SmartCommunity

Communitary Platform of Sensors

Francisco Monteiro Vilaça Pereira

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. Alberto Manuel Ramos da Cunha

Examination Committee

Chairperson: Prof. Ana Maria Severino de Almeida e Paiva
Supervisor: Prof. Alberto Manuel Ramos da Cunha
Member of the Committee: Prof. Renato Jorge Caleira Nunes

November 2019
Acknowledgments

I would like to thank my parents for their friendship, encouragement and caring over all these years, for always being there for me through thick and thin and without whom this project would not be possible. I would also like to thank my grandparents, aunts, uncles and cousins for their understanding and support throughout all these years.

I would also like to acknowledge my dissertation supervisors Prof. Alberto Cunha, Joel Teixeira and Mário Romano for their insight, support and sharing of knowledge that has made this Thesis possible.

Last but not least, to all my friends and colleagues that helped me grow as a person and were always there for me during the good and bad times in my life. Thank you.

To each and every one of you – Thank you.
Abstract

This master thesis addresses the problem of data collection. Current solutions are implemented using technologies like LoRaWAN and GSM. The problem is that it comes attached with infrastructural costs that can be avoided with the utilization of a technology like Bluetooth Low Energy. This study is important because it evaluates a cheaper alternative for data collection that takes advantage of the people that brings their smartphone everywhere they go. To do this, a prototype system for gathering the sensor data called SmartCommunity was developed. The system consists of four components: the Sensor Nodes, an Android application, a Backend and an Endpoint for publishing the transported data. It was concluded that this can be a viable solution for sensor data monitoring that offers security and overall reliability in the delivery of data from sensors to the cloud with the participation of the smartphone as the bridge between the two, evaluating the possibility to use the application in cases where we are walking, driving an electric scooter or bike and also the power consumption of the smartphone while running the service, as well as the possibility to encrypt sensor data.

Keywords

Internet of Things
Resumo

Esta tese de mestrado trata do problema de colheita de dados. As soluções atuais são implementadas usando tecnologias como LoRaWAN e GSM. O problema é que ele vem acompanhado de custos de infraestrutura que podem ser evitados com a utilização de uma tecnologia como Bluetooth Low Energy. Este estudo é importante porque avalia uma alternativa mais barata para a coleta de dados que tira proveito das pessoas que levam seus smartphones aonde quer que vão. Para isso, foi desenvolvido um sistema de protótipo para coletar os dados do sensor chamado SmartCommunity. O sistema consiste em quatro componentes: os nós dos sensores, um aplicativo Android, um back-end e um terminal para publicar os dados transportados. Concluiu-se que essa pode ser uma solução viável para o monitoramento de dados de sensores que oferece segurança e confiabilidade geral na entrega de dados dos sensores para a nuvem com a participação do smartphone como ponte entre os dois, avaliando a possibilidade de usar o aplicativo nos casos em que estamos caminhando, dirigir uma scooter ou bicicleta elétrica e também o consumo de energia do smartphone durante a execução do serviço, além da possibilidade de criptografar os dados do sensor.

Palavras Chave

Internet das Coisas;
## Contents

1 Introduction ......................................................... 1
   1.1 Problem .......................................................... 3
   1.2 Opportunity ....................................................... 3
   1.3 Research Goal .................................................... 4
   1.4 Proposed Idea .................................................... 5
   1.5 Motivation ......................................................... 5
   1.6 Environmental Monitoring ....................................... 6
   1.7 Background ........................................................ 6
      1.7.1 Portugal Climate Changes .................................. 6
      1.7.2 Growth of Population ........................................ 6
   1.8 M2M ................................................................. 8
   1.9 Internet of Things ................................................ 8
      1.9.1 Importance of IoT .......................................... 9
      1.9.2 Future ........................................................ 9
   1.10 Smart Cities ....................................................... 10
      1.10.1 Smart Cities in Portugal .................................... 11
      1.10.1.A Examples ................................................ 11
   1.11 Organization of the Document .................................. 11

2 Literature Overview .................................................. 13
   2.1 Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications .... 15
   2.2 Participatory Sensing enabled Environmental Monitoring in Smart Cities ............................ 15
   2.3 Smart Environmental Monitoring Beacon Study ......................................................... 16
   2.4 Internet of Things for Environmental Monitoring Study ............................................... 17
   2.5 Design of an IoT based Real Time Environment Monitoring System Study .......................... 17
   2.6 A Smart City Environmental Monitoring Network and Analysis Relying on Big Data Techniques .. 18
   2.7 A Comparison of WiFi and Bluetooth Low Energy for Region Monitoring .......................... 19
2.8 Choosing a Microcontroller ........................................... 19
  2.8.1 Easy-to-get Microcontroller .................................. 19
  2.8.2 Energy Efficiency ................................................. 19
  2.8.3 Cost per Unit ...................................................... 19
  2.8.4 Development Kit ................................................ 20
  2.8.5 Documentation .................................................. 20

2.9 Choosing the Wireless Technology .................................. 20
  2.9.0.A ZigBee ............................................................ 20
  2.9.0.B Wi-Fi .............................................................. 20
  2.9.0.C Bluetooth Low Energy ...................................... 21

3 Proposed Solution ......................................................... 23
  3.1 Overview ............................................................. 25
  3.2 Proposed Scenario .................................................. 25

3.3 Logical Architecture .................................................. 25
  3.3.1 Sensor Node ....................................................... 26
    3.3.1.A Architecture ................................................. 26
    3.3.1.B Algorithm .................................................... 27
  3.3.2 Transporter Node ................................................. 28
    3.3.2.A Algorithm .................................................... 28
    3.3.2.B Keeping the device Awake ................................ 31
    3.3.2.C Avoiding Duplicates ....................................... 31
  3.3.3 Backend ........................................................... 32
  3.3.4 Cloud Endpoint .................................................. 32

3.4 Technical Architecture ............................................... 33
  3.4.1 Stack of Technologies ......................................... 34
  3.4.2 PlatformIO ........................................................ 35
  3.4.3 Android ............................................................ 35
    3.4.3.A SQLite ......................................................... 35
  3.4.4 Firestore ........................................................... 35

3.5 Implementation ........................................................ 36
  3.5.1 Sprint 1 ............................................................. 36
    3.5.1.A Outcome ....................................................... 36
  3.5.2 Sprint 2 ............................................................. 36
    3.5.2.A Outcome ....................................................... 36
  3.5.3 Sprint 3 ............................................................. 36
3.5.3.A Outcome ................................................. 37
3.5.4 Sprint 4 .................................................... 37
3.5.4.A Outcome .............................................. 37
3.5.5 Sprint 5 .................................................... 37
3.5.5.A Outcome .............................................. 37
3.5.6 Sprint 6 .................................................... 37
3.5.6.A Outcome .............................................. 37
3.5.7 Sprint 7 .................................................... 38
3.5.7.A Outcome .............................................. 38
3.5.8 Sprint 8 .................................................... 38
3.5.8.A Outcome .............................................. 38
3.5.9 Sprint 9 .................................................... 38
3.5.9.A Outcome .............................................. 38
3.5.10 Sprint 10 .................................................. 38
3.5.10.A Outcome ............................................ 39
3.5.11 Sprint 11 .................................................. 39
3.5.11.A Outcome ............................................ 39
3.5.12 Sprint 12 .................................................. 39
3.5.12.A Outcome ............................................ 39
3.5.13 Sprint 13 .................................................. 39
3.5.13.A Outcome ............................................ 40
3.5.14 Sprint 14 .................................................. 40
3.5.14.A Outcome ............................................ 40
3.6 Used Hardware ............................................... 40
3.6.1 ESP32 ....................................................... 40
3.6.1.A Light Sleep ............................................ 40
3.6.1.B Deep Sleep ............................................. 40
3.6.1.C Firebase Support ..................................... 41
3.6.2 DHT11 ....................................................... 42
3.6.3 Special Attentions ....................................... 42
3.6.4 Huawei P8 Lite .......................................... 42
4 Evaluation ..................................................... 45
4.1 First Prototype Test ........................................ 47
4.2 Multiple Sensor Test ....................................... 48
4.3 Data Collection Time Test .................................. 48
List of Figures

1.1 Classical Sensor Monitoring Architecture. ......................... 3
1.2 Smartphones. ....................................................... 3
1.3 Lime User. ......................................................... 4
1.4 Gira User. .......................................................... 4
1.5 Gira Application. .................................................... 5
1.6 Air Pollution. ....................................................... 7
1.7 Smartphone city users. ............................................. 7
1.8 M2M in the context of this thesis ................................ 8
1.9 M2M and IoT ....................................................... 8
1.10 Internet of Things components. ................................ 9
1.11 Smart Cities components. ....................................... 10
1.12 AYR Credits Application. ....................................... 12

2.1 Structure of the WSN platform. .................................... 15
2.2 Left: The Environmental Sensor Device. Middle and right: Main screen and measurement
details shown in the mobile application. ............................ 16
2.3 System Overview ................................................... 17
2.4 System Architecture. .............................................. 18
2.5 Vehicle with electronic monitoring unit. ......................... 18
2.6 Zigbee technology. ............................................... 21
2.7 Wi-fi technology. .................................................. 21
2.8 BLE technology. ................................................... 22
2.9 BLE vs ZigBee vs Wi-fi ........................................... 22

3.1 Proposed Usage Scenario. .......................................... 25
3.2 Order of Operation. ................................................ 26
3.3 Sensor Node Data Structure. ..................................... 26
3.4 Sensor Loop
3.5 Advertising Stopping Condition
3.6 Starting Scenario
3.7 Device Discovered Scenario
3.8 Device Connected Scenario
3.9 Service Discovered Scenario
3.10 Characteristic Read Scenario
3.11 SQLite data structure
3.12 Wifi Connection Scenario
3.13 Cloud Interaction
3.14 Backend Data Structure
3.15 Firestore Security Rule Implemented
3.16 How collected data is saved
3.17 Proposed Architecture
3.18 Stack of Technologies
3.19 ESP32 Microcontroller
3.20 DHT11 Sensor
3.21 Transporter Node
3.22 Transporter Node Specs
4.1 Sensor readings Test
4.2 Multiple Sensor readings Test
4.3 Test with one person and 2 sensors
4.4 Android Application Scanning Times Log
4.5 Data Collection Time Graphic
4.6 Circumference Characteristics
4.7 Rope Equations
4.8 Real Scenario
4.9 Velocity Equation
4.10 Distance from the center of the circumference to the center of the rope
4.11 Distance from the sensor based equation
4.12 Calculating the length of the rope based on the cathetus
4.13 Supported velocity (m/s) based on distance (m) from the center of the sensor
4.14 Supported velocity (m/s) based on distance (m) from the center sensor
4.15 Two Users Test
4.16 Energy Profiler
Contents

1.1 Problem ................................................................. 3
1.2 Opportunity ......................................................... 3
1.3 Research Goal ...................................................... 4
1.4 Proposed Idea ...................................................... 5
1.5 Motivation .......................................................... 5
1.6 Environmental Monitoring ................................. 6
1.7 Background .......................................................... 6
1.8 M2M ................................................................. 8
1.9 Internet of Things ................................................... 8
1.10 Smart Cities ........................................................ 10
1.11 Organization of the Document ................................. 11
1.1 Problem

The sensor monitoring architecture is divided into three layers: Sensors, Wireless Connectivity and Data Access. The main problem that this thesis addresses is related with the Wireless Connectivity. There are several options to tackle the connectivity between the sensors and the final destination. What is the most practical and affordable way to collect sensor data? Should it be using GSM? Should it be using LoRaWAN or could it be better to use WIFI? Can a ZigBee and Bluetooth take care of the job?

![Figure 1.1: Classical Sensor Monitoring Architecture.](image)

1.2 Opportunity

Nowadays everyone has a smartphone right? Can it be a possible way of transportation between sensors and the Cloud?

![Figure 1.2: Smartphones.](image)
Maybe Wi-Fi, Bluetooth Low Energy or Zigbee could be possible solutions to tackle the Wireless Connectivity problem.

1.3 Research Goal

This study will be focused on evaluating the possibility of gathering sensor data while casually driving an electric bike or a scooter.

![Lime User](image)

**Figure 1.3:** Lime User.

This evaluation is going to be important to validate a scenario where the developed application prototype could be piggybacked on an Application like Gira.

![Gira User](image)

**Figure 1.4:** Gira User.
1.4 Proposed Idea

The proposed idea for this thesis is to design a data gathering system that should provide an alternative way for relaying information from sensors to the Cloud, in order to avoid costs that are inherent to solutions based on technologies like LoRaWAN and GSM [1]. The proposed way of transportation between the sensors and the Cloud is the smartphone.

1.5 Motivation

The motivation for doing this research is to explore a new way for sensor monitoring based on the existence of a transport that should link each sensor to the Cloud.

The purpose for doing this thesis will be fulfilled by answering to this questions:

1. Is it possible to use people’s smartphone as way of transport between the sensors and the Internet?

2. Can I capture sensor data while walking, cycling or even driving?

3. Can the information be encrypted?

4. Can power consumption be an issue?

By evaluating these research questions, this thesis has the goal of serving as a starting point for business value creation for the context of Smart Cities.
1.6 Environmental Monitoring

This study will be more focused on existing solutions based on the sensor data collected from the environment since the sensor used for testing makes temperature and humidity measurements. The importance of this area of research is undeniable.

Environmental Monitoring is an important and highly active research area.

The observation of current values and trends of environmental parameters, such as temperature or level of harmful gases in the air, provides data that can help the detection of hazardous events and the assessment and implementation of appropriate actions in the case of climate change, population growth, urban sprawl, invasive species, and habitat destruction [12].

1.7 Background

In this section, there will be an overview of the importance of the monitoring of big cities data, especially environmental-based ones.

1.7.1 Portugal Climate Changes

The 2030 United Nations climate change scenario predicts fatal changes in the global ecosystem. And Portugal will be one of the countries of the European Union that will be most harmed by these changes.

According to the UN report, the world has until 2030 - 12 years from now - to prevent worsening global warming by 1.5 degrees Celsius, a figure that could be reached due to the increase in greenhouse gas emissions.

The 1.5 Celsius rise could bring sea levels up by 10 centimeters by 2100, which will also impact rainfall and droughts, which will bring even more damage.

According to experts contacted by TVI, “Portugal is the country in Europe that will suffer most from climate change”, due to its “extensive coastal zone”. Rising sea levels are one of the biggest changes if no action is taken against global warming [2].

Experts also speak of the worsening water and air quality caused by rising temperatures and influenced by forest fires that could turn into “mega fires at the expense of drought and heat”.

1.7.2 Growth of Population

The growth of the world population, as well as the increase of the population living in urban areas, results in additional pressures on the spaces, ecosystems, infrastructures and also in the way the population live their lives. The deterioration of the air quality is one of the problems associated with this growth [3].
Therefore, assessing, monitoring and informing populations about urban environmental quality then becomes an essential, particularly important issue as a decision-making tool that contributes to building more sustainable cities [4].

The global population in urban environments is expected to double between 2010 (2.6 billion) and 2050 (5.2 billion) [6]. Cities will have to meet the challenges of growth, performance and competitiveness [7]. Therefore, new strategies will have to be developed to improve cities performance and sustainability.

The transformation of cities into Smart Cities allows for the creation of new socio-economic strategies where Citizens, businesses and governments can access services and resources more efficiently [8].
1.8 M2M

This thesis explores the concept of communication Machine to Machine (M2M), smartphone to sensor more precisely. It can be defined as any technology that allows the devices connected to exchange information and perform actions without human intervention [27].

There is a concept that is based on the axis that connects several M2M vertical solutions. It is called Internet of Things and it represents the potential interconnection of all smart objects, M2M and the way in which they interact with society.

1.9 Internet of Things

Internet of Things is a paradigm was born in 1999 by the hand of Kevin Ashton, sometimes called the "Inventor of IoT", that defined IoT as a system where the Internet is connected to the physical world via ubiquitous sensors [10].

It is basically interconnecting the physical world with the virtual world and applying this concept to all things, opening up new possibilities in the sense of being able to at any time access anything from any place.
“Things” in the physical world are objects that physically exist and from the perspective of IoT we are able to sense, operate, and connect to these things, while in the virtual world “things” are objects that can be stored, accessed, and processed.

1.9.1 Importance of IoT

When something is connected to the internet, it means that it can send and receive information. It can become a smart thing that doesn’t need to have super storage or a super computer inside of it. It just needs to be connected to a super storage or a supercomputer (here enters the concept of cloud computing)

![Figure 1.10: Internet of Things components.](image)

1.9.2 Future

One prediction is that the number of connected things in the world will have a thirtyfold increase between 2009 and 2020.

The reason Internet of Things has become so huge depends partly on Moore's Law and Koomey’s Law.

Kevin Ashton states that these two laws have together enabled us to create powerful and energy efficient computers By turning the graph for Moore’s law upside down it can be interpreted as the size of
a computer is halved every two years. Doing the same thing to Koomey’s law can be interpreted as the amount of energy needed to perform a computation is dropping at a rapid rate.

Combining these interpretations tells us that we can perform the same amount of computations on increasingly a smaller chip, while consuming decreasing amounts of energy - hence computations are becoming more energy efficient.

The potential result is a small, powerful, and energy efficient computer which enables us to provide more advanced services using less chip area and at a lower energy that what has been possible before.

1.10 Smart Cities

The application of the IoT paradigm to an urban environment is of particular interest, as it responds to the governments to adopt information and communications technologies (ICT) solutions in the management of public affairs.

A smart city is a municipality that uses information and communication technologies to increase operational efficiency, share information with the public and improve both the quality of government services and citizen welfare.

This can be achieved by deploying a communication infrastructure that provides cohesive, simple and inexpensive access to a overabundance of public services.

So, it is clear that the accessibility of different types of data, which is collected by a persistent IoT system, may also be utilized to promote actions of the local government toward its residents in order to fully improve the awareness of people about the status of the city their are living in, stimulating the active
participants of the citizens in the management of public administration, and also stimulate the building of new services provided by the IoT.

1.10.1 Smart Cities in Portugal

There is a side, in many Portuguese cities, that the citizens themselves don’t really know about: how the technology is changing their lives [11].

There are regions in Portugal that are investing in the concept of smart city to give the citizens and the tourists a "five star" life quality.

1.10.1.A Examples

1. **Porto** - AYR Credit is a mobile application that simulates a virtual wallet of credits based on CO2 emissions saved. When contributing to this, the user will accumulate credits that can be used to buy products. The current prime-minister António Costa has already had coffee for free thanks to this application.

2. **Faro** - The first Smart City Center of Operations in Algarve is located at Lagoa, and is where monitorization of data that is being collected by sensors in water and waste supply systems.

3. **Portalegre** - Portalegre is preparing for a connected future. The city is betting on a LoRA network for long range low energy communication. There are already garbage bins and water counters connected to the Internet.

4. **Braga** - Transportes Urbanos de Braga - In Braga, the bus is not only a vehicle, it is also an WiFi access point and a mobile sensor of the quality of the air, temperature and humidity. They are also generators of data for the company that manages these vehicles.

5. **Viana do Castelo** - there are more than 10 thousand water smart counters that are able to make real time and remotely sensor collected data.

The recurrent evolution of technology has contributed to the creation of low-cost devices with high processing power and wireless communications like the well-known Raspberry PI and the Arduino.

1.11 Organization of the Document

This thesis is organized as follows: Chapter 1 is a general introduction to the overall study. In chapter 2 there is a more technical approach to the concept of Smart City and an overview of the related work.

In Chapter 3 the proposed solution is described. Hardware used is described in ?? How the designed solution was evaluated is discussed in chapter 4. Chapter 5 gives final thoughts to this study.
Figure 1.12: AYR Credits Application.
2 Literature Overview

Contents

2.1 Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications ............................................. 15
2.2 Participatory Sensing enabled Environmental Monitoring in Smart Cities ................................................................. 15
2.3 Smart Environmental Monitoring Beacon Study .......................................................... 16
2.4 Internet of Things for Environmental Monitoring Study .............................................................. 17
2.5 Design of an IoT based Real Time Environment Monitoring System Study ................................................... 17
2.6 A Smart City Environmental Monitoring Network and Analysis Relying on Big Data Techniques ........................................ 18
2.7 A Comparison of WiFi and Bluetooth Low Energy for Region Monitoring .................................................. 19
2.8 Choosing a Microcontroller .............................................................. 19
2.9 Choosing the Wireless Technology .............................................................. 20
Now there will be a brief description of the studies that were analyzed during my research that contributed to the solution that I proposed.

2.1 Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications

This study shows how a classic Wireless Sensor Network is typically structured.

![Diagram of WSN Platform](image)

**Figure 2.1:** Structure of the WSN platform.

The sensor nodes connect directly to the gateways using a short range communication channel, and each gateway autonomously connects to the server. The gateways process, store, and periodically send the field data to the application server using long-range communication channels. The application server provides long-term data storage, and interfaces for data access and process by end users (either human or other applications) [40].

2.2 Participatory Sensing enabled Environmental Monitoring in Smart Cities

This study is important to my thesis due to the fact that the proposed idea is somehow similar to the one I propose with the slight difference that the sensors are not maintained in fixed positions in specific locations. The sensor is transported by the person, like the smartphone.
A system demonstrator for environmental monitoring based on participatory sensing. Mobile environmental sensors carried by citizens are used to measure pollutant concentrations and provide this data to a Unified Sensing Platform [39].

![Figure 2.2: Left: The Environmental Sensor Device. Middle and right: Main screen and measurement details shown in the mobile application.](image)

The sensor used is a battery and solar cell powered mobile hand held device that is designed to be used by pedestrians and bicyclists. The microprocessor is connected to sensor components to detect pollutants.

Measurements are taken and transmitted to a smartphone that have the Environmental Sensing App installed, via Bluetooth.

The Environmental Sensing app was developed to receive measurements from the sensor device, showing measurements to the user and transmitting the information to the USP.

### 2.3 Smart Environmental Monitoring Beacon Study

This study introduced me to a technology that I did not know and so the paper was able to show me how Firebase could be helpful to design systems for sensor data collection. The downside of this solution is that a GSM module is needed for the microcontroller. So here is a brief description of how the system works.

The designed and created beacon integrates environmental sensors reading, localization, data backup and dispatch to cloud in a portable and stable way. The data is stored in the cloud through an internet connection using a GSM module.

The statistics are available on iOS and Android applications [15].
Using a custom Python library, all the readings are backed-up on the device under an encoded JSON file. After building the JavaScript objects, the library tries to access the Firebase cloud database in order to synchronize last readings.

2.4 Internet of Things for Environmental Monitoring Study

This study showed me another way for monitoring sensor data. The downside of it is that Wi-Fi access is needed and therefore an Access Point. Here is a brief description of how the system works.

The study is based on the existence of an IoT service that acts as an MQTT broker that is responsible for distributing messages to connected clients [23].

2.5 Design of an IoT based Real Time Environment Monitoring System Study

This study was important for my thesis because it made me discover the ESP8266 microcontroller that is a previous version of the ESP32, the microcontroller that I used as sensor node prototype. The downside of this solution is that the microcontroller sends its collected information through HTTP, and therefore using the Wi-Fi module, so there is a need for Internet Access in the sensor side. Here is a brief description of the study.

This study describes the design of a prototype for real time monitoring of environmental conditions using low cost sensors. This study proposes a solution based on captured data that is broadcasted
through internet with an ESP8266 Wi-Fi module. The projected system delivers sensors data to an API called ThingSpeak over an HTTP protocol and allows storing of data. The proposed system works well and it shows reliability [16].

2.6 A Smart City Environmental Monitoring Network and Analysis Relying on Big Data Techniques

This study showed me another way for monitoring real time data, this time the proposed solution is based on an electronic monitoring unit which is mounted on top of a moving vehicle that collects data at an adaptive rate based on the vehicle's speed. The speed is detected via GPS information from a smartphone that is placed inside the vehicle and connected via 3G or 4G.

![Figure 2.4: System Architecture.](image)

![Figure 2.5: Vehicle with electronic monitoring unit.](image)
The smartphone should later relay the sensors readings to a cloud server that contains databases that should store data in real-time [19].

2.7 A Comparison of WiFi and Bluetooth Low Energy for Region Monitoring

This study provided an evaluation regarding power consumption between two wireless technologies, Bluetooth Low Energy and Wi-Fi, both of them candidates as a technology for broadcasting sensor data. Here is a brief summary of the study.

The paper presents results from region monitoring implemented as an app for Android smartphones using WiFi and the low power protocol Bluetooth Low Energy respectively. Both networks are compared regarding accuracy and the power consumption on the mobile device [20].

Regarding power consumption, the benefit of BLE is impressive. BLE nearly doubled the runtime of the mobile device from 37 hours up to 70 hours when the device tries to localize its position.

2.8 Choosing a Microcontroller

This survey showed the important factors to have in mind in order to choose the right microcontroller for the desired job. All the following factors were analysed when I chose the ESP32. It is easy to get, energy efficient, cheap, has a development kit and good documentation all over the Internet.

2.8.1 Easy-to-get Microcontroller

Is it easy to get the microcontroller that you want and in the quantity that you need? This is important to consider at the beginning of the process, especially if the plan is to scale up the system later on.

2.8.2 Energy Efficiency

Energy efficiency is extremely important to consider for IoT applications because there is a need to minimize the need for sending maintenance crews to inspect edge infrastructure.

2.8.3 Cost per Unit

How much does each unit cost? It is important to think about scaling the project up later on. Make sure that the IoT budget support including more of the microcontrollers.
2.8.4 Development Kit

Development kits are an excellent way to get started with the microcontroller because they are designed to give customers an out-of-box experience.

2.8.5 Documentation

Is good documentation for the microcontroller available? These factors are crucial in order to make informed decisions on how to use your microcontroller properly. A good online community can help guide the development [21].

2.9 Choosing the Wireless Technology

This study provided an overview over the available wireless technologies used to communicate between a smartphone and a sensor node. It was important in order to choose Bluetooth Low Energy as the one.

As many of the devices are small low-cost devices, the radio must not add too much additional cost, taking into consideration the fact that the radio and device applications in many cases need to share the same computing engine (micro-controller). Many use cases require a battery or some kind of energy harvesting technology as a power source.

It must be easy to associate a device with the network and with the Internet Services.

The authentication and the encryption must be adequately supported by the wireless technology.

It needs to have the possibility to connect to smart phones.

It is necessary to have the capability to cover enough of a range or to have some capabilities to extend the coverage without having too big of an impact on the system cost.

The most widely used protocols in applications having less tight latency and reliability requirements consist of ZigBee, Wi-Fi and Bluetooth [22].

2.9.0.A ZigBee

ZigBee is extensively used in a wide range of monitoring and control application that require wireless connectivity. These solutions provide energy-efficient designs, but cannot comply with tight latency and reliability requirements and require additional hardware for packaging data and for transmitting them to the Internet.

2.9.0.B Wi-Fi

Wi-Fi is a popular networking technology based on the IEEE 802.11 set of standards that offers higher transmission range and throughput compared to IEEE 802.15.4, with the cost of higher energy con-
2.9.0.C Bluetooth Low Energy

Bluetooth Low Energy was introduced in 2010 with the goal of extending the application domain of Bluetooth to power-constrained devices, such as wireless sensors. The implementation of monitoring applications with Internet connectivity requires the use of Bluetooth Internet gateways for sending and receiving measurement data.

All three technologies have built-in link layer authentication and encryption, which sometimes needs to be completed with end-to-end security from the sensor to the web application. Bluetooth Low Energy has the potential for less power consumption than IEEE 802.15.4. The lack of native support for IEEE 802.15.4 in mobile devices is a problem, especially for mobile or temporary mobile use cases. BLE and IEEE 802.11 devices are easy to associate with a network and mainly with Internet services, while this
is often more difficult with IEEE 802.15.4 devices.
3 Proposed Solution

Contents

3.1 Overview ......................................................... 25
3.2 Proposed Scenario ............................................. 25
3.3 Logical Architecture .......................................... 25
3.4 Technical Architecture ....................................... 33
3.5 Implementation ............................................... 36
3.6 Used Hardware ................................................ 40
3.1 Overview

This chapter describes the solution that was proposed in order to tackle the problem of using the smartphone as a transport between sensor nodes and the cloud, using a low range wireless technology, guaranteeing privacy and avoiding spoofing of sensor nodes in order to provide a reliable way for collecting sensor data in the context of a Smart City.

3.2 Proposed Scenario

The proposed scenario is based on the assumption that is possible to use the citizens smartphone to relay information to the Cloud. The idea is to collect data from Sensor Nodes that are distributed in specific places and that are periodically advertising data that they collected. The information transmitted from the sensor node to the smartphone should be encrypted. The data collected by each smartphone should be saved in a local database in order to be later relayed to the cloud when there is Wi-fi connection available. This is the typical scenario that drove this study.

![Proposed Usage Scenario](image)

**Figure 3.1:** Proposed Usage Scenario.

3.3 Logical Architecture

The usual data transportation scenario is represented in the above figure. This is a simple way of representing what is happening.
The data in the point 3 is encrypted using AES 128 algorithm.

The components were already presented but their functionality wasn’t, so, the next subsections will explain what each component does during the proposed scenario.

### 3.3.1 Sensor Node

The Sensor Node is defined by a micro controller equipped with sensors that is able to collect information from those sensors and display them to mobile gateways that appear in his range.

#### 3.3.1.A Architecture

![Diagram of Sensor Node Data Structure](image)
Every Sensor Node has the same data structure. It is based on the Bluetooth Low Energy specification. Figure 3.3 shows how data is organized inside each device.

### 3.3.1.B Algorithm

Each Sensor Node has a loop running, from time to time, that makes it collect data and later advertise it. It also has times when it isn’t doing anything to in order to lower power consumption.

![Sensor Loop Diagram](image)

**Figure 3.4:** Sensor Loop.

The main loop is represented above. Initially the Sensor Node verifies if the difference between the current date and the initial data collect date was already reached. If the desired value was already reached, the initial date is refreshed.

After that, the micro controller asks the sensor to make data collections. Each collected value is written in persistence memory to avoid data loss if something happens to the Sensor Node.

![AdvertisingStoppingConditionDiagram](image)

**Figure 3.5:** Advertising Stopping Condition.

The condition for stop advertising is represented above. It is based on the verification of the actual situation of the Sensor Node. It basically verifies if it is advertising and if the time interval between the current date and the date of the beginning of the advertisement has already been reached. The time interval is represented like that because for evaluation purposes it was a small one. During the actual scenario it is expected to be much bigger in order to make only advertise several times a day.
3.3.2 Transporter Node

The Transporter Node is defined by each mobile gateway that has the job of collecting data that is being provided by each Sensor Node.

3.3.2.A Algorithm

In order to make possible the collection of sensor data, there are some steps that must be followed.

![Figure 3.6: Starting Scenario.]

This operation is represented by a service that runs in the background and that, when it is triggered, initializes the Bluetooth Adapter. Then it gets the information from the Sensor Nodes that are saved in the Backend and then it starts scanning for those specific devices.

![Figure 3.7: Device Discovered Scenario.]

When the Transporter Node detects a device that is listed in the system it initiates a connection with
him using the technology Bluetooth Low Energy. It creates a callback that later will be called, when the connection is successful established.

![Diagram](image)

**Figure 3.8:** Device Connected Scenario.

When the connection is considered successful the Transporter Nodes starts the service discovery.

![Diagram](image)

**Figure 3.9:** Service Discovered Scenario.

For each service discovered it checks if the service is listed and it iterates over it’s characteristics. Those characteristics are also verified. If some characteristic was already read it will not be read again during this connection. This happens to prevent duplicate readings. Finally a callback is created in order to later read the value that the characteristic is providing.

When a characteristic is successfully read, the Transporter Node proceeds to get it’s value saving it into the an SQLite database, together with the description of the value, the name of the transmitting device and with the specific timestamp. The description can be "Humidity" or “Temperature” for example. When the information is successfully persisted in local storage the android application sends a toast to notify the user if he happens to be with the smartphone screen on. Note that the toast was only made for testing purposes.
Figure 3.10: Characteristic Read Scenario.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Device</th>
<th>Timestamp</th>
</tr>
</thead>
</table>

Figure 3.11: SQLite data structure.

In the figure 5.14 it is possible to see how data is organized for each collected value stored in the local database in each Transporter Node.

Figure 3.12: Wifi Connection Scenario.

The values that are locally stored in the SQLite Database are all still encrypted. In order to decrypt
them, and also, in order to send them to the endpoint in the Cloud there is a need to have Internet connection.

So, the designed solution has a Broadcast Receiver [24], that listens to changes in the Wi-Fi connection. When it is triggered, the local database is queried in order to get all the collected data. For each row in the table, the application calls a Cloud Function that is responsible for decrypting each value. This design choice was made based on the premise that sensitive data should not be exposed to the Transporter Node.

For each Cloud Function call, there is also another call to the Firestore to publish the values that were stored in the local DB with each value already in plaintext.

### 3.3.2.B Keeping the device Awake

In order to keep the smartphone awake, to collect data, even when the screen is off a WakefulBroadcastReceiver had to be implemented.

A WakefulBroadcastReceiver is a broadcast receiver that passes off the work to a Service, while ensuring that the device does not go back to sleep in the transition. If a wake lock is not hold while transitioning the work to a service, the device will go back to sleep before the work completes.

### 3.3.2.C Avoiding Duplicates

During one scan period the transporter node only searches one time for each characteristic. So when a person is passing by a sensor it will only get a value for each registered characteristic. Even if this wasn’t possible, this could be solved by having the advertising intervals that each sensor has, and each transporter node could, when passing near a sensor, see if the difference between the last reading and the current date is bigger than the advertising interval, this can tell if the collected value was already gathered during the respective sensor node advertising interval.
3.3.3 Backend

The third component it’s called the Backend and it is responsible for having all the information regarding the Sensor Nodes that are registered.

![Backend Data Structure](image)

Figure 3.14: Backend Data Structure.

The data structure is represented in Figure 5.15. It is based on the existence of 4 collections.

The top of the hierarchy is represented by the collection of Devices, that in the overall architecture of the solution are represented as the Sensor Nodes. It stores each Device information in documents. Each document is identified by it's ID and each one of them has associated to it two collections. The first collection is called Services. It is populated with documents that represent each Service provided by each Device. Each Service is identified by it's ID, and is characterized by it's description and it's UUID. It also has a collection of the characteristics it provides. Each characteristic is identified in the collection by an ID, and has a description and an UUID as it’s attributes.

3.3.4 Cloud Endpoint

The component is the Endpoint that is characterized by a collection that is referenced by each Device document and it represents the collected data that is being published by each Transporter Node. Each document that represents each value collected is identified by it's timestamp so it can be possible to sort each collected data by it's timestamp.

The technology used as the Backend and the Cloud Endpoint supports Security Rules that avoid not authenticated users to write or read specific fields, documents or even entire collections.

This is important, specifically in the case of the keys. Each Device document has a key associated with it. In order to protect that information the Firebase Security Rules should successfully address that
3.4 Technical Architecture

The solution is based on the existence of four entities. The first one is the Sensor Node that is the source of information. It collects the data and advertises it.

The second one is the Transporter Node that is basically represented by each smartphone that offers the possibility of relaying sensor data to the Internet.

The third one is the Backend that is responsible for saving information related to the Sensor Nodes, including the key of each of them.
The last one is the Endpoint that has the goal of receiving the data transported by each Transporter Node.

### 3.4.1 Stack of Technologies

<table>
<thead>
<tr>
<th>Sensor Node</th>
<th>Transporter Node</th>
<th>Backend</th>
<th>Endpoint</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Sensor Node" /></td>
<td><img src="image" alt="Android" /></td>
<td><img src="image" alt="Cloud Firestore" /></td>
<td></td>
</tr>
</tbody>
</table>

The technologies that helped design the entire solution were the ones that are represented on the previous diagram. The Sensor Node software was developed with the help of PlatformIO.

The application that should run on the Transporter Node was developed with Android Studio.

The Backend, as well as the Endpoint, were developed using the Cloud Firestore provided by Google.
3.4.2 PlatformIO

The main problem which repulses people from the embedded world is a complicated process to setup development software for a specific microcontroller/board: toolchains, proprietary vendor’s IDE and what is more, to get a computer with OS where that software is supported. Multiple hardware platforms require different toolchains, IDEs and, respectively, spending time on learning new development environments. Finding proper libraries and code samples showing how to use popular sensors and actuators. Sharing embedded projects between team members, regardless of an operating system they prefer to work with.

To tackle these problems, PlatformIO is cross platform code builder and library manager with platforms like Arduino support. It takes care of toolchains, debuggers, frameworks that work on most popular platforms like Windows, Mac and Linux. It supports more than 200 development boards along with more than 15 development platforms and 10 frameworks.

3.4.3 Android

Android was the OS chosen to develop the application due to the fact that the smartphone used to develop the solution was an Huawei P8 Lite that runs the Android Operating System.

The platform used for designing the Android application code was Android Studio.

3.4.3.A SQLite

SQLite is a relational database management system contained in a C library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program and it was the main reason that led me to choose this technology to implement the local database to save Sensor Data.

3.4.4 Firestore

Cloud Firestore is a cloud-based NoSQL database, which Google positions as a replacement for the Realtime Database. It allows the storing of data on a remote server that easily allows access to the data and lets the monitoring of changes in real time. Firestore uses collections and documents to store data. A document is an entry that contains any fields. Documents are combined into collections. The collection is a table and the document should be an entry in that table.

This was the chosen technology because it provides a powerful query functionality for specifying which documents to retrieve from a collection, and due to the existence of collections of documents, it is possible to create an hierarchical structure of data, identical to the one that Bluetooth Devices use.
3.5 Implementation

This section describes how the solution was implemented taking a more detailed perspective on how the prototype was evolving during the development of the system. The first Sprint started in February 1st and the last ended at October 15th.

3.5.1 Sprint 1

The first sprint consisted in developing a first prototype for the sensor node in order to test its functionality. More specifically, create a Bluetooth Server on the sensor side and broadcasting a default characteristic only for testing. This basically means that, with an Android Application for Bluetooth Device scanning it was possible to detect the device and see which characteristics it is advertising. It was also important to understand how much data could the Sensor Node advertise (31 Bytes).

3.5.1.A Outcome

The outcome of the first sprint was a simple implementation of the Sensor Node.

3.5.2 Sprint 2

The second sprint consisted in designing a prototype for the transporter node that could gather data from the sensor node and send it to AWS IoT. This was just a simple implementation just to validate the forwarding of data from the mobile phone to the Cloud Provider. Note that AWS IoT was the first option to handle the publishing of collected data. However this was only used for the first tests.

3.5.2.A Outcome

The outcome of the second sprint was a simple implementation of the Transporter Node that can successfully publish data in the Cloud.

3.5.3 Sprint 3

The fourth sprint was dedicated to use Bluetooth Low Energy API from Android to validate a connection scenario between the Transporter Node and the Sensor node without intervention from the user. Note that, comparing with Sprint 1, the difference is that this wasn’t using any already existing Android Application that needs to be controlled by the app user. This was a sprint with huge importance due to the fact that it was based on evaluating solutions based on user interaction connection with remote bluetooth devices and, comparing with Bluetooth Low Energy API for Android, trying to make a simple connecting scenario without interaction with the user.
3.5.3.A Outcome

The outcome of the forth sprint was a simple communication between the Sensor Node and the Transporter Node without intervention from the user, taking advantage of the Bluetooth Low Energy API.

3.5.4 Sprint 4

The third sprint consisted in encrypting a characteristic, with AES 128, that was being sent from the Sensor Node and decrypt it on the transporter node side. The key was hardcoded in order to facilitate the implementation.

3.5.4.A Outcome

The outcome of the fourth sprint was a simple encrypted communication between the Sensor Node and the Transporter Node.

3.5.5 Sprint 5

The fifth sprint was dedicated to find a way to save information about each sensor in the Cloud. This was an important sprint in order to switch from AWS to Google. Firebase Technology offered a set of tools that were more suited for the job, like Security Rules and a NoSQL Database that provided a way for having a non-relational database with the power to implement a hierarchical data structure.

3.5.5.A Outcome

The outcome of the fifth sprint was the integration with Firebase from Google.

3.5.6 Sprint 6

The seventh sprint was dedicated to implement a algorithm to periodically advertise data. This included finding a way for the sensor to infer the current date based on an initial value and a condition to allow the sensor to only advertise data in a specific time interval. This was challenging because I had to search for solutions for getting current data based on Arduino implementations.

3.5.6.A Outcome

The outcome of the sixth sprint was a new algorithm for periodic advertising of data from the Sensor Node.
3.5.7  Sprint 7

This sprint was based on changing the code from Transporter Node in order to take advantage of the integration with Firebase. At this point I could reduce the options of the scan to only get the services and characteristics that were registered in the Backend.

3.5.7.A  Outcome

Fully integrated with Firebase.

3.5.8  Sprint 8

In this sprint the goal was to investigate how the data should be saved, locally in the Transporter Node, to avoid loosing the collected information. This was important to choose SQLite as the right tool to implement the local database. This was a challenge because I never had implemented a database with SQLite. Defining a Schema and a Contract were things that I was not used to work with.

3.5.8.A  Outcome

Integration with SQLite in the Transporter Node in order to save the collected values.

3.5.9  Sprint 9

I dedicated Sprint 10 to investigate the possibility of making the application send the information to Firebase when Wi-fi connection is available. This was challenging in a way that I didn’t know how to make this possible. Luckily I found that a Broadcast Receiver is able to detect a change in wi-fi connection or 4G connectivity and trigger a Service that is responsible to make a call to Firebase to publish collected data.

3.5.9.A  Outcome

Creation of BroadcastReceiver that triggered a Service to publish the information in Firebase.

3.5.10  Sprint 10

This sprint was based on the creation of the Service mentioned in the previous Sprint. This service would query the Local Database and publish all the values in the Backend (Firebase).
3.5.10.A Outcome

The outcome was a new Service that was triggered by having Wi-fi connection in order to publish data to the Backend.

3.5.11 Sprint 11

Sprint 11 was dedicated to found a way to make the application continuing to run while the screen is turned OFF. This was also a challenge because I had no idea how to make this possible. After some research I found the solution. It was based on using a WakefulBroadcastReceiver. It is a special type of broadcast receiver that takes care of creating and managing a partial wake lock for the app. A WakefulBroadcastReceiver passes off the work to a Service (typically an IntentService, like in this case), while ensuring that the device does not go back to sleep in the transition.

3.5.11.A Outcome

Creation of another BroadcastReceiver but this time in order to trigger the Collect Service to start collecting again when the screen is turned off.

3.5.12 Sprint 12

The twelveth sprint was characterized by reflecting on finding an alternative to locally decrypt the received data. After some research I eventually found that a Cloud Function could handle the job by making a call to Firebase to get the key of the specific device and to handle the computation of decrypting each value received.

3.5.12.A Outcome

Implementation of a Cloud Function that would query the Backend in order to get the key from a specific device and decrypt the data received from the Transporter Node. Implementation of a call in the Transporter Node to this Cloud Function.

3.5.13 Sprint 13

This sprint was dedicated to implement another sensor node to further test the scenario where two sensors are advertising data. The challenge was due to the fact that the two models were not exactly the same, despite the fact that the two were ESP32, I bought each one in different places. That happened to make no big difference because the same code worked fine on both microcontrollers.
3.5.13.A Outcome

Implementation of the second Sensor Node for testing purposes.

3.5.14 Sprint 14

This sprint was dedicated to make the tests with the application.

3.5.14.A Outcome

Realization of Application Tests.

3.6 Used Hardware

This section is responsible for giving a brief description of each of the piece of hardware used during the evaluation and design of this solution.

3.6.1 ESP32

The ESP32 was the microcontroller used to represent the Sensor Node in this study. The reasons for this decision are mostly based on the low price, Bluetooth and Wi-fi technology support, power saving features and the wide developer community associated with this particular device.

In order to save battery the ESP32 is capable of preserving energy by putting to sleep some of its components.

The WiFi modules, the Processing Cores, and the Bluetooth module require a lot of current to operate. So, if we want to conserve power we have to disable them when don’t use them.

ESP32 is capable of light sleep and deep sleep power saving modes.

3.6.1.A Light Sleep

In this kind of sleep, digital peripherals, most of the RAM, and CPUs are clock-gated, and supply voltage is reduced.

Upon exiting from light sleep, peripherals and CPU's resume operation (preserve internal state).

3.6.1.B Deep Sleep

CPUs, most of the RAM, and all the digital peripherals are powered off in the Deep Sleep mode.

The only parts of the chip that are still powered on are the RTC controller, peripherals and memories.
Having ESP32 running on active mode with batteries it's not ideal, since the power from batteries will drain very quickly. In deep sleep mode the batteries will last longer.

In this mode the RTC memory also remains powered on, so we can write a program for the ULP co-processor and store it in the RTC memory to access peripheral devices, internal timers and internal sensors.

This mode of operation is useful if you need to wake up the main CPU by an external event, timer or both, maintaining minimal power consumption.

With these features, the ESP32 needs to have some trigger to wake back up the sleepy components. The RTC controller has a built in timer which can be used to wake up the chip after a predefined amount of time (microsecond precision).

3.6.1.C Firebase Support

Google’s Firebase provides support for IoT development, more specifically a Firebase Realtime Database Arduino Client Library for ESP32 [37] that allows for writing values in a database in order to, for example, in deployment time, taking advantage of the Wi-fi module to populate the Backend instead of being dependent on the manual writing of information regarding each Sensor Node.
After this initial population of the Backend is possible to turn off the Wi-fi in order to avoid draining the microcontroller energy [38].

### 3.6.2 DHT11

The sensor used to provide real data for transmission was the DHT11.

DHT11 Temperature and Humidity Sensor features a temperature and humidity sensor complex with a calibrated digital signal output.

![DHT11 Sensor](image)

**Figure 3.20:** DHT11 Sensor.

By using the exclusive digital-signal-acquisition technique and temperature and humidity sensing technology, it ensures high reliability and excellent long-term stability.

This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high-performance 8-bit controller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness.

The utilization of this sensor requires some attention and careful.

### 3.6.3 Special Attentions

Vapor from chemical materials may interfere with DHT’s sensitive-elements and debase its sensitivity.

Relative humidity largely depends on temperature. DHT11 should be mounted at a place as far as possible from parts that may generate heat. Long time exposure to strong sunlight and ultraviolet may debase DHT’s performance.

### 3.6.4 Huawei P8 Lite

Huawei P8 Lite was the smartphone model used during this research. The main reason that led to this option was the fact that this was my personal phone.

Here you can see the specifications of this smartphone. Important aspects are the version of the Bluetooth, 4.1, and information related with the Battery. The information about the battery is relevant because it will be used as reference point to the power consumption test. The tests were made with
this smartphone but, with a Rule of Three, is possible to estimate a possible decrease in the battery percentage of other type of smartphone.

<table>
<thead>
<tr>
<th>Network</th>
<th>Technology</th>
<th>GSM / HSPA / LTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform</td>
<td>OS</td>
<td>Android 7.0 (Nougat), upgradable to Android 8.0 (Oreo); EMUI 9</td>
</tr>
<tr>
<td></td>
<td>Chipset</td>
<td>HiSilicon Kirin 655 (16 nm)</td>
</tr>
<tr>
<td></td>
<td>CPU</td>
<td>Octa-core (4x2.1 GHz Cortex-A53 &amp; 4x1.7 GHz Cortex-A53)</td>
</tr>
<tr>
<td></td>
<td>GPU</td>
<td>Mali-T830MP2</td>
</tr>
<tr>
<td>Communications</td>
<td>WLAN</td>
<td>Wi-Fi 802.11 b/g/n, Wi-Fi Direct, hotspot</td>
</tr>
<tr>
<td></td>
<td>Bluetooth</td>
<td>4.1, A2DP, EDR, LE</td>
</tr>
<tr>
<td></td>
<td>GPS</td>
<td>Yes, with A-GPS, GLONASS</td>
</tr>
<tr>
<td></td>
<td>NFC</td>
<td>Yes (market dependent)</td>
</tr>
<tr>
<td></td>
<td>Radio</td>
<td>FM radio</td>
</tr>
<tr>
<td></td>
<td>USB</td>
<td>microUSB 2.0, USB On-The-Go</td>
</tr>
<tr>
<td>Battery</td>
<td></td>
<td>Non-removable Li-ion 3000 mAh battery</td>
</tr>
</tbody>
</table>

Figure 3.22: Transporter Node Specs.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 First Prototype Test</td>
<td>47</td>
</tr>
<tr>
<td>4.2 Multiple Sensor Test</td>
<td>48</td>
</tr>
<tr>
<td>4.3 Data Collection Time Test</td>
<td>48</td>
</tr>
<tr>
<td>4.4 Range Test</td>
<td>50</td>
</tr>
<tr>
<td>4.5 User Tests</td>
<td>53</td>
</tr>
<tr>
<td>4.6 Power Consumption Test</td>
<td>54</td>
</tr>
<tr>
<td>4.7 Technological Comparision Evaluation</td>
<td>56</td>
</tr>
</tbody>
</table>
This chapter is focused on the evaluation of the proposed solution.

4.1 First Prototype Test

![Sensor readings Test](image)

This was the first test in order to validate the fact that the sensor node could successfully get data from its sensors and also that it has the capability of calculating the current data in order to manage the scheduling of data collection.
In the transporter node side, it was possible to validate the collection of the data and also the fact that it can infer the strength of the signal transmitted from each sensor node.

The transporter node authentication was also validated in order to allow only devices that are registered on the Backend.

4.2 Multiple Sensor Test

![Multiple Sensor readings Test](image)

Figure 4.2: Multiple Sensor readings Test.

A test with two sensors was executed in order to validate the scenario where there are more than 1 sensor transmitting information. The results of the test were the proper collection of data from each of the transmitting sensors. The mobile phone connects to the first sensor it discovers.

![Test with one person and 2 sensors](image)

Figure 4.3: Test with one person and 2 sensors.

4.3 Data Collection Time Test

With the help of Android Studio Logcat [5] it is possible to see the time each value takes to be collected from the time the scan starts.

To understand the time that takes for the smartphone, when inside the range of the sensor, to successfully collect data, values gathered from logs were analyzed in order to understand the tendency.

Figure 4.5 shows a graphic that has 20 consecutive readings. The actual values are in the Appendix A. For the further analysis I will use approximate values to better explain the tendency.
By analyzing the graphic is possible to see the tendency is to take approximately 2000ms to get the data about the first value collected. The second value discovered takes another 2000ms to be collected, thus the 4000ms that appear at each two consecutive readings.

The values that are here shown represent collections of data about two values, humidity and temperature.

If we add three values to read, for example, humidity, temperature and CO2 levels, it would be 2000ms for each collected value and thus 6000ms to collect all the 3 values, assuming that the pattern maintains.

The eleventh measure dictates that the sensor stopped advertising.

In the fifteenth measure I turned the sensor off and turned it on a little bit later.

From 16 to 20 it's possible to see that the pattern it's still the same.
4.4 Range Test

The range test is important to consolidate the conclusions from the previous test section. It was validated a range of approximately 50m. This measurement includes walls so the scenario of indoor monitoring can also be validated.

4.4.1 Rope

![Diagram of a circle with labeled parts: Tangent, Arc, Rope, Radius, Diameter, Secant.]

Figure 4.6: Circumference Characteristics

Each circumference is characterized by the existence of several ropes. Those ropes can be important to measure each possible distances that a person with a smartphone can travel. For each rope length there is a maximum velocity that is supported by the design solution in order to successfully gather sensor data.

\[ c^2 = 2r^2 - 2r^2 \cos \theta \]
\[ c = r\sqrt{2 - 2 \cos \theta} \]
\[ c = 2r \sin \frac{\theta}{2} \]

Figure 4.7: Rope Equations.
The velocity calculated for each rope can determine if it is possible to gather sensor data while walking. It is also possible to infer if it can be feasible to collect data while riding a bike.

![Figure 4.8: Real Scenario](image)

Based on the maximum range of the Sensor Node and based on Theta, that is the angle formed by the ends of the rope from the center of the circumference, it is possible to calculate each length of the rope.

### 4.4.2 Rectilinear Motion

The rectilinear motion is the movement described by objects with constant velocity in a straight trajectory (straight line), for this, it is necessary that the resultant of the forces acting on the body is null. Given an offset Delta s over a time interval Delta t the scalar speed is given by the equation showed in figure 5.6.

If we map the average collecting time from the results to the time interval of the equation, and if we map the displacement with the length of each rope we can have, for each rope, the maximum velocity that is possible to support data collection. From then it is possible to map the results with the cases that I mentioned before like walking, running, cycling or even driving a car.

When you know the distance from the center of the circumference to the center of the rope (which we call d), the length of the rope is calculated using Pythagoras' theorem where a cathetus is half the rope (which we will designate x), the distance from another cathetus and the radius of the circumference is the hypotenuse. Therefore it is possible to use the following equation.

The length of the rope will then be 2 times the cathetus.
So, wrapping up, in a real scenario, knowing how far we are from the sensor, and knowing the average time a smartphone needs to collect data while in the range of the sensor, it is possible to analyse the supported values for the velocity.

### 4.4.3 Results

The equation that allows the calculation of the maximum supported velocity based on the length of each rope is represented below.

\[
v = 2\sqrt{r^2 - d^2}/t\]

The values used are the average scan time for the first and the second reading respectively.

For \(r=50\text{m}\) and \(t=2.2\text{s}\), the relation between \(v\) and \(d\) can be seen by analyzing Figure 4.13.

For \(r=50\text{m}\) and \(t=4.6\text{s}\), the relation between \(v\) and \(d\) can be seen by analyzing Figure 4.14.

The two graphics shows that it is possible to still collect data if our velocity is between 25 m/s and 10 m/s.

According to HealthLine the average walking speed is between 0.94 and 1.34 m/s [31].

According to Lime Wikipedia all of the scooter models that Lime uses can reach speeds of up to 25 km/h or 6.94 m/s.
According to [30] the maximum speed of a Gira electric bike is also up to 25 km/h or 6.94 m/s. All of these values are lower than the maximum supported for the average scan time assuming a range of 50m and a maximum distance from the sensor of 40m. It shows that in these particular cases is possible to collect data.

4.5 User Tests

With two Huawei P8 Lite near the sensor node, each one of them was able to collect data with a difference of approximately 4 seconds between each device. The results show that the smartphone that is closer to the sensor node is able to receive the results beforehand.
4.6 Power Consumption Test

Energy Profiler helps to find where the developed app uses more energy than necessary. It can monitor the use of the CPU, network radio and GPS Sensor displaying a visualization of how much energy each of these components uses [28]. However, Energy Profiler does not directly measure energy consumption. It uses a model that estimates the energy consumption for each resource on the device.

By visualizing the two images of Energy Profiler print screens it is possible to see that the estimated impact that the app has on the performance of the Smartphone is considered to be Light in the beginning and even less during the execution of the data collecting service.

So there is a reason to be optimistic to the next tests that showed the actual impact on the device battery.

The following images show the difference in consumption with 4G turned on and turned off in three difference cases, during the period of 1 hour.

The first graphic can be used as a reference point in order to understand the normal decrease in
percentage of the battery of my smartphone. The other two can then be compared with the first one in order to be possible to make conclusions.

The first one, Figure 4.18 is with the screen locked and without running the application. It shows that there was a decrease in the battery levels in 3 percent with 4G and 2 percent without 4G.

The second graphic, Figure 4.19 shows the decrease in battery life when the app is running but without the sensor running. So the app is not collecting any values. With 4G turned on there is a
decrease in 4 percent and with 4G turned off there is a decrease of 2 percent.

This is already making sense when comparing with the Energy Profiler pictures. It shows that when there is no interaction with the sensor, there is no energy being spent, despite the fact that the device is scanning anyways.

The third graphic, Figure 4.20 shows the decrease in battery life when the app is running and actively collecting data from the sensor. It shows a 6 percent decrease in battery levels with the 4G on and a 5 percent decrease with the 4G turned off.

So it is possible to conclude, comparing the three graphics, that there is indeed a bigger decrease in battery levels while collecting values from the sensor but there isn’t a big difference when comparing with the case when the smartphone isn’t running the application.

4.7 Technological Comparision Evaluation

4.7.1 GSM

GSM needs to know the number of the receiving phone in order so send an SMS with the data [32] [34]. This is something that we should avoid because this would require to have all the registered smartphones number in each sensor which is not a scalable solution. Another disadvantage is the fact that the there is a need for having a SIM900 GSM GPRS Shield, for example in the case of Arduino in order to be able to communicate with smartphones and that comes with a price [35], Adding the price of the Arduino [36].

4.7.2 LoRaWAN

LoRaWAN architecture, as seen in the following figure, requires the existence of LoRa gateways, that are not cheap [33].
4.7.3 BLE

Each ESP32 comes equipped with BLE and Wifi, and all I had to spend were 11€. This is clearly an advantage in terms of cost comparing with the other two technologies. And it is also something practical, due to the fact that there isn't a need for extra hardware, the ESP32 takes care of it.
5

Conclusion

Contents

5.1 Overview ................................................................. 61
5.2 Study Conclusions ...................................................... 61
5.3 Use cases ................................................................. 61
5.4 Future Work ............................................................. 62
5.5 Final Considerations .................................................... 62
5.1 Overview

This chapter consolidates the final thoughts on the proposed solution.

To tackle the problem of sensor data collection in big cities, and also in indoor locations, it was proposed an idea based on the use of Bluetooth Low Energy as the wireless technology that bridged the sensor data with the cloud.

The designed prototype protocol is based on the occurrence of short connections between a smartphone and a microcontroller.

The data transmitted from each Sensor Node is not sent in plaintext, it is encrypted with a key that should be unique for each device.

This specific characteristic allows the collection of data in movement.

5.2 Study Conclusions

According to the evaluation it is reasonable to conclude that the energy consumption of the developed service is not too impacting on the battery life of the smartphone. This answers to the research question related with power consumption.

The results showed that we can collect data while walking, driving an electric scooter or driving an electric bike. If a car drives around the 40km/h mark with a sensor at a distance of approximately 40 meters it is possible to collect data. This answers to the research question related to the situations where we can collect data.

The information that sensor transmits to the smartphone is encrypted so we can also validate the question about securing data, even though it lacks the key management feature that will be left to future work.

So answering to the first research question, yes, it looks like it is possible to use a smartphone as a way of transportation between sensors and the cloud.

There are some questions that will need a more detailed perspective like evaluating the possibility of adding more services to each sensor to see how much time is needed in order to collect all services data.

5.3 Use cases

Associated with an application like Gira, the usage scenario would be triggering the background service in order to start scanning when we unlocked the bike. The app, in the future, could tell us a route that includes sensors in order to make an incentive to the client to follow a specific path in order to collect
sensor data and receive bonus time to use bikes.

5.4 Future Work

The distribution and management of the keys is a theme that should be explored in the future, in order to increase the level of security of the advertised data.

Fault Tolerance should be another theme to be addressed in a future study.

5.5 Final Considerations

To finalize, I hope the Smart City concept doesn’t fade leaving so much potential behind and that the design of future solutions have in consideration the privacy of the people that may want to be part of information relay.
Bibliography

Research Centre for Automatic Control of Nancy, Campus Sciences, BP 70239, Vandoeuvre, 54506, FrancebOKKO SAS, 34 Rue Nationale, Puttelange-aux-Lacs, 57510, FranceReceived 21 September 2017; accepted 20 December 2017Available online 4 January 2018


[10] Internet of Things Exploring and Securing a Future Concept CRISTIAN BUDE and ANDREAS KERVEFORS BERGSTRAND
Last Accessed: August 2019


[13] Everything You Wanted to Know About Smart Cities, Saraju P. Mohanty, University of North Texas

[14] Cloud Edge Computing, Beyond The Data Center, Brian E. Whitaker, Founder, Zettabyte Content

[15] Smart Environmental Monitoring Beacon Pop Alexandru1, Manea Andrei1, Sabau Cristina-Madatina1, Ovidiu Stan1 Department of Automation, Technical University of Cluj Napoca, Cluj, Romania alex.pop1819@gmail.com

[16] Design of an IoT based Real Time Environment Monitoring System using Legacy Sensors, Aparajita Das1*, Manash Pratim Sarma1, Kandarp Kumar Sarma1 and Nikos Mastorakis2 1Department of Electronics and Communication Engineering, GIUST, Guwahati University, Guwahati-781014, Assam, India 2 Technical University of Sofia, Sofia, Kliment Ohridski 8, Bulgaria.

[17] Desenvolvimento de uma APP Android e Plataforma Web para comunicação com Beacons, Nuno Alexandre Lemos de Paiva


[19] A Smart City Environmental Monitoring Network and Analysis Relying on Big Data Techniques Ashraf Tahat, Ruba Aburub, Aseel Al-Zyoude and Chamseddine Tah

[20] Indoor Positioning: A Comparison of WiFi and Bluetooth Low Energy for Region Monitoring Alexander Lindemann, Bettina Schnor, Jan Sohre and Petra Vogel


[22] The Use of Bluetooth Low Energy Smart Sensor for Mobile Devices Yields an Efficient Level of Power Consumption, Khurshid Aliev, Eros Pasero

[23] Environmental Monitoring For IoT Applications Based On Wireless Sensor Network Mrs. Reshma S. Mundhe, Prof. Ravindra G. Dabhade


[27] The evolution of M2M into IoT, Mahbubul Alam; Rasmus H. Nielsen; Neeli R. Prasad

   Last Accessed: October 2019

[29] https://www.healthline.com/health/exercise-fitness/average-walking-speed
   Last Accessed: October 2019

   Last Accessed: October 2019

[31] https://www.healthline.com/health/exercise-fitness/average-walking-speed
   Last Accessed: October 2019

   Last Accessed: October 2019

[33] https://uk.pi-supply.com/products/iot-lora-gateway-hat-for-raspberry-pi
   Last Accessed: October 2019

[34] https://randomnerdtutorials.com/sim900-gsm-gprs-shield-arduino/
   Last Accessed: October 2019

   Last Accessed: October 2019

[36] https://www.fnac.pt
   Last Accessed: October 2019

[37] https://github.com/mobizt/Firebase-ESP32

[38] https://circuits4you.com/2019/01/08/esp8266-turn-off-wifi-save-power/


[40] Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications, Mihai T. Lazarescu
**Figure A.1:** Data Collection Time Values.
Appendix B