Data acquisition system using IoT to monitor sheep and goat mobility

Mateus Costa de Araujo - 57010

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Electrical and Computer Engineering, in the scope of the double diploma programme with the Federal University of Technology - Paraná.

Work oriented by:

Prof. Paulo Leitão

Prof. Miguel Angel Chincaro Bernuy

Prof. José Manuel Castro

Bragança

2024

Data acquisition system using IoT to monitor sheep and goat mobility

Mateus Costa de Araujo - 57010

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Electrical and Computer Engineering, in the scope of the double diploma programme with the Federal University of Technology - Paraná.

Work oriented by:

Prof. Paulo Leitão

Prof. Miguel Angel Chincaro Bernuy

Prof. José Manuel Castro

Bragança

2024

Dedication

I dedicate this work to God and all those who supported me throughout its development, especially my mother and Prof. José Castro who believed in me from the beginning.

Acknowledgement

I want to express my thanks to everyone who participated directly or indirectly in my training journey.

Firstly, to God who gave me strength to face all the challenges. I would also like to thank my mother who supported me throughout this period, and also my friends who accompanied me throughout the journey. I would also like to thank my supervisors for all the support provided during the preparation and development of this work. I would also like to thank the PASTOpraxis project, which supported me in carrying out this work. This work brought me closer to the field of technological development and made me enchanted by the world of IoT.

Finally, I am grateful for having studied at UTFPR and IPB, two great institutions that provided me with training and diverse experiences, both academic and life.

Thank you all.

Abstract

This study investigates the development of an autonomous data acquisition, transport, storage and visualization system based on Internet of Things (IoT) technologies to monitor sheep and goat grazing in the north-eastern region of Portugal. The system was designed with an emphasis on the autonomy, reliability and ergonomics of the device, making use of technologies such as Long Range (LoRa) and Global Navigation Satellite System (GNSS) to enable accurate monitoring of the animals during grazing. By adopting an IoT architecture with distinct functional layers, the system allows data to flow from collection to presentation, thus contributing to efficient land use management and optimizing agricultural productivity in the region. The tests carried out demonstrated the system's effectiveness in various scenarios, highlighting its ability to maintain data integrity even in remote areas with an unstable connection. With an autonomy of up to 37 days, the system proved to be robust and reliable in its operation. The analysis of the collected data revealed patterns in the animals' behavior during grazing, providing a deeper understanding of their habits and allowing for more informed decision-making. In addition, the system was carefully designed with a focus on the animals' well-being, ensuring that they were not hindered in their daily activities. This concern for animal comfort reinforces the system's viability and sustainability in agricultural environments.

Keywords: Internet of Things, GNSS, LoRa, silvopasture, animal monitoring, IoT architecture.

Resumo

Este estudo investiga o desenvolvimento de um sistema autônomo de aquisição, transporte, armazenamento e vizualização de dados baseado em tecnologias de IoT para monitorar o pastoreio de ovinos e caprinos na região do Nordeste de Portugal. O sistema foi projetado com ênfase na autonomia, confiabilidade e ergonomia do dispositivo, fazendo uso de tecnologias como LoRa e GNSS para possibilitar a monitorização precisa dos animais durante o pastoreio. Ao adotar uma arquitetura IoT com camadas funcionais distintas, o sistema permite o fluxo de dados desde a coleta até a apresentação, contribuindo assim para a gestão eficiente do uso do solo e otimização da produtividade agrícola na região. Os testes realizados demonstraram a eficácia do sistema em diversos cenários, destacando sua capacidade de manter a integridade dos dados mesmo em áreas remotas com conexão instável. Com uma autonomia de até 37 dias de funcionamento, o sistema se mostrou robusto e confiável em sua operação. A análise dos dados coletados revelou padrões de comportamento dos animais durante o pastoreio, proporcionando uma compreensão mais profunda de seus hábitos e permitindo tomadas de decisão mais informadas. Além disso, o sistema foi cuidadosamente projetado com foco no bem-estar dos animais, garantindo que não fossem prejudicados em suas atividades diárias. Essa preocupação com o conforto dos animais reforça a viabilidade e sustentabilidade do sistema em ambientes agrícolas.

Palavras-chave: Internet das coisas, GNSS, LoRa, silvipastoril, monitoramento animal, arquitetura IoT.

Contents

List of Tables

List of Figures

Acronyms

ADC Analog-to-Digital Converter.

ANACOM Autoridade Nacional de Comunicações.

API Application Programming Interface.

BJT Bipolar Junction Transistor.

BLE Bluetooth Low Energy.

CPU Central Processing Unit.

CSS Chirp Spread Spectrum.

DAC Digital-to-Analog Converter.

ESA Escola Superior Agrária.

GIS Geographic Information System.

GND Ground.

GNSS Global Navigation Satellite System.

GPIO General Purpose Input/Output.

GPRS General Packet Radio Service.

GPS Global Positioning System.

I2C Inter-Integrated Circuit.

IDE Integrated Development Environment.

IoT Internet of Things.

IPB Instituto Politécnico de Bragança.

IT Information Technologies.

JSON JavaScript Object Notation.

LoRa Long Range.

LPWAN Low Power Wide Area Network.

MQTT Message Queuing Telemetry Transport.

NB-IoT Narrowband Internet of Things.

NFC Near-field Communication.

NMEA National Marine Electronics Association.

PCB Printed Circuit Board.

PLA Polylactic Acid.

PLF Precision Livestock Farming.

RFID Radio Frequency Identification.

RSSI Received Signal Strength Indicator.

SCL Serial Clock Line.

SDA Serial Data Line.

SPI Serial Peripheral Interface.

UART Universal Asynchronous Receiver-Transmitter.

UAV Unmanned Aerial Vehicle.

UTC Coordinated Universal Time.

Wi-Fi Wireless Fidelity.

Chapter 1

Introduction

According to Castro et al. [1], small ruminant production systems in Northeast Portugal are characterized by extensive activities that rely on daily grazing routes through the landscape. The herds explore various areas, utilizing different types of ground cover in varied ways throughout the seasons. The interactive dynamics between shepherds and herds play a crucial role in defining resting and feeding sites for sheep and goats, adjusting according to the biotic and abiotic conditions present.

The authors further assert that in-depth knowledge of the extent of grazing areas is of fundamental importance for the careful management of land use, preservation of vegetation cover, and optimization of productivity in traditional sheep and goat production systems. Thus, these animals perform essential productive, economic, cultural, social, and environmental functions in Northeast Portugal [2].

To acquire this knowledge, commercial or non-commercial devices that utilize Global Positioning System (GPS) or GNSS technology for tracking when attached to the animal have become increasingly common. In addition to these, there is a range of Precision Livestock Farming (PLF) technologies also attached to the device, such as sensors, cameras, among others, collecting further data from the animals [3].

The devices that perform this acquisition may or may not provide real-time data transmission, depending on the application's objectives and the established acquisition structure. When real-time data transmission is required, technologies such as satellite

communication, LoRa, Sigfox, ZigBee, Wireless Fidelity (Wi-Fi), Bluetooth Low Energy (BLE), among others, are utilized [4].

Technologies and data transfer networks are increasingly applied to data acquisition due to the rise of IoT. This technology encompasses a range of innovations in the field of Information Technologies (IT), applicable across various domains by ensuring connectivity and standardization of protocols and technologies [5].

1.1 Justification

The practice of grazing provides various ecosystem benefits, such as fire control [6], benefits to landscape heritage, and nutrient cycling.

Therefore, there are several advantages to recording the trajectory undertaken in this activity. This information enables strategic decision-making, considering not only land use efficiency but also environmental factors and the specific needs of the herds.

In addition to these aspects, recording grazing routes is a challenge that can be solved with data acquisition principles and IoT technologies, since principles of connectivity, sensing, information security and other points must be taken into account when building a IoT structure to acquire data and monitor any grazing route.

By building a data acquisition system that guarantees the recording of grazing routes and other relevant information such as temperature, humidity, relative positioning of the neck of small ruminants, there is a series of improvements in the analyses already carried out with current equipment, since more data can be correlated, providing even more technology and information for the management of this practice.

1.2 Problem Statement

To provide this tracking, some companies have already developed commercial solutions that ensure the acquisition of the geolocation of small ruminants.

The company Domodis [7] presents a solution for types of livestock, horses, sheep, cows

and bulls, goats, wild animals, etc. On its website, the company presents some of the solution's features, such as its communication, which is done via satellites. In addition, the company's devices have an autonomy of approximately two months, depending on the data acquisition and transfer interval.

Although the devices offer interesting autonomy, another point to consider with this solution provided by Domodis is their weight and shape. The models presented to Escola Superior Agrária (ESA) in 2020-2023 do not offer adequate ergonomics for goats, i.e. as they are generic devices, the aspects of this animal's back are not taken into account, and it can even rotate in its neck, which can then generate discomfort for the animal and loss of connection with satellites.

The company Pastoral [8] presents a solution that also aims to track various animals. The company's website does not provide information on the technologies used in its devices, but it does provide information on autonomy and ergonomics. In the tests carried out by ESA, this device, despite containing an integrated solar panel, had a lower autonomy than the device supplied by Domodis.

Unlike the solution presented by Domodis, which presents a solution to accompany the animal's neck, the device developed by Pastoral is positioned on the animal's back, which has some advantages and disadvantages. On the other hand, in the case of goats, the fixing rods, also known as belly pads, can be a risk for the grazing of the tracked animal, allowing the animal to get stuck in some vegetation and end up injuring itself.

In addition to the technological and ergonomic aspects discussed, there is also the cost of implementing a tracking system. Pastoral's solutions start at $95, while Domodis offers devices starting at €397.00 per unit. In addition to the cost of purchasing the devices, Domodis and Pastoral also charge a monthly subscription to use them.

By analyzing these commercial proposals and the experiences of ESA professors and technicians who study and/or herd small ruminants, three main points can be listed that ensure and are capable of measuring the efficiency of a tracking device for these animals:

- **Reliability**: data should be acquired within a pre-defined timeframe and there

should be as few failures in data recording as possible.

- **Autonomy**: the system should work for as long as possible so that it doesn't become inconvenient for administrators.

- **Ergonomics**: the system must not cause inconvenience to the animal and must be easy for administrators to use.

Both the devices supplied by Domodis and Pastoral have been and continue to be used by ESA, and the device presented in this work will be based on the three points mentioned above. More information on the description of the problem will be presented in the course of this work.

1.3 Objectives

Given the context of the problem and its importance, the main objective of this work is to develop a data acquisition system using IoT technologies to monitor the mobility of goats and sheep during grazing, based on **Autonomy**, **Reliability** and **Ergonomy**.

Bearing in mind the main objective of this work, it is possible to define some specific details of where we intend to go with the construction of a system with the aforementioned fundamentals, so the geral requirements of this work are:

- In terms of autonomy, we expect to get at least 20 days of operation with a data transfer rate of every 5 minutes.

- The communication between the device that will be attached to the animal's collar and the gateway that will send the data to the server will be via LoRa and the aim is to achieve a radius of at least 2km from its location.

- The data to be acquired will be: latitude and longitude, additionally ambient temperature and humidity, the relative position of the animal's neck and the system's battery percentage.

- In terms of ergonomics, the system must be suitable for the animal and must not rotate around the animal's neck or interfere with its well-being.
- The system must be low-cost, i.e. all two parts must be built with low-cost components.

1.4 Thesis Structure

This document is structured in six chapters that present how the work was carried out.

Chapter 1 presents an introduction to the work and objectives, contextualizing the project's field of application and the reasons why it should be developed.

Chapter 2 presents the state of the art in the field of application to which the system belongs, presenting similar work being carried out in the area and how it impacts on small ruminant grazing. Technologies such as LoRa, GPS, GNSS, Livestock and others are presented and discussed in this chapter.

Chapter 3 presents the system development methodology, providing all the functional and non-functional requirements and how these should be coupled with an IoT architecture. It also discusses where the system should be installed and how its parts should be distributed.

Chapter 4 presents how the entire system was developed, from the device, which is attached to the animal, to the user/administrator interface technologies. Hardware and software for the device and gateway are presented, as well as the construction and development of Node-RED application and how Grafana application and Telegram bot make it possible to interface with the data.

Chapter 5 presents the tests and results obtained with the system, analyzing the connectivity of the device with the gateway and gateway with MQTT Broker. Data visualization and analysis, device animal behavior and device battery performance are also presented.

Chapter 6 presents the conclusions drawn from the tests and the limitations of the system, as well as guidelines for future work.

Finally, the appendices contain links and files from various other chapters of the work, providing codes, figures and tables that enrich and broaden the perception of the work.

Chapter 2

State of Art

This chapter presents an overview of the technologies and work being developed with the same or similar objectives to this work. Applications made, results obtained and discussions on how acquisition devices are being developed and used in small ruminants are covered.

2.1 Data Acquisition Systems for Livestock

Silvopasture is an integral part of society and a determinant of culture in many parts of the world. It plays a vital role not only for food production, but also contributes to social cohesion, identity, and the functioning of numerous ecological and landscape processes that translate into so-called ecosystem services [9]. With the growing importance of ecosystem services and the need to evaluate and remunerate them, the integration of various technologies and data acquisition systems has become increasingly important in recent decades in this sector, and PLF methodologies can help in this regard [10].

In order to assess the ecosystem services provided by traditional extensive grazing in natural and forest areas, PLF ensures accuracy in recording the extent, intensity and frequency of this process. The location records of grazing herds provide this information, enabling the development of systems for remunerating these services that are adapted to specific local conditions, and which, among other things, help to reduce the risk of

fire, the recycling of nutrients, the use of cultivated fodder, while protecting the valuable ecosystems of mountain regions.

Global Positioning Systems (GPS), as a means of tracking the movement of animals, are among the first and most applied technologies used to study the foraging strategies and decision-making processes of animals. Tracking animals has proven to be an effective way of understanding the interactions between herders and their resources [11], which can be beneficial for animal welfare, the protection of ecosystems and the workload of herders [10][12].

PLF encompasses combined applications of technologies with one or more tools in integrated systems for individual livestock monitoring. In grazing systems, which is the target of this work, some applications of PLF could improve the control of livestock by shepherds, improving the use and management of pastures, as well as the monitoring and control of animals, as already mentioned [10]. Among the stages in the development of a PLF, data acquisition is crucial, as all the decisions made by an intelligent system are based on data collected from the animal.

Data collection is influenced by the parameters selected, which can cover a wide range, including temperature, weight, position, health, grazing, among others. In addition, there are several particularities that each system needs to consider, as the types of sensors, the animals and the farming systems represent determining factors for the implementation of a data acquisition system [10].

The use of GPS or GNSS and Geographic Information System (GIS) to acquire grazing tracking and pasture use monitoring is a highly efficient practice. It can also be combined with accelerometers to monitor grazing behavior, small ruminant behavior, rest, feed monitoring, theft prevention and more [3]. In addition to these, there are several other technologies that can be combined, such as cameras, gyroscopes, solar panels, motion sensors, etc.

2.1.1 Small Ruminant Data Acquisition

Various initiatives aimed at small ruminants are being employed by researchers and livestock farmers around the world. The use of technologies such as GPS, GNSS, Unmanned Aerial Vehicle (UAV)s, Virtual fences, Radio Frequency Identification (RFID), among others, for various purposes. The Figure 2.1 gives an overview of the possibilities that the use of these technologies can provide.

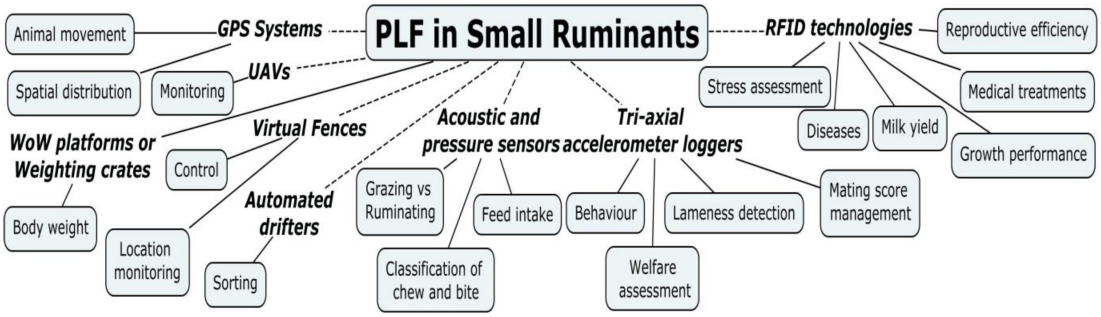

Figure 2.1: PLF applications in grazing sheep and goats [3].

The movement of these animals acquired by means of GPS or GNSS can be linked to various purposes. Borthwick et al. (2024) [13] used GPS to analyze the behavior of 48 sheep in an extensive system, comparing the distribution network of these animals in comparison with your heat stress. Lopez et al. (2024) [14] used the same technology to identify the correlation between the Dry-Salted Violino and Bresaola meats of Bergamasca sheep and the pasture they consumed in the Italian mountains.

Paz-Kagan, T., Alexandroff, V & Ungar, E. D. (2023) [15] analyzed whether the effect of the presence of goat herds in Mediterranean forests in Israel can be detected through the integration of remote sensing and herd tracking at the landscape scale, using GPS and other technologies to analyze the influence of these animals in the region. Plaza et al. (2023) [16] used GPS and GIS on sheep and goats to track all the transhumance routes taken by these animals, including altitudes, longitudes, maximum and minimum daily distances, and grazing time. This study showed the remarkable physiological ability of the Merino de los Montes Universales sheep breed to endure long journeys, even when pregnant.

Castillo-Garcia et al. (2022) [17] used GPS to understand the effect of different grazing intensities of Rasa Aragonesanos sheep on herbivore-plant-soil feedbacks in Mediterranean mountain pastures in the Sierra y Cañones de Guara Natural Park - Spain. This work concluded that grazing management is fundamental to balancing feedbacks in this type of ecosystem. Wild et al. (2023) [18] used the same technology to discuss the possible interrelationships between the movement and selective grazing behavior of free-ranging and ungrazed sheep and the botanical composition of high-altitude mountain pastures in northern Italy. This study analyzed the animals' preference patterns, correlating altitude, preferred species and other data.

Souza et al. (2023) [19] used GPS monitoring to establish preferred routes and grazing locations to find the best feeding strategy (supplementation versus no supplementation) for sheep finished on a Caatinga pasture (seasonally dry tropical forest) in Pernambuco, Brazil, in different seasons (rainy period, transition period).

2.1.2 Integration of PLF Technologies

As seen in Figure 2.1, sensors can return different data about small ruminants and these can be used for different purposes. Combining sensors is also common for investigative and non-investigative purposes. Combining sensors increases the amount of data acquired from the grazing and/or animal and this can provide even more information about it.

Trieua et all. (2022) [20] used GPS and accelerometer to identify ryegrass staggers in sheep in Lincoln, New Zealand. This condition is caused by a toxin present in pastures of perennial ryegrass (Lolium perenne L). Using machine learning, it was possible to identify variations in the behavior of sheep infected with this toxin. Zhao et al. (2022) [21] used the same technology on sheep to identify the circulation of crops provided by flocks. This study concluded that sheep grazing can bring seeds from agricultural fields to grow in natural pastures, causing changes in plant species diversity and community structure.

Chebli et al. (2022) [22] used the same technologies to characterize the activity and protein-energy needs of dairy goats in a Mediterranean forest in northern Morocco. This

study provided data for the development of feeding strategies in semi-extensive systems. Gao et al. (2023) [23] also combined GPS and accelerometer to understand spatial and temporal changes in the grazing trajectory of Sunit sheep for the rational use of pasture resources in the Inner Mongolian desert and for optimal cattle breeding.

Sales-Baptista et al. (2022) [24] used GNSS and a camera on Black Merino sheep in Mitra-PT, to understand time and scale-dependent grazing behavior variables related to explanatory pasture conditions. This study concluded that the use of cameras is suitable for assessing variations in the behavior patterns of sheep grazing complex pastures. Keshavarzi at al. (2023) [25] used GNSS on Merino sheep to identify the animals' preference for social proximity and concluded that there is a preference for proximity to individuals from the same family.

All of these studies combined GPS or GNSS with some other sensor or technique to collect information from their case studies. The geographic points acquired with these technologies are collected in two ways, either in real time or in local storage. Real-time collection presupposes that the GPS or GNSS sends the collected points to a gateway or database with or without a defined time interval. On the other hand, there are devices that store the collected points internally and these are acquired at the end of the experiment or defined period, although the interval defined by the administrator between these points is still respected.

The commercial devices that guarantee the points acquired by the GPS or GNSS are used for both investigative and non-investigative purposes, among the works cited were devices developed by Domodis [7] [16], LikeM13 [26] [18] and Pastoral [8]. Unfortunately, not all the articles indicate which device was used or the technologies involved.

Devices with GPS or GNSS vary the collection of points between 1s and 60m, depending on the data acquisition system, battery, sending architecture (if any) and other details. However, most systems are configured to acquire data between 2m and 5m. When these are sent in real time, technologies such as LoRa, satellite communication, General Packet Radio Service (GPRS), among others, are used [4]. This work uses LoRa technology within an IoT architecture to guarantee the connection between a device and the rest

of the architecture.

Zorawski et al. (2021) [27] developed a prototype using LoRa with LoRaWAN protocol to collect points addressing a system to perform animal tracking, and the development of a test platform, through long-range technologies. This work identified that LoRa is a suitable technology for working with small ruminants in the context of Northern Portugal. The prototype developed in this work takes advantage of the fundamentals explored by Zorawski to develop a data acquisition device with IoT architecture.

The use of LoRa to send data includes the need for an IoT architecture to transport and analyze the data collected. This infrastructure can be built in different ways with various technologies and layers involved.

2.2 Internet of Things

The IoT has rapidly transformed the way we interact with the world around us. The ability to connect devices, collect data and make informed decisions in real time has opened the door to innovations in many areas, from smart homes to industry and environmental monitoring. In this context, the development of efficient IoT nodes, capable of collecting a variety of metrics and communicating them reliably, has become essential. This work focuses on the design and testing of an autonomous IoT node that uses LoRa technology for data transmission. The broader context of IoT, possible applications, specific livestock applications and IoT architecture will be presented below.

2.2.1 IoT Context

The Internet of Things is a new approach to IT that explores a deep integration between the “internet” and the “thing”. The Internet is a global system of computer networks that are connected to each other and use a standard set of communication protocols to ensure this communication. It is a network that connects millions of private, public, corporate, governmental and academic networks [28].

On the other hand, there are Things, which can be any person or everyday object and this includes non-electronic devices [28]. Some examples could be cars, smartphones, cities, traffic lights, industries, food, clothes, furniture, parts and equipment, monuments, works of art, even living things such as calves, dogs, cats and even sheep and goats, which will be the "thing" explored in this work.

Since its emergence, the IoT has grown exponentially, especially since the 2000s, and this is due to a number of factors that have driven its growth, such as the smartphone boom [29], the emergence of Industry 4.0, communication technologies such as Bluetooth, Z-Wave, Near-field Communication (NFC), Wi-Fi, LoRa, Narrowband Internet of Things (NB-IoT), Zegbee, Sigfox and the emergence of various actuators and sensors to digitize various everyday things.

Bhagat [30] states that there are several benefits to using IoT, such as ease of access, i.e. with the advancement and reduction in prices of devices/sensors it becomes easy and affordable to enjoy the benefits of IoT. Furthermore, it makes devices and services smarter, with the implementation of IoT in cities, cars, traffic, coffee machines, among others, a sequence of decision-making takes place that transforms these devices into "smart things", i.e. autonomous and automated.

Saving costs is also one of the benefits cited by the same author. By monitoring and automating various everyday objects, it is possible to set consumption limits, stipulate operating hours, create alerts regarding anomalous situations, etc. Another benefit is the increase in productivity that IoT provides. With its implementation, it is possible to collect various data involving a service/production and identify and predict possible failures or bottlenecks in processes/services, optimizing them intelligently and continuously.

One of the great challenges of the IoT is Information Security (IS), which is related to the protection of information shared or stored by electronic devices. The triad Confidentiality, Integrity and Availability make up the principles of IS [31], where:

- **Reliability** where only authorized persons can access the information.
- **Integrity** is responsible for ensuring that the information is correct and complete

without modification by third parties.

- **Availability** ensures that the information is accessible whenever necessary.

With this wave of advances, technological developments, market growth and concern about information security, two pillars of the IoT will be widely discussed in this paper, namely sensors and connectivity between IoT devices and gateways, other devices responsible for connecting communication technologies, such as LoRa and Wi-Fi, for example.

2.2.2 IoT Applications

IoT can be applied to several domains, e.g., **Smart Home**, **healthcare industry**, **Industrial Automation**, **Smart Cities**, **Agriculture**, **Energy Management**, **Monitoring of the environment** and **Safety and Security** [32]. The Figure 2.2 gives an overview of the areas that are using this technology the most, with a trend for the coming years. There are several other areas where the IoT can be used, such as retail and the supply chain, transportation and logistics, among others [32].

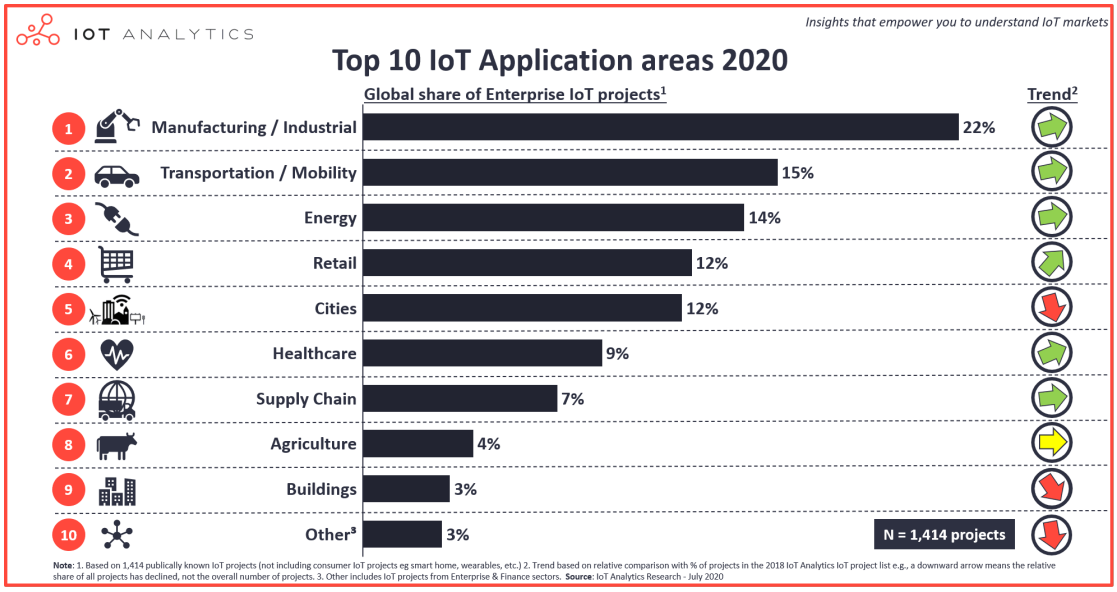

Figure 2.2: Top 10 IoT application areas [33].

As seen in the Figure 2.2, agriculture is eighth among the areas that use IoT the most. Even so, several commercial and non-commercial solutions are being developed for this

area. This domain has begun to revolutionize the agricultural industry, providing intelligent solutions for precision farming, greenhouse management and livestock monitoring [34].

Quinones-Cuenca et al. (2022) [35] used IoT and LoRa to develop a real-time tracking system. In addition, the system relies on a IoT platform, a mobile application and a web server to visualize the location data and necklace status of each node. Quinones-Cuenca covers all the layers also explored in this work, guaranteeing the real-time communication of the entire structure.

Makario, J., Maina, CW. (2021) [36] implements a low-cost IoT BLE solution for farm animal management, which can count, identify and locate animals on the farm with the aim of improving livestock welfare and production. This article has shown that farm animal management can be improved with BLE IoT.

Heni et al. (2023) [37] develops an application that is able to read ear tags on cattle using a RFID reader in which the reader is directed to the cattle's ear tag, reading the animal's identity data. With this identity, it is possible to check and monitor data from the time the animal arrives until it is sold. This ensures better management of the cattle as a whole.

There are also commercial devices that track animals, such as those provided by the companies Domodis [7] and Pastoral [8] mentioned in the problem statement addressed in this paper. These devices are designed to track cows, donkeys, small ruminants and others. Both use satellite communication and have their own data flow structure. Although the Domodis device has good autonomy, it is a large device and has communication faults according to various tests carried out by ESA. The Pastoral product, on the other hand, despite having a solar panel, does not have an interesting autonomy. Both devices presented the problem of rotation around the animal's neck.

All these applications are part of an IoT architecture with layers responsible for transporting and processing data. This work also presents an IoT architecture to ensure proper communication between all its elements.

2.2.3 IoT Architecture

There are a number of layers that usually comprise the IoT and these can be divided into the device layer, the data layer, the service layer and the interface layer [38][39]. These layers are responsible for collecting data from the physical environment, transmitting it over networks, processing and analyzing the data and providing user-friendly interfaces and functionalities. Figure 2.3 graphically shows an architecture with some possibilities in each layer.

- **Device layer** which is the fusion of sensors, actuators, physical components, hardware, connection capabilities and access points that make up a device capable of connecting to and interacting with a network. These devices are also called *node*.
- **Network layer** which is responsible for ensuring that data is sent to the next point in a secure and scalable manner.
- **Service layer** to which data processing is carried out and business rules are applied.
- **Interface layer** where the end user interacts with the system.

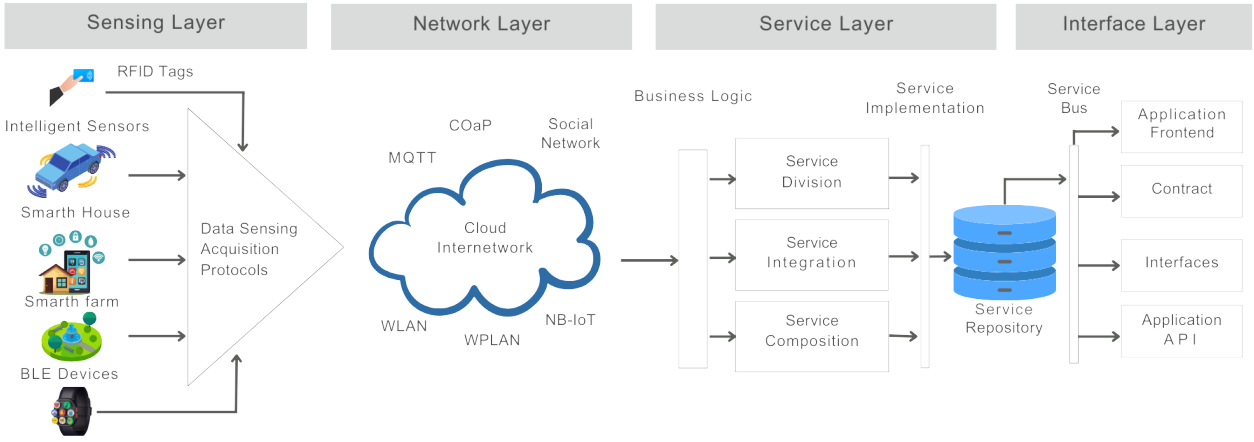

Figure 2.3: Architectural Layers of IoT [40].

An IoT infrastructure is the main element for monitoring and tracking animal behavior. The IoT infrastructure adopted by the livestock acquisition system and its ongoing

support facilitate the transmission and reception of livestock data [34]. Each layer of the infrastructure carries a series of technologies and protocols, but communication technologies are at the heart of the IoT.

2.3 IoT Communication for Livestock

Sensors in the IoT act as the eyes and ears of the digital environment [31], converting physical phenomena into digital data. These devices are designed to capture specific information, such as temperature, humidity, light, movement, geolocation, speed and various other parameters, depending on the application. The evolution of these sensors along with the advancement of IoT has brought with it a sophistication that goes beyond simple data collection, now incorporating local processing and machine learning capabilities.

When it comes to IoT, the data collected by sensors needs to be transmitted to other devices and the transmission technologies and protocols of the various layers of the internet are crucial to ensuring that this happens in a safe, efficient and, above all, scalable way, in order to provide information security. Communication between the *Device Layer* and *Network Layer* in livestock farming is done through various technologies, such as LoRa, BLE, ZigBee, Wi-Fi and GPRS [4] [41].

In this work, the LoRa technology will be explored due to the conditions of the case study, where small ruminants travel great distances, up to 5km between the stable and the furthest point reached by grazing. This technology allows communication over long distances with low consumption, making it cheaper and more efficient than satellite communication, for example [42]. This technology has ensured communication over long distances for tracking animals over wide areas, proving efficient for communications from 500m [43] to 12km [44][27][45][41].

Chapter 3

Methodology

This chapter will present the methodology used to achieve the objectives, specifying the functional and non-functional requirements, the application context, the technologies, the IoT architecture model and the limits of the device, gateway and other elements of the system.

3.1 System Requirements

The system requirements are designed to achieve the specific objectives of the work. These requirements are divided into functional requirements (FR) and non-functional requirements (NFR).

According to Sommerville (2011) [46], the functional requirements of a system describe what it should do, and they also depend on the type of system being developed, who its potential users are and the general approach adopted by the developer. The same author also states that non-functional requirements are requirements that are not directly related to the specific services offered by the system. Therefore, in this work Functional requirements define the system's actions and architecture, while non-functional requirements determine the context and limits they will have.

3.1.1 Functional Requirements

In order to better identify the functional requirements, they were divided into those of the device and those of the general architecture, ensuring a better division between them for searching and association with the rest of this document.

Functional Requirements of the Device

The device's functional requirements define the data that will be collected, its operating mode, operating limits and other details:

- FR01 - The device must work with an external battery and solar panel in parallel.
- FR02 - The device must collect data and send it at a maximum interval of 5 minutes.
- FR03 - The device must be able to collect the following data: geolocation, speed, temperature, humidity, battery level, inertial position and time (if necessary).
- FR04 - The device must receive a delivery confirmation for each data send.
- FR05 - The device must send a signal as soon as it is switched on to check that the entire architecture is working.
- FR06 - The device must contain a memory mechanism for storing undelivered data.
- FR07 - The device must wait for the geolocation data for a maximum of 5 minutes.
- FR08 - The device must reduce its power consumption while idle.
- FR09 - The device must contain an activation key to turn the device on/off.
- FR10 - The device must contain an indicator to check its operation as soon as it is switched on.
- FR11 - The device must display its operating status.

Functional Requirements of the Architecture

Like the device, the architecture to which it belongs also has its functional requirements:

- FR12 - The gateway must pre-process the received data and forward it to the Internet.
- FR13 - The data forwarded by the gateway must contain: data received and processed from the device and the quality of the LoRa and Wi-Fi signals.
- FR14 - There must be an em client to receive the data published by the gateway.
- FR15 - The system will store all data in a database.
- FR16 - The client must organize the data received and store it in a database.
- FR17 - The client must generate alerts on the following data: temperature, battery status, loss of connection and animals outside the defined zone.
- FR18 - The system must contain a data visualization platform.
- FR19 - The visualization platform must display all the data required by the user.
- FR20 - The client must provide the latest information when requested.

3.1.2 Non Functional Requirements

The non-functional requirements will delimit the contextual aspects of the project and its limits, specifying some of the technologies used in the system and some of their restrictions. Like functional requirements, non-functional requirements are also divided into device and architecture non-functional requirements.

Device non-functional requirements

Non-functional requirements relating to the device that collects the data:

- NFR01 - The device must send and receive data via Low Power Wide Area Network (LPWAN) to the gateway.
- NFR02 - The device must be reconfigurable.
- NFR03 - The device must be attached to a collar.
- NFR04 - The device will be tested on the Instituto Politécnico de Bragança (IPB).

Non-functional Requirements of the Overall Architecture

Non-functional requirements relating to the system architecture:

- NFR05 - The Gateway must receive data from the device via LoRa technology.
- NFR06 - All the technologies used in the project will be of the Open Source type.
- NFR07 - The communication model between gateway and client will be of the Publish-Subscribe type.
- NFR08 - The communication protocol used to connect gateway and client must be efficient for IoT applications.
- NFR09 - Communication between the gateway and the rest of the architecture must take place via the Internet.
- NFR10 - The client must be able to handle multiple tasks to optimize system performance.
- NFR11 - The database must be performant in time series for better system performance.
- NFR12 - The data visualization platform must be geared towards IoT applications.
- NFR13 - The alerts generated must be displayed on the visualization platform.
- NFR14 - The collar that will support the device must ensure ergonomics for the neck of the animal that will receive it.

3.2 Established IoT Architecture

In order to better understand the acquisition system and, in particular, the IoT architecture it will be embedded in, this session will explore how the architecture was designed so that the entire data flow was integrated.

Figure 3.1 shows a summary of the established structure and its layers. There are four layers: the acquisition layer, the network layer, the service layer and the interface layer. These make up the entire architecture and manage the flow of data from the animal that will charge the device to the user who will access the data and information generated by the system.

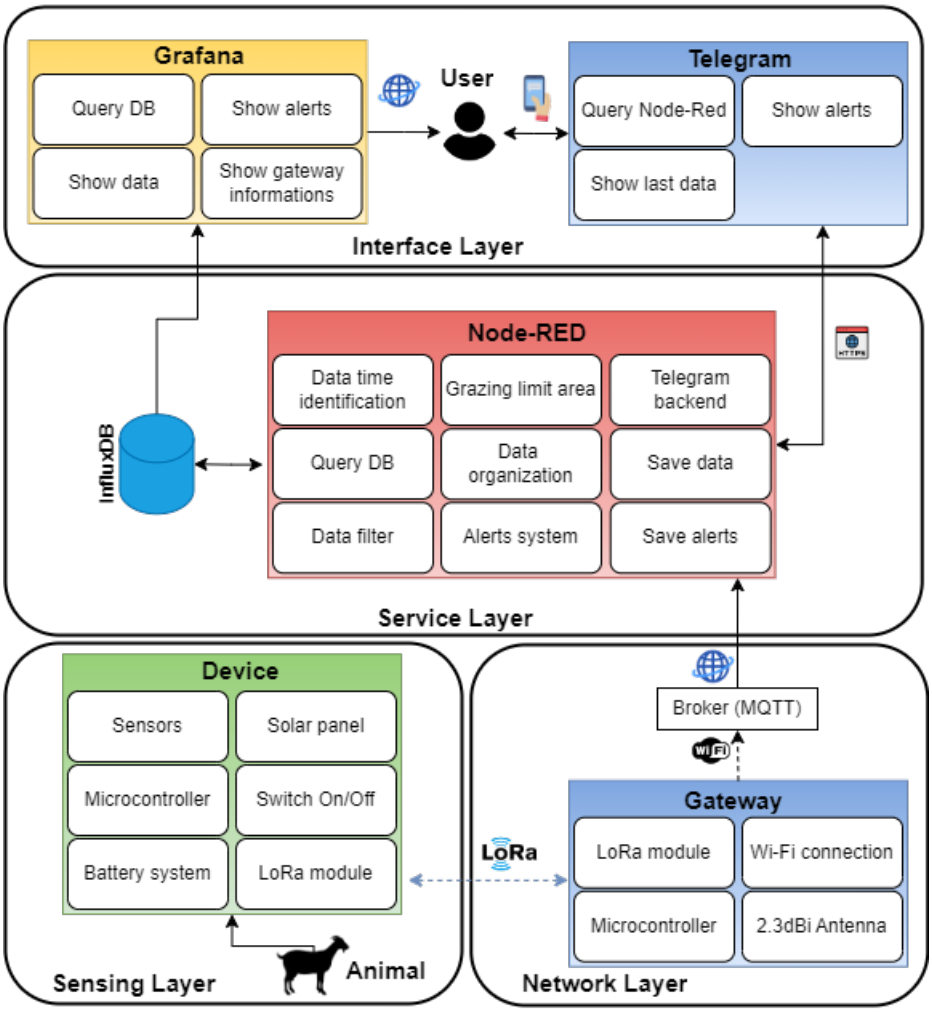

Figure 3.1: System architecture with all the layers and their components.

Each layer plays a crucial role in the architecture, all of whose functionalities, boundaries and contexts follow the requirements presented in the 3.1 session. The following topics will present the role of each layer.

3.2.1 Sensing Layer

The acquisition layer is represented by the device that will be attached to a collar which the animal will wear during its grazing and stay in the stable. The development of this device aims to collect ambient temperature and humidity, inertial state of the neck, geolocation and speed of the animal. In addition to this data, the device also provides battery status.

This device includes sensors, a microcontroller, a solar panel, an on/off switch, a LoRa module, a solar panel and a battery system. It is to be attached to boxes designed and printed with Polylactic Acid (PLA) thermoplastic filament on a 3D printer and finally attached to a collar. The shape of the collar, the dimensions of the boxes that support the device and the weight of the whole set must respect the limits imposed by requirement NFR14.

The device is responsible for fulfilling all the functional and non-functional requirements dedicated to it in the 3.1 session, so its development was based on these requirements. The development of the hardware and software will be further explored in the following sections of this document.

LoRa

LoRa is a spectral spread modulation technique derived from the Chirp Spread Spectrum (CSS) technology developed and patented by Semtech [44]. This technology enables communications up to 10km away with low power consumption, which characterizes it as an LPWAN technology. LoRa stands out as a highly advantageous option for IoT implementations in projects that require secure, long-range connectivity. Its ability to provide long-range communication resources is essential, especially for applications that

cover large geographical areas or are located in remote locations, which is the case of this study.

In addition, its energy-efficient operation facilitates the device's extended functionality, making it particularly suitable for scenarios in which battery longevity is a key concern. LoRa technology offers not only cost-effectiveness, but also scalability and straightforward implementation, making it accessible and practical in a broad spectrum of IoT applications, from environmental monitoring to real-time asset management. In addition, its ability to penetrate dense environments and overcome physical barriers further highlights its versatility and reliability in communication. For these reasons, this technology was selected for this work.

3.2.2 Network Layer

The network layer is responsible for guaranteeing all the architecture's integration, i.e. ensuring that the data received from the device is forwarded to the service layer via various communication and network protocols. It is this layer that categorizes the structure as an IoT architecture.

The Gateway is at the heart of the network layer, it is responsible for receiving data via LoRa from the device, processing it, organizing it in JavaScript Object Notation (JSON) format, and sending it to a Message Queuing Telemetry Transport (MQTT) broker. In addition to this, other data is also forwarded, such as signal quality LoRa and Wi-Fi.

This gateway was developed using a microcontroller with integrated Wi-Fi, a LoRa module and a 2.3dBi gain antenna. As well as the acquisition device, the Gateway was also attached to a box modeled and printed with PLA thermoplastic filament on a 3D printer. The details of the hardware, software and general operation will be presented in the next chapter.

MQTT

MQTT is an OASIS standard messaging protocol for the IoT. This protocol follows the publish/subscribe model, where a device publishes a message, it is forwarded to a broker and a client subscribes to that broker on the specific topic. This protocol is still lightweight, allowing for a small amount of code and minimal network bandwidth. MQTT is known for being extremely resource-efficient, making it ideal for IoT devices that may have power, processing and bandwidth constraints. It is designed to be lightweight and requires only a minimal amount of code and network bandwidth, which is crucial for IoT devices that may have limited resources [47].

MQTT is highly scalable and can be easily scaled to handle a large number of connected devices. What's more, its flexibility allows it to be used in a variety of scenarios, from simple devices to complex IoT systems. MQTT is designed to deal with scenarios where network connectivity can be intermittent. It offers features such as QoS (Quality of Service) that allow you to guarantee message delivery even in unstable network conditions [47].

The publish/subscribe model is highly suitable for IoT systems, where there is a need for asynchronous communication between devices. This allows devices to publish messages on specific topics without needing to know about the subscribers to those messages. Customers can then subscribe to the topics they are interested in and automatically receive the relevant messages.

In this model, since system components are not directly coupled, it's easy to add new publishers and subscribers as needed. This makes the model highly scalable and suitable for systems with a large number of devices and users. It offers flexibility in how data is distributed and consumed. Publishers can send messages to multiple topics and subscribers can subscribe to specific topics of interest. This allows for efficient data distribution and flexible system configuration. It can support high message rates and low latency. This is crucial in IoT systems, where processing and responding to data in real-time is important. For these reasons, this model and protocol was chosen for this

application, as it offers a robust, efficient and scalable solution for communication between IoT devices and the management system.

For this project, a public broker MQTT from hivemq.com was used with the address: broker.mqtt-dashboard.com. In this broker, publications are made in two topics, one to forward the connection status Wi-Fi and the other to forward the data coming from the device. The use of this protocol guarantees requirements NFR07, NFR08 and NFR09.

3.2.3 Service Layer

This layer is responsible for rules, analyzing and redirecting data, creating alerts, exchanging messages with a bot on Telegram and storing data in the InfluxDB database. This layer is made up of an Node-RED application and InfluxDB database. This application must receive the data published in a specific topic in the MQTT broker by the gateway, so it assumes the role of client in the architecture. The database must store all the data sent by the Node-RED application. In addition to the data captured by the device, the bank also receives the quality of the signals sent by the gateway, the alerts generated by the Node-RED application and the record of the type of data received by the gateway.

Node-RED

Node-RED is a programming tool for connecting hardware devices, APIs and services [48]. This is a low-code technology based on flows for visual programming, i.e. for each desired action within Node-RED it is necessary to create a flow of "nodes". Node-RED offers a flow-based graphical interface, allowing developers to create application logic visually by dragging and dropping nodes and connecting them to form complex workflows. This approach significantly reduces the need to write code manually, speeding up the development process.

It also has an extensive library of pre-installed nodes and an active community that contributes additional nodes for integration with a variety of IoT services and devices. This facilitates integration with hardware, communication protocols and cloud services,

providing a comprehensive solution for interoperability requirements.

Node-RED is ideal for rapid prototyping of IoT solutions due to its user-friendly visual interface and the vast library of nodes available. In addition, it can scale to support large-scale implementations, being deployed in cloud environments, containers or edge devices, providing a flexible solution for a variety of IoT scenarios. As such, this platform complies with the NFR10 requirement, and was chosen to manage the entire service layer.

InfluxDB

The database used in this work is InfluxDB, an open source, highly available and scalable time series database designed to handle large volumes of time series data. It is especially suitable for storing, querying and visualizing data that changes over time [49].

This database is highly optimized for time series queries, allowing for fast data ingestion and retrieval, even on large volumes of data. Its distributed, column-oriented architecture contributes to its high performance, making it suitable for scenarios where latency and scalability are essential. It can be easily scaled horizontally to handle large data volumes and growing workloads. It supports data distribution across clusters, allowing you to add more nodes as needed to increase storage and processing capacity.

Offers a variety of features and functionalities that make it flexible and adaptable to different application needs. It supports temporal aggregations and advanced analysis operations, as well as offering integration with other popular tools and services. InfluxDB has an active community of users and developers, which means that there are plenty of resources, tutorials and support available [49]. In addition, the company behind InfluxDB offers commercial support and professional services for organizations that need additional assistance. All these characteristics make this database suitable for the application of this work and make it comply with the NFR11 requirement.

3.2.4 Interface Layer

The interface layer is responsible for presenting the data to the end user. It should present a dashboard with graphs and/or information, depending on the type of data. This layer will be developed with the Grafana application, which makes requests to InfluxDB database, and a Telegram bot, which makes requests to Node-RED application.

Between the panels there should be a heat map for the geolocation points collected, temperature curves, humidity, speed and battery percentage. It should also contain text panels to display alerts and the type of data the gateway is collecting.

This tool should also be able to present the user with data according to a time interval defined by the user, in order to guarantee analysis at a specific time interval. On the other hand, the Telegram bot must provide the latest data stored in the database when requested by the user.

Grafana

Grafana is an open source data analysis and visualization platform. It allows users to create and customize visualization dashboards to monitor and analyze data from a variety of sources, such as databases, monitoring systems and APIs. With its intuitive and flexible interface, Grafana is widely used for system monitoring, application performance analysis and data visualization in a variety of sectors, from information technology to the IoT, which is the case of this work [50]. The use of this platform meets the NFR12 requirement.

This platform is highly interoperable and supports a variety of data sources, including relational databases, time series databases (such as InfluxDB), cloud services and many others. This makes it possible to consolidate data from various sources in one place for more comprehensive analysis. It offers a wide range of visualization options, including time series graphs, tables, maps, gauges and much more. In addition, it has advanced interactivity features, such as zoom, time range selection and drill-down, which allow you to explore the data in detail [50].

Is highly scalable and can handle large volumes of data and growing workloads. Its

distributed and optimized architecture guarantees fast and responsive performance, even when dealing with extensive data sets. Grafana has an active community of users and developers, which means that there is a wide range of features, plug-ins and integrations available [50]. All these features meant that this platform was used in this work.

3.2.5 Telegram

Telegram is an instant messaging app launched in 2013, widely recognized for its fast and secure communication platform. With features such as end-to-end encryption and the option to use the app without revealing phone numbers, it offers users a reliable way to communicate privately. In addition, Telegram allows users to create and interact with automated bots, adding an additional layer of functionality and personalization to [51] conversations.

3.3 System Installation Location

This session will cover where the system will be installed, i.e. specifying the test site and how the technologies should be coupled so that everything works as expected.

3.3.1 Device and Gateway

Herding small ruminants is a common practice in the north-east of Portugal [1]. There are several sites throughout the region where this practice is carried out by shepherds. In order to develop the acquisition system described in this paper, a site was chosen where this practice is similar or equivalent to the practices already adopted by shepherds in the region.

In view of this, the IPB, and specifically the ESA, have a herd of small ruminants with around fifty animals, the majority of which are goats. These animals graze in some defined areas within the campus, the Figure 3.2 shows a map containing the stable where the animals are concentrated and the areas belonging to the IPB where the herd grazes.

This will therefore be the test site.

Figure 3.2: IPB map with the grazing zone and stable.

The collar containing the application system must be attached to an adult animal already adapted to tracking collars, which is defined by the people responsible for the herd. The Gateway, on the other hand, should be attached to a place near the stable.

Defining where the system will be applied is of the utmost importance in order to optimize the entire process, building a prototype to meet the requirements and adapt to the location. The positioning of the gateway is a critical point in the project, since a site must be chosen that guarantees sufficient Wi-Fi signal quality to minimize communication failures. This point must also take into account the entire grazing area of the animals, since it will exchange information with the device periotically. Figure 3.2 shows the point chosen to position the gateway.

3.3.2 Services, Storage and Visualization

The applications made in Node-RED, Grafana and InfluxDB, which together make up the service and user interface layers, must be hosted on a single machine so that they can all be integrated locally. This ensures that the service and interface layer processes work as expected. Access to this machine must be made available to the administrator so that they can access the control panel created in Grafana application.

Chapter 4

Development

This chapter will explore how all the parts of the architecture were developed, presenting the construction process and its functionalities. The device, gateway, data integration and storage, visualization and alerts will be covered, all based on requirement NFR06.

4.1 Device

To present the development of the device, this session is divided into hardware development, software development and adapting the device to a collar. Introductory information on the technologies used in each part will be presented, sufficient to understand their development and operation.

4.1.1 Hardware

This section will present all the components that make up the device's hardware, including the microcontroller, modules, sensors, electronic components, power supply and circuit assembly. Figure 4.1 gives an overview of the device's hardware architecture.

All parts of the architecture were developed to meet the functional requirements of the device, i.e. from FR01 to FR11.

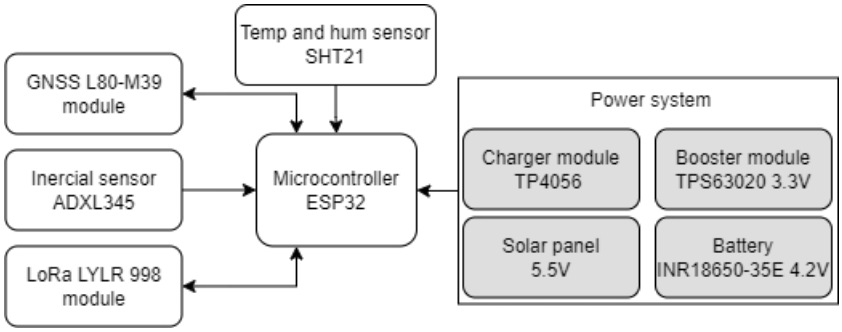

Figure 4.1: Basic electronic architecture overview.

Microcontroller

The ESP32 is a low-cost, high-performance microcontroller developed by Espressif Systems [52]. Because of its versatility of use and advanced features, this microcontroller is widely used in IoT projects. It has Wi-Fi 2.4GHz connectivity and integrated Bluetooth. It also has two Xtensa LX6 cores with a clock frequency of up to 240MHz.

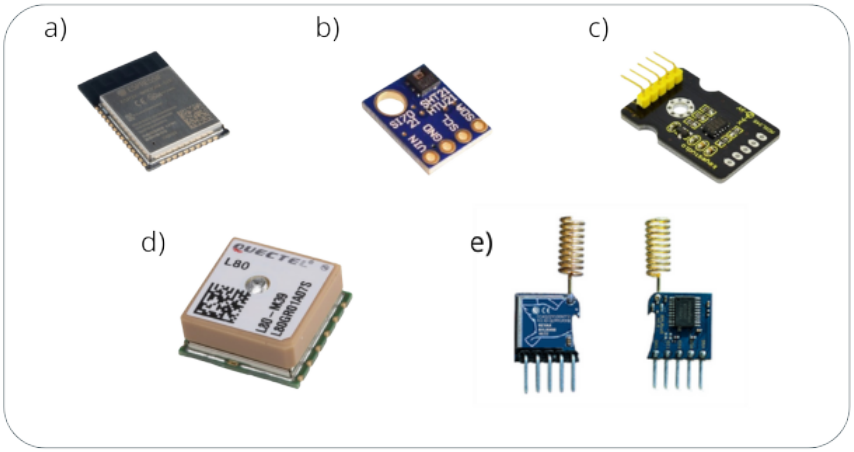

Figure 4.2: Device modules and sensors. a) ESP32-WROOM-32E standalone microcontroller; b) GY-21 module with SHT21 sensor to get temperature and humidity; c) ADXL345 keystudio module to obtain the relative neck angle; d) L80-M39 GNSS module to get geolocalization; e) RYLR998 LoRa module to transfer signal between device and gateway.

This microcontroller also has several peripherals and interfaces, including General Purpose Input/Output (GPIO)s, Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I2C), Digital-to-Analog Converter (DAC) and Analog-to-Digital Converter (ADC). The

model used in the device is the ESP32-WROOM-32E, which can be seen in the Figure 4.2. This model has 4MB of flash memory, 520KB SRAM, operates at 3.0V to 3.6V, has 38 pins GPIOs and can operate at -40°C to 105°C [52].

All these features and operating conditions are essential for integrating sensors and modules. The ESP32 also provides operating modes, such as *deep sleep*, where it reduces power consumption, turns off some GPIOs and starts operating again after some time or a user-defined GPIO change. There is also the option of changing the clock via software, which reduces its processing power and energy consumption. This microcontroller was chosen for all of the above technologies and for its low cost.

Temperature and Humidity Sensor

The SHT21 sensor developed by Senesiron [53] was used to collect temperature and humidity. It has low power consumption and low cost. A module containing this sensor, the GY-21 developed by Silicon Labs [54], was used to integrate it into the project, facilitating its integration with the rest of the system. This module containing the sensor can be seen in Figure 4.2.

This sensor has a I2C connection, 8-bit resolution and an accuracy of 2%RH for humidity and 0.3ºC for temperature. It also has an operating range of 0%RH - 100%RH and -40°C - 125°C for humidity and temperature, respectively. The sensor operates at 2.1V - 3.6V and consumes an average of 300µA of current [53], characterizing it as a low-consumption sensor. This sensor was chosen because of its ease of operation, small physical size and, above all, because it is a low-cost sensor.

Inertial Sensor

The ADXL345 developed by Analog Devices was used as the inertial sensor. This sensor measures the static gravity in tilt detection applications, as well as the dynamic acceleration resulting from movement or shock. As a result, it returns the variation in three axes, X, Y and Z with a resolution of 3.9mg/LSB, allowing the detection of inclination of

less than 1.0º, more than enough for the application of this work, which is interested in collecting the position of the animal's neck [55].

The sensor used in this work is coupled to a module that interfaces it to the microcontroller. Figure 4.2 shows the module developed by Keystudio with the ADXL345 sensor used in this work [56].

This sensor operates at -40ºC to 85ºC, has I2C and SPI connections, but the one used in this project was I2C. It also operates at 2.0V to 3.6V, has a resolution of 13 bits and consumes 23µA of current in read mode, i.e. it is a low-power sensor, ideal for this work. This sensor was chosen because of its high precision, easy handling through its module and low cost.

GNSS Module

To collect the animal's geolocation and speed, the L80-M39 GNSS developed by Quectel [57] was used. This module has an integrated antenna and is capable of collecting data at -165dBm in tracking mode and -148dBm in acquisition mode. Its dimensions of 15.0mm x 15.0mm x 4.0mm make it a small and suitable module for this job. Figure 4.2 shows the module.

In addition, this module operates in -40ºC to 8ºC, 3.0V to 4.3V, has 66 acquisition channels and 22 tracking channels, consumes 25mA in acquisition mode, 20mA in tracking mode and 1mA in Standby mode. It uses the National Marine Electronics Association (NMEA) protocol to communicate, and takes around 35s to collect the first coordinates when switched on in Cold Stard mode and less than 1s when switched on in Hot Stard mode, according to the developer. These characteristics are fundamental to the efficiency and operation of the system and to reducing the consumption of the device. This module was chosen because of its small size, easy handling, standby mode, low consumption and low cost.

LoRa Module

As seen in Figure 4.2 and in accordance with requirement NFR01, the device uses LoRa technology to communicate with the Gateway. This project used the LYLR998 LoRa module developed by REYAX.

This module operates at 2.3V to 3.6V, the frequence radio works with a power range of 0 to 22dBm, with a sensitivity of -129dBm. It also works with a frequency range of 820MHz to 960MHz with an accuracy of ±10ppm. The module is capable of operating at -40ºC to 85ºC. With a power of 22dBm it consumes 140mA to transmit and 17.5mA to receive data [58]. In addition, this module works with an AT command system and contains an integrated spiral antenna, as can be seen in the Figure 4.2.

In Portugal, Autoridade Nacional de Comunicações (ANACOM) defines the purpose of the frequency bands. LoRa is included under "Non specific applications" and can use the 863-870MHz bands with a power of less than or equal to 14dBm [59]. For this work, the settings for the LoRa module were set at 868.5MHz frequency and 14dBm power, where it consumes 115.5mA of current each time it is sent. This module was chosen because it already has an antenna attached, has low power consumption, is easy to operate with AT commands and is inexpensive.

Power System

The power system was designed based on the FR01, i.e. it must contain a solar panel and batteries. The battery model for this project is Lithium-ion INR18650-35E, while the solar panel is 0.66W. Other fundamental modules for this system are the TPS63020 voltage regulator module, TP4056 battery charger module and an on/off switch to turn the entire power system on or off. These are the main components of the power system and can be seen in Figure 4.3.

Each battery provides 3350mAh and three units were used, so when charged the system has a charge of approximately 10050mAh, this amount of batteries, in addition to the primary function of supplying energy to the system, also ensures counterweight to

the collar that supports the device, this will be better discussed in the following sessions. In addition, these batteries have 4.2V when fully charged, 3.6V nominal voltage and stop supplying current at 2.65V, this information will be fundamental for the development of the battery percentage reading system. This battery model was chosen because of its robustness and charging capacity, which are key characteristics for this application. In addition, they can supply up to 2000mA, which is more than enough for the system to function, and this will also be covered in more detail in the following sections.

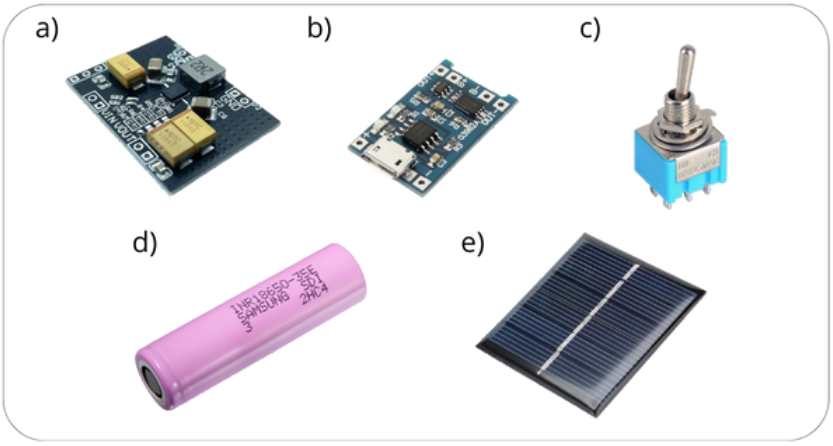

Figure 4.3: Power system parts. a) voltage regulator module TPS63020; b) battery charger module TP4056; c) switch on/off; d) battery INR18650-35E; e) solar panel.

As the batteries oscillate between 2.65V and 4.2V, a voltage regulator is needed to ensure that the power supplied to the system is optimal. All the devices presented above work with a nominal voltage of 3.3V, so the TPS63020 voltage regulator is responsible for guaranteeing 3.3V to the system. As this module is a booster, it is able to regulate the voltage between 1.8V and 5V to 3.3V, with an efficiency of approximately 96%, in addition to being low-cost.

The 5.5V polycrystalline solar panel measures 84.5x55.5mm and has a current of 120mA with a peak power of 0.66W when the light intensity is 38000LUX. The low cost and efficiency of this solar panel fit perfectly into this project. The TP4056 module is used to charge the batteries and integrate them with the solar panel. This module

manages the system's energy, allowing the panel to recharge the batteries and this pair to supply energy to the voltage regulator.

According to FR09, the device must have a switch to turn it on and off, so a 125V on/off switch was added to the power supply system. This switch supports up to 3A and 125V, which is more than enough for the device. It also has a resistor to be switched on or off, which is essential to avoid unexpected state changes.

Circuit Assembly

The assembly of the device was divided into two parts, the first of which will be called the "upper" part of the device, which has the main components for collecting data, such as the microcontroller and modules. The second is called the "lower" part, which contains the batteries, the battery charging module and the on/off switch. This division will be further explored in the 4.1.2 section.

Upper Part

The *upper part* includes the microcontroller, modules, resistors, capacitors, buttons, LEDs, JST connectors, Bipolar Junction Transistor (BJT) NPN BC547 transistors and the solar panel. Figure 4.4 shows the entire electrical circuit and its components. The solar panel is not shown in this electrical diagram because it is a component that must be installed externally, above the box that supports the *top part*.

In the circuit shown in Figure 4.4 there are a series of connections that connect the modules to the microcontroller, voltage regulator module, Ground (GND), BJT NPN transistors and even headers. There are also other subsystems, such as the reset, battery voltage reading and voltage filter. There are also two JST connectors, one with four pins, labeled U13, and the other with two pins, labeled U7, which respectively connect the *bottom* and *top* parts and the *top* and solar panel.

The TPS63020 voltage regulator module is the first module to receive a charge, the voltage coming from the batteries is regulated by it and then supplied to the entire *upper* part shown in the 4.4. The voltage regulator has three main pins, Vin, Vout and GND.

It collects the input voltage from the batteries via the Vin pin and returns the regulated voltage of 3.3V to the Vout pin.

The microcontroller and all the modules have positive and negative terminals represented by VCC and GND, respectively, and these are powered by the voltage regulated by the TPS63020 module. In addition, there are two electrolytic capacitors identified as U8 and U3, both 100µF and 25V, to mitigate voltage drops resulting from the operation of the system. This is essential for the system components to work as expected.

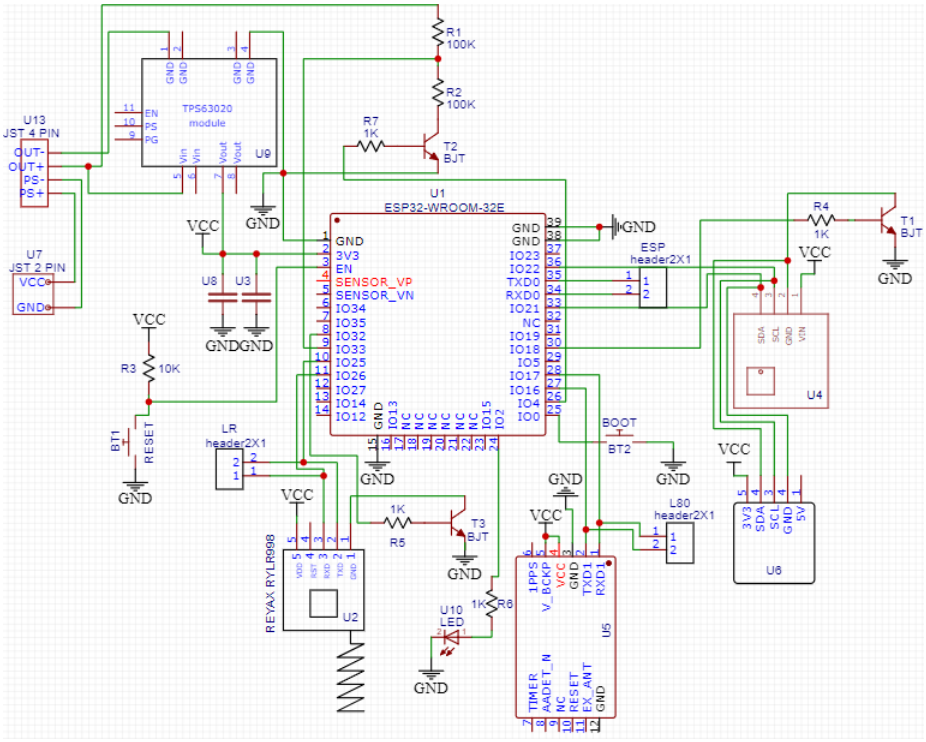

Figure 4.4: Upper part electrical circuit.

The microcontroller is powered by the GPIO identified as 3v3 and also by the GPIO EN, which allows the microcontroller to remain enabled, i.e. switched on. In addition, a BT2 button connected to GND on GPIO 0 has been defined which, together with the BT1 button, is responsible for putting the microcontroller into download mode to receive new software.

The LoRa LYLR998 modules identified as U2 and the GNSS L80-M39 identified

as U5 communicate with the microcontroller using Universal Asynchronous Receiver-Transmitter (UART) protocol, so they are connected with GPIOs that enable this communication. The modules that have the temperature/humidity and inertial sensors communicate with the microcontroller using the I2C protocol. As they are connected in master-slave form, they both connect to the same Serial Data Line (SDA) AND Serial Clock Line (SCL) terminals provided by GPIOs 21 and 22.

The BJT NPN BC547 transistors T3 and T1 are responsible for enabling and disabling the LoRa, keystudio, GY-21 modules identified as U2, U6 and U4. Transistor T2 is responsible for enabling and disabling the battery reading subsystem.

As the battery voltage varies between 2.65V and 4.20V, it is not possible to take a direct reading of its voltage, as the microcontroller is capable of reading up to 3.3V with a resolution of 12 bits. Therefore, two 100k resistors R1 and R2 divide the voltage supplied by the batteries and this is read by GPIO 33. The maximum value read by this GPIO is 2.6V due to the voltage division generated, this value is corrected via software and the percentage value is returned. The T2 transistor enables and disables this reading; it was added to prevent constant current discharge.

You can also find three 2x1 headers connected to the LoRa, GNSS and microcontroller modules. These headers are essential for monitoring the operation of the modules and transferring software to the microcontroller. The circuit also has an indicator led to meet the FR10 and FR11 requirements. The 4.1.3 section shows how this led will be used.

To integrate all the components of the *upper part*, a Printed Circuit Board (PCB) was developed based on the circuit shown in Figure 4.4. The PCB can be seen in Figure 4.5. All the components of the circuit, including its representation, are contained in the PCB, which makes it easier to solder the components.

On the PCB you can see four circular holes, three 4mm in diameter and one 3mm in diameter, which support the screws that attach the board to a support box. There is another hole with a “BJT” indicator, which will be responsible for receiving the cable that connects the *top part* with the *bottom part*. This cable is approximately 25cm long and has four $0.22m^2$ flexible wires. Its function is to provide a connection between the

two parts, supplying power from the batteries to the *upper part* and sending power from the solar panel to the *lower part*.

Another important detail is with regard to the terminals that will receive the modules. For guidance, the rounded terminals are for GND or data and the square ones are for VCC pins (the terminals that will receive the microcontroller and the GNSS do not follow this rule due to their shape). The Figure B.1 in appendix show this terminals with more details.

There are also logos for laboratories such as Sustestec, Cedri, Pastorpraxi and IPB itself. These are part of the context to which this project was applied. All the boards in the project were developed using Easyeda software, an online board design platform [60]. After soldering all the components, the PCB was as shown in the Figure 4.5.

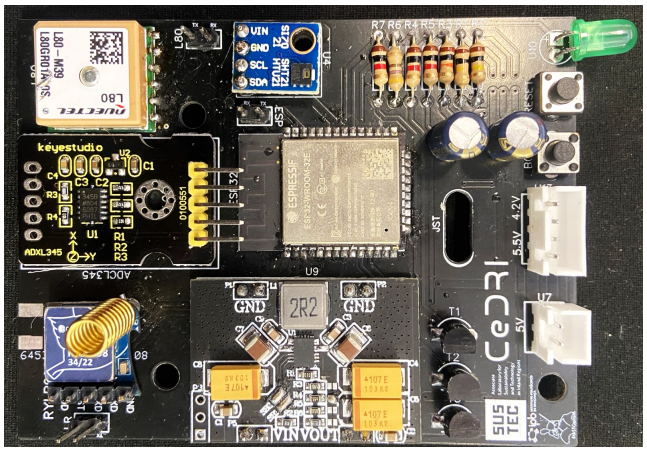

Figure 4.5: Upper PCB soldered.

Bottom Part

To develop the *bottom part*, an electrical circuit was designed that integrates the batteries with all their components. This circuit can be seen in more detail in Figure 4.6 of this document.

In the circuit shown in Figure 4.6 there are three batteries connected in parallel that connect to the battery charger. The on/off switch is connected to the negative pole of the charger which connects to the JST connector, this ensures that when switched off, the

power supplied by the batteries is cut off.

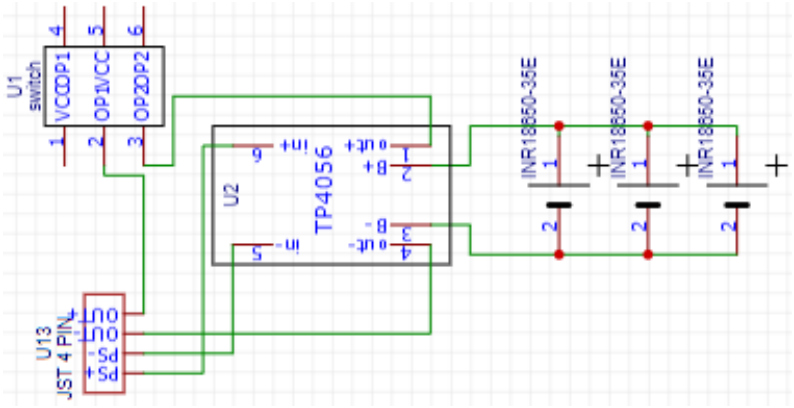

Figure 4.6: Botton part electrical circuit. With this image you can better understand the pins and the layout of the components on the board.

The solar panel that connects to the *top part* supplies power to the battery charger and this is transferred via the PS+ and PS- pins found on the BJT connector to the charger's In- and In+ pins. As the on/off switch only allows current to pass from the tracks that supply power to the *upper part*, the connection between the solar panel and the battery charger is never interrupted, which means that even when the switch is off, the panel can still supply power to the batteries.

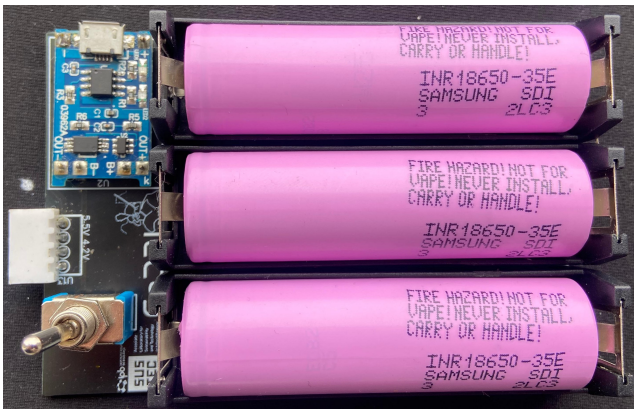

Figure 4.7: Botton PCB soldered.

A PCB was developed to integrate these components, which can be seen in the appendix to this document. This contains six 4mm diameter holes for fixing screws that connect the board to a support box. This board is 96.64mm by 64.90mm and also has

the same PCB logos as the *upper part*. Figure 4.7 shows the PCB with the components already welded on.

4.1.2 Adapting the Device to a Collar

Even though it is widely discussed, ergonomics is one of the main pillars of the project. Therefore, in order for the device to collect data from the animal, it must be coupled to a collar and this set must not be ergonomically harmful to the animal. This complies with requirement NFR03.

The first step was to develop boxes to receive the PCBs of the two parts of the device. Both boxes were printed on PLA at 35% fill. They all have a 3mm wall, giving them protection and resistance.

Figure 4.8 shows the model of the box that supports the *top part*. It has two rods at the bottom for attaching the collar, a 4mm side hole for the LED indicator, a depression in its lid to hold the solar panel and a frame with indicator holes for adding screws to fix the lid to the body of the box.

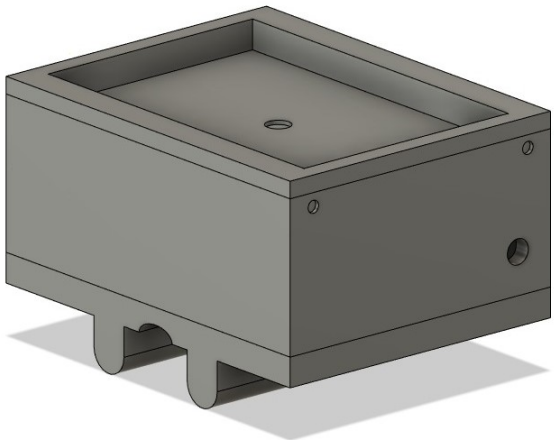

Figure 4.8: Support boxe for upper PCB.

At the base of this box there is also a low relief 10mm long and 5mm high, which makes the goat's neck fit better into the device, since goats have a neck that has a "ridge". At its base there is also a hole that allows the cable connecting the parts to access the pins

of the four-pin JST connector. You can also see this box from different angles with the Figure 4.19.

The case of the *bottom part* can be seen in Figure 4.9. This features a front cut-out to receive the cable that connects to the four-pin JST connector, a side compartment to access the battery charger's micro usb port and a hole at the top that allows the top of the on/off switch to be accessed. There are also four screw holes for fixing the cover.

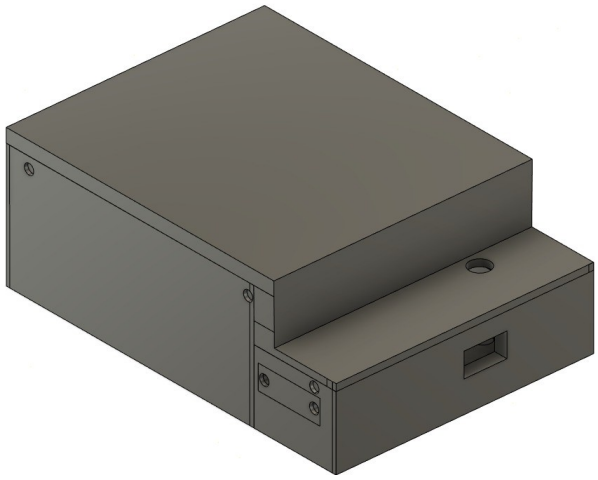

Figure 4.9: Support boxe for bottom PCB.

The collar assigned to this set of boxes is a leather collar measuring approximately 60cm. It has malleable characteristics to provide two turns in the *upper part* box, one on each support rod at the base. Figure 4.10 shows the collar with the upper case, a bag that holds the bottom case and the connection cable on its side. The purpose of this pocket, which holds the bottom case, is to protect the bottom case and connect it to the collar.

You can see that there are two ellipses being formed with this winding format, a larger one that supports the pouch that holds the bottom case and the collar buckle, and a smaller ellipse that wraps around the animal's neck. This construction model means that the connection cable and the pouch that holds the bottom case do not come into direct contact with the animal, protecting the device and ensuring greater comfort for it.

Figure 4.10: Collar with device. Above is the collar with the bag closed and below with the bag open.

4.1.3 Computational Program of Device Actions

The computational program or software was developed in the C/C++ language and presents several important points for the system to function properly, such as a memory system to store data that has not been delivered to the gateway, prior tests on modules to assess their status, among others.

The flowchart shown in Figure B.4 in appendix shows the main instructions carried out by the software. The structure of this flowchart shows a series of pre-defined processes that unfold as you read. However, the basis of the entire algorithm is the exploration of the three initial processes, i.e. *INITIALIZATION*, *SETUP* and *LOOP*, from which all the other processes are called. The code can be found in appendix A and complies with requirement NFR02, i.e. it can be reconfigured.

The first process is "INITIALIZATION", which is responsible for calling all the libraries and defining the program's global variables. These are responsible for facilitating

the interface with the device's modules, providing methods and classes for interacting with them. As the Arduino Integrated Development Environment (IDE) frameworok was used, two pre-defined processes are called that provide the functioning of the entire algorithm, these are *setup()* and *loop()*, these are also called functions. The setup() function defines the initial settings for the algorithm, such as pin settings, communications, etc. The loop() function contains the operation of the device, with its main features, communication with modules, processing, filtering and data conditioning, among others [61].

In general, the setup() function sets the microcontroller's operating clock to 20MHz to reduce its power consumption, sets the GPIOs and checks and activates the first modules. The second decision is to check whether the device has been restarted in any way or has woken up from *deeps leep* mode. If it has woken up, the system displays the information from the LoRa module, collects the battery status and sends a signal via LoRa containing it. This is important in order to check the LoRa module's information such as address and netwoorkID and to check that the entire IoT structure is working and complies with the requirements.

The loop() function starts the process of testing and reading the data provided by the GNSS module. It is only when the geolocation is captured, i.e. latitude and longitude, that the other data can be captured and then sent. If everything goes as expected, the system builds a string with all the data and sends it via LoRa. A confirmation is then expected from the gateway to ensure that this data has been published correctly in the MQTT broker, meeting requirement FR04. This confirmation protocol can be seen in the Figure 4.11. If this confirmation message is not received, the system makes another attempt to send it. If nothing is still received, the program adds the time to which this data was acquired and saves this new string in the microcontroller's flash memory, ensuring that FR06 is met.

If the GNSS module doesn't capture the geolocation after 5 minutes of trying (as per FR07), it checks for any previous acquisition strings; if there are any, the device tries to find the gateway by sending a simple string containing only the battery status, in which case the protocol in Figure 4.11 is also executed. If the device receives the confirmation

message, it sends the previous acquisition string and the confirmation protocol is repeated. This is done until all the strings have been sent or something goes wrong with the protocol, such as a lost connection or poor signal quality. If the geolocation is not captured and there is nothing in memory, the microcontroller goes into *deep sleep*. At the end of all processes, the microcontroller goes into *deeps leep* until it wakes up again and restarts the whole process. This process takes place every five minutes, meeting requirement FR02.

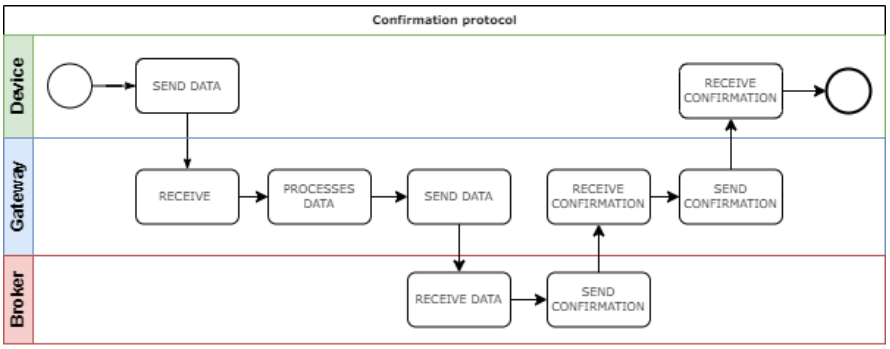

Figure 4.11: Confirmation protocol business process flow.

With this confirmation protocol, no captured data is lost due to a lack of connection between the device and the gateway or between the gateway and the MQTT broker. In addition, the *deep sleep* mode is a way of putting the microcontroller on minimum consumption, turning off the Central Processing Unit (CPU) and leaving only a few peripherals on, which meets requirement FR08.

As you can see, the data sent to the gateway is in string format, and can be delivered in real time or stored in the microcontroller's flash memory. The two possible data string formats can be seen in Figure 4.12. In these two examples, the data shown is fictitious and follows the following sequence and values: latitude: 41.123456; longitude: -6.123456; speed: 1m/s; temperature: 20ºC; Humidity: 45%; Inertial sensor axes X: 0.04, Y: 9.12 and Z: -1.48; Battery percentage: 65% and timestemp: 12/14 at 22:15 (does not contain year or seconds). The content of these strings guarantees FR03.

When the device is able to send the assembled string as soon as the data is collected, it is not necessary to send the timestemp, which is why there are two string formats. You can also check the size of the two types of string in terms of the bytes each string

field occupies, which can vary between 50 and 70 bytes, depending on the data captured. There are also some markers before and after each piece of data, these are letters A to H for variables and K for timestemp, each marker occupies 1 byte in the string. These are important for the gateway to identify and separate the data for each variable contained in the string.

Descrition	-	Lat and Lon	-	Vel	-	Temp	-	Hum	-	X	-	Y	-	Z	-	Bat	-	Time	-
Bytes	1	18	1	~	1	~	1	~	1	~	1	~	1	~	1	~	1	8	1
String with timestemp	A	41.123456-6.123456	B	1	C	20	D	45	E	0.04	F	9.12	G	-1.48	H	65	I	12142215	K
String without timestemp	A	41.123456-6.123456	B	1	C	20	D	45	E	0.04	F	9.12	G	-1.48	H	65	I		

subtitle - : separator ~ : variable

Figure 4.12: Strings formats. Green: string with timestemp; Blue: string without timestemp.

4.2 Gateway

A gateway is responsible for communication between the device and the broker MQTT. For this purpose, hardware and software were developed, designed and programmed to make the connection between the device and the broker.

4.2.1 Hardware

As the gateway has the function of "forwarding" data in a programmed way, it must focus on modules that guarantee the communications necessary for this purpose. In this case, the technologies used are LoRa and Wi-Fi, so the gateway must contain a microcontroller with an integrated Wi-Fi connection and must communicate with a LoRa module to receive messages from the device. It must also contain an antenna to extend the range of the LoRa module, ensuring greater distances between it and the device. The use of LoRa technology guarantees requirement NFR05.

As part of the hardware, an ESP32 WROOM 32E MCU microcontroller was used, similar to the one used in the device, however it is attached to a development board, which allows it to be powered via cable. This microcontroller has 802.11b/g/n wireless

communication standards, A-MPDU and A-MSDU aggregation and Center frequency range of operating channel 2412MHz - 2484MHz [52]. This microcontroller was used because of its technologies and mainly because of its low cost.

In addition to the microcontroller, there are also two LED indicators, which provide the status of the gateway and whether any data is being received or sent to the device. The LoRa module is the same as the one used in the device, but a 195mm long antenna operating at 868MHz with a gain of 2.8dBi has been added to increase the module's range. All of these can be seen in Figure 4.14.

Circuit Assembly

Figure 4.13 shows the electrical diagram of the gateway, containing the microcontroller, LoRa module, LEDs, resistors and headers to ensure communication with the microcontroller and module after soldering in a PCB.

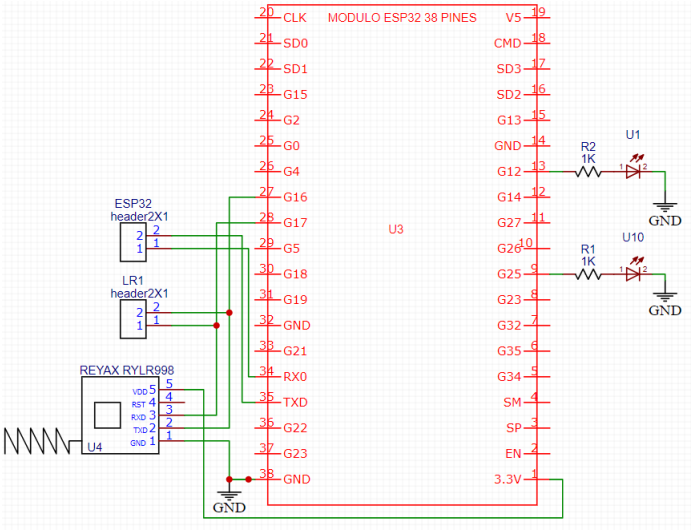

Figure 4.13: Gateway electrical circuit.

As with the device, a PCB was also developed for the gateway, measuring 78.7mm by 45.0mm. This can be seen in more detail in the appendix to this document. This PCB also has the logos shown on the PCB of the device. You can also find the antenna connection terminals (two strips in the left-hand corner of the picture) and four 4mm

diameter holes for screwing the board into its support box.

The PCB with the soldered components can be seen in the Figure 4.14. You can see that this is the same LoRa module as the device, but without the spiral antenna, which has been replaced by a larger antenna with a longer range.

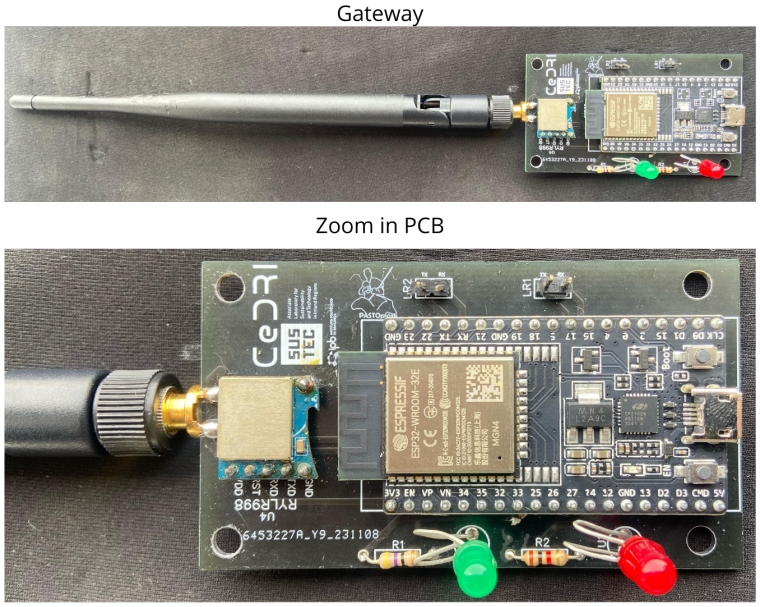

Figure 4.14: Gateway PCB soldered. In top have the complete gateway and in botton a zoom in the PCB.

Support Structure

In order to protect the PCB, a support box was developed. This can be seen in Figure 4.15. The box has two LED indicators on the top, a hole for the power cable on the bottom and another on the front for the antenna.

For the gateway to be installed on a physical site, it needs to be added to a sturdy structure that offers protection and waterproofing. To this end, a PVC structure was developed to support the gateway. It is approximately 40cm long and 75mm in diameter. It has two covers, one that houses the antenna and the other that allows the power cable to access the micro USB input of the microcontroller. This structure can be seen in Figure 4.16.

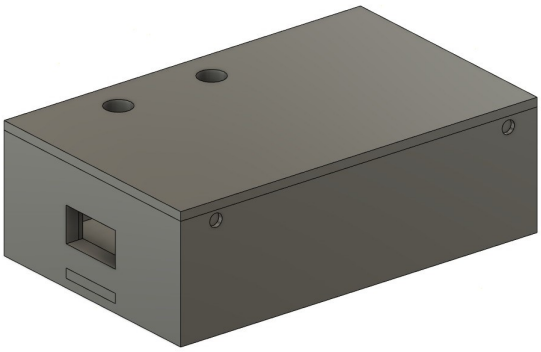

Figure 4.15: Support boxe for PCB to gateway.

Figure 4.16: Gateway struct.

4.2.2 Computational Program of Gateway Actions

The gateway software has been developed for the following purposes: to receive, process and forward data from the device, to remain connected to the broker MQTT and to report its status to the end user.

Flowchart providing the main parts of the software can be found in the appendix of this document. FreeRTOS was used to allow parallelism between tasks. FreeRTOS is an open-source real-time operating system that makes it possible to run parallel tasks on embedded systems. With FreeRTOS, developers can create systems that execute multiple tasks simultaneously, allowing independent operations to be executed concurrently.

In the flowchart there are two main processes, *INITIALIZATION* and *SETUP*. The first process makes the initial settings, such as calling libraries, assigning global variables and so on. The second defines the function of some of the microcontroller's pins, connects it to the Wi-Fi and finally calls the three main tasks.

The first task is to evaluate the connection between the microcontroller and the MQTT

broker, and this is done periodically. If there is a connection, nothing is done, but if the connection is not established, a LED is activated and the software tries to make the connection instantly until the connection is re-established. The second task is responsible for sending the broker the quality of the Wi-Fi, this is important so that the administrator can be informed of the status of the gateway and the quality of the Wi-Fi signal. Finally, the third and main task is responsible for receiving the data via LoRa, this has the highest priority and is responsible for receiving, processing and forwarding the data, guaranteeing FR12. This function also returns confirmation to the device that everything went as expected, according to the confirmation protocol shown in 4.11.

The data received and processed by the third task arrives in string format, which can be seen in the Figure 4.12, so the task processes this string, separating the data according to each variable received (temperature, humidity, latitude, longitude, speed, x y z axes, battery percentage and timestemp, if it is data stored in memory), assembles a JSON, adding the signal quality LoRa and Wi-Fi, according to FR13 and forwards it to the broker MQTT.

4.3 Data Integration and Storage

So far, the components of the acquisition and network layers have been presented, for which hardware and software have been developed so that everything works as expected. From this session onwards, the items developed for the service and interface layer will be presented. As explored in Chapter 3, these layers are made up of Node-RED, InfluxDB, Grafana and Telegram technologies.

4.3.1 Node-RED Services

Node-RED application is the client of the publish/subscribe model that will receive all messages published by the gateway and requests made by Telegram bot, meeting the FR14 and NFR10 requirements. It is also responsible for filtering, processing, analyzing and saving data. It also generates alerts and forwards them to Telegram bot and InfluxDB

database. For all this to happen, the flows have been divided into *global injects*, *processing of data*, *Telegram bot* and *alerts*.

Global Injects

The nodes that make up these flows are responsible for managing and setting some global variables that limit some processes of other nodes. There are three inputs that restart or define global variables, these are repeated at 00h, 08h and 16h of the day. The nodes that run with these inputs set flags limiting alerts, the year (it's important to fill in the data received with timestemp), the Telegram ID of the user who will receive the alerts and the definition of the grazing area and gateway location. This information is important in several other nodes and allows data to be easily updated. This section of nodes can be seen in the Figure 4.17.

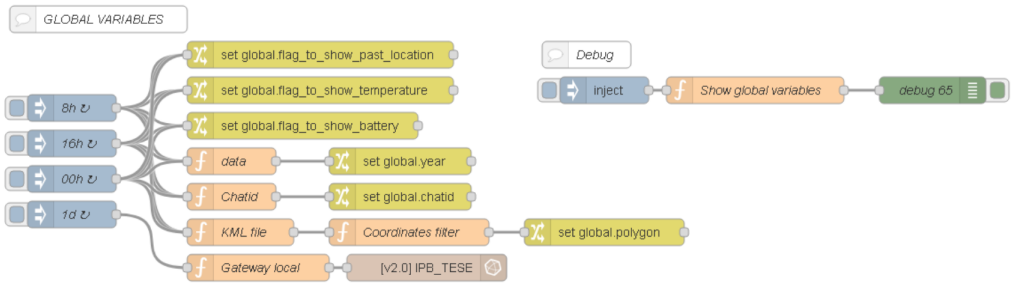

Figure 4.17: Flows responsible for global injections.

To define the grazing area, Keyhole Markup Language (KML) files are used, which are acquired using points captured with the "paths or polygons" tool found in Google Earth. Therefore, in order to define the grazing area and generate alerts from it, it is necessary to define it with the help of Google Earth. This is done by copying all the code from the KML file generated into the *KML file* function. In this work, as the project's application site is IPB, the grazing areas delimited in it were defined exactly.

Processing of Data

The flows responsible for processing and storing the data coming from the broker MQTT admit the main role assigned to the Node-RED application. These are responsible for subscribing to the specific topics of the broker to which the gateway publishes, filtering the incoming data, identifying whether the data is "current" or "stored in memory", organizing it in a format accepted by InfluxDB database and finally saving it. These flows can be seen in Figure 4.18.

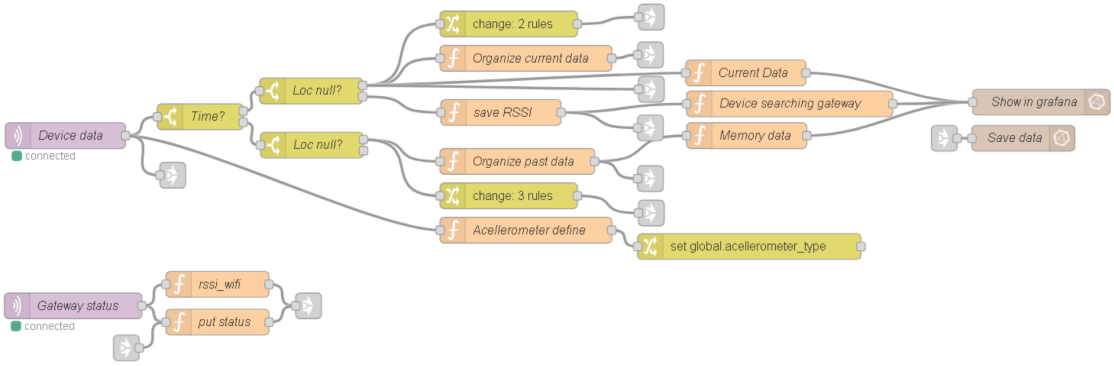

Figure 4.18: Flows responsible for data processing.

There are two main flows, one that starts with a subscription to the thread that receives data from the device and another that receives data generated by the gateway. The first, after receiving the data, performs two filters, analyzing whether the data is "current" or "stored in memory". Based on this, the following flows organize this data to save it in InfluxDB database. This flow also stores data type information in the text database, so that Grafana application can show the administrator the type of data being sent by the device. The first flow also serves as an input for the alerts, which will be covered in more detail in the present flows under *alerts*. These two flows guarantee requirements FR15 and FR16.

Also in the first flow is the interpretation of data from the inertial sensor. As seen above, this sensor returns values for the X, Y and Z axes, which can vary from approximately -10 to +10, depending on the inclination of the device. However, in order to interpret this data, some positions were mapped according to combinations of the values

collected by the sensor, i.e. using the data from the axes to generate a possible position for the animal's neck (or other unexpected event), whether it is inclined, declined, among others. Figure 4.19 shows the five mapped combinations, ranging from 1, 2, 4, 6 and 8. When no position is recognized by the algorithm, the value 0 is recorded.

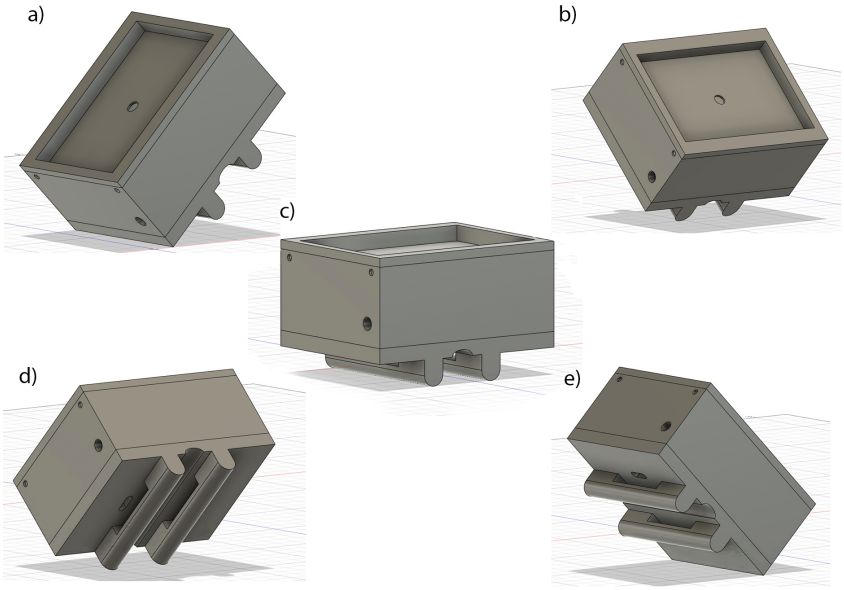

Figure 4.19: Positions mapped according to the data collected by the inertial sensor. a) position 4; b) position 6; c) position 1; d) position 2; e) position 8.

The second flow is responsible for receiving the information posted by the second task executed on the gateway, so it receives the quality of the Wi-Fi, stores it in the database, and presents it to the user in two ways, generating a serial graph containing the quality of the Wi-Fi and a status indication bar that varies between green (on) and red (off).

Telegram Bot

The flows responsible for communicating with a bot in telegram are divided into two: the first provides the user with the request options they want and the second fetches the data they request. Figure 4.20 shows these two flows.

The first flow is executed every time the user sends a "command" to the bot, which returns five options to the user: location, temperature, humidity, speed and battery

percentage. In addition to these options, this flow also searches for the date of the last data stored in the database, providing the user with a correlation between the data they receive and the time it was acquired.

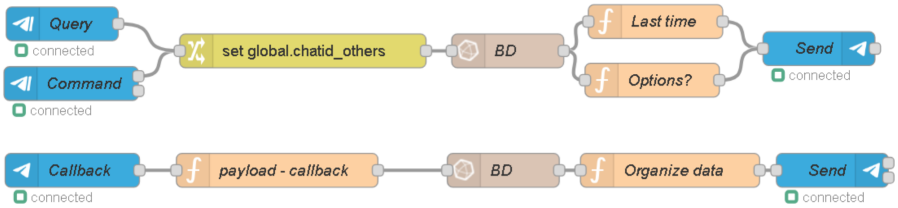

Figure 4.20: Flows responsible for communication between Node-RED application and Telegram bot.

The second flow is executed as soon as the user chooses one of the options provided by the first flow. It searches for the desired data in the database, organizes it according to its type in a message, adds a unit of measurement and sends it to the user. However, for geolocation, the user is returned a map with the point referring to the last location collected. To do this, the flow searches for latitude and longitude and informs the bot that this is data of type *location*. These two flows guarantee requirement FR20.

Alerts

The alert flows are responsible for generating alerts that provide the administrator with important information about the state of the device. The alerts created refer to temperature, battery percentage, location and loss of communication between device and gateway, in accordance with requirement FR17. All alerts are stored in the database to be displayed in Grafana application, and are also sent directly to the administrator via Telegram bot, thus guaranteeing requirement NFR13. The four flows responsible for generating and saving alerts can be seen in Figure 4.21.

The first flow analyzes the device's temperature and battery percentage for data that has been captured at that instant, i.e. that does not have a timestamp. The nominal temperature range was set between -10ºC and 35ºC, otherwise the flow generates an alert. As for the battery percentage, the system generates alerts with values below 25% and 15%.

The alerts are stored and the administrator is notified on his Telegram bot.

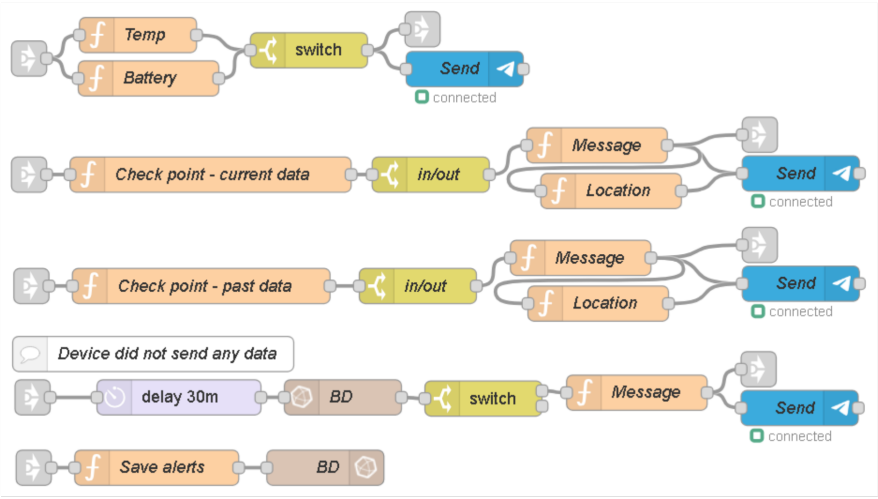

Figure 4.21: Flows responsible for ensuring that alerts are created, stored and sent.

The second and third flows are responsible for generating alerts relating to the boundaries of the grazing zone, in which an alert is generated as soon as the point collected is outside the zone defined by the KML file added by the administrator in the initial flows. For this to happen, a verification algorithm is run as soon as the latitude and longitude data arrive at Node-RED application. Both *memory data* and *current data* are analyzed by the second and third flows, respectively.

The fourth flow is responsible for generating alerts if no data from the device is published by the gateway. This notifies the administrator of a possible loss of range of the LoRa signal. This alert is executed if no data from the device is published for thirty minutes.

The fifth flow is responsible for saving the alerts in the database, which is essential for the administrator's control. By doing this, Grafana application will collect these notifications in the database and display them on one of its dashboards.

4.3.2 Database

InfluxDB database works with buckets, which are logical containers for storing temporal data. These are essential for organizing data. In this project there are two buckets, one for *data* and another for *alerts*. The *data* bucket stores all the data published by the gateway, from that coming from the device to that generated by it. This data is stored according to its type. Each type includes *measurement*, *field*, *location* and *sensorID*. This information allows the system to be scalable, enabling the collection of different data and ensuring its organization according to origin and type. All this information is assigned to the data by the data processing flows.

The *alerts* bucket stores all the alerts generated by Node-RED application alert flows. This allows them to be logged and Grafana application to access them to present them to the administrator. These are also stored according to their type, showing *measurement*, *field*, *location* and *sensorID*.

4.4 Visualization and Alerts

The last layer of the IoT structure is the user interface layer. This layer is made up of Grafana application, to present the data to the administrator, and Telegram bot, to send alerts and allow quick requests. Grafana application displays graphs, alerts, status, etc, according to the period defined by the user, while Telegram bot only displays the alerts generated by Node-RED application and returns, if the administrator requests it, the latest data collected by the system.

4.4.1 Panels at Grafana

Grafana application presents all the data collected by the system in heat maps, time series graphs and texts, which guarantees requirements FR18 and FR19. It searches InfluxDB database com and uses the Flux language to do so. With this tool, a dashboard was built with all the graphs, which are separated into panels. Each panel shows one or more pieces

of information, depending on the data assigned to it. Fifteen panels were built. Figure 4.22 shows the dashboard with all the panels.

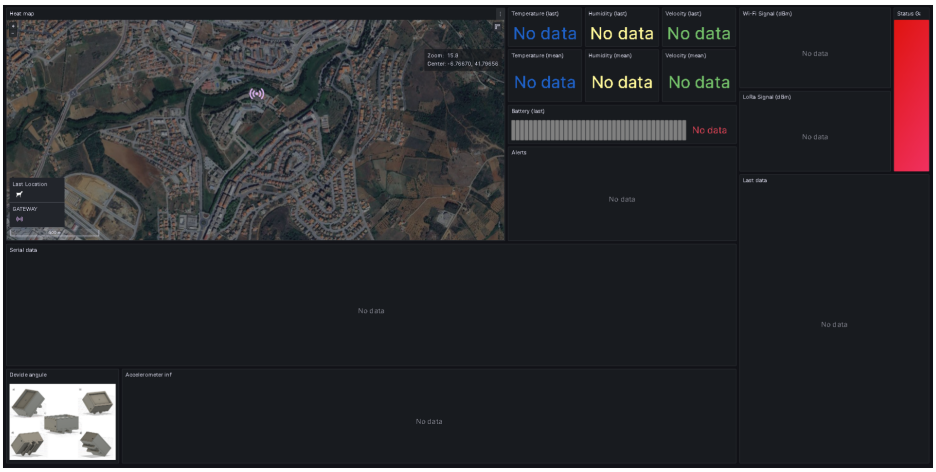

Figure 4.22: Grafana application dashboard with all panels.

The panels generally show *heat map*; *temperature, Humidity and velocity*; *battery status*; *animal neck position* and *alerts*. The period for which the data will be plotted can be changed using one of the tool's top buttons. The panels shown in the image 4.22 do not show any data; these will only be plotted in the Chapter 5.

Heat Map

One of the data points collected by the device is the animal's geolocation, which is represented in longitude and latitude. The *Heat map* panel displays a heat map of the points collected, i.e. according to the density of points and the map settings "radius", "blur" and "opacity" a heat map is generated. This map generates heat regions according to the density of points present, ranging from green (lowest point density) to red (highest point density). Zooming in on the sites also influences the distribution of this heat map, i.e. the less zoomed in, the greater the clustering of points, the opposite is also true.

In addition to the heat map, this panel also shows a route between the points collected, the location of the gateway and the location of the last point collected. There is also a tool for measuring length and area, which is useful for making quick checks on the map.

The base map is provided by Google and includes satellite images, faithfully reflecting the real terrain. The heat map can be seen in Figure 4.22, and is the only panel with a map.

Temperature, Humidity, Speed and Battery Status

Temperature, *humidity* and *velocity* can be viewed in three different ways, namely: their last value, average according to the selected period and in the form of a serial data graph. The *battery level* can be viewed in two ways, namely as a percentage bar and as serial data (the bar shows the last percentage value collected according to the interval selected by the administrator). There are seven panels that provide all these visualizations, six of which show only one piece of data and another which shows all four in the form of a serial data graph.

By mixing the possible ways of visualizing this data, the administrator is provided with a greater understanding of it, which would not be possible to analyze with just one type of visualization. As for the serial data graph, it is possible to quickly select a specific period, which facilitates the process of searching for data and makes it possible to check for possible patterns.

Alerts

The *alerts* panel shows the alerts generated by Node-RED application according to the period selected. This can be seen in the Figure 4.22. All alerts are presented in text format, including the time they were generated and a message stating their type. With this, the administrator can cross-reference the data presented in the other panels and take action on the potential problem.

Arrangement of the Device on the Animal

In the data processing flows there is the interpretation of the data coming from the inertial sensor, which generates five positions. This was presented in session 4.3.1. The values

generated by this processing are shown in the *Accelerometer inf* panel, ranging from 0, 1, 2, 4, 6 and 8. To the left of this panel there is another panel with indications of each mapped position, this is the same image shown in the Figure 4.19.

Gateway Information

Among the panels there are those that are exclusively for displaying data generated by the gateway, such as signal quality LoRa, signal quality Wi-Fi, its status and the type of data it is receiving. There are four panels, two of which show the signal Received Signal Strength Indicator (RSSI) of the communication technologies in the form of a serial data graph, another shows the status in the form of an indication bar, and finally a panel to show the type of data and the time the gateway is receiving it, varying between *current data*, *memory data* and *device searching gateway*. Figure 4.23 shows the layout of these panels on the dashboard.

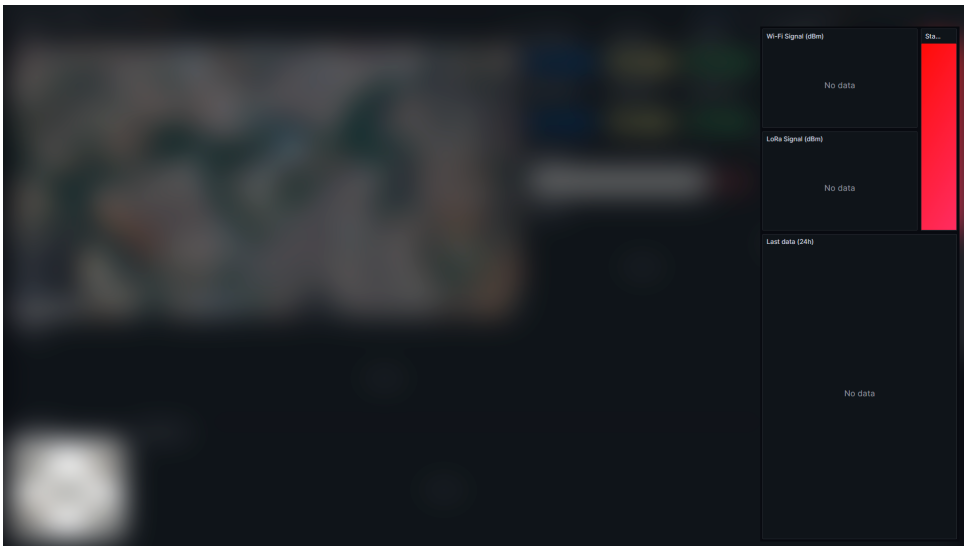

Figure 4.23: Panels to gateway informations. In the blurred background, the complete dashboard was placed, providing a better understanding of the position of the panels.

4.4.2 Telegram Bot

To communicate with Node-RED application, a bot was created with the BotFather tool within Telegram, which is a bot for creating other bots. When a bot is created, it is given a name and a token, which are instantiated in the nodes responsible for sending messages to the bot on Node-RED application. To interact with the bot, all you have to do is press some of the buttons it displays. This bot can be accessed privately or by adding it to a group, making it accessible to more people, but in this work it was only used for private chat. Figure 4.24 shows the bot's interface and the date and buttons it returns, the date following the Coordinated Universal Time (UTC) standard.

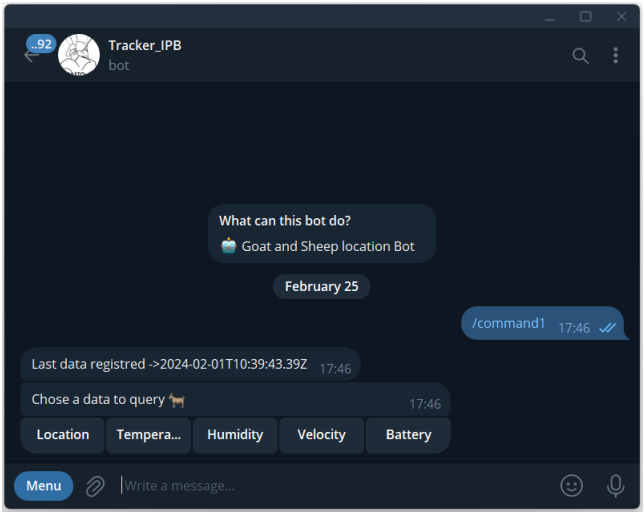

Figure 4.24: Telegram bot with time and buttons returned by Node-RED application.

Chapter 5

Tests and Results

This chapter will present the tests carried out and their results. To do this, tests were carried out at the system's application site and to guarantee the NFR04 requirement, i.e. in the IPB in its grazing zones, shown in Figure 3.2.

The tests and results are divided into *connectivity tests*, *data visualization and analysis*, *animal behavior to the device* and *device battery*. The connectivity tests evaluate both the connection between device-gateway and gateway-broker MQTT, which is essential to ensure the reliability of the system and monitor the functioning of the first and second layers IoT. The visualization tests check the presentation of data to the administrator, which is important to ensure that the interface layer is fulfilling its purpose.

Even though it was not a major focus of the work, animal behavior when using the device was evaluated, which is one of the points that ensures efficiency in terms of the ergonomics of the collar that supports it. Tests were also carried out on the system's autonomy, evaluating its behavior with different configurations.

The tests were carried out in a practical way, i.e. the collar with the device was put on random adult goats that were already used to collars and they grazed normally. This was repeated twice for each zone on different grazing days, i.e. there were four grazing days and it was possible to monitor the entire functioning of the system.

5.1 Connectivity Tests

In order to standardize the tests carried out in a practical way, all the variables for time and number of attempts were defined in advance, which can be seen in Table 5.1. These variables define how the software will behave. The application of each variable can be seen in the flowchart of the complete software at B.4 in appendix.

Type	**Variable**	**Value**	**Un**
System	Deep sleep	10	s
GNSS	Waiting time GNSS	300	s
GNSS	Module reading range	1	s
GNSS	Evaluation time	3	s
GNSS	First data filter	15	-
GNSS	Common data filter	3	-
LoRa	Resend data	10	s
LoRa	Number of attempts	2	-
LoRa	LoRa signal RSSI to send memory data	-80	dbm

Table 5.1: Device operation variables.

With these settings, the device was able to send data with a 30 second interval when confirmation was received, i.e. by removing the 10 seconds consumed by the *deep sleep* mode, the device took an average of 20 seconds to acquire the data, send it, receive confirmation that it had been sent and go to sleep again. However, in conditions where it did not receive confirmation from the gateway, the average time between acquisitions rose to one minute. The choice of 10s of *deep sleep* mode was made to force the device to collect a large volume of data, this is still within the range limited by FR02.

The connection between device and gateway can be checked according to the amount of data received by the gateway during grazing, the RSSI of the LoRa signal published by the gateway provides this information. Table B.1, in appendix, shows the results collected during the four days of grazing.

The first test in zone one had the gateway switched on only after 90 minutes of grazing, so only 15 data points were collected in real time and 114 from memory. This was due to an unexpected break in the gateway's microcontroller connector. The microcontroller was

replaced within 90 minutes and the gateway functioned normally again. The second test in the same zone showed 64 pieces of data collected in real time and 129 in memory; in this test, the gateway was active throughout the grazing. Both tests lasted approximately three hours (nominal grazing time in IPB). The density of data collected per hour in both tests in this zone was 40 and 64, respectively.

For the third test, now in zone two, 215 data were collected in real time and 59 from memory in approximately three hours of grazing. For the fourth test, still in zone two, 861 data were collected in real time and only 15 from memory in seven hours of grazing. Tests three and four had a data density per hour of 123 and 125 respectively.

As there were no power sockets in the stable, the gateway was tested with a powerbank and this was positioned just above the animal barn, as shown in the Figure B.6, in appendix. This location ensured a stable Wi-Fi connection, where its RSSI varied between -75dBm and 85dBm.

It was noted that the connectivity of the device and gateway was maintained throughout the device's stay in the stable. The gateway was programmed to send the status of its connection to Wi-Fi every 30 seconds. During all the tests, and disregarding the physical problem mentioned above, it did not lose its connection and the difference between sending data remained constant, which minimizes the problem with the MQTT protocol.

To analyze LoRa range, a test was carried out outside the IPB areas. The gateway was positioned normally as shown in the Figure B.6, in appendix, on a site on the outskirts of Bragança. The furthest point from which a connection was obtained was approximately 6km. This result was obtained in areas with low physical interference between device and gateway.

5.2 Data Visualization and Analysis

A variety of data was generated during the four tests, from the data captured by the sensors to the alerts generated by the nodes in Node-RED application. All of this was presented in Grafana application, as expected. Figures 5.1 and 5.2 show the four heatmaps

generated during the tests. For this purpose “radius”, “blur”, “opacity” and “zoom” were set in Grafana application to 7, 21, 1 and 17.5 respectively.

Figure 5.1: Test one and two - zone one heatmap.

In these heat maps, you can check the density of points in a colored way, i.e. points that are close together are grouped together and a reddish region is generated. Each setting of “radius”, “blur”, “opacity” and “zoom” defines how this generated heatmap behaves.

Figure 5.1 show the heat map generated in each grazing area in zone one. It can be seen that there was a difference in the grazing regions, while in the first test the animals stayed further west, in the second they stayed to the east.

Figure 5.2 show the heat maps obtained for zone two. As this zone does not have such a diverse area, the two tests carried out were equivalent, with no significant differences between the regions grazed by the animals. However, in test four the animals grazed right up to the edge of the zone, which is due to the length of time these animals have been grazing.

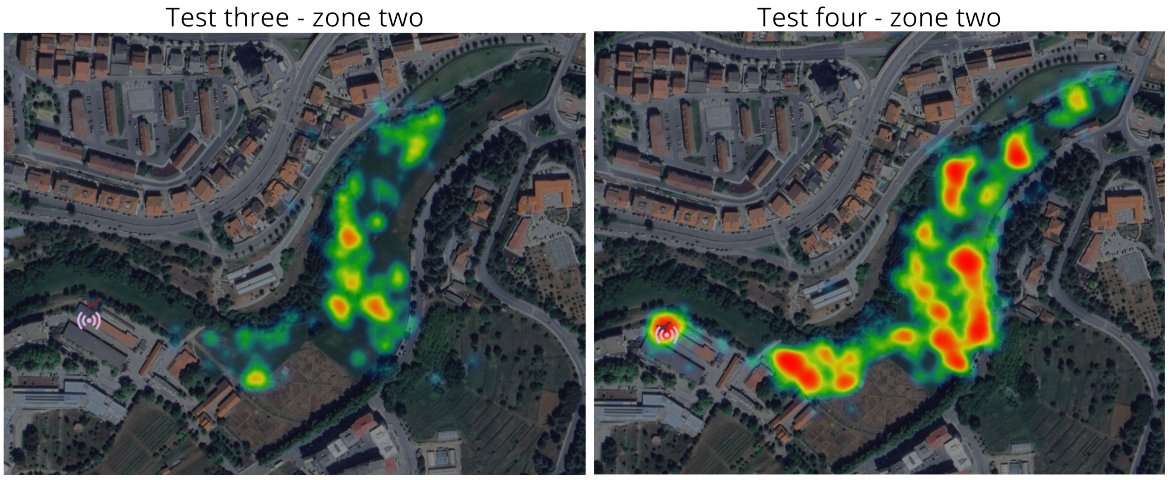

Figure 5.2: Test three and four - zone two heatmap.

The rest of the data was presented as expected by Grafana application. Figure 5.3 shows an overview of the panels with data from the second test carried out in zone two. This shows the heat map, temperature, humidity, battery percentage, speed, device position and data generated by the gateway on the various panels, all as announced in the 4 session.

Figure 5.3: Overview of Grafana application presented data collected from the four test - two zone.

During the tests, several alerts were recorded, among which the most repeated were

that of a point registered outside the defined area and loss of connection between the device and the gateway. As the defined area is concurrent with the edge of the zones and the herd grazed close to these boundaries, some alerts were recorded for *exit* from the predefined zone. In addition, connection loss alerts were repeated when the device and gateway no longer exchanged data. Even so, Grafana application and Telegram bot presented these alerts, the Figure B.7, in appendix, shows two alerts and their presentation by the platforms, referring to test four.

Telegram bot remained active during the tests. Figure B.8, in appendix, shows the data it returned with requests made after test four. As expected, Telegram bot returned the latest data collected on the date and time of the request.

5.3 Animal Behavior on the Device

During the grazing sessions, it was possible to follow the animal that was wearing the collar with the device and check if it showed any signs of discomfort or difficulty in grazing, guaranteeing the NFR14. Figure 5.4 shows some scenes of the animal grazing with the collar on.

Figure 5.4: Collar on the animal.

The aim of this work is not to delve into ergonomics, but the animals did not have any difficulty grazing even with the collar on. During grazing, the animal that was wearing the collar fed itself, interacted with other animals, ran, climbed trees, among other activities, all without any discomfort.

5.4 Device Battery

During the grazing tests, the power supply system showed the data shown in Figure 5.5. All four tests were carried out on sunny days, which contributed to the panel supplying power to the batteries. The serial graph of the battery percentage showed considerable variation during the tests, generated from the influence of the solar panel and the settings shown in Table 5.1.

Figure 5.5: Battery serial data during grazing tests.

To evaluate the graphs shown in Figure 5.5, an average of the first 20 points and the last 20 was calculated in order to check the consumption of the device and/or the contribution of the solar panel. The table B.2, in appendix, shows the results obtained.

Only in the first test did the device end up with a higher percentage than it started with, due to the contribution of the solar panel. The results of the other tests were different, all ending with a reduction of 0.69% per hour.

Another approach to testing the device's consumption was to analyze how much each process consumes, and thus project a relationship between the amount of data and the device's consumption based on the defined *deep sleep* time. To do this, the operation was summarized in five main processes: *wake up*, *acquire data*, *send data*, *wait for confirmation* and *deep sleep*. Each process has an average consumption and duration, which can be seen in the table 5.2. The completion of these processes is characterized as a data collection, i.e. a complete data acquisition that generated and forwarded a string to the gateway.

Process	**Consume (mA)**	**Time (s)**
Wake up	65	3
Acquire data	45	18
Send data	150	0,5
Wait for confirmation	46	4
Deep sleep	8	-

Table 5.2: Consumption of each device process. The time of the deep sleep process has been omitted because its variation will be explored.

The times defined in 5.2 are averages obtained by analyzing the consumption of the device. With this data and knowing the capacity of the 10050 mAh batteries, it is possible to create a relationship between the amount of data and the days of battery life of the device based on the selected *deep sleep*. To generate this relationship, the battery discharge is assumed to be linear and various processes are disregarded, such as the GNSS timeout, data resending LoRa, among others. Finally, Figure 5.6 shows the relationship between battery life in days and the amount of data collected based on the programmed *deep sleep*.

The Figure 5.6 shows that there is a direct inverse correlation between the amount of data collected and battery life, depending on the *deep sleep* selected. With a *deep sleep* of 30s it is possible to acquire approximately 30,000 strings of data in approximately 12 days. With a *deep sleep* of 300s, the number of data strings is approximately 12000, but

the system can operate for up to 37 days.

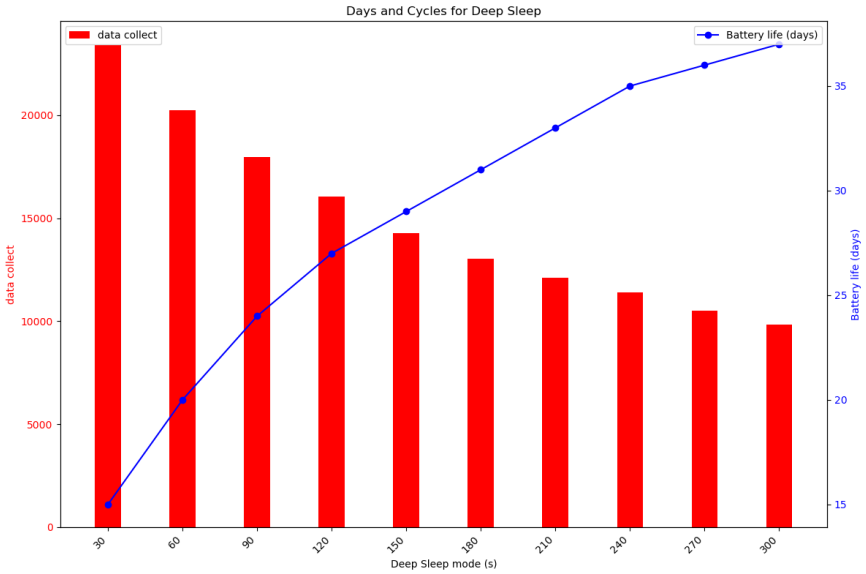

Figure 5.6: Relationship between days duration and device cycles based on programmed *deep sleep*.

Chapter 6

Conclusion and Future Work

This work dealt with the development of a data acquisition system using IoT to monitor sheep and goats. This was based on the importance and benefits that this type of technology can provide for farmers who aim to better monitor and increase the productivity of their flocks, especially in the Northeast of Portugal.

The parts of the system were divided into layers of an IoT architecture and functional and non-functional requirements delimited what, how and where the system should work. These were based on the geral requeriments and characteristics of extensive grazing. As a result, technologies such as LoRa, GNSS, and low-cost modules and sensors were widely discussed in the work.

The tests carried out showed the behavior of the entire system and especially its limitations. The connectivity tests showed that the system guaranteed data integrity, even in areas where the LoRa connection between gateway and device failed. This was due to the device's internal storage system. A distance test was also carried out and the range was 6km, meaning that the LoRa technology is more than efficient for extensive grazing systems where the areas traveled by the animals are limited to approximately 5km from the gateway and device. In addition, the gateway did not show any oscillation during its operation, maintaining an adequate signal for data to be sent.

Data visualization and analysis were also tested. The heat maps presented by Grafana application added value to the geographical points collected and intuitively showed the

areas preferred by the animals in each test carried out. The dashboards built in the tool showed the other data in an appropriate way, allowing the administrator to have control over them.

Telegram bot delivered alerts when necessary, notifying the administrator in real time. As such, the Grafana application and Telegram interface and alert platforms provided the data, status and alerts as expected, i.e. they proved to be suitable for the system.

The system's power supply proved promising in terms of battery life due to its storage capacity and the contribution of the solar panel. The bench tests showed that, with *deep sleep* in 300s, the device can reach 37 days of uninterrupted operation. In the grazing tests, the solar panel had an influence on battery discharge, providing an average of 0.69% per hour. However, this value was influenced by the third test, which had a higher consumption than the others. On the other hand, the first test showed that at the end of grazing the system ended up with 0.35% more battery, reaffirming the positive influence of the solar panel.

The animal's behavior with the collar supporting the device proved to be adequate, i.e. no discomfort was shown. The monitored animal was not restricted in any of its daily activities, such as feeding, running, drinking water, etc. In addition, the way in which the device was divided into top and bottom ensured that the collar did not rotate, allowing the modules to remain above the animal's neck, maximizing the efficiency of the solar panel and data acquisition and sending modules.

For future work, various hardware and software adjustments to the device and gateway could be improved. The size of the device, especially the upper part, can be reduced by better defining the modules and electronic components. Some modules can be replaced by the sensor or printed circuit and attached directly to the PCB along with its electronics, this applies to the temperature and humidity module and the inertial sensor module. It is also advisable to change the microcontroller to an equivalent one that takes up less space and provides the same technologies. A proposal for adapting the design of the device's top case, which is shaped like a "V", would consist of including an additional solar panel while reducing one of the batteries. This measure aims to improve the adaptability of the

collar to the goat's neck.

The device's software can be updated, especially in terms of the way it sends and/or stores data strings. As the size of the string directly influences the LoRa signal, a string format that takes up fewer bytes will ensure leaner and more efficient communication between device and gateway. The gateway must also be adjusted to accommodate this change. The creation of device operating modes could also be added. As the gateway sends confirmation data to the device, it would also be interesting to add changes to its processes along with the confirmation message, which would provide greater control over the device. In addition, security mechanisms such as data encryption can be added to the system, increasing data security and reliability.

As the system presents a lot of data, this could generate more alerts and analysis in Node-RED application, generating more useful information for the administrator. Currently, data analysis is limited and conditional, so it doesn't predict any anomalous animal behavior, but by analyzing a larger volume of data, this could be implemented.

To try to predict the battery life, field and bench tests were carried out, but this data is not enough to determine the actual life of the device; more tests in this direction need to be carried out to have a more reliable prediction. The test carried out to obtain the distance of the LoRa module to the gateway was limited to the geography of Bragança-PT, i.e. to obtain the maximum range of the device to the gateway, tests in wider environments should be carried out.

The general objectives of the work were achieved and its development met all the requirements. The development of this low-cost device is an example of how PLF and IoT work in agriculture. All the data captured generates value for pastoralism and its contexts.

Bibliography

[1] J. Castro, L. Macedo Godoy, J. Castro, and M. Castro, "Sheep Grazing Patterns for Better Land Management: Adjusting GPS Tracking Protocol," Jan. 2022.

[2] M. Castro, "Sistemas de produção animal em regiões de montanha em portugal.," in May 2016.

[3] C. Tzanidakis, O. Tzamaloukas, P. Simitzis, and P. Panagakis, "Precision livestock farming applications (plf) for grazing animals," *Agriculture*, vol. 13, no. 2, 2023, ISSN: 2077-0472. DOI: 10.3390/agriculture13020288.

[4] B. Evstatiev, S. Kadirova, and N. Valov, "Analysis of the Wireless Communication Technologies Used in Livestock Monitoring," in *2022 International Conference on Communications, Information, Electronic and Energy Systems (CIEES)*, 2022, pp. 1–5. DOI: 10.1109/CIEES55704.2022.9990810.

[5] A. Sadeghi-Niaraki, "Internet of thing (IoT) review of review: Bibliometric overview since its foundation," *Future Generation Computer Systems*, vol. 143, pp. 361–377, 2023, ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2023.01.016.

[6] M. Castro, J. P. Castro, and J. Castro, "Understory clearing in open grazed mediterranean oak forests: Assessing the impact on vegetation," *Sustainability*, vol. 14, no. 17, 2022, ISSN: 2071-1050. DOI: 10.3390/su141710979.

[7] Domodis, *Collares GPS*, https://www.domodis.com, 2023.

[8] Pastoral, *Healthier livestock reduced carbon emissions profitable outcomes*, https://www.pastoral.ai, 2023.

[9] I. Köhler-Rollefson, *Hoofprints on the Land*, 9st. Somerset House, London, UK: Chelsea Green Publishing UK, Jan. 2023, ISBN: 9781645021520.

[10] C. Aquilani, A. Confessore, R. Bozzi, F. Sirtori, and C. Pugliese, "Review: Precision livestock farming technologies in pasture-based livestock systems," *Animal*, vol. 16, no. 1, p. 100 429, 2022, ISSN: 1751-7311. DOI: https://doi.org/10.1016/j.animal.2021.100429.

[11] D. Hamidi, M. Komainda, B. Tonn, J. Harbers, N. A. Grinnell, and J. Isselstein, "The effect of grazing intensity and sward heterogeneity on the movement behavior of suckler cows on semi-natural grassland," *Frontiers in Veterinary Science*, vol. 8, 2021, ISSN: 2297-1769. DOI: 10.3389/fvets.2021.639096. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fvets.2021.639096.

[12] I. Gordon, "Animal-based techniques for grazing ecology research," *Small Ruminant Research*, vol. 16, no. 3, pp. 203–214, 1995, ISSN: 0921-4488. DOI: https://doi.org/10.1016/0921-4488(95)00635-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/092144889500635X.

[13] Z. Borthwick, K. Quiring, S. C. Griffith, and S. T. Leu, "Heat stress conditions affect the social network structure of free-ranging sheep," *Ecology and Evolution*, vol. 14, no. 2, e10996, 2024, e10996 ECE-2023-06-00963.R2. DOI: https://doi.org/10.1002/ece3.10996.

[14] A. Lopez, E. Mainardi, E. Beretta, *et al.*, "Characterisation of dry-salted violino and bresaola from grass-fed bergamasca sheep," *Animals*, vol. 14, no. 3, 2024, ISSN: 2076-2615. DOI: 10.3390/ani14030488.

[15] T. Paz-Kagan, V. Alexandroff, and E. D. Ungar, "Detection of goat herding impact on vegetation cover change using multi-season, multi-herd tracking and satellite imagery," *Science of The Total Environment*, vol. 895, p. 164 830, 2023, ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2023.164830.

[16] J. Plaza, J. A. Abecia, N. Sánchez, *et al.*, "The Conquense transhumance route in Spain described by 3D geographical information systems, GPS and remote sensing data," *Small Ruminant Research*, vol. 221, p. 106 953, 2023, ISSN: 0921-4488. DOI: https://doi.org/10.1016/j.smallrumres.2023.106953.

[17] M. Castillo-Garcia, C. L. Alados, J. Ramos, D. Moret, O. Barrantes, and Y. Pueyo, "Understanding herbivore-plant-soil feedbacks to improve grazing management on mediterranean mountain grasslands," *Agriculture, Ecosystems & Environment*, vol. 327, p. 107 833, 2022, ISSN: 0167-8809. DOI: https://doi.org/10.1016/j.agee.2021.107833.

[18] M. Wild, M. Gauly, T. Zanon, J. Isselstein, and M. Komainda, "Tracking free-ranging sheep to evaluate interrelations between selective grazing, movement patterns and the botanical composition of alpine summer pastures in northern italy," *PASTORALISM-RESEARCH POLICY AND PRACTICE*, vol. 13, no. 1, Oct. 2023, ISSN: 2041-7136. DOI: 10.1186/s13570-023-00287-3.

[19] S. Evaristo Jorge Oliveira de, L. M. D. Queiroz, E. I. M. de Lima, *et al.*, "Can GPS monitoring help farmers select the best nutritional management strategy for finishing sheep on pasture?" *Livestock Science*, vol. 272, p. 105 229, 2023, ISSN: 1871-1413. DOI: https://doi.org/10.1016/j.livsci.2023.105229.

[20] L. L. Trieu, D. W. Bailey, H. Cao, *et al.*, "Potential of Accelerometers and GPS Tracking to Remotely Detect Perennial Ryegrass Staggers in Sheep," *Smart Agricultural Technology*, vol. 2, p. 100 040, 2022, ISSN: 2772-3755. DOI: https://doi.org/10.1016/j.atech.2022.100040.

[21] J. Zhao, J. Cao, Z. Che, Y. Guo, C. Ma, and Q. Zhang, "Contribution of sheep grazing to plant diversity in natural grasslands," *Diversity*, vol. 14, no. 6, 2022, ISSN: 1424-2818. DOI: 10.3390/d14060446.

[22] Y. Chebli, S. El Otmani, J.-L. Hornick, J. Bindelle, J.-F. Cabaraux, and M. Chentouf, "Estimation of grazing activity of dairy goats using accelerometers and global

positioning system," *Sensors*, vol. 22, no. 15, 2022, ISSN: 1424-8220. DOI: 10.3390/s22155629.

[23] F. Gao, T. Liu, H. Wang, *et al.*, "Spatial and temporal variation patterns of summer grazing trajectories of sunit sheep," *Ecological Informatics*, vol. 78, p. 102 322, 2023, ISSN: 1574-9541. DOI: https://doi.org/10.1016/j.ecoinf.2023.102322.

[24] E. Sales-Baptista, M. I. Ferraz-de-Oliveira, M. Terra-Braga, J. A. L. de Castro, J. Serrano, and M. C. d'Abreu, "Characterization of grazing behaviour microstructure using point-of-view cameras," *PLOS ONE*, vol. 17, no. 3, pp. 1–28, Mar. 2022. DOI: 10.1371/journal.pone.0265037.

[25] H. Keshavarzi, C. Lee, T. R. Dyall, M. Johnson, and D. L. M. Campbell, "Shared stressful experiences affect social proximity in merino sheep," *BIOLOGY LETTERS*, vol. 19, no. 2, Feb. 2023, ISSN: 1744-9561. DOI: 10.1098/rsbl.2022.0396.

[26] LikeM13, *LikeM13 Customize Your Smart IoT Network*, https://www.likem13.com, 2023.

[27] M. Zorawski, T. Brito, J. Castro, J. P. Castro, M. Castro, and J. Lima, "An IoT Approach for Animals Tracking," in *Optimization, Learning Algorithms and Applications*, Cham: Springer International Publishing, 2021, pp. 269–280, ISBN: 978-3-030-91885-9.

[28] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of things (IoT): A literature review," *Journal of Computer and Communications*, vol. 3, pp. 164–173, 2015. DOI: 10.4236/jcc.2015.35021.

[29] F. Richter, "Have we passed the peak of the smartphone era?" Tech. Rep., 2021.

[30] V. Bhagat, "What are pros & cons of internet of things (IoT)?" Tech. Rep., 2022.

[31] A. Shakdher, S. Agrawal, and B. Yang, "Security Vulnerabilities in Consumer IoT Applications," in *2019 IEEE 5th Intl Conference on Big Data Security on Cloud*

(BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), 2019, pp. 1–6. DOI: 10.1109/BigDataSecurity-HPSC-IDS.2019.00012.

[32] S. Kumar, Kanchan, A. Kumar, and P. Aggarwal, "INTERNET OF THINGS (IOT) APPLICATIONS AND CHALLENGES: A REVIEW," vol. 11, pp. 359–367, Oct. 2023. DOI: 10.1007/s11277-020-07446-4.

[33] I. Analytics, "Top 10 IoT applications in 2020," Tech. Rep., 2020.

[34] M. S. Farooq, O. O. Sohail, A. Abid, and S. Rasheed, "A Survey on the Role of IoT in Agriculture for the Implementation of Smart Livestock Environment," *IEEE Access*, vol. 10, pp. 9483–9505, 2022. DOI: 10.1109/ACCESS.2022.3142848.

[35] M. Quiñones-Cuenca, J. Maldonado, J. Martínez-Curipoma, *et al.*, "Real Time Geolocation System for Livestock based in LoRa," in *2022 17th Iberian Conference on Information Systems and Technologies (CISTI)*, 2022, pp. 1–6. DOI: 10.23919/CISTI54924.2022.9820172.

[36] J. Makario and C. w. Maina, "A Bluetooth Low Energy (BLE) Based System for Livestock Tracking and Localization," in *2021 IST-Africa Conference (IST-Africa)*, 2021, pp. 1–7.

[37] H. Sulistiani, I. Yasin, D. Darwis, S. Samsugi, and N. Reffiandi, "IoT for Monitoring System for Cattle Growth Productivity," in *2023 International Conference on Networking, Electrical Engineering, Computer Science, and Technology (IConNECT)*, 2023, pp. 180–183. DOI: 10.1109/IConNECT56593.2023.10326758.

[38] P. Gokhale, O. Bhat, and S. Bhat, "Introduction to IOT," vol. 5, pp. 41–44, Jan. 2018. DOI: 10.17148/IARJSET.2018.517.

[39] Microsoft, *Tecnologias e protocolos de IoT*, https://azure.microsoft.com/pt-pt/solutions/iot/iot-technology-protocols, 2023.

[40] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, Apr. 2015. DOI: 10.1007/s10796-014-9492-7.

[41] R. Casas, A. Hermosa, Á. Marco, T. Blanco, and F. J. Zarazaga-Soria, "Real-time extensive livestock monitoring using lpwan smart wearable and infrastructure," *Applied Sciences*, vol. 11, no. 3, 2021, ISSN: 2076-3417. DOI: 10.3390/app11031240.

[42] J. G. Panicker, M. Azman, and R. Kashyap, "A LoRa Wireless Mesh Network for Wide-Area Animal Tracking," in *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 2019, pp. 1–5. DOI: 10.1109/ICECCT.2019.8868958.

[43] B. R. dos Reis, Z. Easton, R. R. White, and D. Fuka, "A LoRa sensor network for monitoring pastured livestock location and activity1," *Translational Animal Science*, vol. 5, no. 2, txab010, Jan. 2021, ISSN: 2573-2102. DOI: 10.1093/tas/txab010.

[44] Semtech. "What Is LoRa®?" (2024), [Online]. Available: https://www.semtech.com/lora/what-is-lora.

[45] L. Germani, V. Mecarelli, G. Baruffa, L. Rugini, and F. Frescura, "An IoT Architecture for Continuous Livestock Monitoring Using LoRa LPWAN," *Electronics*, vol. 8, no. 12, 2019, ISSN: 2079-9292. DOI: 10.3390/electronics8121435.

[46] S. Ian, *Engenharia de Software*, 9st. RUa Nelson Francisco, 26, BRASIL: Pearson Education, 2011, ISBN: 978-85-7936-108-1.

[47] MQTT. "MQTT: The Standard for IoT Messaging." (2024), [Online]. Available: https://mqtt.org.

[48] Node-RED. "Node-RED Low-code programming for event-driven applications." (2024), [Online]. Available: https://nodered.org.

[49] InfluxDB. "InfluxDB. It's About Time." (2024), [Online]. Available: https://www.influxdata.com.

[50] G. Labs. "Grafana labs." (2024), [Online]. Available: https://grafana.com.

[51] Telegram. "Telegram - a new era of messaging." (2024), [Online]. Available: https://telegram.org.

[52] Espressif. "Esp32-wroom-32 e esp32-wroom-32ue datasheet." (2024), [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf.

[53] Sensirion. "Sht21." (2024), [Online]. Available: https://www.sensirion.com/products/catalog/SHT21/.

[54] S. Labs. "I2c humidity and temperature sensor." (2024), [Online]. Available: https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf.

[55] A. Devices. "Adxl345." (2024), [Online]. Available: https://www.analog.com/en/products/adxl345.html.

[56] keyestudio. "Ks0012 keyestudio adxl345 three axis acceleration module." (2024), [Online]. Available: https://wiki.keyestudio.com/Ks0012_keyestudio_ADXL345_Three_Axis_Acceleration_Module.

[57] Quectel. "Quectel L80 GNSS." (2024), [Online]. Available: https://www.quectel.com/ProductDownload/L80.html.

[58] RYLR998. "RYLR998 - 868/915MHz LoRa® Antenna Transceiver Module." (2024), [Online]. Available: https://reyax.com/products/RYLR998.

[59] ANACOM, *Quadro nacional de atribuição de frequências*, https://www.anacom.pt, 2010.

[60] Easyeda. "An easier and powerful online pcb design tool." (2024), [Online]. Available: https://easyeda.com.

[61] Arduino. "Arduino ide." (2024), [Online]. Available: https://docs.arduino.cc/software/ide/#ide-v2.

Appendix A

External Links

The codes developed in this work can be found at the following link.

Appendix B

Other Appendices

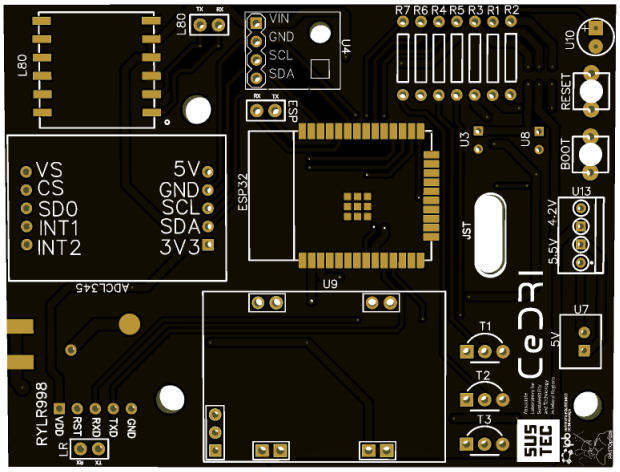

Figure B.1: PCB to device upper part electrical circuit. This image show the pins, figures, local of eletronic components and other PCB details.

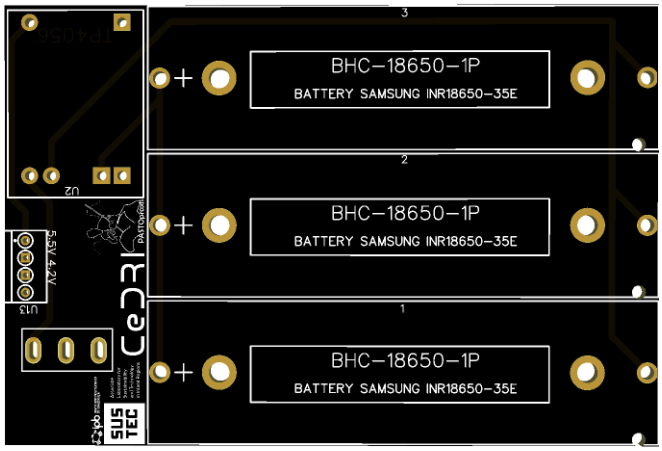

Figure B.2: PCB to bottom part electrical circuit.

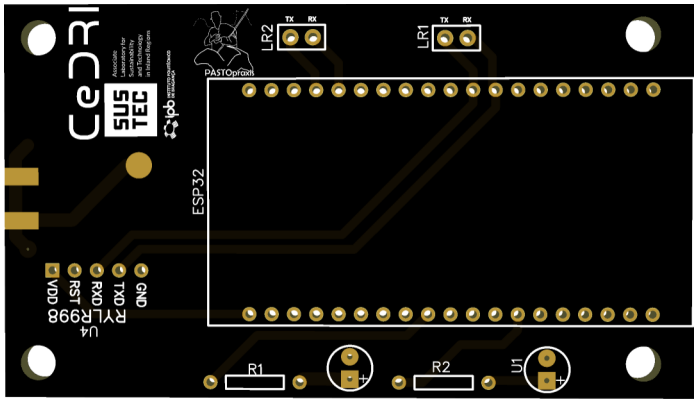

Figure B.3: Gateway PCB. In this image the pins and other details of the PCB in better detail.

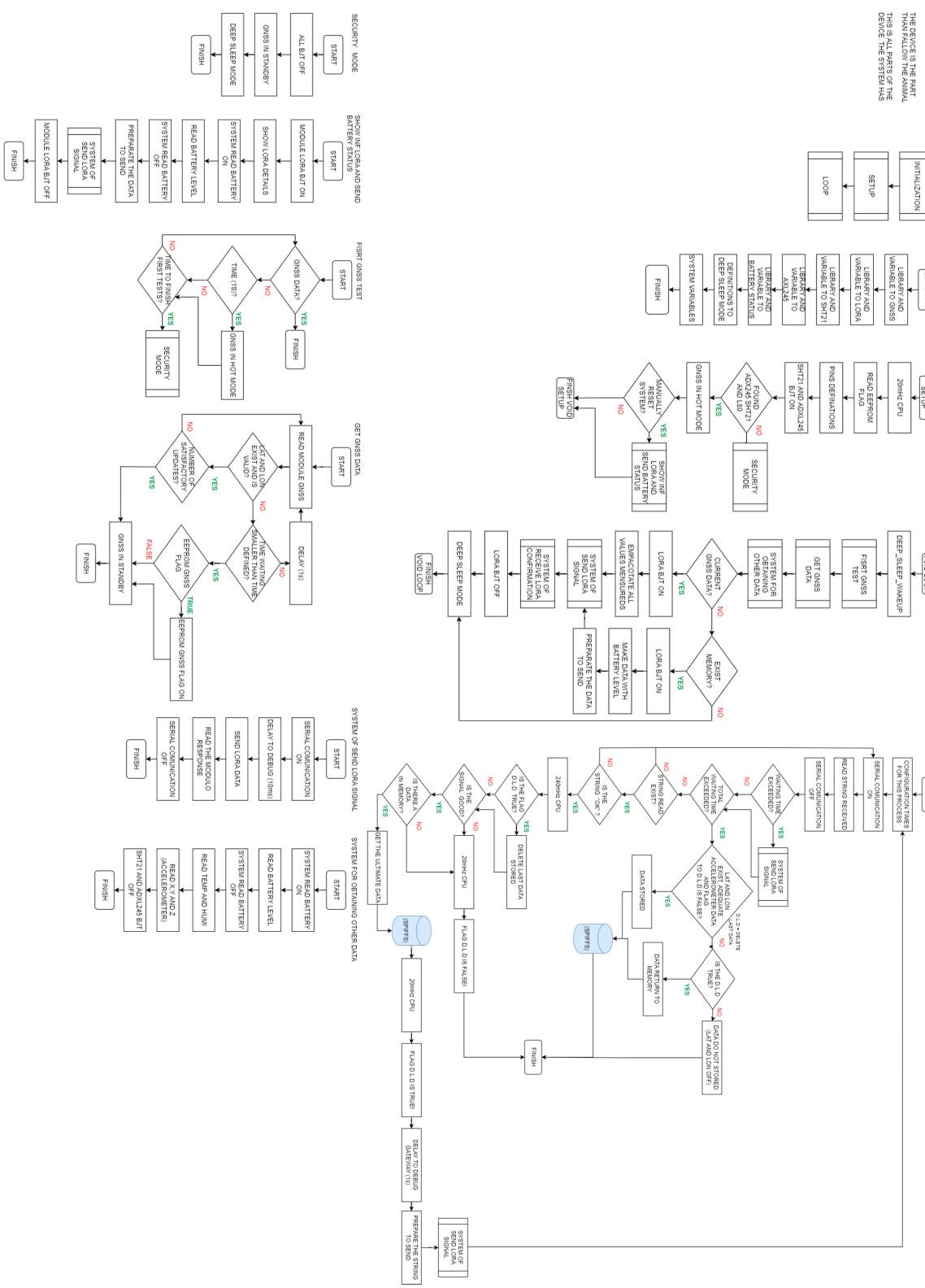

Figure B.4: Device software flowchart.

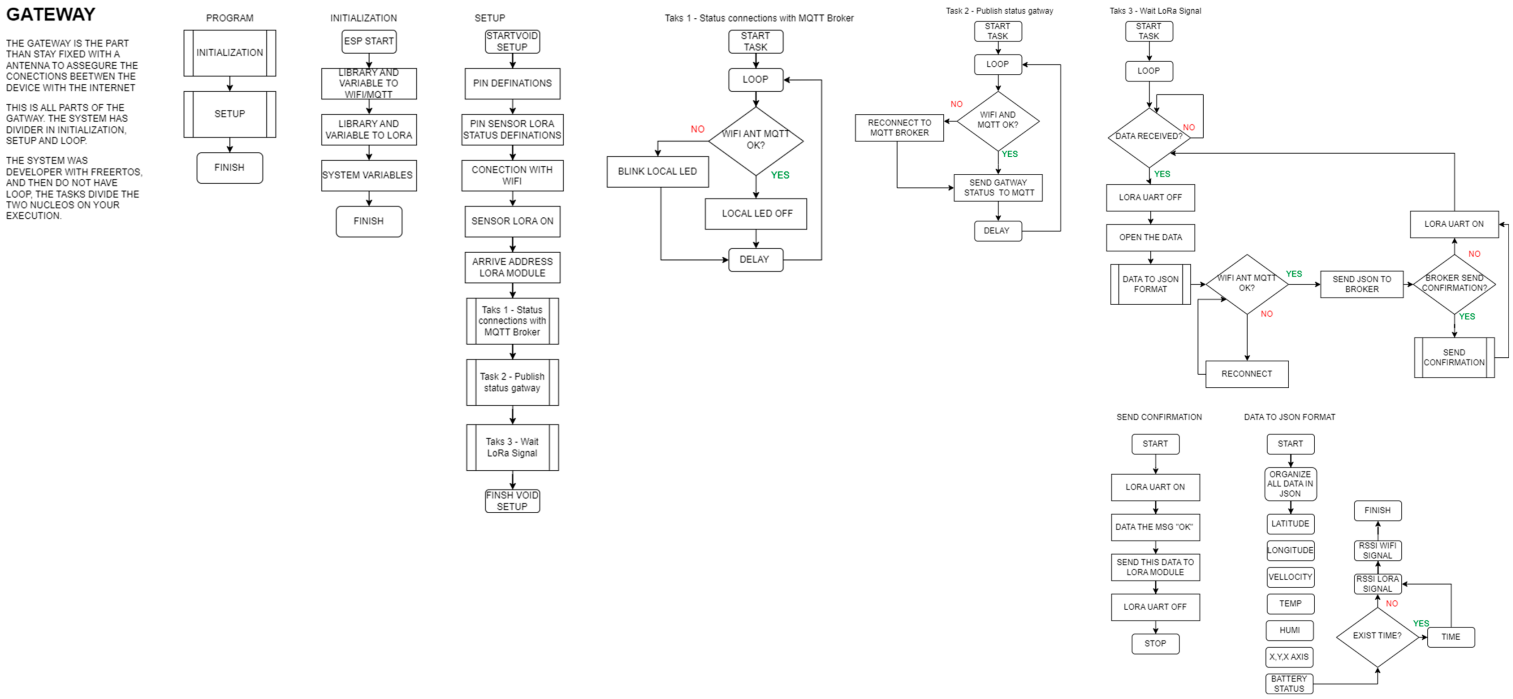

Figure B.5: Gateway software flowchart.

Test	**Zone**	**start (h)**	**finish (h)**	**collected data (qty)**	**real time data (qty)**	**memory data (qty)**	**data/hour (qty)**
1	1	13:17	16:28	129	15	114	40
2	1	13:00	16:01	193	64	129	64
3	2	13:03	15:16	274	215	59	123
4	2	09:04	16:03	876	861	15	125

Table B.1: Amount of data (strings) collected and its type (current or memory) in pastures carried out in the two zones.

Figure B.6: Telegram bot with time and buttons returned by Node-red application.

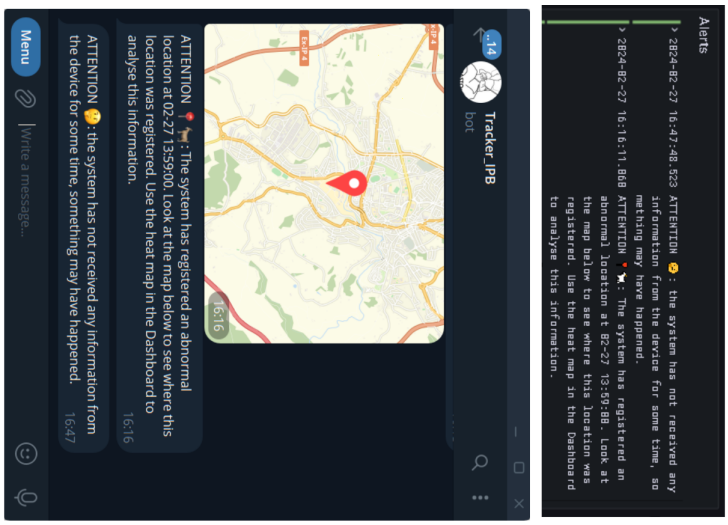

Figure B.7: Alerts presented by Grafana application and Telegram bot simultaneously (test four).

Test	**zone**	**time (h)**	**collected data (qty)**	**start (%)**	**finish (%)**	**variation/hour (%)**	**total variation (%)**
1	1	02:45	129	93.85	94.20	+0.12	+0.35
2	1	03:01	193	93.65	93.10	-0.18	-0.55
3	2	02:13	274	95.00	93.00	-1.81	-2.00
4	2	06:59	876	94.05	93.45	-0.08	-0.60

Table B.2: Results of variation in battery percentage throughout the grazing tests.

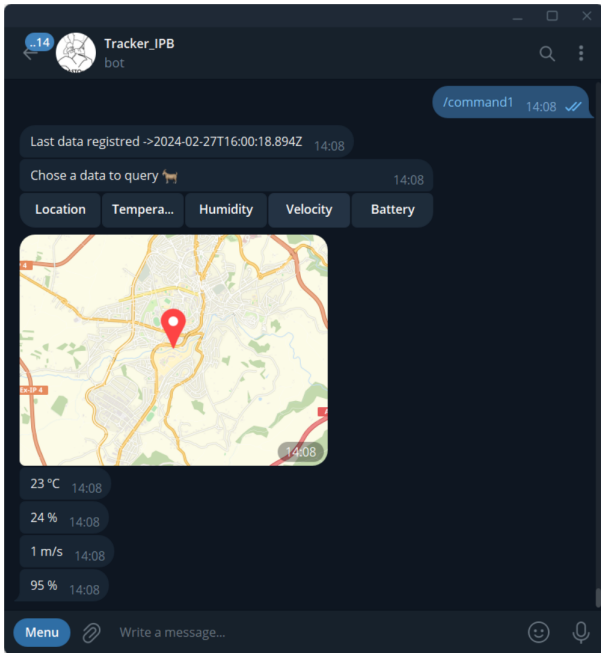

Figure B.8: Data returned by Telegram bot after a request from the administrator.