodexempel 4: Ur valueString-strängen plockas temperaturen, lufttrycket, luftfuktigheten och intervallet ut.

```
valueString = dataRead.split(',', 4);
temperatureVal = valueString[0];
pressureVal = valueString[1];
humidityVal = valueString[2];
intervalValue = valueString[3];
```

3.4.2 Tidsbestämning av mätvärden

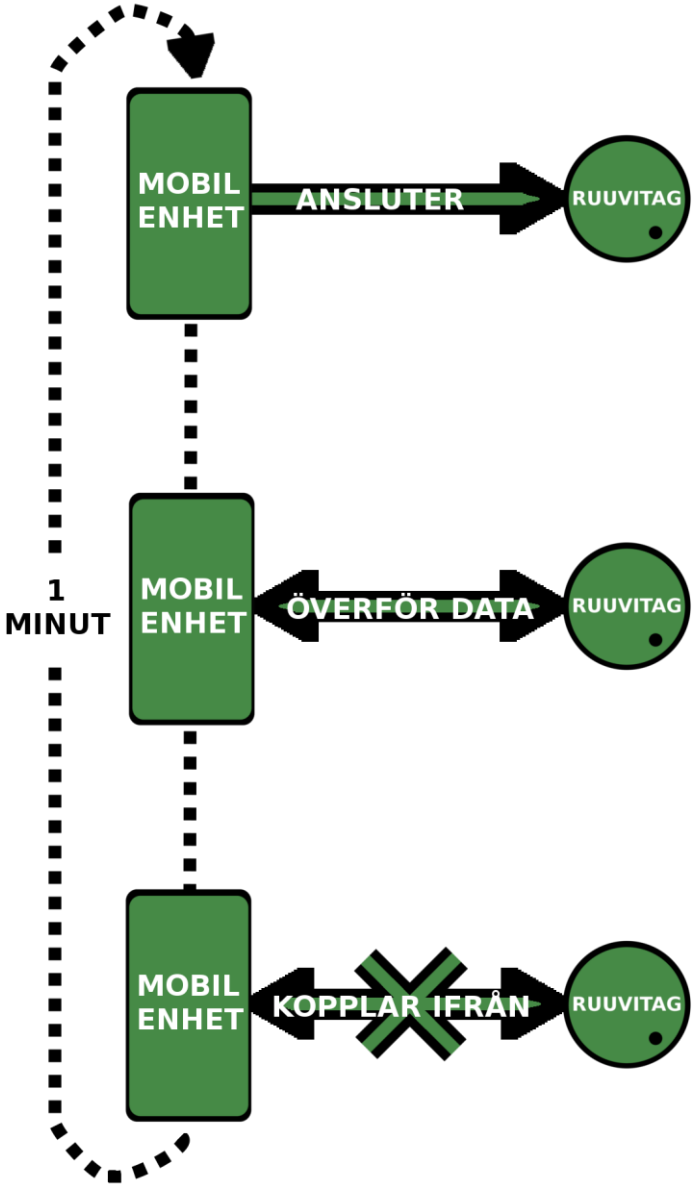
För att applikationen skulle vara användbar för uppföljning samt dokumentering, krävdes tidsbestämning av alla uppmätta värden. Eftersom RuuviTagen skulle växla mellan två olika intervall, det vill säga 30 sekunder och 15 minuter, behövde funktionalitet som räknade ut tidpunkten vid mätningstillfällena implementeras i mobilapplikationen. Den Mobila enheten måste få tag på intervallet mellan de uppmätta värdena, samt tiden sedan det senaste uppmätta värdet vid anslutningstillfället.

Varje position i arrayn innehållande all mätvärdesdata, innehöll också en siffra på sista positionen, denna siffra kunde vara en tvåa (2) för 30 sekunders intervall eller en etta (1) för 15 minuters intervall, mellan mätvärdena. RuuviTagen annonserade också tiden i sekunder sedan det senast uppmätta värdet. Utgående från dessa siffror och tiden sedan det senaste uppmätta värdet, räknade applikationen ut klockslaget när mätvärdena hade blivit uppmätta.

3.5 Automatisering av mobilapplikation

Med automatisering menas att programmet eller applikationen inte behöver någon mänsklig interaktion för att utföra åtgärder. Om användaren vill köra mobilapplikationen medan hen gör något annat eller inte har möjlighet att interagera med applikationen, behövde mobilapplikationen göras mera automatiserad. Men automatiseringen gjordes inte enbart ur användarens perspektiv, utan också ur ett energieffektivt perspektiv.

Då den mobila enheten anslöts, samt då mottagaren hade mottagit data från RuuviTagen framgångsrikt, kopplades den mobila enheten ifrån RuuviTagen. Varefter ett tidsintervall startades på mobilapplikationen. Detta intervall skulle varje minut ansluta den mobila enheten till RuuviTagen och motta mätvärdena som RuuviTagen annonserat. Detta sparar på strömmen i batteriet som driver RuuviTagen, samt gör applikationen mera användarvänlig och automatiserad. Händelseförloppet ses i en illustration i figur 13.



Figur 13: Simpel illustration över automatiseringen av mobilapplikationen.

Eftersom intervallet mellan varje mätning ändrades, beroende på om arrayn innehållande mätvärdena var full eller inte (3.3.2), krävdes att arrayn innehållande mätvärdena skulle tömmas på all data, när all data blivit framgångsrikt annonserad och mottaget på mottagarsidan. Tömningen utfördes med klassbiblioteket ble-centrals (2.5.3) ”write”-funktion på mottagarsidan. Buffern som skickas till RuuviTagen måste skrivas med hexadecimala värden, eftersom data skickas som en ”ArrayBuffer”. Detta kan ses i kodexempel 5. Mottagarsidan skickade ett kommando i form av en funktion som togs emot av RuuviTagen, detta kommando utförde sedan en funktion på RuuviTagen, som hade som uppgift att tömma arrayn och återställa alla variabler till standardvärden som kan ses i kodexempel 6.

Kodexempel 5: Mottagarsidan skickar kommandot ”resetVar()\n” till RuuviTagen.

```
resetCommand = new Uint8Array(12);

resetCommand[0] = 0x72; // r
resetCommand[1] = 0x65; // e
resetCommand[2] = 0x73; // s
resetCommand[3] = 0x65; // e
resetCommand[4] = 0x74; // t
resetCommand[5] = 0x56; // V
resetCommand[6] = 0x61; // a
resetCommand[7] = 0x72; // r
resetCommand[8] = 0x28; // (
resetCommand[9] = 0x29; // )
resetCommand[10] = 0x3b; // \
resetCommand[11] = 0x0a; // n

ble.write(idOfDevice, bluefruit.serviceUUID, bluefruit.txCharacteristic, resetCommand.buffer, writeSuccess, error);
```

Kodexempel 6: Funktionen ”resetVar” utförs då RuuviTagen mottar ”resetVar()\n” från mottagarsidan.

```
function resetVar() {
    measurementsArray = [];
    SimplifiedArray = [];
    TimeSinceLastMeasurement = 0;
    i = 0;
    j = 0;
}
```

3.6 Felhantering av data

För att motta korrekt data, implementerades felhantering på mottagarsidan. Det effektivaste sättet att göra detta, var att kontrollera ifall all relevant data blivit annonserat korrekt från RuuviTagen samt att mottagaren mottagit rätt data.

Eftersom RuuviTagen kunde lagra omkring 70 positioner i en array, var det viktigt att all omgivningsdata som fanns lagrad på RuuviTagen hade blivit skickad framgångsrikt, innan arrayn innehållande all omgivningsdata tömdes. För att möjliggöra tömningen av arrayn implementerades ett simpelt felhanteringssystem på RuuviTag- samt mobilapplikationssidan. Då RuuviTagen hade skickat all omgivningsdata som arrayn innehållit, publicerades slutligen en bekräftelse i form av en sträng till mottagarsidan, att all data har blivit annonserad. Men om mottagarsidan inte fått emot denna bekräftelse, skickade mottagarsidan ett kommando till RuuviTagen att datan inte blivit annonserad eller framgångsrikt mottagen, varvid

RuuviTagen upprepade annonseringen av alla mätvärden. Detta upprepas ända tills mottagarsidan fått bekräftelsen av RuuviTagen att all data med mätvärdena blivit framgångsrikt annonserade från RuuviTagen samt mottagna på mottagarsidan.

3.7 Design av mobilapplikationens användargränssnitt

Användargränssnittet är en viktig del inom applikationstillverkning. Användargränssnittet måste vara tydligt, lätt att använda samt självförklarande. Om dessa mål inte uppfylls får användaren en dålig upplevelse. Detta kan resultera i att användaren håller sig borta eller totalt bojkottar applikationen. För att uppfylla målen för ett bra användargränssnitt måste applikationens utseende samt funktionalitet planeras noggrant.

Eftersom det endast finns en sida på denna applikation, tog planeringen av utseendet inte så lång tid, utan fokus kunde läggas på funktionaliteten samt automatiseringen av applikationen. Men fastän applikationen endast har en sida, måste användargränssnittet tas i beaktande. Mobilapplikationen använder sig av plugin/insticksmodulen Bootstrap, för att få en så responsiv mobilapplikation som möjligt.

Planering och design av utseendet gjordes med hjälp av Pencil Project. Pencil Project är ett gratisprogram med öppen källkod som används för att designa prototyper för grafiska användargränssnitt. I figur 14 ses design av mobilapplikationen i Pencil Project. [34]

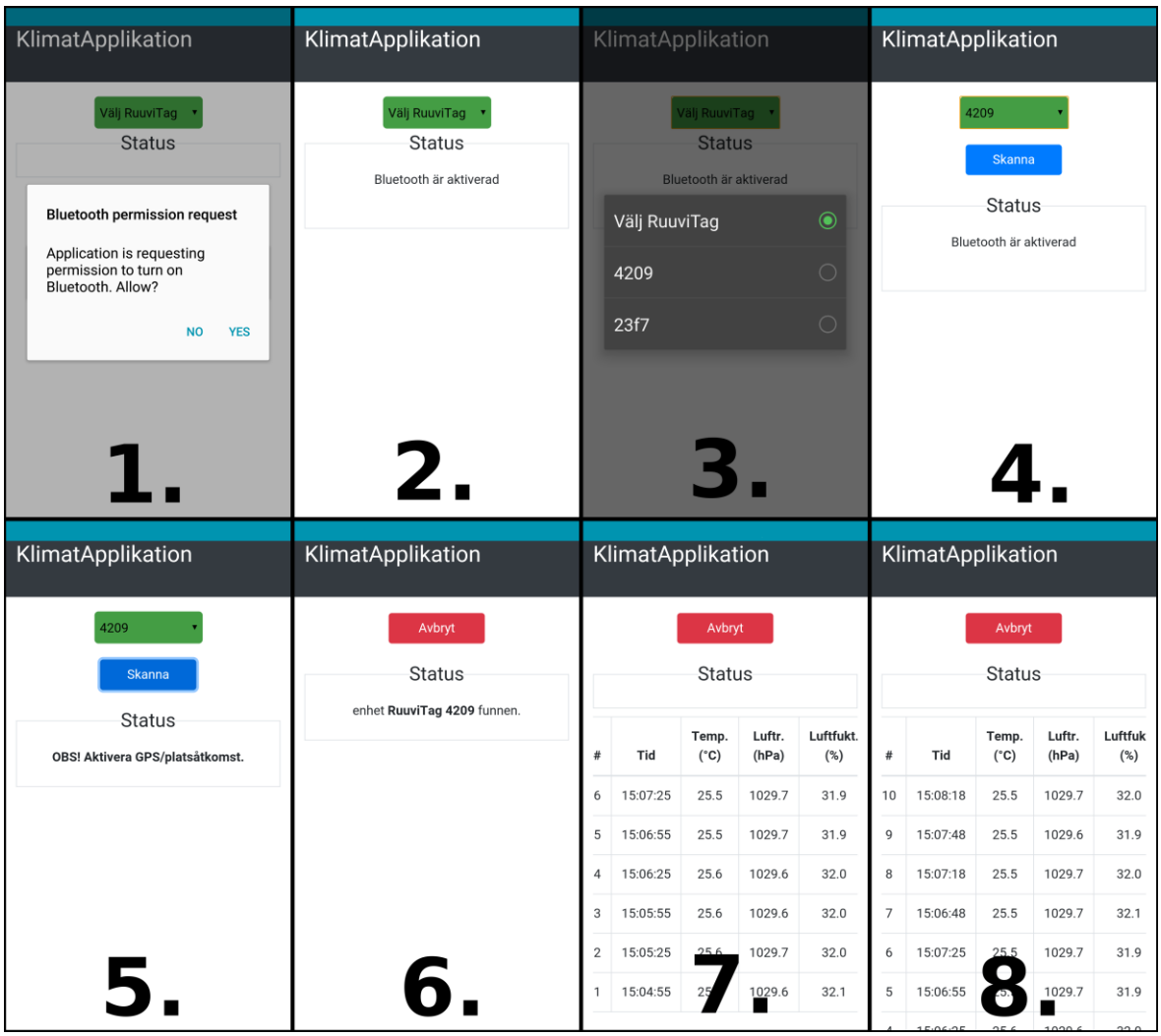
Figur 14: design av mobilapplikationen i programmet Pencil Project.

För att göra applikationen så användarvänlig som möjligt finns endast dessa knappar på applikationen:

- Val av RuuviTag
- Skanning av den valda RuuviTagen, (Denna utför också anslutningen av RuuviTagen då den hittats. Automatiseringen av mobilapplikationen påbörjas)
- Avbryta skanningen och fortsatt mottagning av data från RuuviTagen
- Tillbaka till toppen av tabellen

Den sistnämnda knappen är inte nödvändig, men finns med för att underlätta användningen och för att snabbt komma till de senaste värdena högst uppe i tabellen, om användaren befinner sig längst nere i tabellen. I figur 14 ses också en knapp rensa värden. Denna knapp togs inte med i det slutgiltiga användargränssnittet. Det slutgiltiga Användargränssnittet kan ses i figur 15. Förklaringar av användargränssnittet i figur 15:

1. Applikationen frågar lov om att aktivera Bluetooth på den mobila enheten.
2. Användaren meddelas om att Bluetooth är aktiverat.
3. Användaren väljer vilken RuuviTag hen vill motta data från.
4. Applikationen försöker påbörja skanning. Men GPS/platsåtkomst måste vara aktiverad för att genomföra skanning av BLE-enheter och kommunikation mellan enheterna.
5. Användaren meddelas om att GPS/platsåtkomst måste vara aktiverad.
6. Skanningen lyckades. Mobilapplikationen fann den valda Ruuvitagen.
7. Mobilapplikationen har nu hämtat data från Ruuvitagen och visar datan för användaren.
8. Mobilapplikationen utför automatisk skanning, anslutning samt överföring av data. Detta upprepas med ett intervall på 1 minut, ända tills användaren trycker på knappen ”Avbryt”.



Figur 15: Det slutgiltiga användargränssnittet.

4 Resultat

Målet med examensarbetet var uppföljning samt dokumentering av omgivningens temperatur, lufttryck och luftfuktighet. Med sensorenheten RuuviTag och en mobil enhet med stöd för BLE. Applikationen skulle vara lätt att använda. Applikationen skulle vara automatiserad till viss del. Användaren skulle enkelt kunna kolla upp omgivningsdata på en valfri personlig mobil enhet med stöd för BLE.

Resultatet avspeglar inte fullt ut det ursprungliga målet. Resultatet innehåller fullt fungerande kommunikation mellan sensorenheten RuuviTag och en mobil enhet med stöd för BLE. Omgivningsdata samlas in på sensorenheten och överförs sedan till valfri enhet med stöd för BLE. Från början var avsikten att inkludera överföring av data över internet från mobilapplikationen till en databas på en server. Uppgiften begränsades senare så att man

kunde koncentrera sig på det huvudsakliga i examensarbetet, det vill säga överföring av data med BLE teknologin.

5 Diskussion

Slutresultatet blev inte riktig som jag tänkt mig, på grund av flera olika orsaker samt utmaningar på vägen. Den största orsaken till att examensarbetet inte blev som planerat var tidsbrist samt att efterforskningen tog lång tid. Detta berodde främst på att BLE är en teknologi som jag inte varit bekant med eller kommit i kontakt med tidigare. Utvecklingen av koden till RuuviTagen var utmanande, inte på grund av programmeringsspråket JavaScript, utan eftersom det fanns ett begränsat antal hjälpmedel för sensorenheten RuuviTag.

Fastän slutresultatet inte blev som målet och syftet beskrivits, så är jag helt nöjd med slutresultatet. Det mest utmanande var att förstå hur BLE fungerar samt hur man skulle överföra data över BLE. En annan utmaning var att räkna ut tiden för tillfället omgivningsdata hade blivit uppmätt. Detta är utmanande eftersom batteriet i RuuviTagen kan ta slut utan att användaren vet när. Det är därför viktigt att få en logisk och smidig lösning till detta. Problemet vara att klockan återställs på RuuviTagen när batteriet inte har tillräckligt med ström för att driva RuuviTagen. Detta löstes också, vilket tog ganska lång tid.

Själva designen på mobilapplikationen tycker jag är användarvänlig men kunde förbättras. Design av applikationer är en viktig del i applikationstillverkning och med tillräckligt mycket tid samt kunskap och testning skulle mobilapplikationen kunna göras mera professionell och smidigare.

Examensarbetet har varit mycket intressant men frustrerande ibland och tidskrävande. Eftersom jag inte alltid förstått hur allting skulle fungera samt hänga ihop testade jag många olika lösningar tills jag till sist hittade en fungerande lösning. Jag har lärt mig grunderna inom dataöverföring över BLE, vilket är mycket användbart i dagens samhälle, eftersom trådlös kommunikation implementeras i så gott som alla mobila enheter. Jag är mycket tacksam över att ha fått utföra detta arbete, eftersom det har varit mycket lärorikt, intressant och gett mig en insikt i hur enheter kommunicerar över BLE.

6 Fortsatt utveckling

Den fortsatta utvecklingen av examensarbetet kommer inte att utföras av arbetstagaren som utfört examensarbetet. Tanken med hur examensarbetet kommer användas är att implementera lösningen i en existerande applikation. Denna applikation skall logga temperaturen i ett fraktfordons fraktutrymme, till exempel ett fraktfordon som fraktar kylvaror. Användaren skall se loggad temperatur samt varningar ifall temperaturen avviker de nominella värdena för temperaturen inne i fraktutrymmet.

7 Källförteckning

[1] ”Nyholm Solutions,” Nyholm Solutions, [Online]. Available: http://nyholmsolutions.fi/sv/. [Använd 2019].

[2] P. Lea, ”IoT versus machine to machine,” i *Internet of Things for Architects*, Birmingham, Packt Publishing Ltd., 2018, p. 28.

[3] M. Rouse, ”TechTarget IoTAgenda,” TechTarget, Maj 2018. [Online]. Available: https://internetofthingsagenda.techtarget.com/definition/machine-to-machine-M2M. [Använd 2019].

[4] OPTO 22, ”OPTO 22,” OPTO 22, [Online]. Available: https://www.opto22.com/about-us/history. [Använd 2019].

[5] OPTO 22, ”OPTO 22,” OPTO 22, [Online]. Available: https://www.opto22.com/about-us. [Använd 2019].

[6] M. Rouse, ”TechTarget WhatIs.com,” TechTarget, Juli 2012. [Online]. Available: https://whatis.techtarget.com/definition/sensor. [Använd 2019].

[7] P. Lea, ”Sensing devices,” i *Internet of Things for Architects*, Birmingham, Packt Publishing Ltd., 2018, p. 40.

[8] Ruuvi Innovations Ltd (OY) / Finland, ”Ruuvi,” Ruuvi Innovations Ltd (OY) / Finland, [Online]. Available: https://ruuvi.com/about-us/. [Använd 2019].

[9] Ruuvi Innovations Ltd (OY) / Finland, ”Ruuvi,” Ruuvi Innovations Ltd (OY) / Finland, [Online]. Available: https://ruuvi.com/. [Använd 2019].

[10] Nordic Semiconductor, ”Nordic Semiconductor,” Nordic Semiconductor, [Online]. Available: https://www.nordicsemi.com/Products/Low-power-short-range-wireless/nRF52832. [Använd 2019].

[11] Nordic Semiconductor, ”infocenter.nordicsemi.com,” Nordic Semicondutor, 11 10 2017. [Online]. Available: https://infocenter.nordicsemi.com/index.jsp. [Använd 2019].

[12] STMicroelectronics, ”STMicroelectronics,” STMicroelectronics, 2019. [Online]. Available: https://www.st.com/content/st_com/en/products/mems-and-sensors/accelerometers/lis2dh12.html#quickview-scroll. [Använd 2019].

[13] Robert Bosch GmbH, ”Bosch Sensortec GmbH,” Robert Bosch GmbH, 2019. [Online]. Available: https://www.bosch-sensortec.com/bst/products/all_products/bme280. [Använd 2019].

[14] L. Jämsä, ”Ruuvi Blog,” Ruuvi Innovations Ltd (OY) / Finland, 8 September 2017. [Online]. Available: https://blog.ruuvi.com/datasheet-52fb00265c60. [Använd 2019].

[15] Arduino, ”store.arduino,” Arduino, 2019. [Online]. Available: https://store.arduino.cc/arduino-uno-rev3. [Använd 2019].

[16] spcomputing, ”forum.arduino,” Arduino, 13 Oktober 2012. [Online]. [Använd 2019].

[17] Rapberry Pi foundation, ”Rapberry Pi,” Raspberry Pi foundation, [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/. [Använd 2019].

[18] Wikipedia, ”Wikipedia The free encyclopedia,” Wikimedia foundation, 8 Februari 2019. [Online]. Available: https://en.wikipedia.org/wiki/General-purpose_input/output. [Använd 2019].

[19] Arduino, ”store.arduino,” Arduino, 2019. [Online]. Available: https://store.arduino.cc/grove-temp-humi-barometer-sensor-bme280. [Använd 2019].

[20] P. Viswanathan, ”lifewire,” Dotdash, 19 Februari 2019. [Online]. Available: https://www.lifewire.com/what-is-a-mobile-application-2373354. [Använd 2019].

[21] Statista, ”Statista,” Statista, Maj 2018. [Online]. Available: https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/. [Använd 2019].

[22] Apache Cordova, ”Apache Cordova,” The Apache Cordova Foundation, 2015. [Online]. Available: https://cordova.apache.org/docs/en/latest/guide/overview/index.html.

[23] Apache Cordova, ”Apache Cordova,” The Apache Software Foundation, [Online]. Available: https://cordova.apache.org/docs/en/latest/guide/overview/index.html. [Använd 2019].

[24] Computer Hope, ”Computer Hope,” Computer Hope, 13 November 2018. [Online]. Available: https://www.computerhope.com/jargon/p/plugin.htm. [Använd 2019].

[25] Wikipedia, ”Wikipedia the free encyclopedia,” Wikimedia foundation, 3 April 2019. [Online]. Available: https://en.wikipedia.org/wiki/Bluetooth. [Använd 2019].

[26] Bluetooth SIG, Inc., ”Bluetooth,” Bluetooth SIG, Inc., 2019. [Online]. Available: https://www.bluetooth.com/about-us/bluetooth-origin. [Använd 2019].

[27] [Online]. Available: https://www.bluetooth.com/. [Använd 2019].

[28] Bluetooth SIG, Inc., ”Bluetooth,” Bluetooth SIG, Inc., 2019. [Online]. Available: https://www.bluetooth.com/about-us/our-history. [Använd 2019].

[29] T. GN, ”Thejesh GN,” Thejesh GN, 1 Oktober 2016. [Online]. Available: https://thejeshgn.com/2016/10/01/uart-over-bluetooth-low-energy/. [Använd 2019].

[30] D. Coleman, ”github,” github, Inc., 13 Juni 2018. [Online]. [Använd 2019].

[31] G. Williams, ”GitHub,” GitHub, Inc., 2019. [Online]. Available: https://github.com/espruino/Espruino. [Använd 2019].

[32] Wikipedia, ”Wikipedia the free encyclopedia,” Wikimedia foundation, 31 Juli 2018. [Online]. Available: https://en.wikipedia.org/wiki/Espruino. [Använd 2019].

[33] Nordic Semiconductor, ”nordic semiconductor,” Nordic Semiconductor, [Online]. Available: https://www.nordicsemi.com/?sc_itemid=%7B41FF7A0B-B565-420A-95B7-B32122B5D3AD%7D. [Använd 2019].

[34] Evolus, ”pencil project,” Evolus, [Online]. Available: https://pencil.evolus.vn/. [Använd 2019].

[35] M. Santos och E. Moura, ”What is IoT?,” i *Hands-On IoT Solutions with Blockchain*, Birmingham, Packt Publishing Ltd., 2019, pp. 7-8.

[36] Espruino, ”Espruino ide,” Espruino, [Online]. Available: https://www.espruino.com/ide/. [Använd 2019].