THE COKE CAN PROJECT: DESIGN AND IMPLEMENTATION OF A MULTI-PROTOCOL DECISION ENGINE FOR RELIABLE WSN ROUTING

JAMES WILLIS

being a Master's Engineering Project submitted in partial fulfilment
of the requirements for the *MEng* Honours Degree in Integrated Engineering
2024

New Model Institute for Technology and Engineering (NMITE)
United Kingdom

Student Declaration Form for Submission with Major Projects

Student's Name	James Willis
Student's Number	21221001
Degree Programme	Integrated Engineering, MEng
Supervisor	Tim Belden
Project Title	The Coke Can Project: Design and Implementation of a Multi-Protocol Decision Engine for Reliable Routing in WSNs
Word Count	[WORD COUNT HERE]
Confidential?	No

In submitting this Project report I acknowledge that I understand the definition of, and penalties for, cheating, collusion and plagiarism set out in the assessment regulations. I also confirm that this work has not previously been submitted for assessment for an academic award, unless otherwise indicated.

(Electronic) Signature of student:

Date: November 18, 2024

Executive Summary

This project focuses on developing an edge processing framework tailored for wildfire management and relief, addressing the crucial need for rapid decision-making and resilience in disaster response.

The initial objectives of the project are twofold: to design a framework capable of intelligently routing data based on its Quality of Service (QoS) requirements and network conditions, and to identify a suitable COTS device that could support this functionality. The successful completion of these initial goals led to the integration of both tasks, resulting in the physical implementation of the edge processing framework.

Once implemented, the framework was validated on the selected COTS device, confirming its viability for practical deployment in disaster response scenarios. Key achievements included the development of a decision engine that dynamically adjusts communication protocols, effectively addressing varying data requirements and network conditions. The system's ability to adapt to different scenarios provides a significant advantage for real-time decision-making and data transfer reliability, essential in disaster response situations.

Looking ahead, opportunities for further enhancement include conducting system-wide routing tests, optimising system performance and validating the system under harsh and high-interference environments. These next steps will further strengthen the system's robustness and ensure its optimal performance in real-world applications.

This project underscores the potential of COTS-based edge processing solutions to deliver scalable, cost-effective, and resilient systems, offering critical support for applications like wildfire management and relief operations.

Acknowledgements

Firstly, I would like to express my deepest gratitude to the Project Client for their unwavering commitment and support throughout the duration of this project. Their involvement, from attending weekly client meetings to offering invaluable technical expertise, has been instrumental in shaping the direction and success of this work. Their innovative thinking, coupled with their dedication to the project, allowed us to explore creative solutions and continuously refine the approach. It is their vision and collaboration that have truly made this project possible, and I am immensely grateful for their contribution.

I would also like to extend my heartfelt thanks to my supervisor, Associate Professor Tim Beldon, for his guidance, mentorship, and constant encouragement. His feedback and vast knowledge were essential in driving the project forward and helping me stay focused on the bigger picture.

A special thanks to my second reader, Associate Professor Peter Metcalfe, for his unique and insightful comments, which always brought a fresh perspective.

To my friends, thank you for your support and encouragement. Whether in moments of challenge or celebration, your presence has made all the difference, and I am truly thankful for the joy you've brought into my journey.

Finally, I am deeply grateful to my family for their continuous love, support, and belief in me. Their encouragement and interest in what I do has been a cornerstone of my academic journey, and I cannot thank them enough for always being there.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviated Phrases

ACK	Acknowledgement
ADC	Analog-to-Digital Converter
AoC	Attestation of Conformity
BLE	Bluetooth Low Energy
CA	Collision Avoidance
CE	Conformité Européene
CO2	Carbon Dioxide
COM Port	Communication Port
COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
CSMA	Carrier Sense Multiple Access
CTS	Clear To Send
DIFS	Distributed Coordination Function Interframe Space
DSSS	Direct Sequence Spread Spectrum
EIA	Environmental Impact Assessment
EIRP	Equivalent Isotropic Radiated Power
EM	Electromagnetic
EMC	Electromagnetic Compatibility
EoL	End of Life

FHSS Frequency Hopping Spread Spectrum

GDPR General Data Protection Regulations

GIS Geographical Information System

GNSS Global Navigation Satellite System

GPIO General Purpose Input/Output

GPS Global Positioning System

HadCM3 Hadley Centre Coupled Model Version 3

I2C Inter-Integrated Circuit

IDE Integrated Development Environment

IDNDR International Decade for Natural Disaster Reduction

IFS Interframe Space

IMU Inertial Measurement Unit

IoT Internet of Things

IP Internet Protocol

IR Interference Requirements

ISM Industrial, Scientific, and Medical

KDBI Keetch-Byram Drought Index

LoRa Long Range

LR-WPAN Low Rate Wireless Personal Area Network

MCU Microprocessor Unit

MEMS Micro-Electro-Mechanical System

MVP Minimum Viable Product

NAV Network Allocation Vector

NB Narrow-Band

NiMH Nickel Metal Hydride

NIS Network and Information System

OLED	Organic Light Emitting Diode
PMU	Power Management Unit
PSRAM	Psuedostatic Random Access Memory
QoS	Quality of Service
RED	Radio Equipment Directive
REPL	Read-Eval-Print Loop
RF	Radio Frequency
RoHS	Restriction of Hazardous Substances
RTC	Real-Time Clock
RTS	Request To Send
SoC	System-on-a-Chip
SPI	Serial Peripheral Interface
SPIN	Sensor Protocols for Information via Negotiation
SRAM	Static Random Access Memory
SRD	Short Range Device
SS	Spread Spectrum
SWaP	Size, Weight, and Power
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
UART	Universal Asynchronous Receiver/Transmitter
UKCA	UK Conformity Assessment
USB	Universal Serial Bus
UWB	Ultra Wide-Band
WEEE	Waste Electrical and Electronic Equipment
Wi-Fi	Wireless Fidelity
WSN	Wireless Sensor Network

Abbreviated Names

BACL Bay Area Conformity Laboratories Corp.

CCS Civil Contingencies Secretariat

DEFRA Department for Environment, Food and Rural Affairs

FAA Federal Aviation Administration

FEMA Federal Emergency Management Agency

IAGB Industrieanlagen-Betriebsgesellschaft mbH

IEEE Institute of Electrical and Electronics Engineers

IPCC International Panel on Climate Change

NEIC National Earthquake Information Centre

NFCC National Fire Chiefs Council

Ofcom The Office of Communications

USGS United States Geological Survey

1 Introduction

1.1 Background

1.1.1 Natural Disasters: Wildfires

Natural disasters have become increasingly frequent and severe, posing significant challenges to global resilience. Wildfires, in particular, are rising in both intensity and frequency, with climate change contributing to longer fire seasons and more extreme conditions. Once a less frequent phenomenon in regions such as the UK, wildfires have become a more pressing concern in recent years. This has been particularly evident in areas like Southern Europe, North America, and Australia, where climate models predict worsening fire behaviour due to increased temperatures, drought, and changing vegetation. The resulting environmental destruction and human costs highlight the urgent need for improved disaster management and response strategies.

1.1.2 WSNs in Disaster Management Scenarios

Historically, disaster management was largely reactive, with communities and organisations like the Red Cross providing essential relief without formalised systems or technological support. Over the decades, however, disaster management has evolved significantly, particularly with the advent of Industry 3.0 in the 1970s, which introduced technologies that revolutionised communication during crises. The establishment of disaster management organisations and the implementation of disaster risk reduction strategies in the 1990s marked a shift toward more formal and proactive approaches. With the rise of the 'internet of things' (IoT) and 'wireless sensor networks' (WSNs) in the 2000s, real-time data collection and monitoring became key components of modern disaster management, helping to enhance preparedness and response capabilities. Today, WSNs provide unprecedented precision in monitoring and data collection, revolutionising disaster management with faster, more informed decisions.

In particular, WSNs provide a significant advantage during disaster scenarios, such as wildfires, by offering real-time environmental monitoring through low-energy, autonomous sensors. These networks enable the early detection of disasters, enhance situational awareness, and facilitate improved communication between first responders and affected populations. However, despite the promise of such technologies, critical challenges remain in ensuring reliable communication in remote and disaster-prone areas. The most pressing concern in these situations is ensuring that the right information reaches the right people at the right time, which is often hindered by a lack of reliable communication infrastructure.

To address these challenges, the effective integration of communication technologies such as WSNs into disaster management systems is essential. These networks not only improve real-time data collection but also support more effective resource allocation, allowing for more efficient disaster response. A detailed exploration of the role of WSNs in wildfire disasters, along with the specific communication challenges in these environments, is presented in Appendix A. This highlights the importance of these technologies for effective disaster management and underscores the need for more reliable, resilient communication systems.

1.2 Problem Statement

The project client requires the development of a sophisticated WSN that utilises multiple communication protocols. This network must feature an intelligent routing algorithm capable of dynamically selecting the most suitable protocol for data transmission, based on the specific characteristics of the sensor data and prevailing network conditions. Such a system will enable the network to capitalise on the distinct advantages of each communication protocol, such as extended range or higher data rates. Consequently, the system will facilitate the efficient and timely delivery of critical data to emergency response teams, aiming to improve their responsiveness and effectiveness in providing relief.

The name "Coke Can Project" emerged during early conceptualisation, where different soft drink brands were used to symbolise the various roles and functionalities of nodes within the WSN. For instance, a Coke node could be equipped with environmental sensors, a Pepsi node might function as a data router, and a Fanta node could serve as a gateway to other networks. This analogy provided an intuitive framework to visualise and distinguish between the diverse functionalities of the network's components. Additionally, the reference to a "Coke Can" helped to conceptualise the nodes' physical characteristic—each roughly the size of a standard 330ml can—highlighting their portability and suitability for deployment in diverse environments.

Currently, there is limited research and, more importantly, a lack of practical implementation of dynamic multi-protocol routing systems for WSNs. This project aims to bridge that gap by advancing both the theoretical and practical understanding of how WSNs can provide service differentiation through different wireless communication channels depending on the varying requirements of the sensed environmental data.

1.3 Aim and Objectives

1.3.1 Project Aim

The overarching initiative is to develop a WSN with a dynamic routing protocol that can adapt to varying conditions and data requirements. However, the primary aim of this project is twofold: to conceptualise this dynamic routing protocol and to identify a suitable, commercial off-the-shelf (COTS) product that can facilitate the required functionality. Specifically, the project will focus on how each node in the network will perform edge processing to classify the data it retrieves. This classification will enable each node to transmit data according to its specific 'quality of service' (QoS) requirements, ensuring that data is transmitted via the most suitable communication protocol based on factors such as network conditions and sensor data characteristics. By identifying and implementing a COTS product, the project also aims to reduce development time and cost while ensuring the system is scalable and easily deployable in real-world scenarios.

1.3.2 Project Objectives

To achieve the project aim, the following initial objectives were established:

1. **Conceptualise the design of the edge processing framework** that assess data characteristics and network conditions to dynamically choose the appropriate communication protocol for routing.
2. **Source a COTS component** that could facilitate the required functionality of this edge processing.
3. **Document the findings** and provide recommendations for future enhancements and real-world applications.

Following the successful and early completion of these objectives, additional goals were defined to extend the project's scope. This further development included:

1. **Create a flexible and accessible development platform** on the selected COTS product to enable the implementation of the edge processing framework.
2. **Deploy the edge processing framework** onto COTS devices to demonstrate practical application.
3. **Test and evaluate the implemented framework**, assessing its functionality and performance.

1.4 Research Questions

Based on the outlined problem statement, the research questions guiding this project are as follows:

1. How can an edge processing framework for dynamic routing be designed to select the optimal communication protocol based on data priority and prevailing network conditions?
2. Can the proposed solution be implemented using commercially available components, ensuring both practicality and accessibility?

1.5 Scope and Limitations

1.5.1 Project Scope

This project aims to design, implement, and evaluate a dynamic routing framework for WSNs, with a particular focus on disaster management scenarios where the reliability and efficiency of communication is critical. Among various natural disaster threats, wildfires have been identified as the primary use case due to their increasing impact both locally and globally. The project takes advantage of multiple communication protocols to ensure reliable and timely data transmission, ultimately enhancing emergency response capabilities.

Focusing on wildfires provides a clear and measurable context for identifying specific network requirements and evaluating performance parameters. Whilst the wildfire scenario drives the initial design and development, the framework is intentionally designed to be extensible and adaptable. This flexibility allows for seamless adaptation to other disaster management scenarios or even broader applications, enhancing its value to stakeholders across diverse domains. Such versatility demonstrates the framework's potential for long-term relevance and impact beyond its initial scope.

1.5.2 Project Deliverables

The project deliverables were established in consultation with the client and aligned with the approved 'minimum viable product' (MVP) (refer to Appendix B). These deliverables define the core outputs required to achieve the project's aim and objectives. The key deliverables are as follows:

1. **A routing protocol design**, integrated with a decision engine and implemented as functional code.
2. **Prototype WSN node(s)**, demonstrating the practical application of the decision engine's edge processing capabilities.
3. **A detailed client report**, detailing the system's design and implementation, along with exploratory analysis on potential improvements and future directions.
4. **This project report**, providing a complete account of the project, including the background, methodologies, design, implementation, testing, evaluation, and recommendations for future work.

1.5.3 Assumptions and Limitations

It is essential to highlight the assumptions and limitations of the project, as they provide clarity on the factors that influenced the design and scope, as well as the constraints that may impact the system's applicability to the wider audience of end-users.

Table 1 outlines the key assumptions made during the project:

Table 1: List of assumptions.

No.	Assumption
1	It is assumed that the chosen COTS components are representative of typical WSN nodes and are sufficient for implementing and testing the proposed routing framework.
2	It is assumed that the onboard sensors are calibrated to measure environmental parameters effectively, providing data that is sufficiently accurate for effective disaster relief operations.
3	It is assumed that the communication protocols used are interoperable and adequately support the demonstration of the protocol switching functionality supplied by the implemented routing strategy.
4	It is assumed that the routing algorithm could eventually enhance energy efficiency in WSNs through the optimisation of the protocol switching functionality.

Table 2 highlights the limitations encountered during the project.

Table 2: List of limitations.

No.	Limitation
1	The system's performance is inherently tied to the capabilities of the selected COTS components and specific software libraries used.
2	The project prioritises solving the problem defined in the problem statement rather than creating an optimised product. Consequently, the focus is limited to assessing system functionality, not evaluating system performance.
3	The project spans 14 weeks, with all deliverables to be completed and presented by the end of this period.
4	There is no nominal budget for this project, however, to ensure accessibility to a wider audience for future developments, costs were kept as low as reasonably practicable.

1.6 Report Structure

This report begins with **Section 2: Background Overview**, which introduces the topic of WSNs and explores the challenges and advancements in the field. **Section 3: Methodology** outlines the approach taken to develop, implement and test the routing protocol's edge processing functionality. **Section 4: System Design** details the architecture of the edge processing framework and routing algorithm, while **Section 5: Implementation and Testing** describes the system's construction and functional evaluation. **Section 6: Results and Discussion** presents the outcomes of testing, with a focus on discussing the key findings and proposing directions for future development. Finally, **Section 7: Conclusion** summarises the project outcomes, comparing them to the initial aims and objectives, and evaluates the effectiveness of the methodology in delivering a suitable solution for the client. This section also reflects on the overall project management process, highlighting key insights and lessons learned to take forward to future projects.

2 Background Overview

2.1 Overview of Wireless Sensor Networks (WSNs)

WSNs consist of spatially distributed autonomous nodes that monitor and record physical phenomena, such as environmental conditions, and relay the collected data to a base station, known as the sink (Yick et al., 2008; p.1; Akyildiz and Vuran, 2010; pp.37-38). Within each node lies sensors, a 'microprocessor unit' (MCU), and means for wireless communication, as shown in Figure 1.

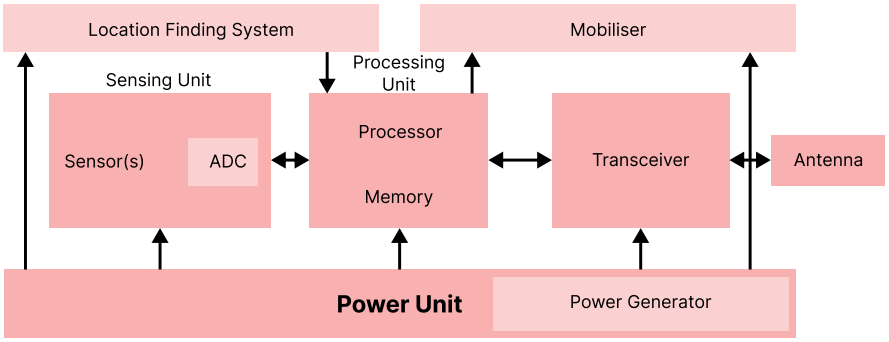

Figure 1: General WSN node hardware architecture (Akyildiz and Vuran, 2010; p.38).

The advancements in 'micro-electro-mechanical systems' (MEMS) and wireless technologies from the 1980s to present day have allowed WSNs to become smaller, more cost-effective, and energy efficient (Algamili et al., 2021). As a result, various related research fields are exploring its application across a wide range of domains, such as environmental monitoring (Werner-Allen et al., 2006), healthcare and biomedical monitoring (Lorincz et al., 2004; Gao et al., 2005), military tracking and surveillance (Simon et al., 2004; Yick et al., 2005; Chipara et al., 2009) industrial applications (Sisinni et al., 2018; Kim and Tran-Dang, 2019), and like the focus of this paper, natural disaster relief (Castillo-Effer et al., 2004; Kandris et al., 2020).

Despite WSNs having seen widespread theoretical advances and growing adoption in recent years through applications such as IoT devices for smart homes (Mouftah et al., 2019; O'Neill, 2023; Amazon, 2024; Google, 2024), challenges remain for their large-scale implementation.

Figure 2: Common smart home IoT devices - Amazon (left) and Google (right).

These challenges mainly revolve around energy consumption and ensuring the reliability of

data transfer (Mahmood et al., 2015). Although modern IoT solutions have made progress in addressing some of these issues, particularly in consumer devices, these two challenges persist across network functions and remain a critical focus.

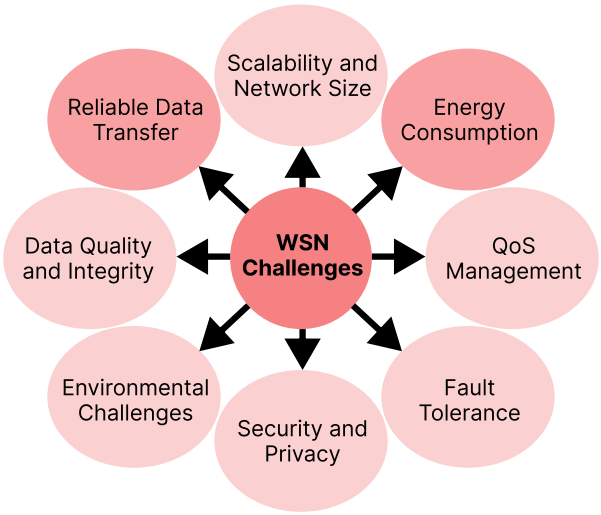

Figure 3: Challenges to the widespread adoption of WSNs (Yin et al., 2022; Landaluce et al., 2020; Elhoseny et al., 2017).

Reliability in data transfer is typically ensured through two main mechanisms: retransmission and redundancy (Wen et al., 2007). However, in the context of disaster relief, where timely communication is vital, QoS management can ensure that critical messages, such as emergency alerts, are prioritised and delivered even when the network is congested or under strain, guaranteeing the reliability of urgent data transmission (Uthra and Raja, 2012). These approaches help maintain data integrity even in the face of transmission failures or network congestion.

On the other hand, energy management in WSNs is a broader concept, extending beyond simply prolonging battery life. It encompasses optimising functions such as routing efficiency, minimising energy consumption during data transmission, and addressing other less obvious aspects of network operations - each of which plays a crucial role in enhancing overall energy efficiency.

To tackle the multifaceted challenge of improving reliability and energy efficiency in these systems, a comprehensive understanding of the WSN architecture and design is essential.

2.2 WSN Architecture and Design

In WSNs, sensor nodes serve dual purposes: they act both as data organisers and data routers (Akyildiz and Vuran, 2010; p.43). Data transmission occurs for two reasons, these being:

- **Sensor function** - the transmission of event information from source nodes to the sink.
- **Router function** - the forwarding of information received from other nodes to the next destination along the path to the sink.

WSNs efficiently manage these functions through a layered architecture, which enables seamless communication and coordination between sensor nodes and routing tasks. The traditional structure of this layered architecture is shown in Figure 4.

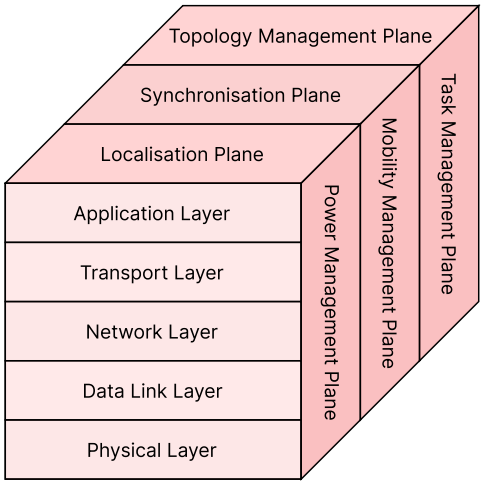

Figure 4: WSN layered architecture.

The architecture of most WSNs typically follows this stack, albeit in slightly different forms depending on the specific requirements of the system. This stack includes several protocol layers, management schemes, and additional components designed to enhance performance.

Starting from the bottom, the physical layer handles modulation, transmission, and reception, ensuring reliable performance even in noisy environments. The data link layer manages communication reliability through error control and channel access using the 'medium access control (MAC) protocol to minimise collisions. The network layer is responsible for routing data, while the transport layer ensures data flow continuity for sensor network applications. The application layer supports software for various sensing tasks and provides user interfaces for network interaction. Additionally, the power, mobility, and task management planes optimise power usage, node movement, and task distribution to improve efficiency.

The following sections provide a detailed exploration of the five core layers, examining their functions and the techniques developed over time to enhance performance and functionality of WSNs.

2.2.1 Physical Layer

The physical layer encompasses all hardware components within the networked nodes, such as sensors, processors and power sources (as seen in Figure 5).

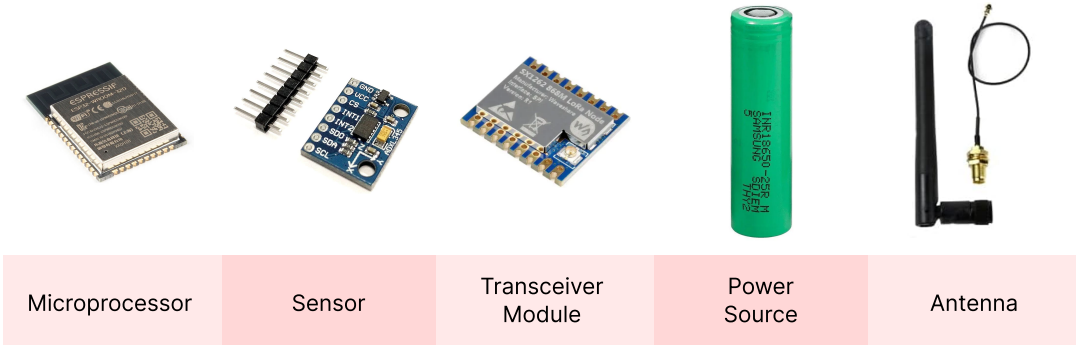

Figure 5: Examples of WSN node hardware components.

The layer's primary function is to convert data into a radio signal, which is then transmitted wirelessly through the air via electromagnetic (EM) waves (Faludi, 2011; pp.23-25). 'Radio frequency' (RF) communication uses various technologies including 'narrow-band' (NB), 'spread-spectrum' (SS), and 'ultra-wide band' (UWB) categories (Akyildiz and Vuran, 2010; p.54):

Communication Type	Description
Narrow-band (NB)	Early WSN systems used narrow-band techniques, which involve transmitting signals within a small, specific frequency range.
Spread-spectrum (SS)	Improves data transmission speed and resistance to interference by spreading the signal over a wider range of frequencies.
Frequency hopping spread spectrum (FHSS)	Switches between different frequency channels, like how Bluetooth works.
Direct sequence spread spectrum (DSSS)	Spreads the signal over a wider bandwidth using special codes, commonly used in WSNs, especially in IEEE 802.15.4 for 'low-rate wireless personal area networks' (LR-WPANs) (Yang, 2017; pp.147-164).
Ultra-wide band (UWB)	Sends signals directly without needing a carrier frequency, using a technique called pulse position modulation (PPM) to transmit data efficiently.

Table 3: Types of RF communication in WSNs.

The variety of RF communication techniques addresses different trade-offs in WSN applications. NB is energy-efficient for low-interference environments, while spread-spectrum methods like FHSS and DSSS improve reliability in noisy or congested settings. UWB offers high data rates with low power, ideal for precise tracking and high-throughput applications. These techniques enable WSNs to be optimised for specific needs, balancing energy efficiency, reliability, and scalability.

2.2.2 Data Link Layer

The data link layer plays a key role in organising data into packets and managing how devices share the communication channel. This layer is closely tied to the design of MAC protocols, which aim to prevent devices from "talking over" each other. Since WSN nodes run on limited power, saving energy is a top priority in MAC protocol design. These protocols are typically grouped into three types: contention-based, reservation-based, and hybrid (Ye et al., 2002; Raghavendra et al., 2006). These protocols generally use two multiple access schemes: 'carrier sense multiple access' (CSMA) and 'time division multiple access' (TDMA).

2.2.2.1 CSMA and CSMA/CA

Most WSN MAC protocols use CSMA, which requires a node to listen to the channel before transmitting. If the channel is clear, the node transmits; if not, it waits and retries. CSMA/CA enhances 'collision avoidance' (CA) by reserving the channel with 'request-to-send' (RTS) and 'clear-to-send' (CTS) packets to prevent devices that can't detect each other from trying to send messages to the same destination.

In CSMA, if the communication channel is busy, the device waits for a short pause, called the 'interframe space' (IFS). Once the channel is clear for a certain time ('distributed coordination function' IFS (DIFS)), the device picks a random wait time to avoid clashing with others. After this wait, it tries to send its data. If the device receives an 'acknowledgement' (ACK) response from the receiver, the data reception was successful. If not, the data packet is retransmitted.

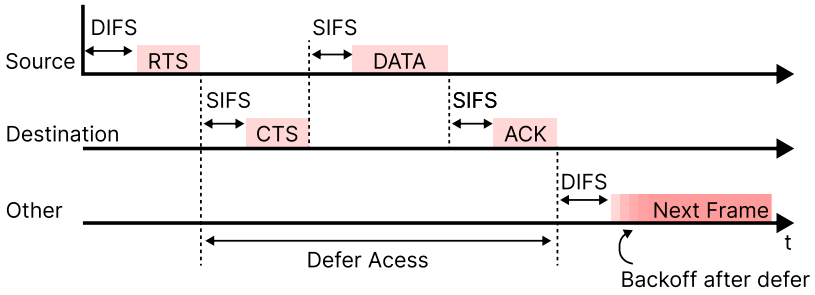

Figure 6: CSMA/CA mechanism.

Although this mechanism helps manage communication, data packets can still collide if they send CTS signals at the same time. To reduce this, a technique called binary exponential backoff makes devices wait longer and longer before trying again. Another tool, the 'network allocation vector' (NAV), acts like a timer, telling devices when the network will be free. This way, devices can stay off and save power until it's their turn to send data.

2.2.2.2 TDMA

As an alternative to CSMA, TDMA avoids collisions by dividing the communication into time slots. Nodes reserve slots during a reservation period, and data is transmitted during the designated time slots, ensuring collision-free communication.

2.2.3 Network Layer

The network layer in WSNs focuses primarily on routing algorithms and protocols (Lewis, 2004; pp.17-19). Due to the large number of nodes, address-based protocols are impractical, so data-centric routing is used, where queries are based on attributes of the sensed phenomenon, and only relevant nodes respond. This layer is crucial for efficiently routing data between nodes and the sink. Unlike ad hoc networks, WSNs often use data-centric protocols due to the difficulty of assigning unique IDs to a large number of sensor nodes. Instead of querying specific nodes, users query attributes of the observed phenomena. This data-centric approach allows for more relevant, attribute-based routing.

2.2.3.1 Flat-Architecture Protocols

One common approach to implementing data-centric routing is through flat-architecture protocols, where all nodes are treated equally in terms of routing responsibilities. In such systems, protocols like flooding and gossiping are widely used. However, these protocols suffer from inefficiencies, such as message redundancy and high energy consumption, particularly in large-scale networks. Flooding, for example, broadcasts data to all nodes indiscriminately, leading to issues like message implosion, overlap, and resource blindness. Whereas gossiping mitigates these problems by randomly selecting a single neighbour to forward data, reducing energy consumption but increasing latency (Chandna and Singla, 2015).

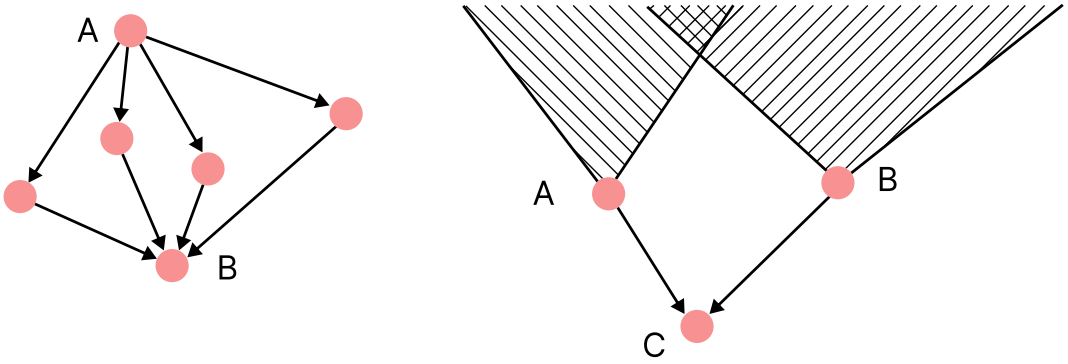

Figure 7: Main problems with flooding: implosion (a) and data overlap (b) (Akyildiz and Vuran, 2010; p.144).

More advanced flat protocols, like 'sensor protocols for information via negotiation' (SPIN), improve upon flooding and gossiping by introducing negotiation mechanisms and energy-awareness, sending data only to interested nodes and adjusting participation based on energy levels (Akyildiz and Vuran, 2010; pp.145-146). Similarly, directed diffusion builds routes dynamically based on the sink's interests, reinforcing paths for optimised data delivery. Both mechanisms are illustrated in Figure 8.

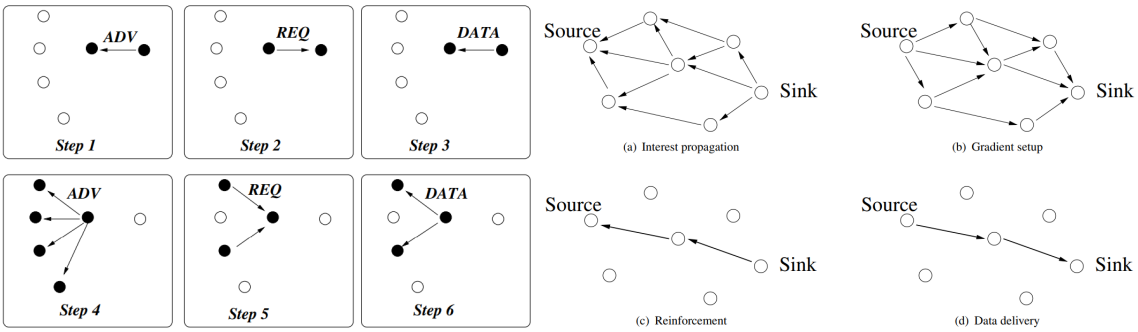

Figure 8: SPIN (left) vs directed diffusion (right) (Akyildiz and Vuran, 2010; pp.145-146).

Despite the advancements in flat-architecture protocols, such as improved efficiency and reduced redundancy, they still face significant scalability and energy consumption challenges. More advanced routing protocols are classified into hierarchical, geographical, and QoS-based categories to address these limitations and optimise performance for different applications.

2.2.3.2 Hierarchical Routing

Hierarchical routing organises nodes into clusters, where cluster heads are responsible for aggregating data from their members and transmitting the aggregated data to the sink or base station. This method significantly reduces the number of transmissions and the size of transmitted data, resulting in better energy efficiency, particularly in large-scale networks. By localising communication within clusters, hierarchical routing extends the network's lifespan, making it more suitable for networks with many sensor nodes (Haque et al., 2018).

2.2.3.3 Geographical Routing

Geographical routing uses nodes' physical locations to make routing decisions. Each node typically relies on its position, often provided via GPS, to determine the next hop in the network. This approach simplifies routing by eliminating the need for complex routing tables, as data is forwarded to the node closest to the destination. Geographical routing is particularly useful in dynamic networks where node locations change frequently, as it adapts easily to topology changes without extensive recalculations (Grover et al., 2014).

2.2.3.4 QoS-based Routing

QoS-based routing ensures performance requirements such as latency, bandwidth, and packet delivery ratio are met, prioritising traffic for applications like real-time monitoring. One core capability of this type of routing is its ability to assign priority to different traffic types, optimising resource management (Asif et al., 2017). By selecting routes that meet QoS criteria, critical data is reliably delivered. However, maintaining performance can complicate energy management, requiring a balance between high performance and energy efficiency.

These different strategies optimise data delivery in combination with the transport layer, ensuring reliability and integrity as data moves through the network.

2.2.4 Transport Layer

The success and efficiency of WSNs hinges on dependable communication between sensor nodes and the sink. In a network where data travels through several nodes, ensuring this reliability requires a robust transport mechanism. The transport layer is key in controlling network congestion, ensuring reliable data transfer, and bridging the gap between the application and network layers through (de)multiplexing (the combining and distribution of event data across the network (Akyildiz and Vuran, 2010; p.167)).

While many transport layer solutions exist for conventional wireless networks, they often fall short when applied to WSNs due to the latter's unique characteristics. TCP/IP, a foundational suite for traditional networks, includes the 'transmission control protocol' (TCP) and the 'internet protocol' (IP). TCP provides end-to-end reliability through connection establishment, acknowledgements, and retransmissions, while IP handles addressing and routing across networks. However, these mechanisms introduce considerable overheads which can be inefficient in the resource-constrained environments typical of WSNs (Kurose and Ross, 2000; pp.171-177). Furthermore, the assumption of end-to-end global addressing common in TCP/IP is not feasible in WSNs, where sensor nodes may lack unique addresses. Instead, WSNs often use attribute-based naming and data-centric routing, which require transport layer protocols specifically designed to handle these conditions effectively (Raghavendra et al., 2006; pp.27-29). Several such protocols have been developed to address these challenges and are both illustrated and referred to in Figure 9 by Akyildiz and Vuran (2010; p.169).

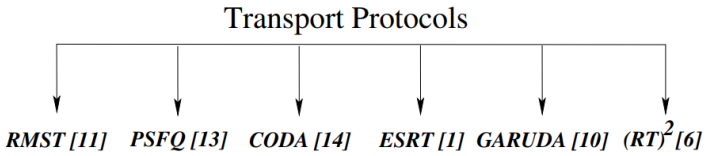

Figure 9: Transport protocols for WSNs (Akyildiz and Vuran, 2010; p.169).

2.2.5 Application Layer

The application layer acts as the interface between the WSN and the real world. It facilitates the interaction with collected data for the user, without needing to understand the details of the network. The tasks of this layer include compressing data (source coding), processing user queries (query processing), and managing the network's operation (network management), to ensure the system runs smoothly and effectively.

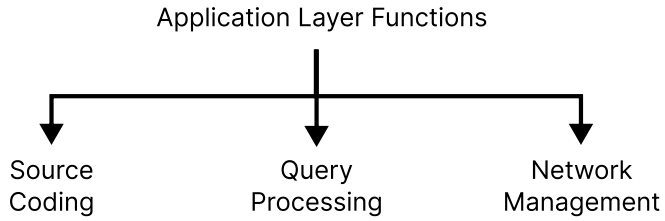

Figure 10: Application protocols for WSNs.

2.2.5.1 Source Coding

Source coding involves the compression of data at the sensor nodes to reduce the amount of data transmitted across the network. By minimising the data size, source coding conserves bandwidth and energy, critical resources for WSNs. Efficient coding schemes can significantly extend the network's lifespan by reducing the communication load.

2.2.5.2 Query Processing

Query processing allows the users to request specific data from the network using high-level queries. The network processes these queries and returns the relevant data, often filtering or aggregating information to meet the user's needs. This approach streamlines data retrieval and reduces unnecessary data transmissions, enhancing overall network efficiency.

2.2.5.3 Network Management

Network management protocols focus on maintaining and optimising the network's performance. This includes tasks like monitoring node health, managing energy resources, and configuring network settings. Effective network management ensures that the network operates smoothly, adapts to changes, and remains resilient over time, even as nodes fail, or new nodes are added.

With the architecture and design of WSNs providing the necessary framework for efficient communication and data management, the next crucial step is understanding the communication protocols that facilitate data transfer across these networks. These protocols play a vital role in ensuring the efficient operation and performance of WSNs, especially in dynamic environments such as disaster response, where timely and reliable communication is paramount.

2.3 Key Communication Protocols for WSNs

Designing a WSNs is an inherently cyclical process, in which each design choice influences and is influenced by a myriad of other system factors. Although this iterative process can be lengthy and at times complex, it ensures the effective integration of key components - such as sensor nodes, communication protocols, and data management strategies - aligning them with the specific requirements of the application.

One critical aspect of WSN design is the choice of communication protocol(s). Without an effective means of wireless communication, sensor nodes, regardless of their sensing capabilities, would be rendered practically useless. The ability to transmit data between nodes and to external systems is fundamental to the functionality of any WSN.

In the realm of WSNs, several communication protocols are commonly employed, each with its unique advantages and limitations. Table 4 shows a summary of three common wireless communication standards.

Table 4: Overview of communication standards: LoRa, IEEE 802.15.4, and Wi-Fi.

Communication Protocol	Description
LoRa	The 'long range' (LoRa) communication standard excels in providing extended range while maintaining low power consumption, making it ideal for wide-area networks like environmental monitoring and disaster management. However, its key limitation is that it operates with a relatively low data rate.
IEEE 802.15.4	This standard underpins the design of LR-WPANs, including protocols like ZigBee and Thread. It is optimised for low-power, short-range communication, ensuring reliable data transmission over limited distances, therefore rendering it unsuitable for networks containing long-range or widely disperse nodes.
Wi-Fi	Wi-Fi facilitates communication with higher data rates over short distance with robust and high-speed connectivity, suitable for dense environments. Therefore, it can facilitate transmission of multimedia (such as video footage), however is limited similarly to IEEE 802.15.4 in terms of range.

Each protocol offers distinct QoS features suited to specific applications. For instance, Wi-Fi and Bluetooth excel in high-data-rate applications, while LoRa is optimal for long-range, low-power applications and IEEE 802.15.4 for real-time, reliable communication over short distances. Selecting the right protocol involves balancing data rate, power, range, and QoS needs, ensuring the WSN design meets its goals efficiently and effectively.

2.4 Hardware Constraints for WSNs

As previously discussed, designing WSNs is a complex, interconnected process that involves balancing communication protocols with available hardware. Beyond protocols, technical specifications such as 'size, weight, and power' (SWaP), processing capacity, memory, and energy requirements are critical factors. Regulatory constraints also play a vital role, as compliance with legal standards for frequency use, data privacy, and environmental impact must be integrated into hardware choices and network design.

The following sections explore these considerations in more detail, highlighting their impact on WSN design and performance.

2.4.1 SWaP Requirements

The limitations imposed by SWaP are crucial factors in selecting hardware for WSNs. Nodes need to be adaptable to various deployment scenarios, such as being lightweight and portable enough to be carried by users or dropped in by drones. These nodes should be small enough to fit comfortably in a person's hand and operate efficiently in confined spaces, facilitating easier deployment in challenging environments.

Additionally, nodes may need to function wirelessly and operate on battery power, or leverage different energy harvesting methods. These methods include photovoltaic (solar), piezoelectric/vibration, thermal, flow or wind harvesting (Shaikh and Zeadally, 2016).

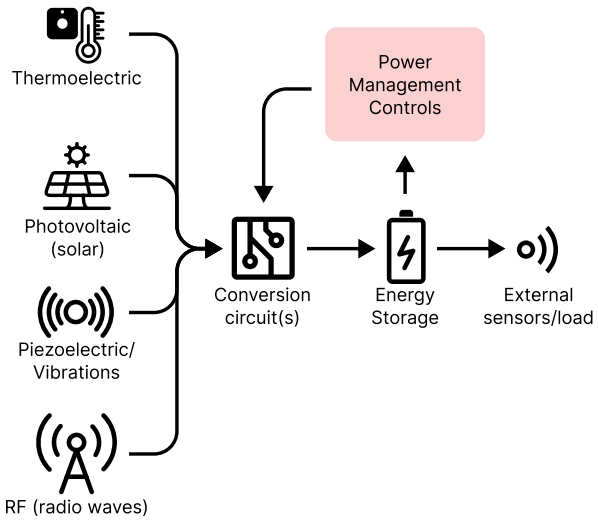

Figure 11: Energy harvesting process.

Such techniques reduce the dependency on batteries, allowing nodes to operate autonomously for extended periods of time in the remote locations typical of WSN deployment. Nevertheless, the incorporation of these harvesting techniques may prove difficult in combination with the previously advised small form factor. Therefore, it is often more effective to distribute network functionality across multiple node types, each tailored to a specific task. This reduces hardware density per node and optimises performance in large-scale WSN deployments.

2.4.2 Processing Power and Memory

Hardware that fulfils the SWaP requirements for WSN sensor nodes typically have limited processing capabilities and memory compared to traditional computing devices. This limitation affects the level of complexity of the edge processing (where data is processed near its source) that can be implemented on these nodes. To address this, lightweight data processing techniques are often employed, but these must be simple and energy-efficient due to the limitations of the hardware.

2.4.3 Energy Consumption

Energy efficiency is a primary concern in WSN design, as reiterated appropriately. Sensor nodes are often battery-powered and need to operate for extended periods without frequent recharging or replacement. Consequently, power consumption must be carefully managed, influencing choices in communication protocols, data processing, and routing strategies. High energy consumption can arise from frequent data transmissions, complex processing tasks, or inefficient communication protocols. As illustrated in Figure 12, most of the energy consumption in a WSN node is due to the hardware components involved in wireless communication being active or operational.

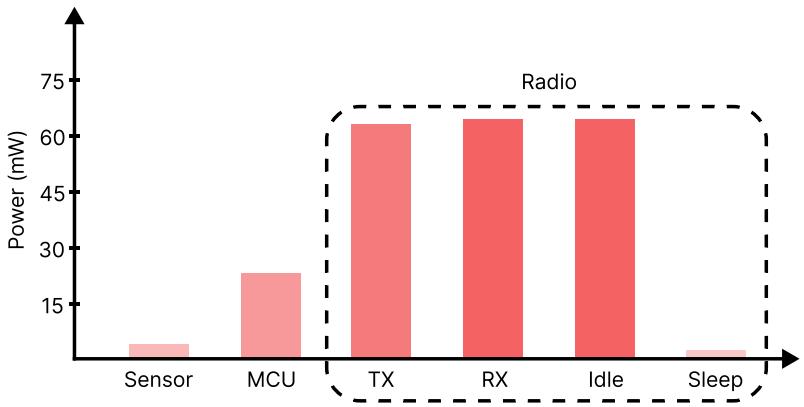

Figure 12: Approximate node energy consumption during routine WSN activities (Akyildiz and Vuran, 2010; p.44).

Therefore, to significantly reduce energy consumption, data should be transmitted using the most efficient communication protocol, and radios should be powered down when not in use.

2.5 Legal Constraints for WSNs

As the deployment of WSNs expands across various sectors, it becomes increasingly important to address the regulatory frameworks that govern their use. These regulations are critical for ensuring the safe, legal, and efficient operation of WSNs. The following sections explore key regulatory considerations that impact WSN design and deployment: communication regulations, which govern frequency usage and network operations; data regulations, which address privacy and security concerns; health and safety regulations, which ensure that WSNs do not harm users or the environment; and environmental regulations, which ensure the sustainable use of WSN technology. Understanding these regulatory policies is essential for designing WSNs that are both compliant and effective in real-world applications.

2.5.1 Communication Regulations

2.5.1.1 Communication Act 2003

The Communications Act 2003 sets the overarching regulatory framework for the communications sector in the UK and established 'The Office of Communications' (Ofcom) as the regulatory authority. The Act addresses spectrum management, competition, and consumer protection (UK Parliament, 2023).

2.5.1.2 Wireless Telegraphy Act 2006

Operating within the framework set out by the Communications Act 2003, the Wireless Telegraphy Act 2006 provides the specific regulations for licencing and spectrum management; it is unlawful to establish wireless telegraphy station(s), or to install or use wireless telegraphy apparatus except under and in accordance with a wireless telegraphy licence granted by Ofcom (UK Parliament, 2006).

Globally, the 2.400-2.4835 GHz band is license exempt and free to use for many purposes, including Wi-Fi networks, Bluetooth short range links, etc. However, certain countries and continents have specific bands within which more of the spectrum is allocated for licence-free use.

While the use does not require a licence, the devices must meet strict regulations and be certified. Using equipment that doesn't meet the conditions of the license exemption is an offense under the Act, which can result in fines of up to £5,000 and/or six months' imprisonment.

2.5.1.3 Radio Equipment Regulations 2017 (SI 2017/1206)

The Radio Equipment Regulations 2017, which are part of UK law, implement the EU Radio Equipment Directive (RED) (European Parliament, 2014; UK Parliament, 2017). These regulations require all radio equipment to meet essential requirements regarding health and safety, electromagnetic compatibility (EMC), and the efficient use of the radio spectrum to avoid harmful interference.

Alongside this, the UKCA (UK Conformity Assessment) marking must be present on all COTS components, ensuring conformity with UK regulations - tested and certified for UK safety, EMC, and radio spectrum regulations (Department for Business and Trade, 2024).

2.5.1.4 Electromagnetic Compatibility (EMC) Regulations 2016

The EMC regulations were established to ensure that electronic devices do not generate excessive EM interference and are resistant to interference from other sources (UK Parliament, 2016). These regulations help devices operate correctly and reliably within their intended environments.

2.5.1.5 UK Interference Requirements 2030 (IR 2030)

As part of the Radio Equipment Regulations 2017, the Interface Requirements 2030 sets out the requirements for the use of short-range-devices (SRDs) in the specified frequency bands (Ofcom, 2023). Although the name implies small transmission distances, the SRD classification applies to devices that use technology such as LoRa, which can transmit over several kilometres (which is short relative to devices facilitating satellite communication).

For networked SRDs, as seen in WSNs, the IR 2030/1 and IR 2030/31 specifications outline the interface requirements for these systems while operating in the 868 MHz and 2.4 GHz bands. The specific requirements within these bands are displayed in Table 5.

Table 5: IR 2030 requirements.

	Frequency Bands	
Nominal Frequency Range	863-870 MHz	2.4000-2.4835 GHz
Maximum Power Output	14 dBm (25mW) Some sub-bands permit 20dBm (100mW) EIRP	20 dBm (100mW) EIRP
Power Adaptive Control	Required (in some sub-bands)	Not Required
Duty Cycle Restrictions	Range from 0.1% - 10%	None
Channel Access and Interference Avoidance Techniques	Required	Required

In addition, the IR 2030 also outlines requirements for SRDs in sub-bands within and outside these major bands.

2.5.2 Data Regulations

Since any WSN is data-centric, it is imperative that this data is handled, processed and disposed of in accordance with legislation, ensuring compliance and ethical operations. The regulations below detail the regulatory requirements that should be adhered to when creating and using these systems.

2.5.2.1 GDPR (General Data Protection Regulation)

The General Data Protection Regulation (GDPR), passed and enforced by the EU, governs the processing of personal data of individuals within the EU and UK (European Parliament, 2024). Therefore, for any WSNs that collect, store, or process personal data, this set of regulations apply.

The regulations mandate strict data protection measures which include data subject rights, minimisation, and security regarding the handling of personal data.

2.5.2.2 Data Protection Act 2018

The Data Protection Act 2018 complements GDPR, specifying data protection provisions and regulations enforceable in the UK (UK Parliament, 2018a).

2.5.2.3 Data Retention Regulations 2014

For any WSN storing data for legal or regulatory purposes, the Data Retention Regulations 2014, dictate how long it must be stored and the method for handling this data (UK Parliament, 2014).

2.5.2.4 Network and Information Systems Regulations 2018 (NIS Regs)

As the wider initiative is for a WSN providing emergency services, and potentially establishing temporary critical communications infrastructure, the Network and information Systems Regulations 2018, should be considered (UK Parliament, 2018b). These regulations set out the security measures required to increase the network's resilience against cyber-attacks.

2.5.2.5 Freedom of Information Act 2000

Finally, as with all information systems operated by public authorities, the created WSN may be subject to requests for information under this Act (UK Parliament, 2000). Therefore, the system must be able to provide documents of the data processed in order to adhere to these requests.

2.5.3 Health and Safety Regulations

The Health and Safety at Work etc. Act 1974 ensures that all systems are designed and operated in a safe manner, protecting workers and the general public (UK Parliament, 1974). These regulations should be adhered to throughout the system development lifecycle.

2.5.4 Environmental Regulations

As the nodes of the WSN are to be deployed in harsh and potentially inaccessible conditions, compliance with the environmental regulations is crucial.

2.5.4.1 The Waste Electrical and Electronic Equipment (WEEE) Regulations 2013

The WEEE Regulations ensure proper disposal and recycling of electronic waste, minimising the environmental impact (UK Parliament, 2013). The WSN nodes should therefore be designed with end-of-life disposal in mind.

It is also important to note that, for future works (i.e., when the system is at a larger scale), it would be good practice to conduct environmental impact assessments (EIAs) to help minimise ecological disruption during system deployment.

2.5.4.2 Restriction of Hazardous Substances (RoHS) Regulation

Alongside the WEEE Regulations, the Restriction of the Use of Certain Hazardous Substances in Electrical and Electronic Equipment Regulations 2012, restricts the use of certain hazardous materials in electrical components, reducing both potential environmental and health risks (UK Parliament, 2012). To ensure RoHS compliance, suppliers should be checked and verified for appropriate certification prior to purchasing of COTS components.

2.6 Summary of Literature: Project Trajectory

The background overview provides a concise exploration of the key factors influencing the design and deployment of WSNs, laying the knowledge foundation in order to fulfil the project's aim and objectives. It covers essential topics such as communication protocols, routing methods, edge processing, and hardware constraints, ensuring the WSN functions effectively to support the development of a dynamic routing protocol. Additionally, it highlights the need to consider regulatory constraints to ensure compliance with relevant standards.

By exploring these areas, the literature review sets the stage for the project's trajectory and highlights the technical and regulatory considerations that will influence its success.

The next stages of the project will detail the methodology, system design, implementation, and testing phases. The methodology will detail how the project was ran, ensuring sufficient tools and techniques are used to minimise uncertainty and ensure project success. The system design will outline the routing protocol and edge processing functionalities, ensuring that each node in the network can autonomously classify and transmit data based on real-time network conditions, while the implementation phase will involve integrating these features with the selected COTS product. Finally, testing will evaluate the system's functionality against the project's goals, validating the effectiveness of the edge processing framework's implementation.

3 Methodology

3.1 Methodology Synthesis

The following sections outline various project and product methodologies, exploring their origins, contexts of use, and respective strengths and weaknesses. These methodologies are then synthesised into a unified approach, which was subsequently applied throughout the project detailed in this report.

3.1.1 Design Thinking

The design thinking process, a methodology that has evolved over the last few decades, was first mentioned by Simon (1996), however ill-defined at the time. Former president of the Design Management Institute, Lockwood (2010) proposed a detailed definition of the methodology, describing it as a human-centred process that emphasises collaboration, rapid concept prototyping, and concurrency, to name a few aspects.

The methodology follows an iterative flow through the stages: emphathise, define, ideate, prototype, and test, as shown in Figure 13, and elaborated upon in line with the 'Design Thinking Process Guide' (University of Stanford, 2024).

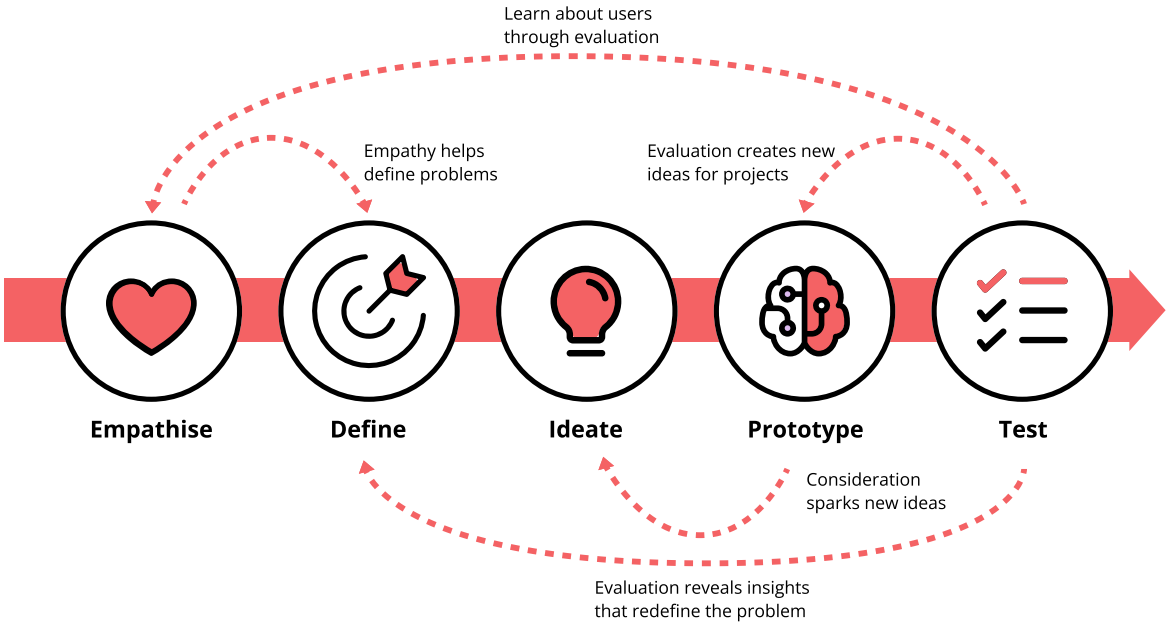

Figure 13: The design thinking methodology.

The following sections detail the functions of and activities involved within each phase of the 'Design Thinking' methodology.

3.1.1.1 Empathise

The empathise stage places emphasis on cross-collaboration between "designers" and "users" which promotes a highly effective, user-centred approach to projects, where the primary goal is to meet specific requirements rooted in the needs of the client or end-user. This collaboration ensures designers make informed, well-grounded decisions throughout the process - imperative for project success (Hanan and Karp, 1989). A strong focus on the empathise phase is crucial, as it lays the foundation for a well-defined project, synthesising user needs and ensuring the solution effectively addresses the correct problem.

3.1.1.2 Define

The define stage is about bringing clarity and focus to the design challenge based on the insights gathered during the empathise phase. It is the designer's responsibility to distill the information collected about the users and the context or application of use, making sense of the widespread data. The ultimate goal of this stage is to craft a meaningful, actionable problem statement, which clearly defines the challenge to address. The define stage acts as the primary instance convergence within the project lifecycle, narrowing down the system design focus, which paradoxically leads to more creative and impactful solutions.

3.1.1.3 Ideate

The ideate stage focuses on generating a broad spectrum of ideas and solutions. It encourages designers to explore concepts ranging from the most sophisticated and innovative to the simplest and most unconventional, ensuring a comprehensive exploration of possibilities.

Ideation is crucial as it provides the foundation for building prototypes and developing solutions that can be tested with users. The goal is to transition from problem identification to solution generation by blending a deep understanding of the problem space with creativity. Rather than seeking a single best solution, ideation aims to generate a wide array of ideas, expanding the scope of potential innovations. This process enables teams to move beyond obvious solutions, leverage diverse perspectives, and uncover unexpected areas of creativity, ensuring both flexibility and variety in the final solution set.

3.1.1.4 Prototype

The prototype stage involves creating iterative artifacts to explore and refine solutions. Early prototypes are low-resolution and inexpensive, designed to address broad components of the problem. As the project progresses, prototypes and components become more specific.

3.1.1.5 Test

The test stage involves gathering user feedback on prototypes to refine solutions and deepen understanding. This stage transitions from initial empathy to evaluating how well prototypes address the problem and meet user needs. The methodology emphasises that, rather than solely assessing user preferences, focus should be on probing "Why?" to uncover deeper insights into user behavior and the effectiveness of the solution.

Testing ideally occurs in the user's natural environment or through realistic simulations to ensure genuine feedback. This phase is crucial for refining prototypes, validating the problem framing, and making necessary adjustments based on real-world insights.

3.1.1.6 The Rules

These project stages are built around four essential rules of design thinking conceptualised by Wolniak (2017), and detailed in Table 6.

Table 6: The rules of design thinking (Wolniak, 2017).

Rules	Explanation
The human rule	Innovation is fundamentally social; while individual contributions are crucial, a diverse and agile focus group is essential for creating groundbreaking innovations. People are the most valuable asset in the design process.
The ambiguity rule	Designers must embrace ambiguity and avoid design fixation. Innovation requires experimentation, pushing boundaries, and the freedom to see things differently. The process is often lengthy and uncertain but essential for creating alternative futures.
The re-design rule	All innovation is a form of re-innovation. Understanding past solutions helps inform future developments. By learning from previous approaches and adapting to evolving technologies and social conditions, we can better anticipate and address future needs.
The tangible rule	Make innovation tangible through rapid learning and conceptual prototyping, which is crucial for effective design thinking.

These rules form the foundation for effective innovation throughout the project lifecycle.

3.1.2 Systems Engineering V-Model

The Systems Engineering V-Model was developed independently in both Germany and the United States during the 1980s. In the US, it first emerged at Hughes Aircraft in 1982, where it was applied to the Federal Aviation Administration (FAA) Advanced Automation System program. The model aimed to detect latent defects in software, particularly for automating air traffic control processes. The U.S. Department of Defense later adopted the V-Model, extending its use across systems engineering projects, particularly in defense and aerospace (Forsberg and Mooz, 1991).

In Germany, the V-Model originated in the late 1980s through collaboration between Industrieanlagen-Betriebsgesellschaft mbH (IABG) and the Federal Office for Defense Technology and Procurement, primarily for defense projects. By 1992, the model was adopted by the Federal Ministry of the Interior for civilian projects and remains a standard for federal administration and defense systems (IABG, 1992).

The V-Model follows a 'V' shape, progressing from requirements decomposition on the left to integration and testing on the right. It excels at linking design and testing, ensuring components are iteratively built, tested, and validated against predefined objectives.

Despite its growing popularity, the V-Model has faced criticism for not accommodating the iterative nature of software development, where each phase affects the previous ones (Marick, 1999). Additionally, some argue that its rigid framework limits its testing methodologies (Harmonic Software Systems, 2015).

Nevertheless, the V-Model remains adaptable in its scope and application, proving valuable for various systems engineering projects. Over time, it has evolved into the iterations we use today, as depicted in Figure 14.

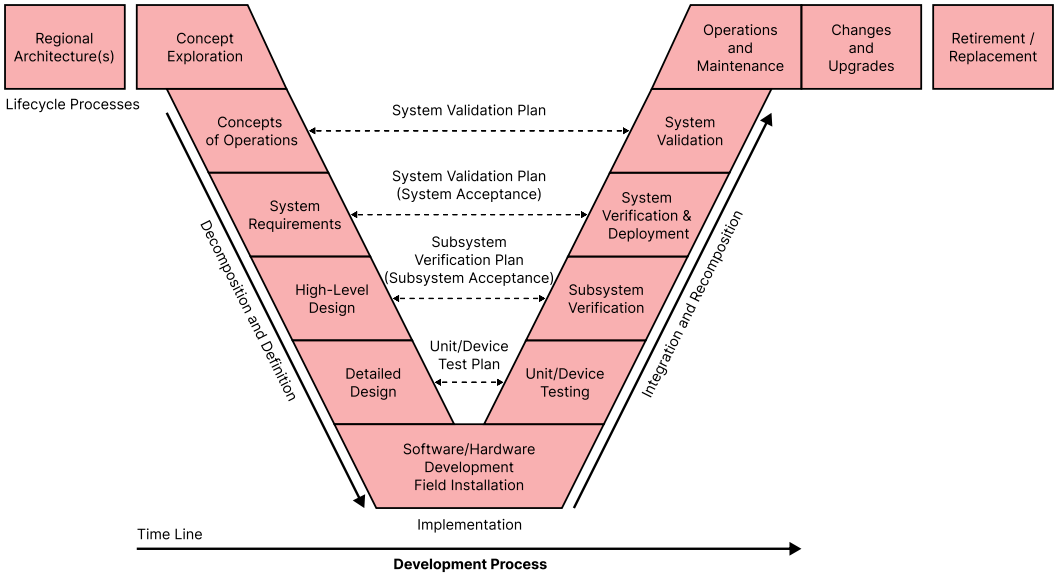

Figure 14: Systems engineering V-model.

3.1.3 The Sprint Scrum Structure

Scrums, first developed in the 1990's by Schwaber and Sutherland (2020), are small subcycles of a larger sprint structure, built on empiricism and lean thinking, ensuring that decisions are made based on newfound knowledge rather than pre-existing plans. Sprints are fixed-duration periods, typically lasting a month or less, during which value is added through structured project phases: planning, daily scrums, implementation, review, and retrospection. These two agile frameworks are combined to form the following flow structure.

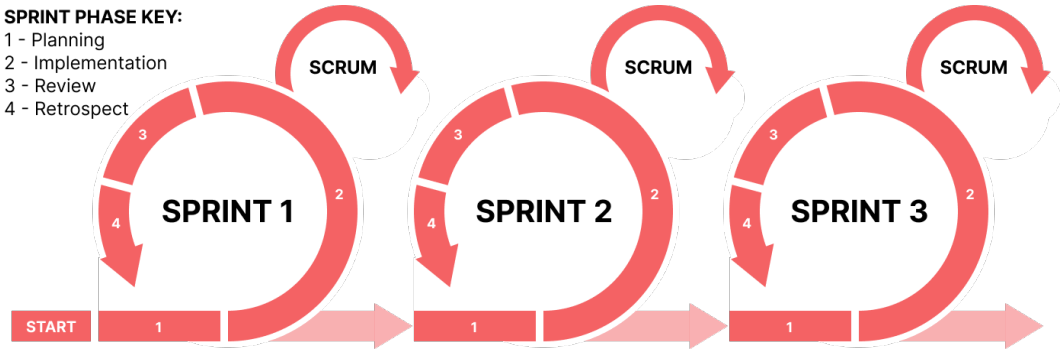

Figure 15: Agile sprints and scrums.

This structure enables sprint and scrum teams to be agile throughout the project lifecycle.

3.1.4 Adopted Methodology

3.1.4.1 Methodology Creation

By integrating Design Thinking, the V-Model, and Sprint Scrum techniques into a cohesive methodology, this approach provides a robust framework for managing complex systems engineering projects. It effectively addresses the evolving nature of these projects by ensuring adaptability, iterative development, and a user-centered focus throughout the project lifecycle. Each of the individual methodologies brings unique strengths that, when combined, enhance overall project effectiveness. The methodology is illustrated in Figure 16.

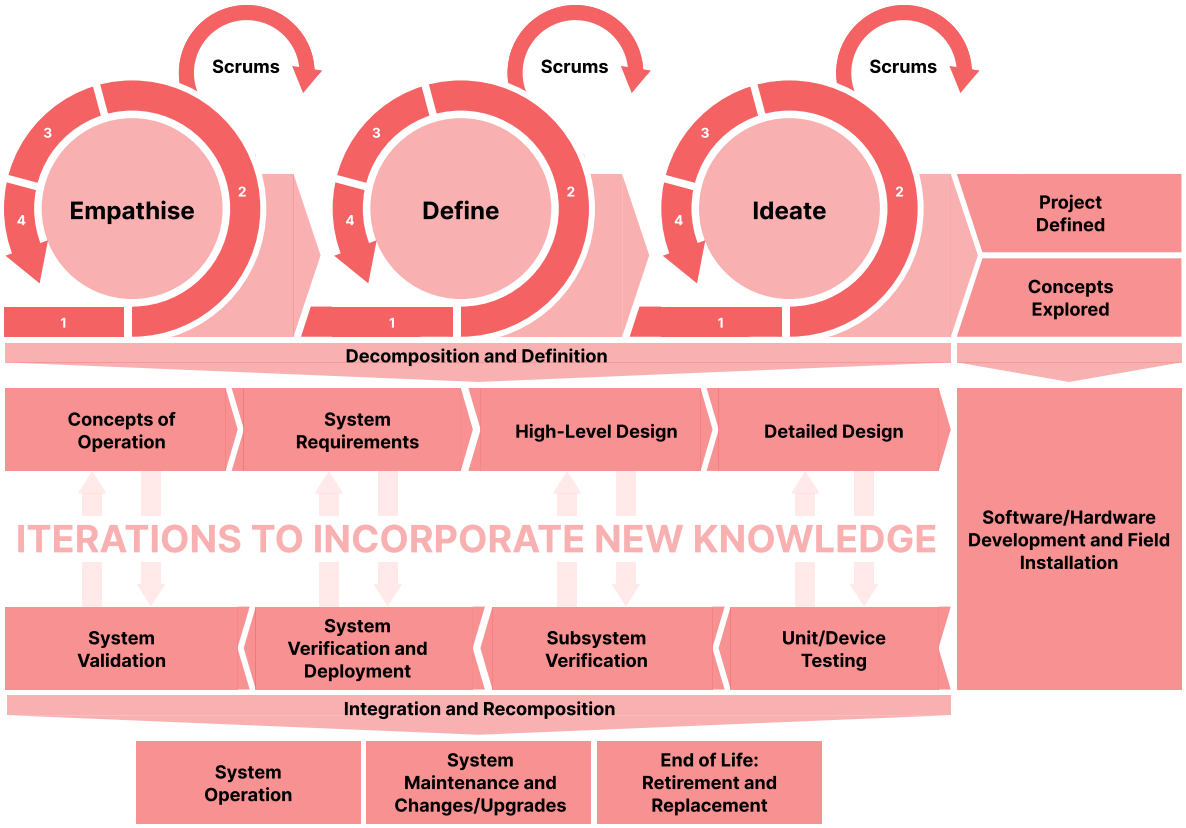

Figure 16: Adopted methodology.

The Design Thinking's methodology ensures a user-centered approach, emphasising empathy and iterative refinement to deeply understand and address user requirements. Its focus on problem-solving through creativity and innovation ensures that solutions are tailored to real-world challenges and are continuously improved based on end-user and client feedback.

The V-Model complements this with its structured approach to system development, offering a clear framework for managing requirements, design, and testing. By aligning each phase of development with corresponding verification and validation steps, the V-Model ensures that systems are built and tested iteratively, thus helping manage complexity and mitigating risks.

The sprint structure introduces agility into the methodology, providing a flexible framework

for managing project tasks and adapting to changes quickly. The iterative nature of sprints and the focus on regular reviews and adjustments align well with the evolving insights gained through Design Thinking. Scrums further enhance this agility by breaking work into manageable subcycles, incorporating new knowledge in an effective and timely manner.

By integrating these methodologies, the adopted approach addresses the limitations of each individual method and offers a dynamic, responsive strategy for project management. This cohesive framework ensures that solutions are not only effectively developed but also continuously validated and refined throughout the entire project lifecycle.

3.1.4.2 Methodology Application

In the initial stages of the project, namely the empathise, define, and ideate stages, a sprint and scrum team must be established. The sprint team, consisting of the project client, supervisor, and lead, meets at the end of each week-long sprints for an agile review, assessing the outcomes and setting goals for the next sprint. Meanwhile, the project lead conducts daily scrums, breaking down the sprint into subtasks. These are reviewed daily via a logbook, which accompanies this document. This process ensures that all stakeholders stay informed, and project knowledge and requirements are continuously captured and managed.

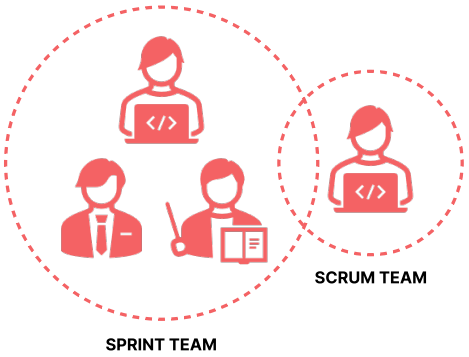

Figure 17: Sprint and scrum teams.

As the project advances into the prototyping and testing phases, the project's purpose and definition should be clearly set, and an MVP agreed upon (see Appendix B). The MVP serves as the baseline for success, with subsequent iterations refining and improving it (Holiday, 2015; p.82). Instead of adhering to a rigid structure, the methodology emphasises iterating through the V-Model to incorporate new knowledge into the testing phases, aligning better with the evolving nature of software development.

Exiting the iterative loop is possible once the MVP is achieved. If not, the project will still have made valuable contributions, following a logical process that future teams can build upon. If the loop is exited, then the project's outcome - service or product - will then go into operation, and the subsequent lifecycle stages of maintenance, upgrades, and end-of-life (EoL).

This methodology has been adopted for this project, and the following sections of this report will outline the activities undertaken at each stage.

3.2 Tools and Techniques

Delivering a project on time and within budget alone is not enough to define it as a success, therefore other tools and techniques aside from the main methodology must be employed to ensure holisitc project success (Wateridge, 1998). Holistic success encompasses factors beyond the core "iron triangle" of time, cost, and scope. Infact, as Joslin (2016) acknowledges, project success is a multidimensional construct in which stakeholders select from various criteria that they feel crucial for evaluating project success, no matter how subjective. Therefore it is more appropriate to expand on the original iron triangle of project management when displaying a project's success criteria, as shown in Figure 18.

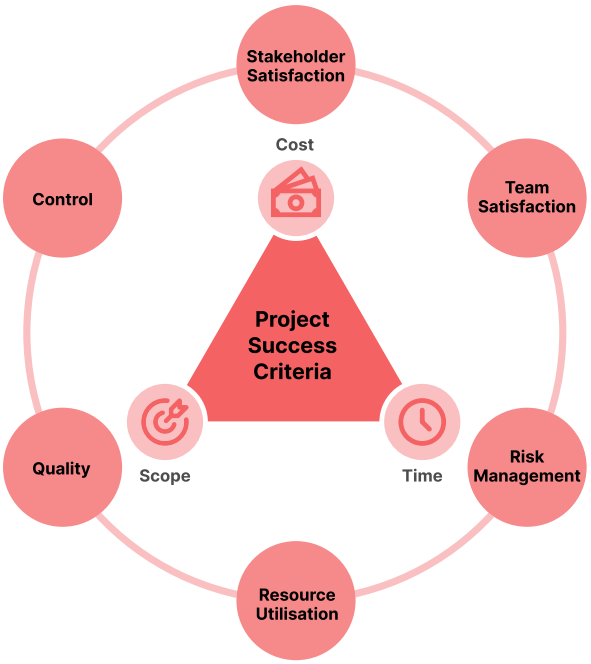

Figure 18: The updated iron triangle of project management [Modified from (Eby, 2023)].

Based on these criteria, specific tools and techniques are applied to manage and achieve the defined metrics effectively. The following sections detail how various project tools and methods were employed to mitigate risks and ensure project success across each criterion outlined in Figure 18.

3.2.1 Time and Scope

Effective management of time and scope was achieved through the use of a project plan and an ongoing logbook, alongside the foundational establishment of the aforementioned MVP. These tools ensured that all project objectives were tracked and met, with regular updates provided to the client on progress and any necessary adjustments. The logbook played a central role in daily and weekly reflection and planning: each day, a mood indicator and mood statement were recorded to capture personal reflections, completed tasks were documented, and key takeaways were noted. Additionally, plans for the following day or days were outlined, ensuring a continuous workflow and proactive task management.

At the end of each week, a comprehensive review was conducted in a similar format, with all completed tasks logged, reflections summarised, and a strategic plan developed for the upcoming week. This review formed the basis of the Monday morning client meetings, where it was used to align upcoming goals with client expectations and provide transparent updates on project progress (see Appendix C for all project meeting minutes). This structured approach enabled clear visibility into the project's trajectory, supporting alignment on expectations and maintaining the project within the agreed-upon scope and timeline.

To illustrate this process further, sample pages from the project logbook, showcasing daily and weekly review entries, are provided in Appendix D. These examples highlight the format used to track daily progress, set immediate goals, and conduct weekly reviews, offering additional insight into the logbook's role in maintaining structured project management and communication.

3.2.2 Quality, Control, and Team Satisfaction

To maintain high standards of quality and control, along with sustaining sprint and scrum team satisfaction, regular communication with the client and project appraiser was prioritised. This included consistent liaison sessions to discuss project progress and ensure requirements were captured and managed throughout the project lifecycle (see Appendix E for requirements).

3.2.3 Stakeholder Satisfaction

It is important in any project to recognise that there will be both aligned and non-aligned stakeholders. *Aligned stakeholders* are those whose goals and interests align with the project's objectives, such as collaborators, end-users, or sponsors who benefit directly from its success. *Non-aligned stakeholders* are those whose interests may diverge or even conflict with the project, such as competitors, regulatory bodies, or groups affected by its implementation.

To address both groups, a stakeholder map and engagement plan (Appendix F) were developed, supported by an analysis of macro-environmental factors (Appendix G) that could impact project success. These tools were used to ensure stakeholder satisfaction by identifying key stakeholders, defining their roles and expectations, and guiding how project information was documented and communicated to meet their needs effectively.

Additionally, a focused case study on wildfires (Appendix A) - the primary use case - was conducted to gain deeper insights into the problem space and refine system requirements. This

multi-faceted approach ensured that the project not only addressed real-world challenges but also added value through system flexibility, scalability, and accessibility.

3.2.4 Risk Management

Risk and uncertainty was proactively addressed through a structured risk analysis and mitigation plan (Appendix H). Potential project risks were identified early, with mitigation strategies in place to prevent and manage issues as they arose. This approach minimised disruptions and ensured that any arising risks could be handled effectively according to the predetermined mitigative action.

3.2.5 Resource Utilisation and Cost

Finally, resource utilisation and cost considerations were managed with efficiency in mind. While cost minimisation was not the primary focus, resources were strategically allocated to reduce waste (both material and time-based) and keep the project on track. A key decision was to select a system-on-a-chip (SoC) that fulfilled the majority, if not all, of the required functionalities. This approach allowed the project to concentrate on the design and optimisation of the routing algorithm rather than diverting effort toward developing a custom hardware system. Sourcing a COTS SoC not only minimised costs and resource demands but also mitigated risks associated with longer lead times and potential delays. Additionally, this approach ensured seamless compatibility with a broad range of sensors, achieving a level of integration and performance likely superior to a custom-built solution. By adopting an established SoC, the nodes adhered to industry standards and benefited from access to high-quality components at a lower cost, enabled by economies of scale.

By addressing these factors through the outlined tools and techniques, the project ensures a well-organised and methodical approach to managing uncertainty, which enhances its chances of success. This includes aligning stakeholder expectations, proactively mitigating risks, and optimising resource allocation to drive efficient progress and meet objectives.

4 System Design

Building on the established MVP, the project has two core objectives. Firstly, it aims to develop an innovative routing algorithm capable of classifying data according to specific requirements and transferring it across the network using the most appropriate of several wireless communication protocols. Secondly, it involves identifying a suitable COTS system, or combination of systems, that can reliably support and enable the required network functionality. The following sections outline the approach and methods used to achieve these objectives, detailing both the design of the routing algorithm and the selection process for appropriate hardware.

4.1 Routing Protocol Design

Through a comprehensive review of the literature and ongoing collaboration with the client, a clear understanding of the system requirements was developed. The literature consistently highlights that energy efficiency is paramount in WSNs, with other factors either supporting or enhancing this objective. This project addresses a critical aspect of the energy challenge by offering data transmission across three distinct wireless communication channels, each with their own strengths and limitations. By aligning the choice of communication channel with the specific needs of the data, the system intelligently optimises power usage while prioritising the urgency and importance of the information being transmitted.

Data routing involves the process of receiving or sensing environmental data, followed by its transmission across the network to a designated destination, typically the sink node. To visually demonstrate the flow of data within this system, refer to Figure 19.

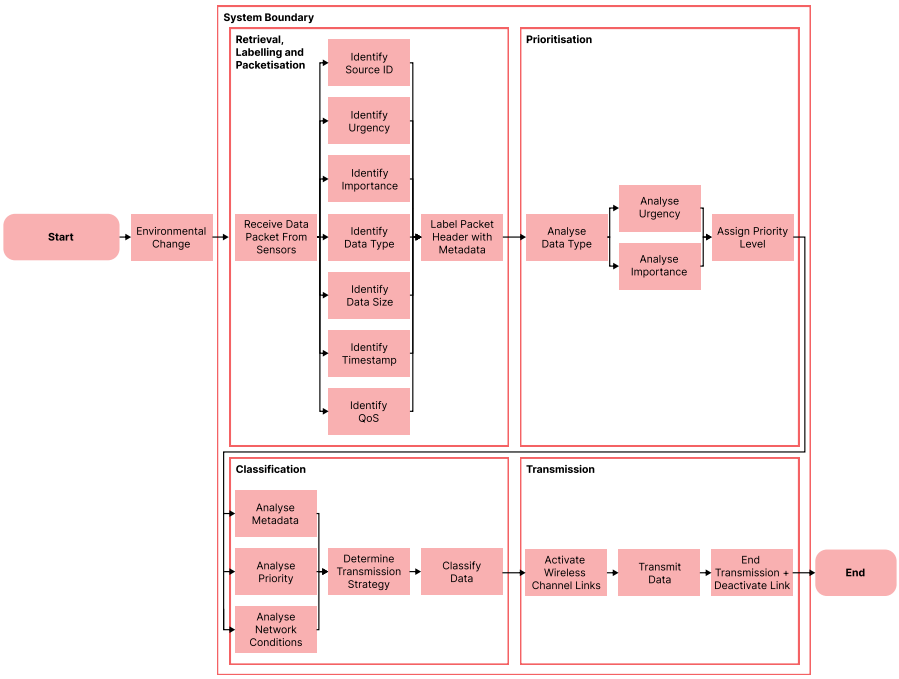

Figure 19: System data flow diagram.

As shown in Figure 19 each node performs local edge processing, which is conducted in the following four key phases:

1. Data Retrieval
2. Labelling and Packetisation
3. Prioritisation and Classification
4. Transmission

The following sections provide a detailed explanation of the specific functions and operations within each of these phases of the routing process.

4.1.1 Data Retrieval

The intial phase of the routing process is the point at which data enters the system. This occurs through the various onboard sensors detecting environmental conditions. To ensure accurate, efficient and timely acquisition of this data, several factors need to be considered, starting from the sensors themselves, to the way in which the algorithm processes this data. The following paragraphs explain the main factors considered in this process.

4.1.1.1 Sensor Selection and Configuration

To fulfill the requirements of both the client and potential use case stakeholders, a variety of sensors are required. These sensors should gather data in real-time, covering environmental (extrinsic), internal (intrinsic) and network parameters. Table 7 shows the parameters suggested to be captured by the system.

Table 7: Parameters to be measured.

Parameter Type	Parameters
Extrinsic	Temperature, pressure, humidity, CO2 concentration, wind speed, video footage and sound.
Intrinsic	Acceleration and velocity, latitude and longitude, battery life, and CPU usage.
Network	Transmission queue length, available bandwidth, signal strength, local node health, data throughput, latency, packet loss rate, link quality, and radio availability.

These parameters have been chosen to meet the specific needs of wildfire monitoring; however, the system should be designed with modularity in mind, allowing for the integration of additional sensors as needed. This flexible approach ensures the system can adapt to various applications beyond its initial scope.

Alongside the types of parameters being measured, the routing algorithm must ensure compatibility with a variety of different sensor interfaces, data types and be designed in a modular fashion for ease of scalability.

4.1.1.2 Data Quality and Integrity

Ensuring the quality and integrity of the data passing through the system is paramount to maintaining system reliability. To mitigate potential sensor errors, calibration processes were conducted for each sensor type, ensuring accurate readings were being retrieved. Additionally, the routing algorithm, where possible, has been designed to have built-in error handling mechansisms to address any anomalous data.

4.1.1.3 Communication and Protocols

The data retrieved from each sensor must be efficiently transferred across the network. This requires selecting the appropriate communication protocols for sensor-to-MCU communication. To streamline integration, a SoC with pre-integrated sensors was prioritised, reducing the need for complex hardware configurations. This allowed the focus to remain on optimising the algorithm to manage communication, rather than diverting resources to address hardware-level challenges.

As such, the system algorithm is required to interface with a range of peripherals (external or onboard devices connected to the MCU) through a range of different wired communication protocols such as I2C, SPI, UART and USB.

4.1.1.4 Power and Resource Constraints

Power remains the main constraint with WSNs due to their wireless nature and reliance on limited resources. As such, the algorithm should aim to minimise the time that components are on and optimise the processes conducted whilst they are on.

4.1.2 Labelling and Packetisation

Once data has been retrieved from the onboard sensors or through communication with other nodes, it needs to be labelled with appropriate metadata, typically in the form of a packet as illustrated in Figure 20.

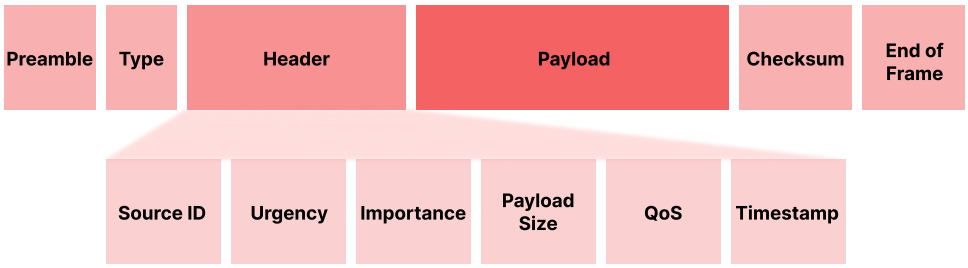

Figure 20: Data packet format.

The metadata contained within the header ensures that each packet is accurately classified and transmitted through the most suitable wireless channel.

The table below highlights the typical size and purpose of each section within these packets.

Table 8: Packet fields, their size and purpose.

Packet Field	Size	Content/Purpose
Preamble	2-4 bytes	Used for synchronisation between sender and receiver (e.g., “0xAA55”).
Header	4-8 bytes	Contains the source and destination addresses, packet length, and protocol-specific flags.
Type	1 byte	Specifies the type of data being transmitted (e.g., “0x01” for temperature, “0x02” for humidity, etc.).
Payload	Variable	The actual sensor data being transmitted (e.g., temperature, humidity, and image data).
Checksum	2 bytes	Error-checking field to ensure data integrity during transmission (e.g., “CRC16”).
End of Frame	1 byte	Marks the end of a packet (e.g., “0xFF”).

It is important to emphasise that the primary goal of this project is to develop a routing algorithm capable of providing service differentiation by selecting the appropriate wireless communication protocol based on the data’s QoS requirements, rather than creating a market-ready product. As such, while hexcode is often employed in commercial systems to optimise packet size, this project opts for simplicity, ease of interpretation, and efficiency within the given timeframe. Instead, short descriptive variables and floats are used to assign qualitative and quantitative values to the data, streamlining development and enhancing clarity.

4.1.3 Prioritisation and Classification

Once the packets have been created, they get passed through the decision engine, from which an urgency and importance score are assigned to the data based on certain predetermined thresholds (made customisable for the user).

When packets enter the decision engine, they are equipped with the following metadata:

- Sensor ID
- Urgency
- Importance
- Data Type
- Data Size
- Timestamp
- Quality of Service (QoS)
- Priority (which a value gets assigned to once processed by the decision engine).

Data is categorised into one of four classes based on these parameters, as shown in Table 9.

Table 9: Data classes.

	Classification/ Data Type			
	Critical	Data Intensive	Reliable	Standard
Delay	Critical	Moderate	Boundless	Boundless
Loss	Intolerant	Low Tolerance	Intolerant	Tolerant
Priority	Highest	High	High Reliability	Best Effort
QoS	Guaranteed Delivery	High Bandwidth	High Reliability	Standard
Wireless Comms. Protocol	Wi-Fi/LoRa (range dependant)	Wi-Fi	BLE	LoRa

Each class dictates a specific transmission strategy, ensuring the selected communication protocol aligns with the data's QoS requirements and system priorities.

4.1.4 Transmission

The following table outlines the transmission strategies for each data class, detailing how the system selects the most suitable communication protocol to meet the defined QoS requirements.

Table 10: Transmission strategy for each data class.

Data Class	Primary Protocol	Fallback Protocol	Additional Notes
Critical	Wi-Fi	LoRa	If all protocols fail, the system retries or uses nearby nodes to relay the message.
Data Intensive	Wi-Fi	LoRa (byte arrays)	Large data split into smaller packets for LoRa if Wi-Fi is unavailable.
Reliable	BLE	Wi-Fi/LoRa	BLE preferred for energy efficiency; Wi-Fi or LoRa used for extended coverage.
Standard	LoRa	None	Buffers data and retransmits at intervals if LoRa fails; minimal impact from data loss.

To complement the transmission strategy table, Figure 21 provides a flow diagram illustrating the generic process for transmitting packets within the system. This visual representation highlights the decision-making pathways and protocol selection stages, offering a clearer understanding of how data moves from classification to transmission.

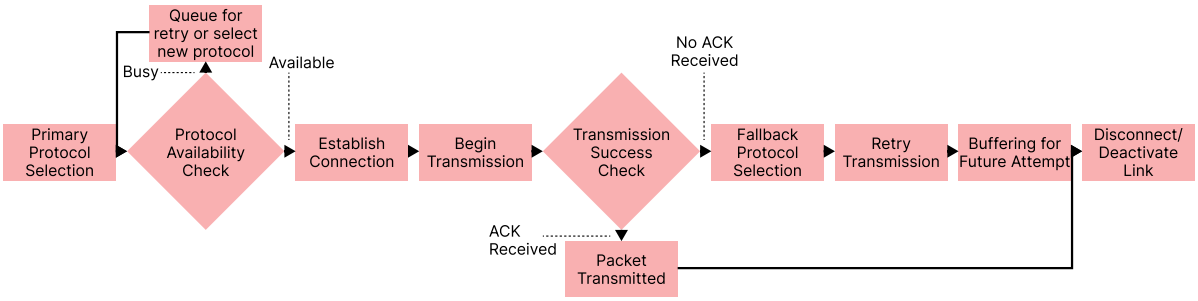

Figure 21: Generic transmission strategy flow diagram.

4.2 Component Selection and Evaluation

The selection of the SoC is critical to ensuring the functionality, performance, and scalability of the WSN. The decision was made to carefully evaluate multiple options to find the most suitable COTS SoC that would meet the system's requirements and allow for efficient implementation of the routing algorithm. The selection process considered the following criteria:

1. Number of wireless communication protocols available
2. SoC functionality and reprogrammability
3. Sensor functionality and integration capability
4. Cost
5. Availability and lead times
6. Conformance to regulations and industry standards
7. Community support and documentation
8. Supported development platforms

After establishing the selection criteria, an evaluation of wireless communication protocols was conducted, focusing on their applicability to the project's requirements. This was followed by an assessment of commercially available SoCs options that supported the identified protocols. The evaluation prioritised how each SoC addressed the criteria listed above, ensuring compatibility with the project's functional, performance, and scalability needs. Detailed comparisons of the communication protocols and SoCs are presented in Appendix I and Appendix J, respectively.

After thorough assessment, the LilyGO T-Beam Supreme (see Figure 22) emerged as the optimal choice for this project. This decision was based on its extensive feature set and how well it aligned with the project's requirements, which includes multiple communication protocols, pre-integrated sensors, and several supported development platforms. The subsequent sections provide a detailed overview of the device and demonstrate how its capabilities meet the established selection criteria.

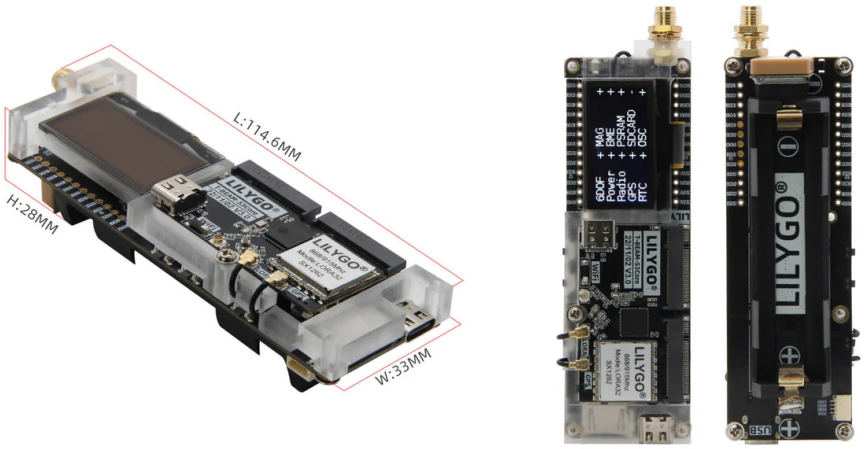

Figure 22: LilyGO T-Beam Supreme (LilyGO, 2024).

4.2.1 Node Specification: LilyGO T-Beam Supreme

The LilyGO T-Beam Supreme has been selected as the SoC for all nodes due to its comprehensive feature set, which meets the requirements of this project (full specification can be found in Appendix K).

These include integrated GPS, long-range LoRa communication, and dual high-performance wireless protocols: Wi-Fi and Bluetooth 5.0 Low Energy (BLE). Additionally, it provides 8MB of onboard flash memory, which supports fast processing, and an integrated micro-SD card reader for handling larger data sets.

At the heart of the LilyGO T-Beam Supreme is the ESP32-S3 microcontroller, which is built around a dual-core processor operating at 240MHz. Its low-power design helps in prolonging the battery life of the device, making it ideal for applications requiring long operational periods, such as disaster management.

Key features of the ESP32-S3 include:

- Dual-core Xtensa LX7 processor (240MHz)
- 512KB SRAM (expanded by an 8MB PSRAM in the LilyGO T-Beam Supreme)
- 8MB flash memory
- 2.4GHz Wi-Fi (IEEE 802.11 b/g/n)
- Bluetooth 5.0 Low Energy (BLE)
- GPIO Pins: including UART, I2C, SPI, and ADC peripherals for connecting various modules

The LoRa communication module (SX1262) provides reliable, long-range communication in the 868MHz ISM band, supporting low-power and extended-range communication between nodes. Due to the nature of the data expected to be retrieved by the system, such as general telemetry and lower-priority information, this communication channel is ideal for transmitting data over longer distances with minimal power consumption.

Other notable features of the LilyGO T-Beam Supreme include:

- L76K global navigation satellite system (GNSS) module
- AXP2101 power management unit (PMU)
- Battery holder
- QMI8658 inertial measurement unit (IMU) - triaxial accelerometer and gyroscope
- Air pressure sensor
- Magnetometer
- Real-time clock (RTC)
- USB-C Interface

The integration of these components simplifies system hardware development, reducing the need for separate external modules, which streamlines the overall design and shortens the development timeline.

4.2.1.1 Product Compliance

The LilyGO T-Beam Supreme is CE certified for EMC and RF transmission compliance. The CE marking ensures that the device adheres to all applicable EU regulations for safe operation in various environments, including wireless communication and electromagnetic emissions. The CE Attestation of Conformity (AoC), unlike a Declaration of Conformity, is conducted by a third-party accreditation organisation to verify that the product conforms to the essential requirements outlined in the relevant EU directives, such as:

- Radio Equipment Directive (RED) 2014/53/EU, covering the compliance of RF components such as the Wi-Fi, Bluetooth, and LoRa modules.
- Electromagnetic Compatibility (EMC) Directive 2014/30/EU, ensuring that the device does not emit harmful electromagnetic interference.

The CE compliance testing for the LilyGO T-Beam Supreme was conducted by Bay Area Compliance Laboratories Corp. (BACL) and the AoCs can be found in Appendix L.

4.2.2 Selection Criteria Fulfillment

The LilyGO T-Beam Supreme was selected due to its ability to meet all the essential criteria, as shown in Table 11.

Table 11: Node device comparison against selection criteria.

Criteria	Device Alignment
1	The LilyGO T-Beam Supreme supports Wi-Fi, Bluetooth 5.0 Low Energy (BLE), and LoRa. This range of protocols ensures versatility in wireless communication matching that of the system requirements.
2	The device is powered by the ESP32-S3, a dual-core processor operating at 240MHz, with 512KB SRAM and 8MB flash memory. It supports reprogramming via development platforms such as the Thonny and Arduino IDE, allowing for flexible and easy modifications to meet project needs.
3	The LilyGO T-Beam Supreme integrates several sensors, including a GPS module (L76K), an IMU (QMI8658), an air pressure sensor, and a magnetometer. These sensors are directly integrated into the board, reducing the need for additional components and simplifying the system design.
4	The LilyGO T-Beam Supreme offers a cost-effective solution given its comprehensive feature set, making it suitable for rapid prototyping. The overall price is competitive compared to other SoCs with similar capabilities.
5	The LilyGO T-Beam Supreme is readily available from various online suppliers, with relatively quick lead times for orders. This makes it a viable option for development with minimal delays.
6	The device is CE certified, ensuring compliance with EMC and RF transmission regulations, including the Radio Equipment Directive and Electromagnetic Compatibility Directive.
7	The LilyGO T-Beam Supreme is supported by a diverse community, with resources available on platforms like GitHub, user forums, and official datasheets. However, this project aims to further enhance and contribute to this support.
8	The device supports popular development platforms, including Thonny, the Arduino IDE and PlatformIO, facilitating the integration of custom code and modules in various programming languages.

Evaluation of the LilyGO T-Beam Supreme against the selection criteria confirms its suitability as the selected hardware for implementing the algorithm, which will be described in the following section.

5 Implementation and Testing

5.1 System Implementation

The implementation phase centers on transforming the LILYGO T-Beam Supreme from a device originally constrained by Meshtastic firmware (a platform designed for mesh communication but limited in flexibility) into a fully configurable system node serving as the core hardware within the WSN. This transformation involved replacing the default firmware with a MicroPython-based solution, offering greater adaptability and accessibility. By utilising the Thonny IDE, the system was programmed to establish a user-friendly platform capable of executing the custom routing algorithm. This section details the implementation process, highlights the rationale behind key decisions, and concludes with testing aligned with the V-model approach adopted in the project's methodology.

5.1.1 Reprogramming the LILYGO T-Beam Supreme

The LILYGO T-Beam Supreme comes pre-installed with Meshtastic firmware, designed for simple mesh networking applications. However, this project required a more versatile and customisable platform to support the broader objectives, including the development of a tailored routing algorithm. To achieve this, the device was transitioned to an ESP32-based MicroPython framework, providing a flexible foundation for system development.

This shift offered several key advantages, such as MicroPython being an accessible, beginner-friendly language with an interactive REPL (Read-Eval-Print Loop) for real-time debugging. Additionally, MicroPython's extensive documentation and simplicity enable rapid development, making it an ideal choice for this project's fast-paced implementation. However, community support for configuring the LILYGO T-Beam Supreme with MicroPython proved limited.

To address this gap, a comprehensive user guide has been developed and is available on GitHub at this link:

GitHub User Guide: https://github.com/Pythons-And-Ladders/the-coke-can-project/.

Together with this document and its appended resources, the guide serves as a primary reference, offering essential support for anyone aiming to expand or build upon this project's foundation.

5.1.2 Development Environment and Tools

For the LILYGO T-Beam Supreme devices to perform as intended, several additional components and a robust IDE capable of interpreting MicroPython are necessary. The prerequisites for this project are detailed in Table 12, including the specific models used at this stage, along with hyperlinks for easy access.

Table 12: Prerequisites (per node).

Hardware/Software	Specific Model Used (Hyperlinked)
TTGO T-Beam Supreme	TTGO T-Beam Supreme (868MHz) Meshtastic with L76K GNSS
USB to USB-C Data Cable	Amazon Basics Data Cable
2.4GHz Antenna	The PiHut 2.4GHz WiFi/BLE Antenna With U.FL Adapter
Flat Top 18650 Rechargable Battery	Aukido 3.7V 3500mAh NiMH Battery
Development Platform (Laptop)	HP ProBook 450 G7
Thonny IDE (latest stable version)	thonny-4.1.6.exe for Windows 64-bit (Python 3.0)

The IDE selected for reprogramming the devices into flexible, MicroPython-based nodes was Thonny IDE. Thonny was chosen due to its beginner-friendly interface, which simplifies the process of writing, testing, and debugging MicroPython scripts. Its built-in support for MicroPython includes automatic detection of connected devices via COM ports, an interactive REPL for real-time feedback, and an intuitive layout, making it an excellent choice for rapid development and efficient code deployment.

The full guide on how to install and setup the development platform and flash the MicroPython firmware can be found on the main README.md file on the GitHub page.

5.2 Testing

The adopted methodology follows the V-Model for the testing phases, aligning development stages with corresponding validation phases. This section covers unit testing, which ensure successful re-establishing of MCU-peripheral communication using the MicroPython-based solutions; subsystem testing, which verifies the functionality integrated peripheral edge processing; and system testing, which assesses the WSN and routing protocol. Full test documentation is in Appendix M.

5.2.1 Unit Testing

Unit testing focused on validating the functionality of individual components and their interactions with the system. Specifically, this phase involved testing the onboard peripherals, such as the GNSS module, PMU and IMU. Each peripheral was tested independently to ensure it was properly initialised, configured, and able to interact correctly with the main system. Test reports for the unit testing phase are provided in Appendix M.2, where detailed results from each test are included for reference.

During the unit testing phase, some peripheral functionalities did not work as expected on initial tests. These issues were addressed through version control, with the code being iteratively improved. As testing progressed, newfound knowledge was incorporated to resolve problems and refine the system, ensuring functionality was achieved.

Despite these efforts, certain peripherals were either not fully tested within the project's timeframe due to deadlines, or were deferred for future development. The table below indicates the peripherals that were successfully interfaced with, as well as those that were not integrated at the current stage of the project.

Table 13: Peripheral interface status.

Peripheral	Interface Status
ESP32-S3 MCU	Interfaced
SH1106 OLED Display	Interfaced
BME280 Sensor	Interfaced
SX1262 LoRa Transceiver Module	Interfaced
WiFi Transceiver	Interfaced
BLE Transceiver	Interfaced
AXP2101 PMU	Interfaced
L76K GNSS Module	Interfaced
MicroSD Card Reader	Interfaced
QMI8658 IMU	NOT Interfaced

Additionally, several key findings emerged during unit testing, including the discovery of incorrect pin assignments in the manufacturer's pinmap and the necessity of powering the UART bus to interface properly with the GNSS module. These findings, along with their corrective measures, are summarised in the unit testing overview table (Appendix M.1).

5.2.2 Subsystem Testing

Subsystem testing focused on verifying the integration of the different peripheral components that have been interfaced with within the unit testing phase. This phase particularly focuses on the decision engine and edge processing conducted individually by each node.

The verification of these codes involved ensuring that the decision-making logic - encapsulating data collection, processing, and transmission - was functioning correctly. The test code developed for subsystem testing specifically processes data from the following onboard peripherals: the GNSS module, BME280 module, and AXP2101 PMU.

The code is designed with a modular structure, allowing for seamless integration of additional peripherals in future expansions of this project. This ensures adaptability and simplifies the process of extending the edge processing framework.

The code used and tested is both provided in Appendix M.3 and can be found on the GitHub user guide.

5.2.3 System Testing

At this stage of the project, further subsystem testing is necessary before progressing to system testing, which has not yet been conducted. System testing will involve evaluating the WSN, comprising multiple nodes running the validated edge processing framework, to ensure efficient data routing across the network.

Following functional testing, the system will be ready for optimisation, focusing on key factors such as energy efficiency, reliability of data transfer, security, and adaptability to various scenarios. One way performance can be better optimised is through the reprogramming of the nodes to utilise both cores of the onboard MCUs. Employing techniques such as multithreading or multiprocessing would enable each node to simultaneously receive and transmit data, improving communication efficiency within the network.

Before deployment, each node should undergo rigorous physical testing. Appropriate, robust, and environmentally friendly weatherproof casing must be designed to meet the necessary IPX ratings and durability standards for the specific use case.

6 Results and Discussion

This section reflects on the outcomes of the project relative to its primary aim: to conceptualise the edge processing framework for a dynamic routing protocol and implement an the framework onto a COTS device. Additionally, the tools, techniques, and performance of the adopted methodology are critically evaluated, with future work identified to address limitations and expand system functionality.

6.1 Comparison with Initial Objectives

The project successfully met its initial objectives. The edge processing framework was designed to dynamically classify and route data according to its QoS requirements, selecting the appropriate communication protocol in line with varying network conditions. A suitable COTS device was identified, providing a scalable and cost-effective platform for implementing the framework onto.

Further to the early completion of these goals, the project was extended to include practical implementation on the selected COTS device. This demonstrated the feasibility of using commercial hardware to achieve sophisticated data handling, thereby validating the framework's practical application. However, system-wide testing and optimisation remains an area for future work, as the current evaluation primarily focused on unit and subsystem functionality.

6.2 Evaluation of Methodology, Tools, and Techniques

The adopted methodology, integrating Design Thinking, the V-Model, and Sprint Scrum techniques, created a robust and iterative approach to managing the complexity of this project. This combination of methodologies provided a flexible, user-centreed framework for development, while allowing for continuous validation and refinement through the project lifecycle. Although the project was successful in delivering the required outcomes to satisfy the client, it is important to critically evaluate the strengths and weaknesses of the methods used to achieve this.

6.2.1 Strengths and Effectiveness

The following strengths highlight the areas in which the methodology proved particularly effective:

- The iterative Sprint Scrum cycles allowed the project to accommodate the evolving requirements of the client, especially as the scope expanded from the MVP to include practical implementation. Weekly reviews ensured alignment within the sprint team, mitigating the risk of scope creep.
- Design Thinking ensured the system met real-world needs by incorporating stakeholder needs through the development of user stories, particularly in the define and ideate stages, refining the project's focus.
- The V-Model provided a clear framework for aligning development with validation, ensuring the edge processing framework was robust, well-tested, and aligned with the

overarching system requirements.
- The logbook entries and meeting minutes facilitated continuous tracking of progress, goal setting, and motivation, ultimately ensuring effective recording of the work completed throughout the project.

6.2.2 Limitations and Challenges

The following points outline the challenges encountered with the adopted methodology, which could potentially be addressed and mitigated in future projects.

- Testing was constrained to functional and conducted within a small-scale environment. This limited the ability to evaluate performance and scalability in large-scale deployments involving multiple nodes.
- The methodology relied on manufacturer-provided calibrations for sensors, assuming accuracy without further verification. This could affect performance under specific conditions, highlighting the need for additional calibration and verification steps in future implementations.
- The system was not tested in environments comparable to its intended use, such as disaster-prone or high-interference areas, limiting validation of its resilience under real-world conditions.

6.3 Future Work

Several opportunities for further development and refinement have been identified:

- Expand and test subsystem functionality by integrating more peripherals for well-rounded node capabilities.
- Perform comprehensive system-level functional testing.
- Optimise the routing algorithm for efficiency and scalability.
- Incorporate mesh-networking capabilities to improve system redundancy.
- Implement advanced security protocols to safeguard data transmission.
- Carry out physical deployment and rigorous environmental testing.
- Integrate machine learning to enable adaptive responses to changing network conditions.

By addressing these areas, the project can evolve from a proof of concept into a fully operational system, ready for deployment in disaster response and other critical scenarios.

7 Conclusion

7.1 Project Conclusion

In conclusion, the project successfully met both its primary and secondary objectives, including the development and implementation of a dynamic edge processing framework on a COTS device. The integration of various methodologies, such as Design Thinking, the V-Model, and the iterative Sprint Scrum cycles, provided a balanced approach that was both structured but also flexible, allowing for agile changes according to the evolving project needs. The use of these methods alongside the aforementioned tools and techniques culminated in a system capable of dynamic data processing, classification and transmission. Thus, demonstrating the feasibility of using commercial hardware for complex real-time operations.

While the project validated the framework's potential, limitations in testing environments and scalability underscore the need for further refinement. Future work should focus on comprehensive system-level testing, algorithm optimisation, and real-world deployment to solidify the system's operational readiness.

7.2 Personal Conclusion and Lessons Learnt

This project provided invaluable experience in managing complex, multidisciplinary tasks within a dynamic development framework.

Key lessons include:

- Regular iterative reviews, facilitated through the logbook and client meetings, ensured alignment with project goals, highlighting the importance of adaptive planning and ongoing feedback in achieving success.
- The transition from theoretical design to practical implementation proved challenging, demonstrating the complexities involved in real-world application beyond conceptual exercises.
- Defining an MVP provided a clear and achievable project goal, focusing efforts on delivering essential functionality first and refining the system in subsequent iterations.
- The integration of structured methodologies with agile practices underscored the importance of customising the approach to fit the specific project needs.

These lessons will inform future endeavors, particularly in enhancing project management skills, optimising technical solutions, and fostering collaborative and consultative innovation.

References

Adeel, A. et al. (2019), A survey on the role of wireless sensor networks and iot in disaster management, *in* T. Durrani, W. Wang and S. Forbes, eds, 'Geological Disaster Monitoring Based on Sensor Networks', Springer, Singapore, pp. 57–66.

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y. and Cayirci, E. (2002), 'Wireless sensor networks: A survey', *Computer Networks* **38**, 393–422.

Akyildiz, I. F. and Vuran, M. C. (2010), *Wireless sensor networks*, John Wiley & Sons.

Alexander, D. E. (2002), *Principles of Emergency Planning and Management*, 4th edn, Oxford University Press, Oxford.

Algamili, A., Khir, M., Dennis, J. and et al. (2021), 'A review of actuation and sensing mechanisms in mems-based sensor devices', *Nanoscale Research Letters* (16).
URL: *https://doi.org/10.1186/s11671-021-03481-7*

Amazon (2024), 'Meet Alexa'.
URL: *https://www.amazon.co.uk/b?ie=UTF8&node=12728352031*

Asif, M., Khan, S., Ahmad, R., Sohail, M. and Singh, D. (2017), 'Quality of service of routing protocols in wireless sensor networks: A review', *IEEE Access* **5**, 1846–1871.
URL: *10.1109/ACCESS.2017.2654356*

Atzori, L., Iera, A. and Morabito, G. (2010), 'The internet of things: A survey', *Computer Networks* **54**, 2787–2805.

Bai, Y., Du, W., Ma, Z., Shen, C., Zhou, Y. and Chen, B. (2010), Emergency communication system by heterogeneous wireless networking, *in* 'International Conference on Wireless Communications, Networking and Information Security (WCNIS)', IEEE.

Baird, A., O'Keefe, P., Westgate, K. and Wisner, B. (1975), *Towards an Explanation and Reduction of Disaster Proneness*, Disaster Research Unit, University of Bradford, Bradford.

Bartoli, G., Fantacci, R., Gei, F., Marabissi, D. and Micciullo, L. (2015), 'A novel emergency management platform for smart public safety', *International Journal of Communication* **28**(5), 928–943.

Carli, M., Panzieri, S. and Pascucci, F. (2014), 'A joint routing and localization algorithm for emergency scenario', *Ad Hoc Networks* **13**, 19–33.

Castillo-Effer, M., Quintela, D., Moreno, W., Jordan, R. and Westhoff, W. (2004), Wireless sensor networks for flash-flood alerting, *in* 'Proceedings of the Fifth IEEE International Caracas Conference on Devices, Circuits and Systems, 2004.', Vol. 1, pp. 142–146.
URL: *http://dx.doi.org/10.1109/ICCDCS.2004.1393370*

Chandna, M. and Singla, B. (2015), 'Comparative analysis of flooding and gossiping in wireless sensor networks using sir', *International Journal of Computer Science and Information Technologies* (4), 4020–4023.
URL: *https://api.semanticscholar.org/CorpusID:201073963*

Chen, D., Liu, Z., Wang, L., Dou, M., Chen, J. and Li, H. (2013), 'Natural disaster monitoring with wireless sensor networks: A case study of data-intensive applications upon low-cost scalable systems', *Mobile Networks and Applications* **18**(5), 651–663.

Chipara, O., Lu, C., Bailey, T. C. and Roman, G.-C. (2009), 'Reliable patient monitoring: A clinical study in a step-down hospital unit', *All Computer Science and Engineering Research* (wucse-2009-82).
URL: *https://openscholarship.wustl.edu/cse_research/33*

Civil Contingencies Secretariat (2013), 'The Role of Local Resilience Forums'.

Department for Business and Trade (2024), 'Using the UKCA marking'.
URL: *https://www.gov.uk/guidance/using-the-ukca-marking*

Dupuy, J.-L., Fargeon, H., Martin-StPaul, N., Pimont, F. and Ruffault, J. (2020), 'Climate change impact on future wildfire danger and activity in southern europe: A review', *Annals of Forest Science* **77**(35), 1–25.

Eby, K. (2023), 'Hone Your Skills With This Guide to Project Success Criteria'.
URL: *https://www.smartsheet.com/content/project-success-criteria*

Elhoseny, M., Farouk, A., Batle, J., Shehab, A. and Hassanien, A. E. (2017), *Secure Image Processing and Transmission Schema in Cluster-Based Wireless Sensor Network Secure Image Processing and Transmission Schema in Cluster-Based Wireless Sensor Network*.
URL: *10.4018/978-1-5225-2229-4.ch045*

Erdelj, M., Król, M. and Natalizio, E. (2017), 'Wireless sensor networks and multi-uav systems for natural disaster management', *Computer Networks* **124**.
URL: *10.1016/j.comnet.2017.05.021*

Erman, A. T., v. Hoesel, L., Havinga, P. and Wu, J. (2008), 'Enabling mobility in heterogeneous wireless sensor networks cooperating with uavs for mission-critical management', *IEEE Wireless Communications* **15**(6), 38–46.

European Parliament (2014), 'Directive 2014/53/EU (Radio Equipment Directive)'.
URL: *https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32014L0053*

European Parliament (2024), 'Regulation (EU) 2016/679 (General Data Protection Regulation)'.
URL: *https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679*

Faludi, R. (2011), *Building Wireless Sensor Networks: A Practical Guide to the ZigBee Mesh Networking Protocol*, O'Reilly, chapter Up and Running.

Forsberg, K. and Mooz, H. (1991), The relationship of system engineering to the project cycle, *in* 'Proceedings of the First Annual NCOSE Conference, NY, USA: Wiley', pp. 57–65.

Fragkiadakis, A., Askoxylakis, I., Tragos, E. and Verikoukis, C. (2011), 'Ubiquitous robust communications for emergency response using multi-operator heterogeneous networks', *EURASIP Journal on Wireless Communications and Networking* pp. 1–16.

Fried, J. S., Torn, M. S. and Mills, E. (2004), 'The impact of climate change on wildfire severity: A regional forecast for northern california', *Climatic Change* **64**, 169–191.

Frigerio, S., Schenato, L., Bossi, G., Cavalli, M., Mantovani, M., Marcato, G. and Pasuto, A. (2014), 'A web-based platform for automatic and continuous landslide monitoring: The rotolon (eastern italian alps) case study', *Computers and Geosciences* **63**, 96–105.

Fujiwara, T. and Watanabe, T. (2005), 'An ad hoc networking scheme in hybrid networks for emergency communications', *Ad Hoc Networks* **3**(5), 607–620.

Gao, T., Greenspan, D., Welsh, M., Juang, R. and Alm, A. (2005), Vital signs monitoring and patient tracking over a wireless network, *in* '2005 IEEE Engineering in Medicine and Biology 27th Annual Conference', pp. 102–105.
URL: *https://doi.org/10.1109/IEMBS.2005.1616352*

George, S. M., Misra, S., Xue, G. and Krunz, M. (2010), 'Distressnet: A wireless ad hoc and sensor network architecture for situation management in disaster response', *IEEE Communications Magazine* pp. 128–136. Issue: December.

Google (2024), 'What is Google Home'.
URL: *https://home.google.com/intl/en_uk/what-is-google-home/*

Grover, J., Shikha and Sharma, M. (2014), Location based protocols in wireless sensor network — a review, *in* 'Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT)', pp. 1–5.
URL: *10.1109/ICCCNT.2014.6962990*

Haque, M., Ahmad, T. and Imran, M. (2018), *Review of Hierarchical Routing Protocols for Wireless Sensor Networks*, pp. 237–246.
URL: *10.1007/978-981-10-5523-2_22*

Harmonic Software Systems (2015), 'The Death Of The V-Model'.
URL: *https://harmonicss.co.uk/hss/the-death-of-the-v-model/*

Heath, R. L. and O'Hair, H. D. (2009), *Handbook of Risk and Crisis Communication*, 1st edn, Routledge, New York.

HM Government (2023), 'The Third National Adaptation Programme (NAP3) and the Fourth Strategy for Climate Adaptation Reporting'.
URL: *https://www.gov.uk/government/publications/the-third-national-adaptation-programme-nap3-and-the-fourth-strategy-for-climate-adaptation-reporting*

Holiday, R. (2015), *The Obstacle Is The Way: The Ancient Art of Turning Adversity to Advantage*, Profile Books Ltd.

Horsley, J. S. (2015), The history of disaster communication: The untold story, *in* 'The Proceedings of the International History of Public Relations Conference 2015', Bournemouth University.

IABG (1992), The V-Model Development Standard for Software and System Development (Version 1), Technical report, Federal Ministry of Defense, Germany. Initial V-Model release for German federal projects created in cooperation with the Federal Office for Defense Technology and Procurement (BWB).

Intergovernmental Panel on Climate Change (IPCC) (2023*a*), 'Climate Change 2023: Synthesis Report'.
URL: *https://www.ipcc.ch/report/ar6/syr/*

Intergovernmental Panel on Climate Change (IPCC) (2023*b*), 'IPCC Sixth Assessment Report: Summary for Policymakers'.
URL: *https://www.ipcc.ch/report/ar6/syr/downloads/report/IPCC_AR6_SYR_SPM.pdf*

Joslin, R. (2016), 'The impact of project methodologies on project success in different project environments', *International Journal of Managing Projects in Business* **9**(2), 364–388.
URL: *https://doi.org/10.1108/IJMPB-03-2015-0025*

Kandris, D., Nakas, C., Vomvas, D. and Koulouras, G. (2020), 'Applications of wireless sensor networks: An up-to-date survey', *Applied System Innovation* **3**(1).
URL: *https://www.mdpi.com/2571-5577/3/1/14*,

Kim, D.-S. and Tran-Dang, H. (2019), *Wireless Sensor Networks for Industrial Applications*, Springer International Publishing, pp. 127–140.
URL: *https://doi.org/10.1007/978-3-030-04927-0_10*,

Kumar, V., Rus, D. and Singh, S. (2004), 'Robot and sensor networks for first responders', *IEEE Pervasive Computing* **3**(4), 24–33.

Kureshi, I., Theodoropoulos, G., Mangina, E., O'Hare, G. and Roche, J. (2015), Towards an info-symbiotic decision support system for disaster risk management, *in* '19th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)', IEEE/ACM.

Kurose, J. F. and Ross, K. W. (2000), *Connection-Oriented Transport: TCP*, Addison Wesley, chapter Transport Layer: Overview of the Transport Layer in the Internet.

Laituri, M. and Kodrich, K. (2008), 'On line disaster response community: People as sensors of high magnitude disasters using internet gis', *Sensors* **8**, 3037–3055.

Landaluce, H., Arjona, L., Perallos, A., Falcone, F., Angulo, I. and Muralter, F. (2020), 'A review of iot sensing applications and challenges using rfid and wireless sensor networks', *Sensors* **20**, 2495.
URL: *10.3390/s20092495*

Lewis, F. L. (2004), *Wireless Sensor Networks*, John Wiley & Sons, Ltd, chapter 2, pp. 11–46.
URL: *https://doi.org/10.1002/047168659X.ch2*

Li, L. and Goodchild, M. F. (2010), 'The role of social networks in emergency management: A research agenda', *International Journal of Information Systems for Crisis Response and Management* **2**, 49–59.

LilyGO (2024), 'T-Beam SUPREME Meshtastic', Online.
URL: *https://lilygo.cc/products/t-beam-supreme-meshtastic?variant=43067943944373*

Liu, Y., Goodrick, S. L. and Stanturf, J. A. (2013), 'Future u.s. wildfire potential trends projected using a dynamically downscaled climate change scenario', *Forest Ecology and Management* **294**, 120–135.

Lockwood, T. (2010), *Design thinking: Integrating innovation, customer experience, and brand value*, Simon and Schuster.

Lorincz, K., Malan, D., Fulford-Jones, T., Nawoj, A., Clavel, A., Shnayder, V., Mainland, G., Welsh, M. and Moulton, S. (2004), 'Sensor networks for emergency response: challenges and opportunities', *IEEE Pervasive Computing* **3**(4), 16–23.
URL: *https://doi.org/10.1109/MPRV.2004.18*

Mahmood, M. A., Seah, W. K. and Welch, I. (2015), 'Reliability in wireless sensor networks: A survey and challenges ahead', *Computer Networks* **79**, 166–187.
URL: *https://www.sciencedirect.com/science/article/pii/S1389128614004708*

Marick, B. (1999), 'New models for test development', *Testing Foundations* .

Met Office (n.d.), 'HadCM3: Met Office Climate Prediction Model'.
URL: *https://www.metoffice.gov.uk/research/approach/modelling-systems/unified-model/climate-models/hadcm3*

Minh, Q., Nguyen, K., Borcea, C. and Yamada, S. (2014), 'On-the-fly establishment of multi-hop wireless access networks for disaster recovery', *IEEE Communications Magazine* **52**(10), 60–66.

Mohr, C. (2018), 'Clara Barton', International Committee of the Red Cross.
URL: *https://blogs.icrc.org/cross-files/clara-barton/*

Mori, A., Okada, H., Kobayashi, K., Katayama, M. and Mase, K. (2015), Construction of a node-combined wireless network for large-scale disasters, *in* '12th Annual IEEE Consumer Communications and Networking Conference (CCNC)', IEEE.

Mosterman, P. J., Sanabria, D. E., Bilgin, E., Zhang, K. and Zander, J. (2014), 'A heterogeneous fleet of vehicles for automated humanitarian missions', *Computing in Science and Engineering* **16**(3), 90–95.

Mouftah, H. T., Erol-Kantarci, M. and Rehmani, M. H. (2019), *Wireless Sensor Networks in Smart Cities: Applications of Channel Bonding to Meet Data Communication Requirements*, Wiley Telecom, pp. 247–268.
URL: *https://doi.org/10.1002/9781119360124.ch9*

NASA (2024), 'Global Temperature'.
URL: *https://climate.nasa.gov/vital-signs/global-temperature/?intent=121*

National Fire Chiefs Council (2024), 'Wildfires Position Statement'.
URL: *https://nfcc.org.uk/our-services/position-statements/wildfires-position-statement/*

Ochoa, S. F. and Santos, R. (2015), 'Human-centric wireless sensor networks to improve information availability during urban search and rescue activities', *Information Fusion* **22**, 71–84.

Ofcom (2023), 'IR 2030 – UK Interface Requirements 2030: Licence Exempt Short Range Devices (SRDs)'.
URL: *https://www.ofcom.org.uk/siteassets/resources/documents/spectrum/interface-requirements/ir-2030.pdf?v=335258*

Okubo, P. G., Jennifer, N. S. and Koyanagi, R. Y. (2014), The evolution of seismic monitoring systems at the hawaiian volcano observatory, Technical report, U.S. Geological Survey.

O'Neill, S. (2023), 'Everything you need to know about Amazon Sidewalk'.
URL: *https://www.aboutamazon.com/news/devices/everything-you-need-to-know-about-amazon-sidewalk*

Palenchar, M. J. (2009), *Historical Trends of Risk and Crisis Communication*, Routledge, pp. 31–52.

Pogkas, N., Karastergios, G. E., Antonopoulos, C. P., Koubias, S. and Papadopoulos, G. (2007), 'Architecture design and implementation of an ad-hoc network for disaster relief operations', *IEEE Transactions on Industrial Informatics* **3**(1), 63–72.

Raghavendra, C. S., Sivalingam, K. M. and Znati, T. (2006), *Wireless sensor networks*, Springer, chapter Transport Layer.

Ritchie, H. and Rosado, P. (2022), 'Natural Disasters', Our World in Data.
URL: *https://ourworldindata.org/natural-disasters*

Sardouk, A., Mansouri, M., Merghem-Boulahia, L., Gaïti, D. and Rahim-Amoud, R. (2010), Multi-agent system based wireless sensor network for crisis management, *in* 'Global Telecommunications Conference (GLOBECOM 2010)', IEEE.

Scanlon, T. J. (2023), 'Rewriting a living legend: Researching the 1917 halifax explosion', *International Journal of Mass Emergencies & Disasters* **15**, 147–178.

Schwaber, K. and Sutherland, J. (2020), The scrum guide. the definitive guide to scrum: The rules of the game, Technical report, ScrumGuides.

Shaikh, F. K. and Zeadally, S. (2016), 'Energy harvesting in wireless sensor networks: A comprehensive review', *Renewable and Sustainable Energy Reviews* **55**, 1041–1054.
URL: *https://doi.org/10.1016/j.rser.2015.11.010*

Silva, B. N., Han, K. and Han, D. (2014), 'Disaster management: Evolution of emergency communication and iot-based approaches', *International Journal of Computer Networks and Communications* **6**(4), 1–15.

Simon, G., Maróti, M., Lédeczi, A., Balogh, G., Kusy, B., Nádas, A., Pap, G., Sallai, J. and Frampton, K. (2004), Sensor network-based countersniper system, *in* 'Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems', SenSys '04, Association for Computing Machinery, New York, NY, USA, p. 1–12.
URL: *https://doi.org/10.1145/1031495.1031497*

Simon, H. (1996), *The Sciences of the Artifical*, 3rd edn, MIT Press.

Sisinni, E., Saifullah, A., Han, S., Jennehag, U. and Gidlund, M. (2018), 'Industrial internet of things: Challenges, opportunities, and directions', *IEEE Transactions on Industrial Informatics* **14**(11), 4724–4734.
URL: *https://doi.org/10.1109/TII.2018.2852491*

Tuna, G., Gungor, V. and Gulez, K. (2014), 'An autonomous wireless sensor network deployment system using mobile robots for human existence detection in case of disasters', *Ad Hoc Networks* **13**, 54–68.

Ueyama, J., Freitas, H., Faical, B. S., Filho, G. P., Fini, P., Pessin, G., Gomes, P. H. and Villas, L. A. (2014), 'Exploiting the use of unmanned aerial vehicles to provide resilience in wireless sensor networks', *IEEE Communications Magazine* **52**(12), 81–87.

UK Health Security Agency (2023), 'Health Effects of Climate Change (HECC) in the UK: 2023 report. Chapter 10. Wildfires and health'.
URL: *https://assets.publishing.service.gov.uk/media/65705b7a7391350013b03bbc/HECC-report-2023-chapter-10-wildfires.pdf*

UK Parliament (1974), 'Health and Safety at Work etc. Act 1974'.
URL: *https://www.legislation.gov.uk/ukpga/1974/37*

UK Parliament (2000), 'Freedom of Information Act 2000'.
URL: *https://www.legislation.gov.uk/ukpga/2000/36*

UK Parliament (2004), 'Civil Contingencies Act 2004'.

UK Parliament (2006), 'Wireless Telegraphy Act 2006'.
URL: *https://www.legislation.gov.uk/ukpga/2006/36*

UK Parliament (2012), 'The Restriction of the Use of Certain Hazardous Substances in Electrical and Electronic Equipment Regulations 2012'.
URL: *https://www.legislation.gov.uk/uksi/2012/3032*

UK Parliament (2013), 'The Waste Electrical and Electronic Equipment Regulations 2013'.
URL: *https://www.legislation.gov.uk/uksi/2013/3113*

UK Parliament (2014), 'The Data Retention Regulations 2014'.
URL: *https://www.legislation.gov.uk/ukdsi/2014/9780111118894*

UK Parliament (2016), 'The Electromagnetic Compatibility Regulations 2016'.
URL: *https://www.legislation.gov.uk/uksi/2016/1091*

UK Parliament (2017), 'The Radio Equipment Regulations 2017'.
URL: *https://www.legislation.gov.uk/uksi/2017/1206*

UK Parliament (2018a), 'Data Protection Act 2018'.
URL: *https://www.legislation.gov.uk/ukpga/2018/12*

UK Parliament (2018b), 'The Network and Information Systems Regulations 2018'.
URL: *https://www.legislation.gov.uk/uksi/2018/506*

UK Parliament (2023), 'Communications Act 2003'.
URL: *https://www.legislation.gov.uk/ukpga/2003/21*

United Nations (2015*a*), 'Paris Agreement'.

United Nations (2015*b*), 'Transforming Our World: The 2030 Agenda for Sustainable Development'.

United Nations Office for Disaster Risk Reduction (2024), 'DRR and UNDRR's History', Online.
URL: *https://www.undrr.org/our-work/history*

University of Stanford (2024), 'An Introduction to Design Thinking PROCESS GUIDE'.
URL: *https://web.stanford.edu/~mshanks/MichaelShanks/files/509554.pdf*

USGS (2017), 'Significant Earthquakes, 1965-2016'.
URL: *https://www.kaggle.com/datasets/usgs/earthquake-database?select=database.csv*

Uthra, R. A. and Raja, S. V. K. (2012), 'Qos routing in wireless sensor networks—a survey', *ACM Comput. Surv.* **45**(1).
URL: *https://doi.org/10.1145/2379776.2379785*

van Oldenborgh, G. J., Otto, F., Haustein, K. and et al. (2021), 'Attribution of the australian bushfire risk to anthropogenic climate change', *Natural Hazards and Earth System Sciences* **21**, 941–960.

Wateridge, J. (1998), 'How can is/it projects be measured for success?', *International Journal of Project Management* **16**(1), 59–63.
URL: *https://www.sciencedirect.com/science/article/pii/S0263786397000227*

Wen, H., Lin, C., Ren, F., Yue, Y. and Huang, X. (2007), Retransmission or redundancy: Transmission reliability in wireless sensor networks, Vol. 55, pp. 1–7.
URL: *10.1109/MOBHOC.2007.4428620*

Werner-Allen, G., Lorincz, K., Ruiz, M., Marcillo, O., Johnson, J., Lees, J. and Welsh, M. (2006), 'Deploying a wireless sensor network on an active volcano', *Internet Computing, IEEE* **10**, 18–25.
URL: *https://doi.org/10.1109/MIC.2006.26*

Willis, J. (2023), 'Analysing the frequency trends of earthquakes: A comparative study of modern and historical seismic data', *NMITE* .

Wolniak, R. (2017), The design thinking method and its stages, *in* 'Methods and tools for improving products and services on the selected examples'.
URL: *https://api.semanticscholar.org/CorpusID:195729028*

Wotton, B. M., Flannigan, M. D. and Marshall, G. A. (2017), 'Potential climate change impacts on fire intensity and key wildfire suppression thresholds in canada', *Environmental Research Letters* **12**(9), 095003.

Yang, H.-C. (2017), *Introduction to Digital Wireless Communications*, Institution of Engineering and Technology (IET).

Ye, W., Heidemann, J. and Estrin, D. (2002), An energy-efficient mac protocol for wireless sensor networks, *in* 'Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies', Vol. 3, pp. 1567–1576.

Yick, J., Mukherjee, B. and Ghosal, D. (2005), Analysis of a prediction-based mobility adaptive tracking algorithm, *in* '2nd International Conference on Broadband Networks, 2005', Vol. 1, pp. 753–760.
URL: *https://doi.org/10.1109/ICBN.2005.1589681*

Yick, J., Mukherjee, B. and Ghosal, D. (2008), 'Wireless sensor network survey', *Computer Networks* **52**(12), 2292–2330.
URL: *https://doi.org/10.1016/j.comnet.2008.04.002*

Yin, H., Yang, H. and Shahmoradi, S. (2022), 'Eatmr: an energy-aware trust algorithm based the aodv protocol and multi-path routing approach in wireless sensor networks', *Telecommunication Systems* **81**.
URL: *10.1007/s11235-022-00915-0*

Appendices

Part 1

Main Report Appendices

A Case Study Exploration

A.1 Natural Disasters and Wildfires

In recent years, there has been a significant rise in the frequency and intensity of natural disasters worldwide. Natural disasters be can defined as sudden events occurring at a specific time and place, causing severe danger, loss and disruption of societal functions, ultimately preventing the fulfilment of essential social and economic activities (Silva et al., 2014; Horsley, 2015).

Among these disasters, wildfires have become increasingly prominent. Historically, the UK experienced fewer wildfires compared to other regions, but recent trends indicate that this is changing. The rise in global surface temperatures (GTS) is a key driver behind this increase, both in terms of wildfire frequency and intensity. The following figure illustrates the rise in temperature anomalies (temperature differences relative to the long-term average) since 1880, with a notable increase in recent years.

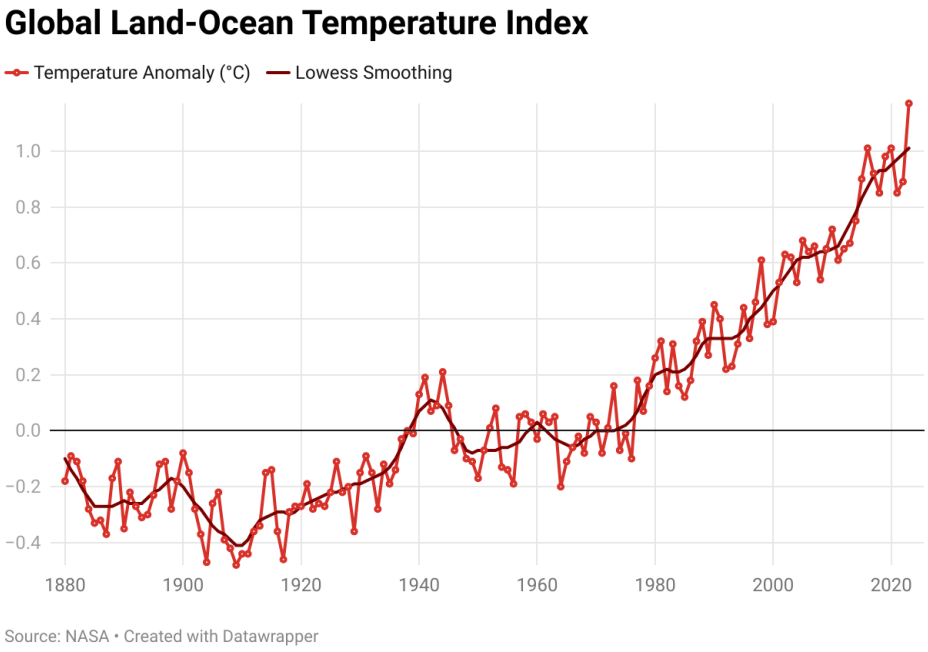

Figure 23: GTS Index from 1880 to present (NASA, 2024).

In 2022, the UK experienced an unprecedented heatwave, resulting in a significant number of wildfires. This event highlighted health risks and placed additional strain on emergency services (UK Health Security Agency, 2023). Globally, the impact of increased GTS on wildfire severity is also becoming more evident. For instance, a study in Northern California predicted that under a 2x CO2 climate change scenario, fires would burn more intensely and spread faster due to warmer and windier conditions (Fried et al., 2004). Similarly, in Canada, future

fire behaviour is expected to worsen, with the proportion of days with unmanageable fire potential more than doubling in some northern and eastern boreal forest regions (Wotton et al., 2017).

In North America as a whole, future wildfire potential is projected to rise significantly, with fire seasons potentially extending by several months due to increased temperatures and decreased relative humidity (Liu et al., 2013). The use of climate models such as HadCM3 (Met Office, n.d.) and KDBI models (Liu et al., 2013) suggests that increased fire potential is a major concern, requiring more resources and management efforts for disaster prevention and recovery.

In Southern Europe, changes in fuel, vegetation, and anthropogenic factors are expected to increase fire danger and the length of fire seasons (Dupuy et al., 2020). Meanwhile, in Australia, extreme heat and prolonged droughts are amplifying fire risk, although climate change primarily worsens these conditions rather than directly causing them (van Oldenborgh et al., 2021).

A.1.1 The Wildfire Feedback Loop

A concerning phenomenon associated with wildfires is the feedback loop. Wildfires release significant amounts of CO2 into the atmosphere, exacerbating global warming, which in turn creates conditions more favorable for wildfires (Intergovernmental Panel on Climate Change (IPCC), 2023*a*). This vicious cycle perpetuates the problem, as shown in the figure below.

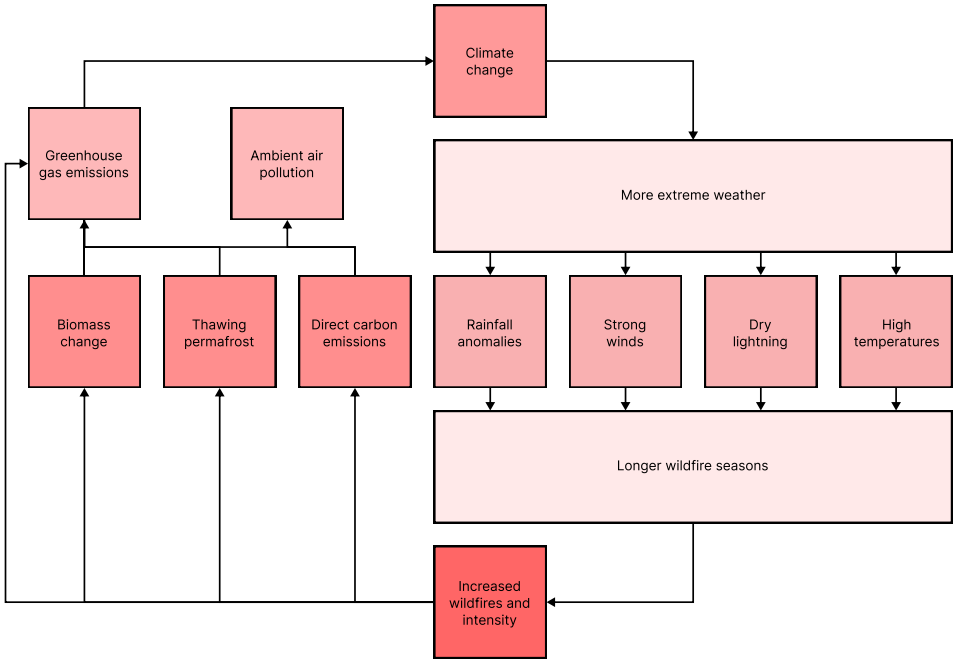

Figure 24: Wildfire feedback loop.

Typically, ecosystems such as forests and oceans act as carbon sinks, absorbing CO2 from the atmosphere. However, with increasing ecosystem stress caused by spikes in CO2 levels, the ability of these carbon sinks to absorb will diminish, potentially reversing their functionality entirely - turning them into sources that release more CO2 than they absorb. This feedback loop highlights the importance of addressing wildfire risk as part of the broader climate change mitigation strategies (Intergovernmental Panel on Climate Change (IPCC), 2023*a*).

A.1.2 Societal Impact of Wildfires and Natural Disasters

The impact of natural disasters is not uniformly distributed; although one might assume that natural disasters are non-discriminatory in their impact, the reality is that the severity and outcomes of such events are often exacerbated by the development status of different regions. Countries with less advanced infrastructure, lower economic resources, and limited capacity for disaster management frequently face more severe consequences when disasters strike. The National Fire Chiefs Council National Fire Chiefs Council (2024) has highlighted that while wildfires are becoming a more prominent issue globally, the UK's approach to managing them is still evolving. The NFCC points out that current funding for wildfire management is inadequate and that more accurate reporting and forecasting are needed to better prepare for and respond to emerging risks. This need for improved data is echoed in the International Panel on Climate Change (IPCC) reports, which indicate that ecosystem responses to warming, such as increased wildfire frequency and intensity, are not fully incorporated into current climate models (Intergovernmental Panel on Climate Change (IPCC), 2023*a*;*b*).

The forthcoming UK Wildfire Strategy and Action Plan, planned for release by Department for Environment, Food and Rural Affairs (DEFRA), aims to address these issues by improving wildfire risk assessment and response capabilities (HM Government, 2023; p.58). This strategy underscores the urgent need to address the forecasted increase in wildfires, considering both the increasing risk and the significant impacts on health, property, and natural resources.

A.2 Natural Disaster Trends and Reporting

Natural disasters are inherently unpredictable, as their occurrence is infrequent, and their locations, communication needs, and sensing requirements vary greatly from one event to the next (George et al., 2010). Despite their relative rarity, research shows that the frequency and intensity of phenomena causing these events, as well as the number of people affected and material damage caused, are increasing (Silva et al., 2014; Ritchie and Rosado, 2022).

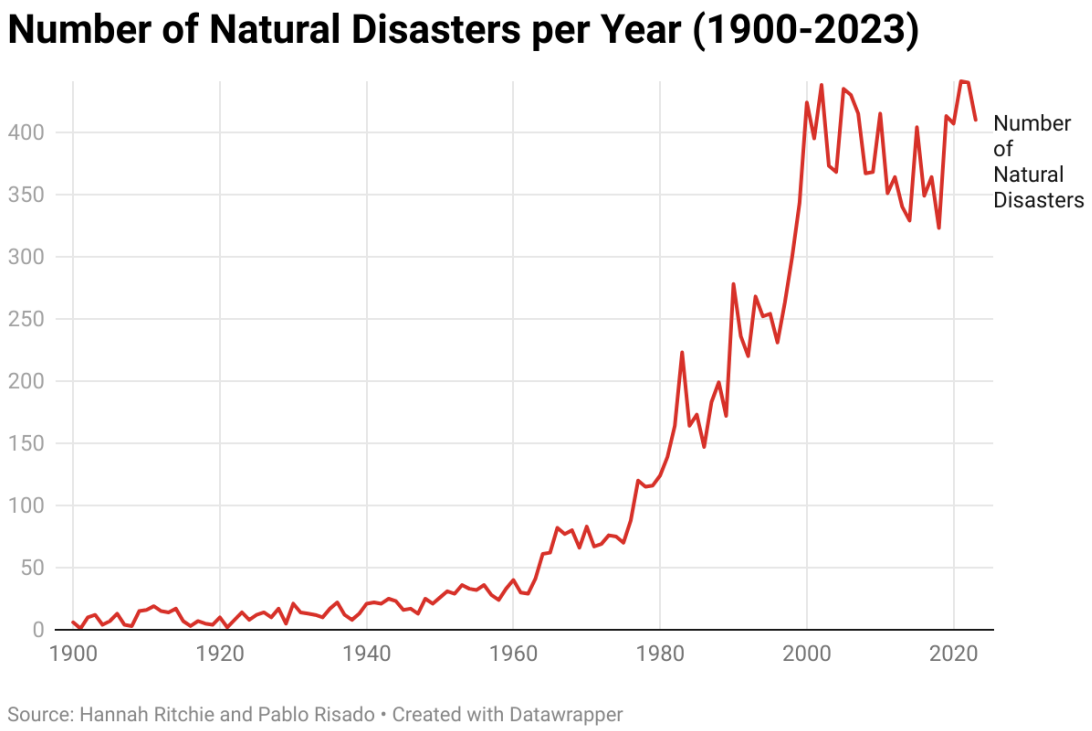

Figure 25: Number of recorded natural disaster events from 1900 to 2024 (Ritchie and Rosado, 2022).

While some literature attributes the need for improved disaster management to the rising frequency of such events, it is crucial to recognise that factors such as enhanced reporting and more accurate measurement often skew this data. For instance, a Mann-Kendall test conducted on significant (>5.5-magnitude) earthquakes occurring annually from 1965 to 2016 (using data from the National Earthquake Information Centre (NEIC) of the (USGS, 2017) suggested no statistically significant positive trend in the data (see Figure 26).

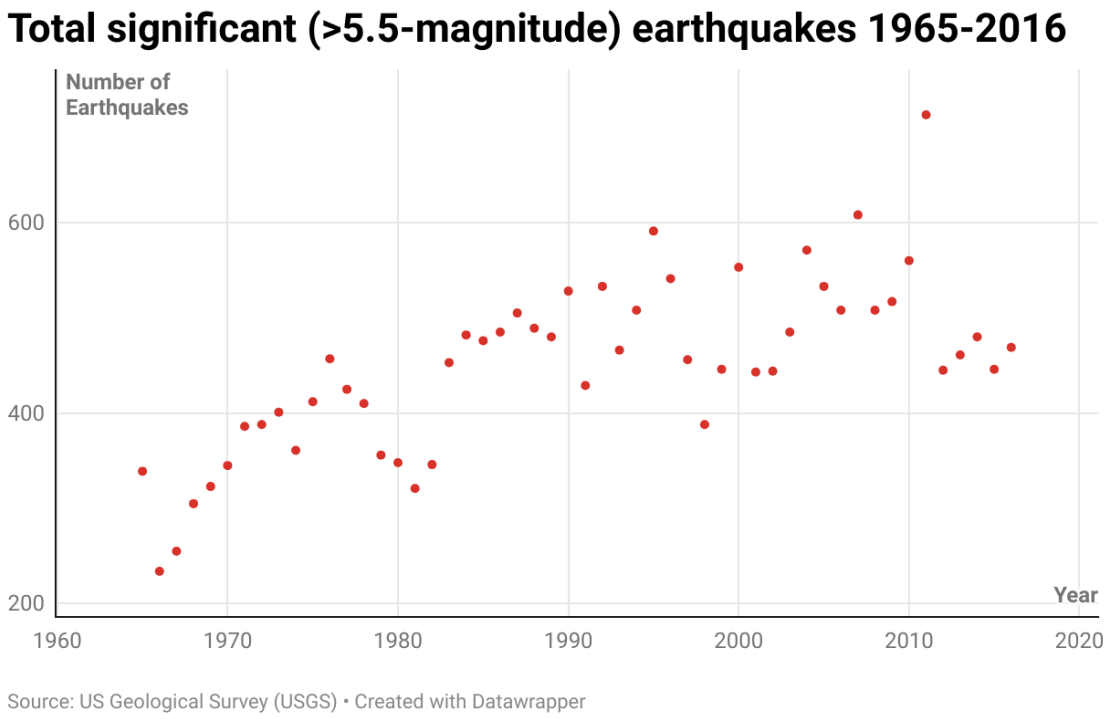

Figure 26: Number of >5.5-magnitude earthquakes per year from 1965-2016 (USGS, 2017).

The test accepted the null hypothesis, indicating no temporal trend in significant earthquake occurrences (Willis, 2023). Further research confirms the concept that the adoption of new monitoring technologies and increased incident reporting in recent years has influenced the data, creating the appearance of a positive trend (Okubo et al., 2014).

While this phenomenon might apply to earthquake data, there is a clear increase in the reported occurrences of other natural phenomena. Therefore, it is proposed that the case for enhancing disaster management should not be based solely on the frequency of these events but rather on the devastating consequences they bring - loss of life, economic disruption, and material damage - which are the fundamental drivers of the need for this improvement. Over the past 25 years, more than 6 million deaths have been attributed to natural disasters, with £140 billion spent on infrastructure reconstruction and over £14 trillion in total economic damages worldwide (statistics calculated from the data provided by Ritchie and Rosado (2022)). Figure 27 illustrates the evolution of total economic damages over time, with every natural disaster represented by a dot on the graph.

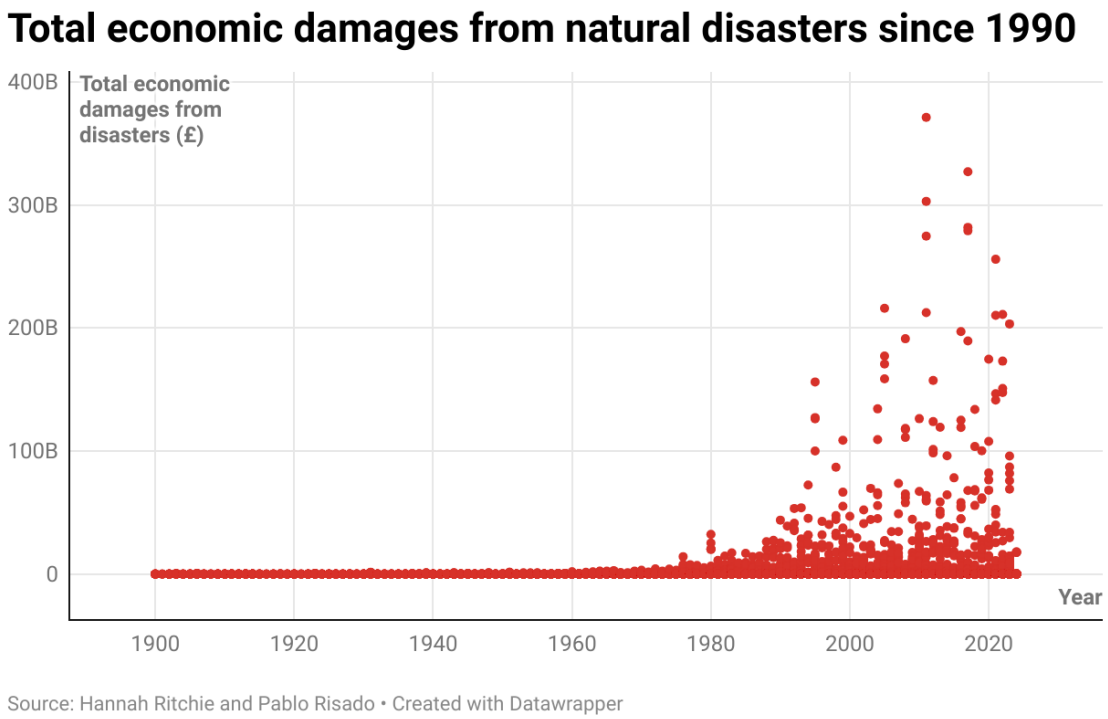

Figure 27: Total economic damage from disasters between 1990 and 2024.

This alarming data emphasises the need to rethink our approach to disaster management.

A.3 Overview of Disaster Management and Communication

Before the 1970s, disaster management was predominantly reactive with limited formal structures and ad-hoc responses, driven by local communities and international organisations like the Red Cross (Horsley, 2015; Scanlon, 2023). Clara Barton's advocacy for the Red Cross highlighted the need for effective communication in disaster relief, laying the groundwork for modern disaster response (Horsley, 2015; Mohr, 2018).

The 1970s introduced significant changes with the advent of Industry 3.0, bringing technological advancements that revolutionized crisis communication and spurred increased research in disaster management (Palenchar, 2009). During this era, disaster management began to formalise with the development of key management stages (see Figure 28) and the introduction of crisis communication theories. This period also marked significant institutional advancements, including the establishment of the Federal Emergency Management Agency (FEMA) in the US and the creation of the Civil Contingencies Secretariat (CCS) and the Civil Contingencies Act in the UK (Baird et al., 1975; Heath and O'Hair, 2009; Civil Contingencies Secretariat, 2013; UK Parliament, 2004).

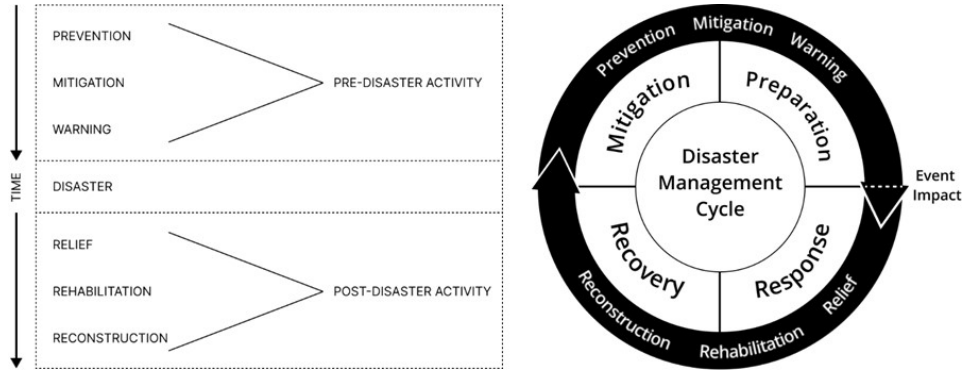

Figure 28: Disaster management stages: Baird et al. (1975) - left, Alexander (2002) - right.

Following this, the 1989 International Decade for Natural Disaster Reduction (IDNDR) marked a global push for disaster risk reduction (United Nations Office for Disaster Risk Reduction, 2024). The 1990s integrated new technologies like satellite imagery and 'geographic information system' (GIS), enhancing early warning systems and response coordination (Laituri and Kodrich, 2008).

The turn of the millennium brought a proactive, technology-driven approach with the rise of the Internet of Things (IoT) and wireless sensor networks (WSNs), enabling real-time data collection and analysis. This period saw the adoption of the Sendai Framework for Disaster Risk Reduction in 2015, emphasising technology and data in disaster management (United Nations, 2015*a*;*b*).

Today, disaster management benefits from advanced communication and monitoring systems facilitated by IoT and WSNs. These technologies, employing autonomous, low-energy

sensors, have transformed disaster anticipation, monitoring, and response, offering unprecedented precision and efficiency throughout the management cycle (Akyildiz et al., 2002; Yick et al., 2008; Chen et al., 2013; Kandris et al., 2020).

A.4 The Challenge of Energy Efficiency and Communication Reliability

As highlighted by Baird et al. (1975) and reiterated by Erdelj et al. (2017), the most critical factor in any disaster is the population—disasters, by definition, require people to be impacted. Therefore, disaster management systems must prioritise the needs of those affected in their design. Central to addressing these needs is effective communication, which plays a crucial role in ensuring that the right information reaches the right people at the right time.

However, one of the major challenges during a disaster is the lack of communication and situational awareness. As Ochoa and Santos (2015) note, this often forces first responders to improvise, undermining the efficiency of the disaster response. Similarly, Li and Goodchild (2010) emphasise that the primary challenges in emergency response are information-related; inadequate knowledge, information sharing, and communication channels can severely hinder disaster response efforts.

To overcome these communication barriers, effective monitoring and information-sharing infrastructure is crucial. This is where IoT technology, particularly WSNs, offers significant advantages by providing diverse and real-time data from multiple sources (Atzori et al., 2010). Current uses of this technology are shown in Table 15 (adapted from Erdelj et al. (2017)), delineating their use into two major categories, pre and post event, and then further refining their use to five typical use cases. Table 14 shows the key for the use cases seen in Table 15.

Table 14: Key: WSN use cases in disaster management.

Stage No.	Description
1	Monitoring, forecasting and early warning systems.
2	Disaster information fusion and sharing.
3	Situational awareness, logistics, and evacuation.
4	Standalone communications system.

Table 15: WSN use cases in disaster management (Erdelj et al., 2017) [Modified].

Pre-event	Post-event	1	2	3	4	References
•		•				(Frigerio et al., 2014)
•		•	•			(Chen et al., 2013)
•	•	•		•		(Erman et al., 2008)
	•	•			•	(Ueyama et al., 2014)
•	•	•	•			(Kureshi et al., 2015)
	•		•			(Mosterman et al., 2014)
•	•		•			(Bartoli et al., 2015)
	•		•			(Sardouk et al., 2010)
	•		•			(Kumar et al., 2004)
	•			•		(George et al., 2010)
	•			•		(Pogkas et al., 2007)
	•			•	•	(Mori et al., 2015)
	•				•	(Fujiwara and Watanabe, 2005)
	•				•	(Bai et al., 2010)
	•				•	(Fragkiadakis et al., 2011)
	•				•	(Tuna et al., 2014)
	•				•	(Carli et al., 2014)
	•				•	(Minh et al., 2014)

Table 15 highlights the extensive research in disaster management using WSNs. Despite their potential benefits, energy efficiency remains a key barrier to widespread adoption (Akyildiz and Vuran, 2010). A study on WSNs and IoT in disaster management suggests further research should focus on improving managing data flow, data integrity and optimising power usage (Adeel et al., 2019).

B Minimum Viable Product (MVP)

The table below presents the minimum viable project outcomes as a set of objectives essential to meeting the client's requirements.

Table 16: Minimum Viable Product (MVP).

MVP Objective	Description
1	To design the routing system for a wireless sensor network, specifically addressing: 1. Sensor data retrieval and labelling. 2. Priority and pre-empt assignment based on (but not limited to) the following factors: (a) Intrinsic property thresholds (battery life, internal node temperature, node connectivity, GPS, etc.). (b) Extrinsic property thresholds (external temperature, humidity, air quality, vibration, noise, camera footage, etc.). (c) Network conditions (congestion levels, etc.). 3. Decision engine to determine which radio/protocol is used to transmit the data and how to transmit the data.
2	To source and develop an appropriate set of nodes that make up a basic WSN that could deliver the above-described functionality.

C Meeting Minutes

The following tables provide a comprehensive overview of the agendas, outcomes, and action plans from all meetings held with the client and project appraiser throughout the project's duration.

Table 17: Meeting Summary: MEP-M1

Meeting ID:	MEP-M1
Date:	15/08/2024
Meeting Type:	Phone Call
Meeting Duration:	1 hour 27 mins
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Confirm project proposal is sufficiently indicative of the client's needs. 2. Schedule weekly meetings for the duration of the project. 3. Ask if a known COTS product serving a similar purpose for a different application is known or currently available.
Meeting Outcome:	• All points on the agenda were addressed. • Project proposal form approved and signed. • NDA requested. • Project further discussed and defined.
Plan:	1. Document meeting outcomes. 2. Set up Teams meeting invites for the duration of the project. 3. Finalise project proposal form and submit to module lead for final review.
Next Scheduled Meeting:	19/08/2024 (Postponed until 26/08/2024 to allow for more significant progress to be made)

Table 18: Meeting Summary: MEP-M2

Meeting ID:	MEP-M2
Date:	22/08/2024
Meeting Type:	In-person
Meeting Duration:	1hr (approx.)
Person(s) Involved:	James Willis, Project Client, and Tim Beldon
Agenda:	1. Discuss project and minimum viable product (MVP).
Meeting Outcome:	• Agenda was addressed. • MVP identified. • Project system outlined further – intelligent routing protocol that can label, assign priority and pre-empt levels, and select appropriate transmission methods/protocols to communicate specific data within the network.
Plan:	1. Record MVP and confirm with client in writing – identify levels/additions that could be employed if the project was to go smoothly. 2. Define system FBD and process flow diagrams for hardware, data, and overall system.
Next Scheduled Meeting:	26/08/2024 (Client busy, agenda sent via email to keep updated – rescheduled for 02/09/2024)

Table 19: Meeting Summary: MEP-M3

Meeting ID:	MEP-M3
Date:	26/08/2024
Meeting Type:	Email Update (Bank Holiday)
Meeting Duration:	N/A
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Conduct agile project update meeting – report on completed tasks, current tasks, and plans for the coming week. • **Completed Tasks:** (a) Read textbook on WSNs to bring general knowledge of WSNs up to speed. (b) Meeting with client last week to discuss MVP and additions that could be completed if the project deadline allows for it. (c) Began brainstorming system design. • **Current Position:** (a) Wrote up first draft of MVP. (b) Began to write up system requirements document. • **Plans:** (a) Continue creating system functional block diagrams and process flow charts to aid system design thinking. (b) Research regulations and standards surrounding the problem space. (c) Identify and map all possible decision factors involved in the decision engine.
Meeting Outcome:	• All meeting agenda points addressed. • Documents sent to client for approval.
Plan:	1. Continue conceptualising system design. 2. Research regulations and standards. 3. Identify decision factors. 4. Act on feedback received on documents.
Next Scheduled Meeting:	02/09/2024

Table 20: Meeting Summary: MEP-M4

Meeting ID:	MEP-M4
Date:	02/09/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	1 hour (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Discuss previous week's work. 2. Discuss process for defining components.
Meeting Outcome:	• All meeting agenda points were addressed. • Sensor components not necessary for MVP; dummy data can be used to demonstrate protocol switching functionality. • STM32 MCUs are preferred in the industry, but ESP32s are sufficient due to their user-friendliness, available documentation, and suitability for rapid development. Code should be as software/hardware agnostic as possible. • The system must be flexible and extensible across different platforms. • Employ time-division multiplexing for parallel processing with MCUs. • Ordering four nodes ensures redundancy and facilitates mesh networking demonstrations. • MCUs should support long-range protocols to enable routing and multi-hop functionality in the decision engine. • Additional discussions covered routers, WRT packages, APIs, beaconing, and leveraging existing infrastructure.
Plan:	1. Identify and select appropriate components for WSN nodes. 2. Order components. 3. Create detailed system and system-of-systems maps to display connections between components within nodes. 4. Research non-blocking sequence algorithms/parallel processing for MCUs.
Next Scheduled Meeting:	16/09/2024 (Postponed from 09/09/2024 due to client's unavailability).

Table 21: Meeting Summary: MEP-M5

Meeting ID:	MEP-M5
Date:	16/09/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	1 hour (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Update on previous two weeks' progress. 2. Send system requirements to client for feedback.
Meeting Outcome:	• All agenda points were addressed. • Scheduled meeting with IT department to discuss project again, aiming to avoid unnecessary constraints initially set due to security concerns.
Plan:	1. Wait for and act on feedback from the client on the system requirements document. 2. Update client report. 3. Follow up with technicians regarding component delivery status.
Next Scheduled Meeting:	23/09/2024

Table 22: Meeting Summary: MEP-M6

Meeting ID:	MEP-M6
Date:	16/09/2024
Meeting Type:	Phone Call
Meeting Duration:	30 mins (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Discuss feedback on system requirements.
Meeting Outcome:	• System requirements document amended to include a full requirements list below user stories. • Updated system requirements document approved by the client.
Plan:	1. Continue with the plan from meeting MEP-M5.
Next Scheduled Meeting:	23/09/2024

Table 23: Meeting Summary: MEP-M7

Meeting ID:	MEP-M7
Date:	23/09/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	1hr (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Update client on the outcome of IT meeting – constraints dropped, provided the system is designed with security in mind (e.g., passwords and addresses for secure node-to-node communication). 2. Update client on delay in component delivery and mitigation plans: begin coding on similar chips from the University's inventory to gain familiarity with interfacing the MCU and running MicroPython.
Meeting Outcome:	• Both agenda points were addressed. • Client approved the work-around for component delays.
Plan:	1. Continue coding on alternative SoCs.
Next Scheduled Meeting:	30/09/2024

Table 24: Meeting Summary: MEP-M8

Meeting ID:	MEP-M8
Date:	30/09/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	25mins (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Update client on the arrival of components and the success with coding. 2. Explain that the OLED and LoRa modules have been successfully interfaced.
Meeting Outcome:	• All agenda points were addressed.
Plan:	1. Continue to create code to display for HRH DofE's visit. 2. Continue to interface with onboard peripherals.
Next Scheduled Meeting:	07/10/2024

Table 25: Meeting Summary: MEP-M9

Meeting ID:	MEP-M9
Date:	07/10/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	1hr (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Discuss narrowing the focus on the routing protocol by the end of the coming week. 2. Update client on progress on interfaced modules and showcasing the project to HRH DofE. 3. Expectation management – ensuring MVP has been reached and setting goals for further development within the coming weeks.
Meeting Outcome:	• All agenda points were addressed. • Aspirations in terms of further development goals were set and agreed on.
Plan:	1. Continue to interface with onboard peripherals. 2. Read through information sent by the project client on routing algorithms.
Next Scheduled Meeting:	14/10/2024

Table 26: Meeting Summary: MEP-M10

Meeting ID:	MEP-M10
Date:	14/10/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	25mins (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Discuss week's progress and the fact that all three communication modules have been successfully interfaced with and operate within a singular script. 2. Update on successfully interfacing PMU and GPS modules.
Meeting Outcome:	• Agenda points were addressed. • Discussed routing algorithm information that the client had provided. • Client satisfied with progress.
Plan:	1. Take functional code and turn it into usable code. 2. Begin coding decision engine. 3. Begin unit testing.
Next Scheduled Meeting:	21/10/2024

Table 27: Meeting Summary: MEP-M11

Meeting ID:	MEP-M11
Date:	21/10/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	35mins (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Discuss week's progress. 2. Discuss how to add value to project – through user guide (GitHub page).
Meeting Outcome:	• Client approves the use of GitHub to provide user guide and publicly display code. • Client happy with project state and agrees that due to project report deadlines it is a good idea to begin write-up of reports and user guide.
Plan:	1. Create GitHub user guide. 2. Comment and upload all code, libraries and drivers. 3. Continue to advance decision engine code.
Next Scheduled Meeting:	04/11/2024 (Week off on the week commencing 28/10/2024 so meeting set for week after).

Table 28: Meeting Summary: MEP-M12

Meeting ID:	MEP-M12
Date:	04/11/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	25mins (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Update client on GitHub user guide page completion. 2. Discuss project and client reports and feedback.
Meeting Outcome:	• GitHub page published publicly for client to see and for public to access. • Set out the type of feedback required from the client at the end of the project.
Plan:	1. Write up reports.
Next Scheduled Meeting:	11/11/2024

Table 29: Meeting Summary: MEP-M13

Meeting ID:	MEP-M13
Date:	11/11/2024
Meeting Type:	Email Update
Meeting Duration:	N/A
Person(s) Involved:	James Willis and Project Client
Agenda:	1. Explain the reason for switching focus to project report for this week rather than remaining on client report – deadlines and the ability to take extracts of project report and insert into client report in a concise fashion.
Meeting Outcome:	• Client understood reasoning and is happy to provide feedback whenever documents are ready to be sent and reviewed.
Plan:	1. Finish academic report and send off to technical and non-technical audiences for reviews whilst working on client report.
Next Scheduled Meeting:	18/11/2024

Table 30: Meeting Summary: MEP-M14

Meeting ID:	MEP-M14
Date:	18/11/2024
Meeting Type:	Microsoft Teams
Meeting Duration:	30 mins (approx.)
Person(s) Involved:	James Willis and Project Client
Agenda:	• N/A
Meeting Outcome:	• N/A
Plan:	• N/A
Next Scheduled Meeting:	18/11/2024

D Logbook Example Pages

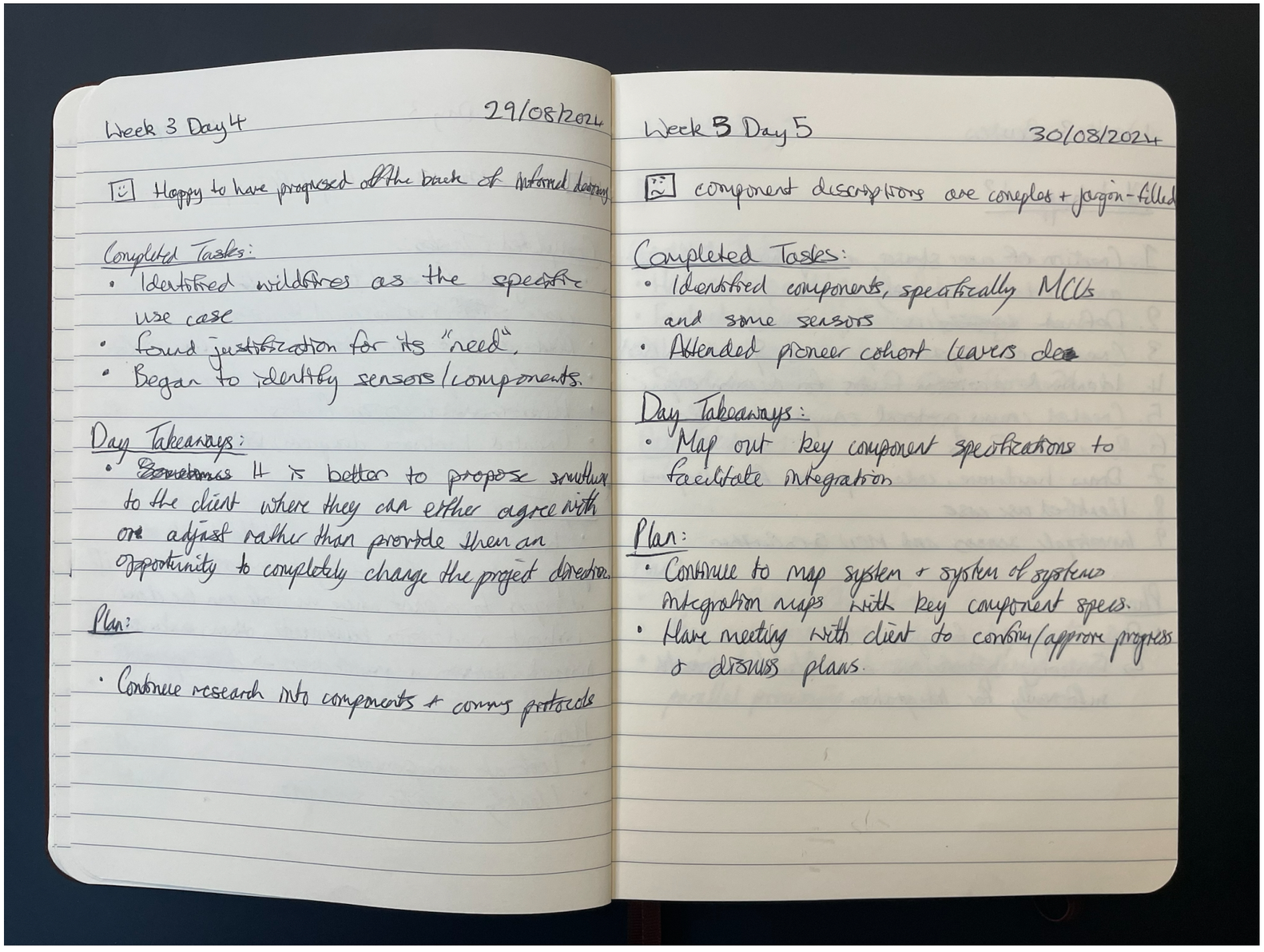

Figure 29: Logbook example pages [1].

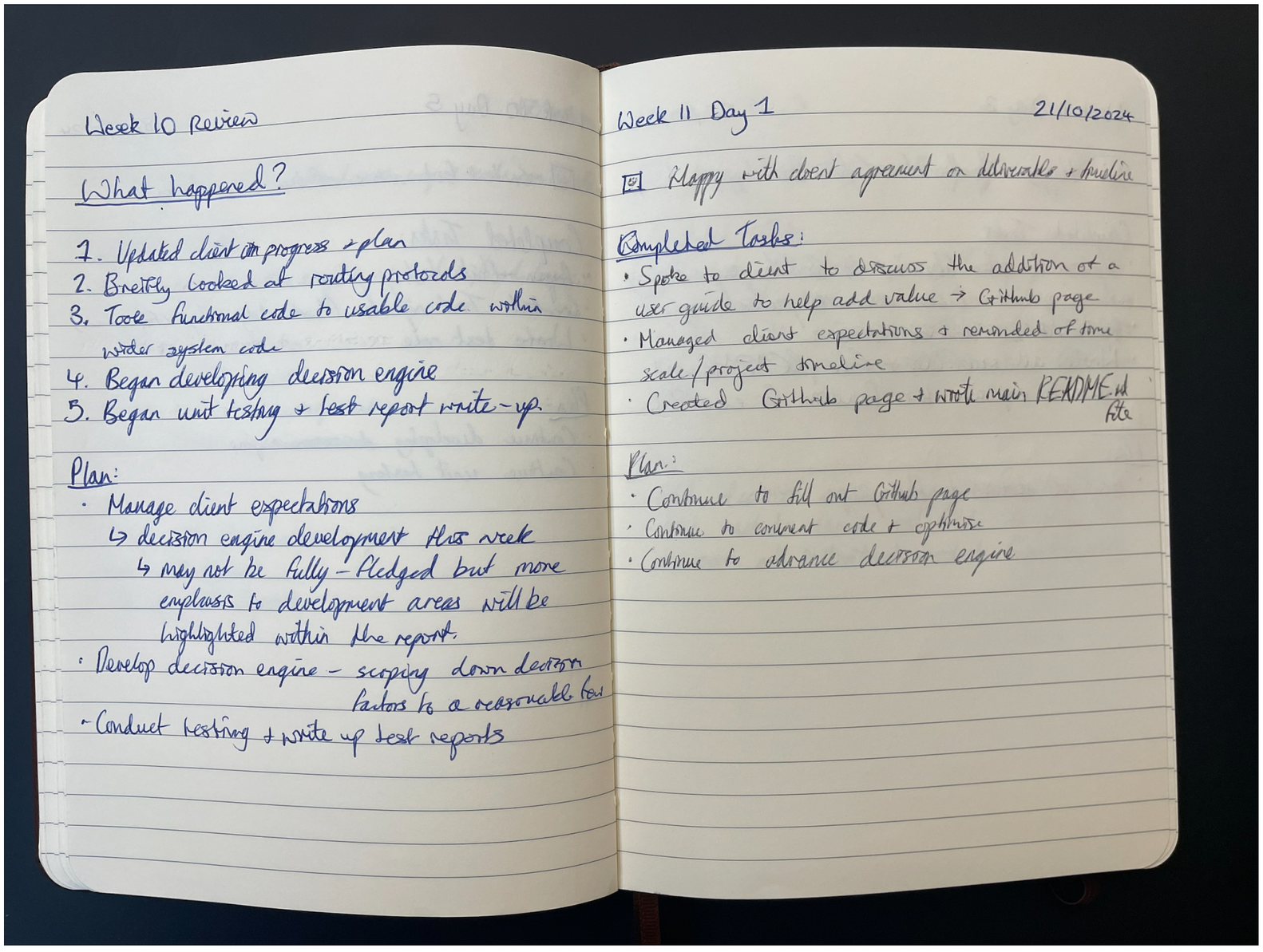

Figure 30: Logbook example pages [2].

E System Requirements Document

A clear, concise and approved set of requirements is essential for the success of any project. However, as projects progress, and the project team gains a deeper understanding of the problem space, new requirements and opportunities are likely to arise. To ensure these requirements are captured and incorporated in a systematic and controlled manner, the Dynamic System Development Method (DSDM) has been adopted (Agile Business Consortium Ltd., 2024). This method acknowledges the need for enhanced flexibility, allowing for requirements to be added to or fleshed out when appropriate.

User stories are a key aspect in defining this set of requirements early in the project. They provide a practical way to capture specific needs from the perspective of potential end users in a simple and goal-oriented format. The "I am…", "I need…", "so that…" format has been used to focus on what needs to be achieved rather than how it will be achieved, ensuring the solution is kept open to innovation (Agile Business Consortium Ltd., 2024).

In this project, the user stories are categorised into the following key functional areas:

1. Data Collection and Labelling
2. Decision Engine
3. Communication and Networking
4. Flexibility and Extensibility
5. Performance
6. Cost and Resource Efficiency

These user stories serve as the foundation for defining the system requirements. Through the consolidation of client feedback, feasibility assessments, and the application of MoSCoW (Must have, Should have, Could have, Won't have) prioritisation scores (Clegg & Barker, 1994). The table below shows the feasibility and MoSCow keys.

Table 31: Feasibility and MoSCoW scoring key.

Feasibility (F)	Meaning	MoSCoW (M)	Meaning
4	Feasible	M	Must have
3	Within scope	S	Should have
2	Feasible but not essential	C	Could have
1	Not within scope	W	Will not have

E.1 User Stories

E.1.1 Data Collection and Labelling

Table 32: Data collection and labelling related user stories.

I am...	I need...	So that...	F	M
I am a disaster relief coordinator	I need the system to automatically collect data from various sensor nodes in real-time	so that I can monitor the environmental conditions and coordinate relief efforts effectively.	4	M
I am a data analyst	I need the system to process and label the collected data with appropriate metadata	so that I can accurately interpret the data.	4	M

E.1.2 Decision Engine

Table 33: Decision engine related user stories.

I am...	I need...	So that...	F	M
I am a network administrator	I need the system to assign priority levels to data based on predefined criteria	so that critical information is transmitted first to support timely decision-making.	4	M
I am a system operator	I need the system to select the appropriate communication protocol based on the type and size of data	so that data is transmitted effectively while conserving energy and optimising network traffic.	4	M
I am an emergency first responder	I need the system to route data to the correct destination within the network based on its priority and protocol requirements	so that I receive information to my operational needs without delay.	3	S
I am a disaster relief coordinator	I need the system to pre-empt lower priority data transmission in favour of higher priority data when network resources are limited	so that urgent messages are delivered promptly.	3	S

E.1.3 Communication and Networking

Table 34: Communication and networking related user stories.

I am...	I need...	So that...	F	M
I am a communications engineer	I need the system to use different transmission protocols	so that data transmission is optimised based on data size and urgency.	4	M
I am a network operator	I need the system to support multiple methods of transmission such as direct node-to-node communication and relay through gateways	so that data can be routed effectively across different network topologies.	3	S

E.1.4 Flexibility and Extensibility

Table 35: Flexibility and extensibility related user stories.

I am...	I need...	So that...	F	M
I am a network security specialist	I need the system to use standard communication protocols for network communication	so that data transfer is reliable and interoperable.	4	S
I am a project manager	I need the system to be adaptable for use in various applications beyond disaster relief	so that it can contribute knowledge to a wider field and provide long-term value.	3	S
I am a system architect	I need the system to be modular and scalable	so that upgrades, replacements and increasing number of nodes is feasibility for various applications.	3	S
I am an operations manager	I need the system to be small and easily deployable	so that it can be quickly set up and reliably used in harsh and unpredictable conditions.	3	S
I am a disaster relief coordinator	I need the system's interfaces to be intuitive and user-friendly	so that I can quickly access and interpret critical data without extensive training.	2	S
I am a network engineer	I need the system to be interoperable with existing communication infrastructures	so that it can enhance collaborative operations.	2	C
I am an environmental engineer	I need the system to be durable and weather-resistant	so that it can operate in harsh environmental conditions during disaster scenarios.	2	C

E.1.5 Form and Performance

Table 36: Form and performance related user stories.

I am...	I need...	So that...	F	M
I am a system designer	I need the system to route data with minimal latency	so that response time can be quick enough to address the needs of disaster scenarios	4	M
I am a procurement officer	I need the system to be built using commercial-off-the-shelf (COTS) products	so that it minimises costs and ensures ease of replacement, maintenance and integration.	4	M
I am a disaster relief coordinator	I need the system to be highly reliable with minimal downtime	so that there is continuous data flow and communication during operations.	4	S
I am a system maintenance engineer	I need the system to detect and handle errors in data transmission	so that system reliability is maintained, and issues are quickly addressed.	3	S

E.1.6 Resource Efficiency

Table 37: Resource efficiency related user stories.

I am...	I need...	So that...	F	M
I am an energy management specialist	I need the system to be power-efficient with long battery life	so that it can sustain operations in remote areas without frequent recharging or battery replacements.	4	S
I am a project manager	I need the system to be cost-effective in both initial setup and ongoing maintenance	so that it remains within budget constraints.	2	S

E.1.7 Regulatory and Compliance

Table 38: Regulatory and compliance related user stories.

I am...	I need...	So that...	F	M
I am a compliance officer	I need the system to ensure all collected data is protected in accordance with data privacy regulations	so that sensitive information is secure and complies with legal standards.	3	M
I am a systems integrator	I need the system to adhere to relevant industry standards (such as ISO 802.15.4, etc.)	so that it meets appropriate quality and safety benchmarks.	3	S
I am a disaster relief coordinator	I need the system to generate reports that meet regulatory requirements for data collection and management	so that the documentation can be submitted for audits and compliance checks.	2	S

E.2 System Requirements

Based on the feasibility and MoSCoW scores of the user stories outlined above, the key system requirements have been developed. To ensure these requirements are effective, clear and flexible, Bill Wake's INVEST model (2003) has been adopted; confirming that they are: Independent, Negotiable, Valuable, Estimable, Small, and Testable.

E.2.1 Data Collection and Labelling

Table 39: Data collection and labelling requirements.

ID	Requirements
R1	The system must automatically collect data from sensor nodes in real-time to enable continuous/live monitoring and alerting of environmental conditions.
R2	The system must process and label the collected data with appropriate metadata for accurate interpretation and manipulation.

E.2.2 Decision Engine

Table 40: Decision engine requirements.

ID	Requirements
R3	The system must assign priority levels to data based on predefined criteria, ensuring that data is transmitted via an appropriate communication protocol.
R4	The system must dynamically select the appropriate communication protocol based on the data type, size, and priority, optimising the data transmission to conserve energy.

E.2.3 Communication and Networking

Table 41: Communication and networking requirements.

ID	Requirements
R5	The system must support multiple communication protocols/methods of transmission to enable service differentiation functionality.

E.2.4 Flexibility and Extensibility

Table 42: Flexibility and extensibility requirements.

ID	Requirements
R6	The system must be flexible, extensible and modular in design to enable functionality across different applications.

E.2.5 Form and Performance

Table 43: Form and performance requirements.

ID	Requirements
R7	The nodes must be of small size and weight (specifically close to that of a standard 330ml drinks cannister).
R8	The system must be made from commercial-off-the-shelf components.
R9	The system must provide the end users with appropriate alerts in real-time to ensure quick and reliable disaster response.
R10	The system must allow for end users to customise alert thresholds according to situational and operational requirements.

E.2.6 Resource Efficiency

Table 44: Resource efficiency requirements.

ID	Requirements
R11	The system must be energy efficient to ensure prolonged operation for various applications.

E.3 Client Approval

To verify and validate that these system requirements are sufficiently indicative of the client's needs, approval is required and will be used as evidence that the client has reviewed, understood, and agreed upon the documented requirements. This approval serves as a formal agreement and will guide the development process, ensuring that the delivered system aligns with their expectations.

Approval Status: **APPROVED**

By approving this document, you [THE CLIENT/POTENTIAL END USER] are verifying and validating the presented client requirements, approving their representativeness of your needs.

This approval is solely to be used as part of this NMITE Masters Engineering Project and will have no power for wider contractual or other legal effect.

F Stakeholder Map and Engagement Plan

F.1 Stakeholder Map

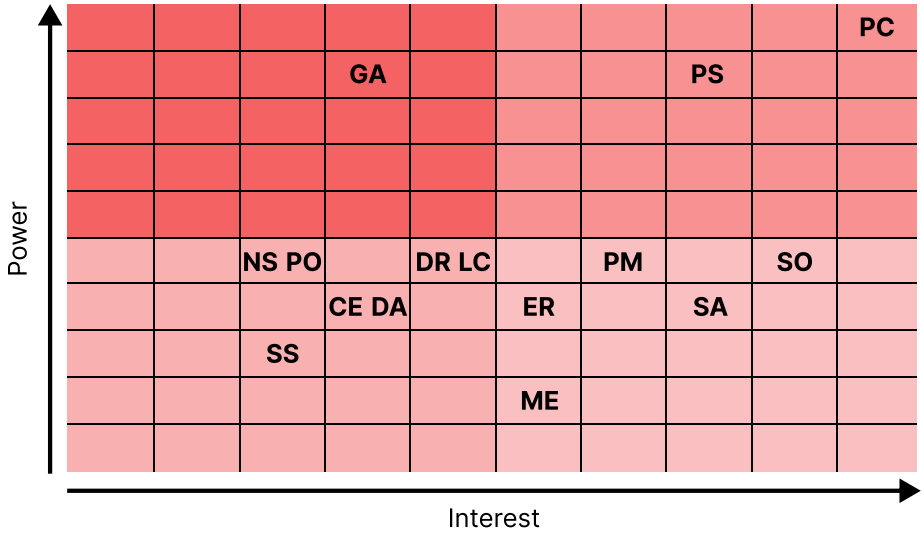

Figure 31: Stakeholder Map.

F.2 Stakeholder Engagement Key

Table 45: Strategy of engagement key.

Section Colour	Stakeholder Need	Strategy of Engagement (SoE)
1	Keep satisfied	Comply with regulatory bodies.
2	Engage closely and actively influence.	In-person discussions, Microsoft Teams meetings and email conversations. Collaboratively working on the project and ensuring all information and documents align with the client's requirements.
3	Keep Informed	Consideration into of all of the aspects that may affect the WSN's acceptance among potential end users and detail within the report.
4	Monitor	Ensure report is transparent and accessible to both technical and non-technical audiences. Provide justifications towards the need for the project and the decisions made throughout.

F.3 Table of Analysis

Table 46: Stakeholder analysis.

ID	Stakeholder	Power (1-10)	Interest (1-10)	Expectation/Role	SoE
GA	Government Agencies	9	4	Ensure compliance with regulations and standards. Oversee project execution.	1
PC	Project Client	10	10	Define project scope and requirements, approve deliverables, and ensure alignment with business goals.	2
PS	Project Supervisor	8	9	Provide guidance on project methodology and monitor project progress.	2
DR	Disaster Relief Coordinator	5	5	Monitor environmental conditions and coordinate relief efforts.	3
DA	Data Analyst	4	4	Process and label collected data for accurate interpretation.	3
NA	Network Administrator	4	5	Assign priority levels to data, monitor performance, and ensure critical info is transmitted.	3
CE	Communications Engineer	4	4	Implement and maintain various communication protocols.	3
SS	Security Specialist	3	3	Ensure secure and standard protocol communication.	3
PO	Procurement Officer	5	3	Use COTS products to minimise costs and ensure ease of maintenance.	3
LC	Local Communities	5	5	Benefit from early warnings and better resource management. Ensure WSN meets public needs and is user-friendly.	3
NS	Non-aligned Stakeholders	5	3	Require flexibility and extensibility in system design for applicability across non-disaster relief related applications.	3
PM	Project Manager	5	7	Oversee project implementation, ensure adaptability for broader applications.	4
SA	System Architect	4	8	Design modular systems, ensure scalability, and maintain system integrity.	4
ME	Maintenance Engineer	2	6	Detect and address errors in data transmission.	4
SO	System Operator	5	9	Select communication protocols to optimise transmission and conserve resources.	4
ER	Emergency First Responder	4	6	Receive timely instructions from coordinator to make informed decisions and provide relief.	4

G Analysis of Macro-Environmental Factors

The following table highlights the macro-environmental factors, namely PESTLE (Political, Economic, Social, Technological, Legal, and Environmental) considerations, that influence the project's success.

Table 47: PESTLE analysis.

Factor	Consideration
Political	Compliance with appropriate communication, data processing, environmental, and health and safety regulations.
	Data privacy laws, such as GDPR, influence how edge devices handle and store sensitive data locally.
	Regulatory approval for deploying edge devices in disaster-prone areas must be secured prior to deployment.
Economic	COTS components help reduce costs, but initial investment in infrastructure may be high.
	Potential funding opportunities from disaster resilience initiatives and technology grants.
	Effective response and mitigation could provide significant long-term savings.
Social	Project aims to enhance public safety and increase awareness through early warning systems.
	Training for first responders, coordinators, and operators is essential.
	Transparency regarding data collection and privacy is necessary to gain public trust.
Technological	Advances in artificial intelligence and machine learning could improve real-time decision making beyond the scope of this project.
	System will reduce latency by processing critical data locally, improving response times.
	The possiblity of integrating with existing WSNs and cloud systems could further enhance functionality and redundancy.
Legal	Data security standards must be adhered to when system is operation to mitigate legal risk around unauthorised access to sensitive data.
	Product conformity documents should be requested from manufacturer before use.
	Contracts and agreements with stakeholders must define roles, responsibilities, and deliverables.
Environmental	The system should try to reduce energy consumption where possible.
	Early detection and response can minimise environmental damage during disasters.
	Increased frequency of natural disasters raises demand for more robust disaster management systems.

H Risk Management

H.1 Impact Key

Table 48: Risk Impact Key.

Impact Key		
1	**Very Low**	Insignificant change to the cost, schedule, scope, and quality of the project.
2	**Low**	Minor change to the cost (<10% increase), schedule (<5% increase in time), scope and quality (barely noticeable quality decrease).
3	**Moderate**	Moderate change to the cost (10-20% increase), schedule (5-10% increase in time), scope and quality (client approval is needed).
4	**High**	Major change to the cost (20-40% increase), schedule (10-20% increase in time), scope (unacceptable reduction) and quality (unaccepted reduction).
5	**Very High**	Catastrophic change to the cost (>40% increase), schedule (>20% increase in time), scope (project end result doesn't fit the scope at all) and quality (result is deemed effectively useless).

H.2 Likelihood Key

Table 49: Risk Likelihood Key.

Likelihood Key		
1	**Very Low**	The risk is highly unlikely to occur but may still occur in exceptional or unforeseen circumstances.
2	**Low**	The risk will most likely not occur. Risk is not expected.
3	**Moderate**	The risk could occur.
4	**High**	The risk is likely to occur. There is a strong possibility that this risk will occur – frequent occurrence in similar projects.
5	**Very High**	The risk is almost certain to occur. Risk is expected to occur – regular occurrences of this risk in similar projects.

H.3 Risk Assessment and Mitigation Plan

Table 50: Risk assessment and mitigation plan for PCRs.

Risk ID	Risk	Risk Level		Mitigation Plan	Revised Risk Level	
		Impact	Likelihood		Impact	Likelihood
R1	Project misses deadline.	5	3	Understand system requirements, client requirements and use project management methods, tools and techniques to deal with uncertainties.	5	1
R2	Project lead not sufficiently skilled	4	4	Research and develop necessary knowledge and skills foundation. Utilise client and supervisor specialist knowledge to build on foundation.	4	2
R3	Project does not achieve or strays away from agreed MVP.	4	3	Regularly engage with client to understand and manage their requirements and expectations.	3	2
R4	Project outputs do not provide sufficient information to both technical and non-technical audiences.	4	3	Ensure project report is understandable by both audiences. Provide supplementary technical client report and publicly release an accessible user guide.	4	1
R5	System implemented does not provide the functionality required by the client.	4	3	Ensure methodology iteratively revisits requirements to ensure both COTS product and edge processing framework reflect the client's needs. Update client on project progress, managing expectations.	4	1
R6	Project does not provide opportunity for client feedback	4	2	Continual interaction with client - weekly Teams meetings and emails. Request feedback at the end of the project to evaluate the efficacy of the method.	4	1
R7	Project does not establish pertinent need.	3	4	Conduct case study into a pressing area for potential deployment of such system.	3	2
R8	Adopted methodology is too rigid and does not account for criticisms of the synthesised methodologies.	3	3	Incorporate sprint/scrum structure throughout project lifecycle to iteratively flow through project. Ensure criticisms are addressed and amendments made to ensure methodology is effective.	3	1
R9	Project success considerations are limited.	3	3	Use tools and technique to account for a range of uncertain factors both internal and external, aligned and non-aligned to the project.	3	1
R10	Project motivation is lost	5	3	Ensure logbook is updated frequently to detail completed tasks and ensure a structured plan is set out.	5	1

I Communication Protocol Comparison Table

Due to the different types of data expected to be processed and communicated within the network, different protocols are required to transfer this information in an effective and energy-efficient manner. The table below shows a range of different communication protocols applicable to WSNs and their properties.

***Range** dependant on physical layer properties of transmitter and receiver - antenna, gain, transmission power and interference*

Table 51: Communication protocol comparison table.

Technology		Frequency	Max Range (outdoors)	Max Data Rate	Power Usage	Available Topologies	Proprietary
IEEE 802.11	Wi-Fi HaLow	sub-GHz	Up to 3km	4Mbps	Low	Star	Wi-Fi Alliance
	Wi-Fi 4	2.4GHz	Up to 1km	600Mbps	High	Star	Wi-Fi Alliance
	Wi-Fi 6	2.4 and 5GHz	Up to 35m	9.6Gbps	High	Star	Wi-Fi Alliance
IEEE 802.15.1	Bluetooth 3.0	2.4GHz	10m	24Mbps	High	P2P, Mesh	Bluetooth SIG
	Bluetooth 4.0 LE	2.4GHz	100m	1Mbps	Low	P2P, Mesh	Bluetooth SIG
	Bluetooth 5.0 LE	2.4GHz	1km	2Mbps	Low	P2P, Mesh	Bluetooth SIG
IEEE 802.15.4	ZigBee	2.4GHz	90m	250kbps	Low	Mesh, Star, Cluster	ZigBee Alliance
	WirelessHART	2.4GHz	90m	250kbps	Moderate	Mesh	
	6LoWPAN	2.4GHz	100m	250kbps	Moderate	Star	
	ISA 100.11a	2.4GHz	500m	250kbps	Moderate	Mesh, Star, Cluster	
	Thread	2.4GHz	30m	250kbps	Low	Mesh	ThreadGroup
IEEE 802.16	WiMAX	2.3-5.8GHz	15km	75Mbps	High	Star, Tree, P2P	Wi-Fi Alliance
Cellular	GPRS	380 to 1900 MHz	Several kms	128.4kbps	High	Star	
	EDGE	380 to 1900 MHz	Several kms	355.5kbps	High	Star	
	3G-HSDPA	700MHz to 3GHz	Several kms	14.4Mbps	High	Star	
	3G-HDPA+	700MHz to 3GHz	Several kms	337Mbps	High	Star	
	LTE-Cat1	600MHz to 6GHz	Several kms	10Mbps	Moderate	Star	
	LTE-CatM1	600MHz to 6GHz	Several kms	1Mbps	Low	Star	
	NB-IoT	600MHz to 6GHz	Several kms	250kbps	Low	Star	
	5G	600MHz to 6GHz	Several kms	20Gbps	High	Star	
LoRaWan		868.42MHz	Several kms	50kbps	Low	Star	LoRa Alliance
Z-Wave		868.42MHz	30m	40kbps	Low	Mesh	Silicon Labs
SigFox		868MHz	800m	600bps	Low	P2P	SigFox
ISO 18000	RFID	2.4GHz	1-2m				

In evaluating the available options, it is evident from the table that there is a broad array of communication technologies, each with its unique set of benefits and limitations. For a rapid development project, minimising the integration of disparate components is crucial, as this is where the majority of complications and time delays typically arise (especially as the project focus is centred on the routing algorithm).

Therefore, a critical factor in selecting appropriate communication protocols is the availability and suitability of COTS hardware, specifically systems on chips (SoCs). SoCs are pre-integrated development boards that encompass microprocessors, communication modules, and interfaces necessary for connecting sensors and accessing network features. By focusing on SoCs that offer a comprehensive, all-in-one solution, the system development process can be streamlined, reducing integration challenges, and expediting project timelines.

J SoC Comparison Table

Recognising that integration is typically a lengthy process, the decision was made to use a SoC with all necessary components pre-integrated. This approach allows more time and focus to be dedicated to the core objective of the project: developing the routing algorithm. The table below outlines the specifications of various SoCs that are capable of supporting the functionality needed to meet the client's requirements.

Table 52: SoC comparison table.

Category	TTGO T-Beam Supreme	TTGO T3 v1.6.1	TTGO T-Beam v1.2	Heltec LoRa32 v3	Adafruit Feather ESP32-S3 with LoRa
Price (approx.)	£47.00	£32.00	£33.00	£28.00	£48.00
Processor	ESP-S3	ESP32 Dual-core 240MHz	ESP32 Dual-core 240MHz	ESP32-S3FN8 Dual-core 240MHz	ESP32-S2 Single-core 240MHz
Flash	8MB	4MB	4MB	8MB	4MB
RAM	520KB SRAM	520KB SRAM	520KB SRAM	520KB SRAM	320KB SRAM
LoRa Module	SX1262 (longer range, lower power)	SX1276	SX1276	SX1262 (longer range, lower power)	SX1276
Supported Protocols	LoRa, Wi-Fi, Bluetooth 5 (LE), GPS	LoRa, Wi-Fi, Bluetooth (LE)	LoRa, Wi-Fi, Bluetooth (LE), GPS	LoRa, Wi-Fi, Bluetooth (LE)	LoRa, Wi-Fi, Bluetooth (LE)
On-Board GPS	L76K	No (external GPS module required)	u-blox NEO-6M GPS Module	No (external GPS module required)	No (external GPS module required)
Power Management	AXP2101	No PMU	AXP2101	AXP192	No PMU
Battery Connector	Battery Holder	Yes (JST 2.0 for LiPo)	Battery Holder	Yes (JST 2.0 for LiPo)	Yes (JST 2.0 for LiPo)
MicroSD Card Reader	Yes	Yes	No	No	No
Certifications	CE	CE	CE	CE	CE
Additional Features	QMI8658 IMU, 1.3" OLED Display, Air pressure sensor, Magnetometer, Real-time clock (RTC)	MicroSD Card Reader, 0.96" OLED Display	GPS, AXP2101, PMU, 0.96" OLED Display	0.96" OLED Display, PMU	USB-C, Feather form-factor
Size	60x25mm	51x25mm	68x25mm	50x25mm	41x15mm
Additional Components Required	None	GPS Module	MicroSD Card Reader	GPS module, MicroSD Card Reader	GPS module, MicroSD Card Reader
USB Interface	USB-C	Micro-USB	Micro-USB	USB-C	USB-C

The TTGO T-Beam Supreme was chosen over other SoCs as it fulfils all the critical project requirements: integrated GPS, long-range LoRa communication, high-power Wi-Fi/Bluetooth communication, 8GB random-access memory (RAM), onboard data storage via a microSD card reader, and robust sensor capabilities. In addition to these core features, it offers advanced functionality, including a power management unit (PMU), battery holder, 1.3" OLED display, triaxial inertial measurement unit (IMU), air pressure sensor, magnetometer, and a real-time clock (RTC).

K LilyGO T-Beam Supreme Specification

Table 53: LilyGO T-Beam Supreme Specification Table

Property	Specification
General Properties	
Brand	LilyGO LilyGO
Manufacturer model/SKU	H661
Model revision	V3.0
Features	Battery charger Display Microcontroller SD-card adapter Battery holder Temperature sensor Humidity sensor Pressure sensor Acceleration sensor Gyroscope Battery protection circuit RTC Programmable button(s)
Physical Properties	
Main color	Black
Weight [g]	56 (incl. antennas)
Dimensions (L x W x H) [mm]	114.6 x 33 x 28
Form factor	Module (general)
Electrical Properties	
Minimum supply voltage [V DC]	5
Maximum supply voltage [V DC]	5
IO-pin input/output voltage [V]	3.3
Maximum IO-pin source current [mA]	40
Maximum IO-pin sink current [mA]	28

Table 54: LilyGO T-Beam Supreme Specification Table

Property	Specification
MCU	
Manufacturer	Espressif
Chip series	ESP32-S3FN8
Maximum clock frequency [MHz]	240
Number of cores	2 (Xtensa LX7)
Internal Memory	
SRAM [KB]	512
PSRAM [MB]	8
Flash memory [MB]	8
EEPROM type	Emulated
Communication Interfaces	
Hardware interface(s)	I2C, I2S, SPI, Native USB, USB CDC, UART, PWM, ADC and GPIO
Number of IO-pins	10
Hardware PWM channels	8
Analog input pins (ADC)	2
Analog output pins (DAC)	0
Hardware UART interfaces	3
Hardware I2C interfaces	2
Hardware SPI interfaces	2
USB to serial converter	Integrated into microcontroller
Supported SD card size	Micro SD
Wireless Communication	
Interface(s)	GNSS, Wi-Fi, Bluetooth, LoRa
Wi-Fi chip/SoC	ESP32-S3
Wi-Fi frequency [GHz]	2.4
Bluetooth version	5.0 (LE)
GNSS chip/SoC	Quectel L76K
Supported GNSS systems	GPS, Beidou, GLONASS
LoRa chip/SoC	SX1262
Display	
Type	OLED
Driver	SH1106
Screen diagonal [inch]	1.3
Resolution (pixels)	128 x 64
Display shape	Rectangular
Pixel colors	White

Table 55: LilyGO T-Beam Supreme Specification Table

Property	Specification
Sensors	
Temperature Sensor	BME280 (-40 to 85°C, 0.01°C resolution, ±1.0°C accuracy)
Humidity Sensor	BME280 (0-100% R.H., 0.008% resolution, ±3% accuracy)
Pressure Sensor	BME280 (300-1100 hPa, 0.0018 hPa resolution, ±1.0 hPa accuracy)
Acceleration Sensor	QMI8658C (±16g, 3 DOF, 16-bit resolution)
Gyroscope	QMI8658C (±2048°/s, 3 DOF, 16-bit resolution)
Magnetometer	QMC6310 (0-3 mT, Compass, 3 DOF, ±2° acc.)
RTC	
RTC chip	PCF8563
Back-up battery	Yes
Battery Circuit	
Battery charger chip/SoC	AXP2101
Supported battery chemistry	Li-ion, Li-Po
Maximum charging voltage [V]	4.2
Maximum charging current [mA]	1000
Adjustable charging current	Yes
Safety features	Short circuit Overvoltage Overcurrent Polarity Undervoltage (UVP 2.4V)
Battery Holder	
Battery form factor	18650 (Flat top preferred)
Suitable number of batteries	1
Connectors	
Power supply connector(s)	USB-C Through hole solder pin(s) (2.54mm)
IO-connector(s)	JST-SH compatible male Through hole solder pin(s) (2.54mm)
Antenna connector(s)	LoRa (IPEX MHF 1/uFL, SMA female) GNSS (Built-in, IPEX MHF 1/uFL)
Package Contents	
Package contents	1x T-Beam-S3 Supreme 1x LoRa SMA antenna 2x Male pin header (13p) 1x Nut 1x Tooth lock washer 1x Spring washer

L CE Attestations of Conformity: TTGO T-Beam Supreme

Upon request, the manufacturer of the selected SoC provided the following attestations of conformity (AoCs). The devices have been awarded the CE mark after successfully passing the required tests.

L.1 RED AoC

BAY AREA COMPLIANCE LABORATORIES CORPORATION
www.baclcorp.com www.baclcorp.com.cn
bacltest@baclcorp.com

ATTESTATION OF CONFORMITY

Attestation Number:SZ1240319-13904E-RF
Date of Issue:2024-08-21

Manufacturer:
Company name: Shenzhen Xin Yuan Electronic Technology Co., Ltd.
Address: Room 801-803, Yousuowei Building, No.2000 JiaXian Road, Bantian Street, Longgang District, Shenzhen, Guangdong China

Product:
Name: T-BEAM-S3
Model(s): T-BEAM-S3, T-BEAM-S3(L76K GPS)

Trade Mark: LILYGO

Bay Area Compliance Laboratories Corp. (Shenzhen) hereby declares that the submitted sample(s) of the above equipment has been tested for CE regulations and in accordance with the European Directives and Standards:

Radio Equipment Directive 2014/53/EU

Essential Requirements		Harmonized Standards	Test Report Number
RED Article 3.2	Radio	ETSI EN 300 328 V2.2.2 (2019-07); ETSI EN 300 220-1 V3.1.1 (2017-02); ETSI EN 300 220-2 V3.2.1 (2018-06) ETSI EN 303 413 V1.2.1 (2021-04)	SZ1240319-13904E-RF-22A SZ1240319-13904E-RF-22B SZ1240319-13904E-RF-22C SZ1240319-13904E-RF-22D
RED Article 3.1(b)	EMC	ETSI EN 301 489-1 V2.2.3 (2019-11); ETSI EN 301 489-3 V2.3.2 (2023-01); ETSI EN 301 489-17 V3.2.4 (2020-09); ETSI EN 301 489-19 V2.2.1 (2022-09)	SZ1240319-13904E-EM-02
RED Article 3.1(a)	Safety	EN IEC 62368-1:2020+A11:2020	SZ1240319-13904E-SF
RED Article 3.1(a)	Health	EN IEC 62311:2020; EN 50665:2017	SZ1240319-13904E-RF

CE

Mark is permitted only after all applicable requirements are met in accordance with the CE regulation requirements, including the manufacturer's issuance of a "Declaration of Conformity. The Declaration of Conformity is issued under the sole responsibility of the manufacturer. This attestation is specific to the standard(s) stated above and compliance with additional standards and/or CE regulations are applicable.

Attestation by:
RF Engineer: Nancy Wang
Signature:

C03-C2 (210115) PAGE1 OF 1

L.2 EMC AoC

BAY AREA COMPLIANCE LABORATORIES CORPORATION
www.baclcorp.com www.baclcorp.com.cn
baclltest@baclcorp.com

ATTESTATION OF CONFORMITY

Attestation Number:SZ1240319-13904E-EM
Date of Issue:2024-08-21

Manufacturer:

Company name: Shenzhen Xin Yuan Electronic Technology Co., Ltd.
Address: Room 801-803, Yousuowei Building, No.2000 JiaXian Road, Bantian Street, Longgang District, Shenzhen, Guangdong China

Product:

Name: T-BEAM-S3
Model(s): T-BEAM-S3, T-BEAM-S3(L76K GPS)

Trade Mark:

LILYGO

Bay Area Compliance Laboratories Corp. (Shenzhen) hereby declares that the submitted sample(s) of the above equipment has been tested for CE regulations and in accordance with the European Directives and Standards:

EMC Directive 2014/30/EU

Essential Requirements		Harmonized Standards	Test Report Number
EMCD Clause 1(a)	Emission	EN 55032:2015+A1:2020	SZ1240319-13904E-EM-01
EMCD Clause 1(b)	Immunity	EN 55035:2017+A11:2020	SZ1240319-13904E-EM-01

Mark is permitted only after all applicable requirements are met in accordance with the CE regulation requirements, including the manufacturer's issuance of a "Declaration of Conformity. The Declaration of Conformity is issued under the sole responsibility of the manufacturer. This attestation is specific to the standard(s) stated above and compliance with additional standards and/or CE regulations are applicable.

Attestation by:

EMC Engineer: Moon Liu

Signature:

C02-C2 (210115) PAGE1 OF 1

M Test Reports and Findings

M.1 Unit Testing Overview

Table 56: Overview of unit testing.

Test Type	Test Objective	Tested Unit/Subsystem/System	Desired Outcome	Status	Remarks
Unit	Verify communication with the ESP32-S3 microprocessor and ensure proper execution of the loaded code.	ESP32-S3 Microprocessor Unit	The IDE can detect, connect, compile, upload, and run the code on the MCU without fault in communication between the devices.	Pass	N/A
Unit	Verify communication with the SH1106 OLED display and validate correct rendering of visual outputs.	SH1106 OLED Display	The MCU can connect to the OLED display via the I2C bus and run code that displays text clearly.	Pass	N/A
Unit	Verify communication with the BME280 sensor to ensure accurate readings of temperature, humidity, and pressure.	BME280 Sensor	The MCU can connect to the BME280 sensor via the I2C bus and run code that reads valid ambient temperature, pressure, and humidity values.	Pass	N/A
Unit	Verify communication with the SX1262 LoRa transceiver module to validate data transmission and reception capabilities.	SX1262 LoRa Transceiver Module	A message is successfully transmitted from one node and correctly received by another node via LoRa communication, with no packet loss or errors in the transmitted data.	Pass	N/A
Unit	Verify the functionality of the WiFi/BLE transceiver module, validating reliable connection and data exchange with devices over WiFi.	WiFi/BLE Transceiver Module	A message is successfully transmitted from one node and correctly received by another node via Wi-Fi communication, with errors in the transmitted data.	Pass	N/A
Unit	Verify the functionality of the WiFi/BLE transceiver module, validating reliable connection and data exchange with devices over BLE.	WiFi/BLE Transceiver Module	A message is successfully transmitted from one node and correctly received by another node via BLE communication, with no packet loss or errors in the transmitted data.	Pass	N/A
Unit	Verify communication with the AXP2101 to confirm proper voltage regulation and power supply management.	AXP2101 PMU	1. The PMU is successfully identified by the MCU. 2. The PMU can return intrinsic power parameters. 3. The PMU is able to power peripherals.	Pass	Manufacturer pinmap states PMU pins as SDA=41 and SCL=42, however, PMU is only interfaceable when assigning the pins as SDA=42 and SCL=41.
Unit	Verify communication with the L76K GNSS module and ensure accurate acquisition of satellite signals for location data.	L76K GNSS Module	The MCU successfully initialises the L76K GNSS module and valid latitude and longitude values are read.	Pass	GNSS module only interfaceable when the appropriate voltage regulator is enabled and GNSS powered to 3.3V by PMU.
Unit	Verify functionality of the MicroSD card reader, ensuring successful read and write operations with stored data.	MicroSD Card Reader	The MCU successfully initialises the MicroSD Card Reader and successfully writes to and reads from the mounted filesystem.	Pass	Test pass was achieved with a minor code change during the write-up, not implemented in the decision engine, but uploaded to GitHub.
Unit	Verify communication with the QMI8658 Inertial Measurement Unit (IMU) and validate the accuracy of motion and orientation data.	QMI8658 IMU	The MCU successfully initializes the QMI8658 IMU and valid sensor values are read.	Fail	Device WHO_AM_I Control register not returning correct hexcode value (should be 0x05 as shown in the sensor datasheet). It is suggested that the Meshtastic firmware is downloaded and searched through to find C++ code that interfaces with the IMU.

M.2 Unit Testing

M.2.1 MCU Test

M.2.1.1 MCU Test Report

Table 57: Unit Test Report: ESP32-S3 Communication Test

Test Type:	Unit
Module:	ESP32-S3
Test Details:	The test verifies the communication between the integrated development environment (IDE) (Thonny) and the microprocessor unit (MCU) (ESP32-S3). This test ensures that code can both be written onto, and be processed by, the MCU.
Pass Criteria:	The IDE can detect, connect, compile, upload, and run the code on the MCU without fault in communication between the devices.
Fail Criteria:	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) • Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code that interacts with the MCU. 4. Upload and run the code on the MCU. 5. Monitor the response and record the results based on the terminal outputs.
Test Code Output:	ESP32 Chip Information: Unique ID (MAC address): 48ca435ba4a4 Flash Size: 8 MB
Test Results:	The Unique ID (MAC address) differs from the development platform's, and the flash size matches the specifications of the ESP32-S3 MCU. Successful retrieval of these parameters indicates that communication between the MCU and the IDE is properly established. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.1.2 MCU Test Code

```
import machine
import esp
import ubinascii

def print_chip_info():
    print("ESP32 Chip Information:")

    unique_id = machine.unique_id()
    mac_address = ubinascii.hexlify(unique_id).decode()
    print(f"Unique ID (MAC address): {mac_address}")

    flash_size = esp.flash_size()
    print(f"Flash Size: {flash_size // (1024 * 1024)} MB")

print_chip_info()
```

M.2.2 OLED Dislay Test

M.2.2.1 OLED Test Report

Table 58: Unit Test Report: SH1106 OLED Display Communication Test

Test Type:	Unit
Module:	SH1106 OLED Display
Test Details:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the OLED display (SH1106) connected via I2C. It ensures that the MCU successfully interfaces with the OLED screen over the I2C bus, displaying visual outputs accurately.
Pass Criteria:	The MCU can connect to the OLED display via the I2C bus and run code that displays text clearly.
Fail Criteria:	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) • Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code (shown below) that interfaces with the SH1106 OLED display. 4. Upload and run the code on the MCU. 5. Monitor the response and record the results based on the OLED screen's output.
Test Code Output:	The specified text "Testing123" is displayed on the OLED screen.
Test Results:	The OLED screen is successfully interfaced via the I2C bus, and an oled object is created using the SH1106_I2C class from the sh1106 driver library. The specified text is displayed clearly on the OLED screen. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.2.2 OLED Test Code

```
from machine import Pin, I2C
import sh1106  # OLED display driver library
import time

# Define I2C pins for TTGO T-BEAM SUPREME
sda = Pin(17)  # SDA pin
scl = Pin(18)  # SCL pin

# Create I2C interface
i2c = I2C(0, scl=scl, sda=sda, freq=400000)

# Initialise the SH1106 OLED display (128x64 resolution)
oled = sh1106.SH1106_I2C(128, 64, i2c)

# Clear the display
oled.fill(0)

# Display text
oled.text("Testing123", 0, 0)

# Update the display to show the text
oled.show()
```

M.2.3 BME280 Sensor Test

M.2.3.1 BME280 Test Report

Table 59: Unit Test Report: BME280 Sensor Communication Test

Test Type:	Unit
Module:	BME280 Sensor
Test Details:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the temperature, pressure, and humidity sensor (BME280) connected via I2C. It ensures that the MCU successfully interfaces with the BME280 sensor over the I2C bus, reading valid ambient values.
Pass Criteria:	The MCU can connect to the BME280 sensor via the I2C bus and run code that reads valid ambient temperature, pressure, and humidity values.
Fail Criteria:	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) • Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code (shown below) that interfaces with the BME280 sensor. 4. Upload and run the code on the MCU. 5. Monitor the response and record the results based on the terminal outputs.
Test Code Output:	Temperature: 22.6 °C Pressure: 114270.0 Pa Humidity: 75.0 %
Test Results:	The BME280 sensor is successfully interfaced via the I2C bus, and a tph_sensor object is created using the BME280 class from the bme280 driver library. The outputted ambient temperature, pressure, and humidity values are displayed in the terminal (as shown above). These values were validated against another calibrated sensor. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.3.2 BME280 Test Code

```
from machine import I2C, Pin
from bme280 import BME280  # BME class from bme280 driver library

sda = Pin(17)  # SDA pin
scl = Pin(18)  # SCL pin

# Create I2C interface
i2c = I2C(0, scl=scl, sda=sda, freq=400000)

# Setup bme280 sensor
tph_sensor = BME280(i2c)

# Read all values from sensor
sensor_data = tph_sensor.read_all()

# Format the individual readings to 2 decimal places for readability
temperature = f"{sensor_data['temperature']:.1f}"
# conversion to Pa from hPa
pressure = float(f"{sensor_data['pressure']:.1f}") * 100
humidity = f"{sensor_data['humidity']:.1f}"

# Print current temperature, pressure, and humidity readings to the
# output terminal.
print("Temperature: ", temperature)
print("Pressure: ", pressure)
print("Humidity: ", humidity)
```

M.2.4 LoRa Test

M.2.4.1 LoRa Test Report

Table 60: Unit Test Report: SX1262 LoRa Transceiver Communication Test

Test Type:	Unit
Tested Module:	SX1262 LoRa Transceiver
Test Description:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the SX1262 LoRa transceiver module. It ensures that the MCU successfully transmits and receives messages via LoRa communication, with no packet loss or data errors.
Pass Criteria:	A message is successfully transmitted from one node and correctly received by another node via LoRa communication, with no packet loss or errors in the transmitted data.
Fail	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) •Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code (shown below) that interfaces with the SX1262 LoRa module. The transmitter code should be uploaded to one node, and the receiver code to another. 4. Upload and run the code on the MCU - the receiver code should be run first to ensure the receiver is listening before the transmission occurs. 5. Monitor the response and record the results based on the terminal outputs.
Test Code Output (Receiver):	`Receiver initialised successfully."` `Received message from source \x02: testingTXRX."`
Test Code Output (Transmitter):	`Transmitter initialised successfully.` `Sent message to node: \x01 Message: testingTXRX`
Test Results:	The SX1262 LoRa transceiver modules successfully transmitted and received messages via LoRa communication, with no packet loss or errors in the transmission. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.4.2 LoRa Test Receiver Code

```
from machine import Pin
import time
from sx1262 import SX1262

# Pin assignments for SX1262
SPI_BUS = 2                 # SPI bus
CLK_PIN = Pin(12)           # SCLK pin
MOSI_PIN = Pin(11)          # MOSI pin
MISO_PIN = Pin(13)          # MISO pin
CS_PIN = Pin(10, Pin.OUT)   # Chip Select
RST_PIN = Pin(5, Pin.OUT)   # Reset
GPIO_PIN = Pin(4, Pin.IN)   # DIO1
IRQ_PIN = Pin(1, Pin.IN)    # IRQ pin

# Create SX1262 LoRa transceiver object
LORA = SX1262(spi_bus=SPI_BUS, clk=CLK_PIN, mosi=MOSI_PIN,
              miso=MISO_PIN, cs=CS_PIN, irq=IRQ_PIN,
              rst=RST_PIN, gpio=GPIO_PIN)

# Reset the module
RST_PIN.value(0)  # Set RST low
time.sleep(0.1)   # Wait for 100 ms
RST_PIN.value(1)  # Set RST high
time.sleep(0.5)   # Wait for 500 ms

# Begin communication with SX1262
LORA.begin(freq=868, bw=500.0, sf=12, cr=8, syncWord=0x12,
         power=-5, currentLimit=60.0, preambleLength=8,
         implicit=False, implicitLen=0xFF,
         crcOn=True, txIq=False, rxIq=False,
         tcxoVoltage=1.7, useRegulatorLDO=False, blocking=True)

print("Receiver initialised successfully.")

# Define the unique addresses for the target and source node(s)
TARGET_ADDR = b'\x01'
SOURCE_ADDR = b'\x02'

# Function to handle received messages
def cb(EVENTS):
    if EVENTS & SX1262.RX_DONE:
        MESSAGE, ERR = LORA.recv()
        ERROR = SX1262.STATUS[ERR]
        if ERROR == 'ERR_NONE':
            # Extract the destination and source addresses
            # First byte is the destination address
            TARGET = MESSAGE[0:1]
            # Second byte is the source address
            SOURCE = MESSAGE[1:2]
            # The rest of the message is the actual payload
            PAYLOAD = MESSAGE[2:]
```

```
            # Check if the message is for this node
            if TARGET == TARGET_ADDR and SOURCE == SOURCE_ADDR:
                print('Received message from source', SOURCE, ':',
                      PAYLOAD)
            else:
                print('Message not for this node or from unexpected
                      source. Ignored.')

        else:
            print('Received message with error:', ERROR)
        print('Received message:', MESSAGE, 'with status:', ERROR)

# Set callback for receiving messages
LORA.setBlockingCallback(False, cb)  # Use non-blocking callback

while True:
    time.sleep(1)  # Keep the loop running
```

M.2.4.3 LoRa Test Transmitter Code

```
from machine import SPI, Pin
import time
from sx1262 import SX1262

# Pin assignments for SX1262
SPI_BUS = 2
CLK_PIN = Pin(12)
MOSI_PIN = Pin(11)
MISO_PIN = Pin(13)
CS_PIN = Pin(10, Pin.OUT)  # Chip Select
RST_PIN = Pin(5, Pin.OUT)  # Reset
GPIO_PIN = Pin(4, Pin.IN)  # DIO1
IRQ_PIN = Pin(1, Pin.IN)   # IRQ pin

# Create SX1262 LoRa transceiver object
LORA = SX1262(spi_bus=SPI_BUS, clk=CLK_PIN, mosi=MOSI_PIN,
              miso=MISO_PIN, cs=CS_PIN, irq=IRQ_PIN,
              rst=RST_PIN, gpio=GPIO_PIN)

# Reset the module
RST_PIN.value(0)  # Set RST low
time.sleep(0.1)  # Wait for 100 ms
RST_PIN.value(1)  # Set RST high
time.sleep(0.5)  # Wait for 500 ms

# Begin communication with SX1262
LORA.begin(freq=868, bw=500.0, sf=12, cr=8, syncWord=0x12,
         power=-5, currentLimit=60.0, preambleLength=8,
         implicit=False, implicitLen=0xFF,
         crcOn=True, txIq=False, rxIq=False,
         tcxoVoltage=1.7, useRegulatorLDO=False, blocking=True)

print("Transmitter initialised successfully.")

# Define the unique addresses for the target and source node(s)
TARGET_ADDR = b'\x01'
SOURCE_ADDR = b'\x02'

while True:

    # Create a message with addressing
    MESSAGE = TARGET_ADDR + SOURCE_ADDR + b'testingTXRX'

    # Send the LoRa message
    LORA.send(MESSAGE)
    print("Sent message to node:", TARGET_ADDR, "Message:", MESSAGE)

    time.sleep(5)  # Wait before sending the next message
```

M.2.5 Wi-Fi Test

M.2.5.1 Wi-Fi Test Report

Table 61: Unit Test Report: Wi-Fi Transceiver Communication Test

Test Type:	Unit
Tested Module:	Wi-Fi Transceiver
Test Description:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the on-chip Wi-Fi transceiver. It ensures that the MCU successfully transmits and receives messages via Wi-Fi communication, with no packet loss or data errors.
Pass Criteria:	A message is successfully transmitted from one node and correctly received by another node via Wi-Fi communication, with no packet loss or errors in the transmitted data.
Fail	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) •Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code (shown below) that interfaces with the on-chip Wi-Fi transceiver. The station (transmitter) code should be uploaded to one node, and the access point (receiver) code to another. 4. Upload and run the code on the MCU - the access point code should be run first to ensure the access point is listening before the transmission occurs. 5. Monitor the response and record the results based on the terminal outputs.
Test Code Output (Receiver):	`Receiver: Listening for message...` `Received message: "Hello, Server!"`

Test Code Output (Transmitter):	`Transmitter: Sending message "Hello, Server!"`
	`Request sent: GET /?msg=Hello,Server!HTTP/1.0\r\nHost: 192.168.4.1\r\n\r\n`
	`Response from server: HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n<html><body><h1>Message received by Access Point!</h1></body></html>\r\n`

Test Results:	The Wi-Fi transceivers successfully transmitted and received messages via Wi-Fi communication, with no packet loss or errors in the transmission. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.5.2 Wi-Fi Access Point Test Code

```
import network
import socket

# Set up Access Point (AP) credentials
# SSID of the AP
SSID = 'The Coke Can Project'
# Password for the AP
PASSWORD = 'Coke Can'
# Port number for the web server
PORT = 80

def SETUP_ACCESS_POINT(SSID, PASSWORD):
    AP = network.WLAN(network.AP_IF)
    AP.active(True)
    AP.config(essid=SSID, password=PASSWORD,
              authmode=network.AUTH_WPA2_PSK)

    # Print the IP address of the access point
    print('Access Point IP Address:', AP.ifconfig()[0])
    return AP

def START_WEB_SERVER():
    ADDR = ('0.0.0.0', PORT)  # Bind to all interfaces on port 80
    S = socket.socket()
    S.bind(ADDR)  # Bind the socket to the address
    S.listen(1)
    print('Listening on', ADDR)

    return S

# Allow connection from client then receive and decode GET request
def SERVE_CLIENT(S):
    CL, ADDR = S.accept()
    print('Client connected from', ADDR)
    REQUEST = CL.recv(1024)

    # Decode the request
    REQUEST_STR = REQUEST.decode()

    # Parse the request to get the message if it exists
    MESSAGE = None
    if "GET" in REQUEST_STR:
        # Look for the 'msg=' parameter
        MSG_START = REQUEST_STR.find("msg=")
        if MSG_START != -1:
            # Move past 'msg=' to get to the start of the message
            MSG_START += len("msg=")
            # Find the next '&' or end of line
            MSG_END = REQUEST_STR.find("&", MSG_START)
            # If no '&' is found, take everything until the end of the line
            if MSG_END == -1:
```

```
                # In case no '&', look for space
                MSG_END = REQUEST_STR.find(" ", MSG_START)
                # If no space, take till the end of the string
                if MSG_END == -1:
                    MSG_END = len(REQUEST_STR)
            # Extract the message
            MESSAGE = REQUEST_STR[MSG_START:MSG_END]
            print("Received message:", MESSAGE)

    # Serve a simple HTML page
    RESPONSE = """HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n
                <html><body><h1>Message received by Access Point!<
                /h1></body></html>\r\n"""
    CL.send(RESPONSE)
    CL.close()

# Deactivate the AP
def SHUTDOWN_ACCESS_POINT(AP):
    print('Shutting down access point...')
    AP.active(False)
    print('Access point shut down.')

def MAIN():
    # Set up the access point
    AP = SETUP_ACCESS_POINT(SSID, PASSWORD)
    # Start the web server
    SERVER_SOCKET = START_WEB_SERVER()

    try:
        # Serve a client request
        SERVE_CLIENT(SERVER_SOCKET)
    finally:
        # Ensure access point is shut down
        SHUTDOWN_ACCESS_POINT(AP)

# Run the main function
if __name__ == "__main__":
    MAIN()
```

M.2.5.3 Wi-Fi Station Test Code

```
import network
import socket
import time

# Define Access Point (AP) credentials
# SSID of the Access Point (AP)
AP_SSID = 'The Coke Can Project'
# Password for the AP
AP_PASSWORD = 'Coke Can'
# IP address of the web server
SERVER_IP = '192.168.4.1'
# Port number for the web server
PORT = 80

# Setup station (STA) node and try to connect to AP
def SETUP_WIFI(SSID, PASSWORD):
    WLAN = network.WLAN(network.STA_IF)
    WLAN.active(True)
    WLAN.connect(SSID, PASSWORD)
    print('Connecting to Access Point...')
    while not WLAN.isconnected():
        time.sleep(1)
    print('Connected to Access Point!')

    # Print the IP address assigned by the access point
    print('IP Address:', WLAN.ifconfig()[0])
    return WLAN

# Connect to the correct server and port, and send a GET request.
def SEND_HTTP_REQUEST(SERVER_IP, PORT, MESSAGE):
    # Setup and connect to the AP
    S = socket.socket()
    S.connect((SERVER_IP, PORT))

    # Prepare the HTTP GET request with a message as a query parameter
    REQUEST ='GET /?msg={}&HTTP/1.0\r\nHost: {}\r\n\r\n'.format(
                                                        MESSAGE,
                                                        SERVER_IP
                                                        )

    # Send the request
    S.send(REQUEST.encode())
    print('Request sent:', REQUEST)

    # Receive the response
    RESPONSE = S.recv(1024)
    print('Response from server:', RESPONSE)

    # Close the socket
    S.close()
    return RESPONSE
```

```
# Decode the resposne recieved from the AP
def DECODE_RESPONSE(RESPONSE):
    RESPONSE_STR = RESPONSE.decode()

    # Separate the headers and body of the HTTP response
    HEADER_END = RESPONSE_STR.find("\r\n\r\n")
    if HEADER_END != -1:
        # The body starts after "\r\n\r\n"
        BODY = RESPONSE_STR[HEADER_END + 4:]

        # Find the message within the HTML tags
        START = BODY.find('<h1>')
        END = BODY.find('</h1>')
        if START != -1 and END != -1:
            # Extract and print the message between <h1> and </h1>
            MESSAGE = BODY[START + 4:END]
            print('Response message:', MESSAGE)
        else:
            print("Message not found in response.")
    else:
        print("Malformed HTTP response.")

# Deactivate the station interface
def SHUTDOWN_WIFI(WLAN):
    print('Shutting down Wi-Fi connection...')
    WLAN.active(False)

def MAIN():
    # Set up Wi-Fi connection
    WLAN = SETUP_WIFI(AP_SSID, AP_PASSWORD)
    # Message to send
    MESSAGE_TO_SEND = "Hello, Server!"
    # Send HTTP request
    RESPONSE = SEND_HTTP_REQUEST(SERVER_IP, PORT, MESSAGE_TO_SEND
    # Decode response from AP
    DECODE_RESPONSE(RESPONSE)
    # Shut down Wi-Fi connection
    SHUTDOWN_WIFI(WLAN)

# Run the main function
if __name__ == "__main__":
    MAIN()
```

M.2.6 BLE Test

M.2.6.1 BLE Test Report

Table 62: Unit Test Report: BLE Transceiver Communication Test

Test Type:	Unit
Tested Module:	BLE Transceiver
Test Description:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the on-chip BLE transceiver. It ensures that the MCU successfully transmits and receives messages via BLE communication, with no packet loss or data errors.
Pass Criteria:	A message is successfully transmitted from one node and correctly received by another node via BLE communication, with no packet loss or errors in the transmitted data.
Fail Criteria:	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) • Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code (shown below) that interfaces with the on-chip BLE transceiver. The client (transmitter) code should be uploaded to one node, and the server (receiver) code to another. 4. Upload and run the code on the MCU - the server code should be run prior to the client code to ensure the server is listening before a transmission occurs. 5. Monitor the response and record the results based on the terminal outputs.

Test Code Output (Receiver):	`Waiting for a connection...`
	`Connected to: <device_info>`
	`Received message: TestingBLE`

Test Code Output (Transmitter):	`Connecting to <device_info>...`
	`Sending message: TestingBLE`
	`Message sent`

Test Results:	The BLE transceivers successfully transmitted and received messages via BLE communication, with no packet loss or errors in the transmission. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.6.2 BLE Server Test Code

```
import asyncio
import aioble
import bluetooth

# Define UUIDs for the BLE service and characteristic.
MESSAGE_SERVICE_UUID = bluetooth.UUID(0x1234)
MESSAGE_CHARACTERISTIC_UUID = bluetooth.UUID(0x1235)

# Create a GATT server with a characteristic to receive messages
# MESSAGE_SERVICE is the BLE service, and MESSAGE_CHARACTERISTIC is the
# BLE characteristic
MESSAGE_SERVICE = aioble.Service(MESSAGE_SERVICE_UUID)
MESSAGE_CHARACTERISTIC = aioble.Characteristic(
                                            MESSAGE_SERVICE,
                                            MESSAGE_CHARACTERISTIC_UUID,
                                            write=True
                                            )

# Register the GATT service so that the BLE server is aware of this
# service and characteristic
aioble.register_services(MESSAGE_SERVICE)

# Asynchronous task to handle received messages from the client
async def MESSAGE_TASK(CONNECTION):
    try:
        # Keep the connection open and listen for incoming messages
        # Timeout set to None (but configurable to any time)
        with CONNECTION.timeout(None):
            while True:
                # Wait for the client to write data to the
                # characteristic (non-blocking)
                await MESSAGE_CHARACTERISTIC.written()

                # Read the message written to the characteristic,
                # decode it, and print it
                MSG = MESSAGE_CHARACTERISTIC.read().decode()
                print(f"Received message: {MSG}")

    # If the client disconnects, this exception will be caught, and
    # the task will stop
    except aioble.DeviceDisconnectedError:
        print("Client disconnected.")
        return

# Asynchronous task to handle advertising and waiting for connections
async def PERIPHERAL_TASK():
    while True:
        print("Waiting for a connection...")

        # Start BLE advertising to allow clients to
        # find and connect to this server
```

```
        # The server advertises the name "BLE-Message-Server" and the
        # service UUID
        CONNECTION = await aioble.advertise(500000,
                                            name="BLE-Message-Server",
                                            services=[MESSAGE_SERVICE_UUID]
                                            )

        # Once a client connects, print the device information
        print("Connected to:", CONNECTION.device)

        # Start the task to handle receiving messages after
        # a connection is established
        await MESSAGE_TASK(CONNECTION)

# Main function to start the peripheral task.
async def MAIN():
    # Start the peripheral task, which will handle the advertising
    # and connections
    await PERIPHERAL_TASK()

# Run the BLE server and handle KeyboardInterrupt to allow clean exit
try:
    # Start the asynchronous loop with the main task
    asyncio.run(MAIN())
except KeyboardInterrupt:
    # Handle case where the program is interrupted (Ctrl+C), and exit.
    print("\nProgram interrupted. Exiting...")
```

M.2.6.3 BLE Client Test Code

```
import asyncio
import aioble
import bluetooth

# UUIDs for the BLE service and characteristic
MESSAGE_SERVICE_UUID = bluetooth.UUID(0x1234)
MESSAGE_CHARACTERISTIC_UUID = bluetooth.UUID(0x1235)

class MessageClient:
    # Initialise the MessageClient
    def __init__(self, device):
        # Define the BLE server device to connect to
        self._device = device
        # Placeholder for the connection object
        self._connection = None

    # Asynchronously connect to the specified device
    async def connect(self):
        print(f"Connecting to {self._device}...")
        # Establish connection with BLE server device
        self._connection = await self._device.connect()
        print("Discovering services...")
        # Discover the service on the connected device using its UUID
        MESSAGE_SERVICE = await self._connection.service(
                                                    MESSAGE_SERVICE_UUID
                                                    )
        # Discover the characteristic (for sending/receiving data)
        # within the service
        self._message_characteristic = await MESSAGE_SERVICE
                            .characteristic(MESSAGE_CHARACTERISTIC_UUID)

    # Asynchronously send a message to the device via the characteristic
    async def send_message(self, MESSAGE):
        print(f"Sending message: {MESSAGE}")
        # Encode the the message and write to the characteristic
        await self._message_characteristic.write(MESSAGE.encode())

    # Disconnect from the device when called if a connection exists
    async def disconnect(self):
        if self._connection:
            # Disconnect from connected device
            await self._connection.disconnect()

# Scan for the server and send a message.
async def main():
    # Scan for BLE devices
    async with aioble.scan(5000, 30000, 30000, active=True) as scanner:
        # Loop through each device within scan results
        async for result in scanner:
            # Check if the device matches the "BLE-Message-Server"
            # and service UUID
```

```
            if result.name() == "BLE-Message-Server" and
                                MESSAGE_SERVICE_UUID in result.services():
                # Store the device for connection
                device = result.device
                # Break out of loop when desired device has been found
                break
        else:
            # If no devices with the desired name and service UUID are
            # found then print results
            print("BLE server not found.")
            return

    # Message to be sent
    message = 'TestingBLE'
    # Create instance of the MessageClient class with result device
    client = MessageClient(device)
    # Connect to the BLE server device
    await client.connect()
    # Send the message to the BLE server device
    await client.send_message(message)
    # Print confirmation of sent message
    print("Message sent")
    # Disconnect from the BLE server device
    await client.disconnect()

# Run the main function in an asynchronous loop
asyncio.run(main())
```

M.2.7 PMU Module Test

M.2.7.1 PMU Test Report

Table 63: Unit Test Report: PMU Communication and Peripheral Control Test

Test Type:	Unit
Tested Module:	AXP2101 PMU
Test Description:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the AXP2101 PMU. It ensures that the MCU is capable of identifying the peripheral, reading intrinsic power metrics, and successfully configuring the power of other peripherals on the SoC through the PMU.
Pass Criteria:	1. The PMU is successfully identified by the MCU. 2. The PMU can return intrinsic power parameters. 3. The PMU is capable of configuring the power of other peripherals.
Fail Criteria:	The tested module fails to meet all of the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) • Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the three separate test codes (shown below): a. PMU Identification Test Code b. PMU Intrinsic Parameters Test Code c. PMU Power Peripherals Test Code 4. Upload and run each code on the MCU. 5. Monitor the response and record the results based on the terminal outputs.
Test Code Output (a):	`PMU initialized, Chip ID: 0x40`

Test Code Output (b):	`Battery: 4.10 V`
	`System: 3.30 V`
	`Battery%: 95%`
	`Charging: Yes`
	`Discharging: No`
	`Standby: No`

Test Code Output (c):	`Setting LDO voltage to 3300 mV (Steps: 28)` `LDO for I2C enabled.`
Test Results:	The PMU is successfully identified and capable of both returning intrinsic power parameters and powering other on-board peripherals. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.7.2 PMU Identification Test Code

```
from AXP2101 import *
from machine import Pin, I2C

# Define pins and initialise I2C bus
SDA = 42
SCL = 41
I2C_BUS1 = I2C(1, scl=Pin(SCL), sda=Pin(SDA))

# Initialise the PMU object with I2C bus
PMU = AXP2101(I2C_BUS1, addr=AXP2101_SLAVE_ADDRESS)

# Check if the PMU is connected and working
id = PMU.getChipID()
if id != XPOWERS_AXP2101_CHIP_ID:
    print("PMU is not online...")
else:
    print('PMU initialised, Chip ID: {}'.format(hex(id)))
```

M.2.7.3 PMU Intrinsic Parameters Test Code

```
from AXP2101 import *
import time
from machine import Pin, I2C
from sh1106 import SH1106_I2C

# Define I2C Pins and Addresses
OLED_SCL_PIN = 18
OLED_SDA_PIN = 17
PMU_SCL_PIN = 41
PMU_SDA_PIN = 42
PMU_ADDR = AXP2101_SLAVE_ADDRESS

# Initialise the OLED I2C bus and OLED display
def INIT_OLED():
    I2C_BUS0 = I2C(0, scl=Pin(OLED_SCL_PIN), sda=Pin(OLED_SDA_PIN))
    OLED = SH1106_I2C(128, 64, I2C_BUS0)
    return OLED

# Initialise the I2C bus for PMU
def INIT_PMU_I2C():
    I2C_BUS1 = I2C(1, scl=Pin(PMU_SCL_PIN), sda=Pin(PMU_SDA_PIN))
    return AXP2101(I2C_BUS1, addr=PMU_ADDR)

# Convert millivolts to volts
def MV_TO_V(MV):
    return MV / 1000

# Convert voltage to battery percentage
def VOLTAGE_TO_PERCENTAGE(VOLTAGE):
    VOLTAGE_PERCENT_MAP = [
        (4.20, 100), (3.60, 0)
    ]
    if VOLTAGE >= 4.20:
        return 100
    elif VOLTAGE <= 3.56:
        return 0
    for i in range(len(VOLTAGE_PERCENT_MAP) - 1):
        V_HIGH, P_HIGH = VOLTAGE_PERCENT_MAP[i]
        V_LOW, P_LOW = VOLTAGE_PERCENT_MAP[i + 1]
        if V_LOW <= VOLTAGE <= V_HIGH:
            PERCENTAGE = P_LOW + (P_HIGH - P_LOW) * ((VOLTAGE - V_LOW) /
                                                     (V_HIGH - V_LOW))
            return round(PERCENTAGE)
    return 0

# Display intrinsic system metrics on OLED
def DISPLAY_SYSTEM_METRICS(PMU, OLED):
    # Retrieve battery and system parameters from PMU
    BATT_VOLTAGE = MV_TO_V(PMU.getBattVoltage())
    SYS_VOLTAGE = MV_TO_V(PMU.getSystemVoltage())
    BATT_PERCENTAGE = VOLTAGE_TO_PERCENTAGE(BATT_VOLTAGE)
```

```
    # Display the parameters onto the OLED screen
    OLED.fill(0)
    OLED.text("Battery: {:.2f} V".format(BATT_VOLTAGE), 0, 0)
    OLED.text("System: {:.2f} V".format(SYS_VOLTAGE), 0, 10)
    OLED.text("Battery%: {}%".format(BATT_PERCENTAGE), 0, 20)
    OLED.text("Charging: {}".format("Yes" if PMU.isCharging()
                                    else "No"), 0, 30)
    OLED.text("Discharging: {}".format("Yes" if PMU.isDischarge()
                                    else "No"), 0, 40)
    OLED.text("Standby: {}".format("Yes" if PMU.isStandby()
                                    else "No"), 0, 50)
    OLED.show()

# Main function to initialise I2C buses, OLED screen and PMU.
# Display metrics to OLED screen.
def MAIN():
    OLED = INIT_OLED()
    PMU = INIT_PMU_I2C()
    if PMU.getChipID() != XPOWERS_AXP2101_CHIP_ID:
        print("PMU is not online...")
        return

    print('PMU initialized, Chip ID: {}'.format(hex(PMU.getChipID())))

    # Continuous loop with exception handling for clean exit
    try:
        while True:
            DISPLAY_SYSTEM_METRICS(PMU, OLED)
            time.sleep(5)
    except KeyboardInterrupt:
        print("Program interrupted by user, exiting...")
    finally:
        # Clear screen on exit
        OLED.fill(0)
        OLED.show()
        print("Exited cleanly.")

if __name__ == "__main__":
    MAIN()
```

M.2.7.4 PMU Power Peripherals Test Code

```
from machine import I2C, Pin

# Define I2C pins and AXP2101 address
# SCL pin
SCL_PIN = 41
# SDA pin
SDA_PIN = 42
# PMU address
AXP2101_ADDR = 0x34

# Register Definitions
# Register to enable linear dropout (LDO) voltage regulator
_AXP2101_LDO_ONOFF_CTRL0 = 0x90
# Register to set LDO voltage
_AXP2101_LDO_VOL0_CTRL = 0x92
# Minimum voltage in mV
_AXP2101_LDO1_VOL_MIN = 500
# Step size in mV
_AXP2101_LDO1_VOL_STEPS = 100

# Initialize I2C communication
I2C_BUS = I2C(1, scl=Pin(SCL_PIN), sda=Pin(SDA_PIN))

def ENABLE_LDO_FOR_I2C(VOLTAGE_MV):
    # Clamp voltage to minimum to ensure module does not receive too
    # much voltage
    VOLTAGE_MV = max(VOLTAGE_MV, _AXP2101_LDO1_VOL_MIN)

    # Calculate the number of steps needed to set the desired voltage
    STEPS = (VOLTAGE_MV - _AXP2101_LDO1_VOL_MIN) //
                _AXP2101_LDO1_VOL_STEPS
    print(f"Setting LDO voltage to {VOLTAGE_MV} mV (Steps: {STEPS})")

    # Set the LDO voltage and enable it
    I2C_BUS.writeto_mem(AXP2101_ADDR,
                        _AXP2101_LDO_VOL0_CTRL,
                        bytes([STEPS]))
    # Enable LDO0
    I2C_BUS.writeto_mem(AXP2101_ADDR,
                        _AXP2101_LDO_ONOFF_CTRL0,
                        bytes([0x01]))
    print("LDO for I2C enabled.")

# Enable the LDO powering I2C bus with a voltage of 3300 mV
ENABLE_LDO_FOR_I2C(3300)
```

M.2.8 GNSS Module Test

M.2.8.1 GNSS Test Report

Table 64: Unit Test Report: GNSS Communication and Location Acquisition Test

Test Type:	Unit
Tested Module:	L76K GNSS Module
Test Description:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the L76K GNSS module. It ensures that the MCU successfully initialises the onboard L76K GNSS module and reads valid latitude and longitude values.
Pass Criteria:	The MCU successfully initialises the L76K GNSS module and valid latitude and longitude values are read.
Fail Criteria:	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) • Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code (shown below). 4. Upload and run each code on the MCU. 5. Monitor the response and record the results based on the terminal outputs.

Test Code Output:	`I2C Bus 1 initialised successfully.`
	`GPS is enabled with voltage:  3300 mV`
	`Waiting for fix...`
	`Lat:  51°28.133'N`
	`Lon:  0°0.345'W`

Test Results:	The L76K GNSS is successfully initialised and valid latitude and longitude values are read. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.8.2 GNSS Test Code

```
import time
from machine import Pin, I2C, UART
from AXP2101 import *
from micropyGPS import MicropyGPS

def INIT_I2C1(SDA_PIN=42, SCL_PIN=41):
    I2C_BUS1 = I2C(1, scl=Pin(SCL_PIN), sda=Pin(SDA_PIN))
    print("I2C Bus 1 initialised successfully.")
    return I2C_BUS1

def INIT_PMU(I2C_BUS1, ALDO4_VOLTAGE=3300):
    # Replace with actual I2C address if different
    PMU = AXP2101(I2C_BUS1, addr=0x34)
    # Set the ALDO4 voltage
    PMU.setALDO4Voltage(ALDO4_VOLTAGE)
    # Enable ALDO4
    PMU.enableALDO4()
    print("GPS is enabled with voltage: {} mV".format(ALDO4_VOLTAGE))

# Initialise UART for GPS
def INIT_GPS_UART(TX=8, RX=9):
    UART_GPS = UART(1, baudrate=9600, tx=Pin(TX), rx=Pin(RX))
    return UART_GPS

# Read GPS data from the UART
def READ_GPS(UART):
    # Check if any data is available data is on the UART
    if UART.any():
        # Read available data
        GPS_DATA = UART.read(UART.any())
        # Return the GPS data
        return GPS_DATA
    return None

# Define main function to parse NMEA sentences and print Lat and Lon
# data to the shell
def MAIN():
    # Initialise GPS object for NMEA sentence parser
    GPS = MicropyGPS()
    # Initialise the GPS UART
    UART_GPS = INIT_GPS_UART()

    while True:
        # Read data from GPS
        GPS_DATA = READ_GPS(UART_GPS)
        # Check if any data has been received
        if GPS_DATA:
            # Iterate over each character in the received data
            for CHAR in GPS_DATA:
                # Update the GPS object with each character
                GPS.update(chr(CHAR))
```

```
            # Check if valid GPS latitude and longitude values
            # have been received
            # Validate latitude and longitude
            if GPS.latitude and GPS.longitude:
                # Get latitude as a string
                LATITUDE = GPS.latitude_string()
                # Get longitude as a string
                LONGITUDE = GPS.longitude_string()
                # Print latitude to the shell
                print("Lat: {}".format(LATITUDE))
                # Print longitude to the shell
                print("Lon: {}".format(LONGITUDE))
            else:
                # Indicate that GPS fix is not been established yet
                print("Waiting for fix...")

        # Delay between reads for stability
        time.sleep(1)

# Execute the main function when the script is run directly
if __name__ == "__main__":
    I2C_BUS1 = INIT_I2C1()
    INIT_PMU(I2C_BUS1)
    MAIN()
```

M.2.9 MicroSD Card Reader Test

M.2.9.1 MicroSD Card Reader Test Report

Table 65: Unit Test Report: MicroSD Card Reader Communication and File Operations Test

Test Type:	Unit
Tested Module:	MicroSD Card Reader
Test Description:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the MicroSD Card Reader. It ensures that the MCU successfully initialises the module and can write to and read from the filesystem.
Pass Criteria:	The MCU successfully initialises the MicroSD Card Reader and successfully writes to and reads from the mounted filesystem.
Fail Criteria:	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) • Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code (shown below). 4. Upload and run each code on the MCU. 5. Monitor the response and record the results based on the terminal outputs.

Test Code Output:	`Filesystem mounted successfully`
	`SD Card files: ['System Volume Information', 'test.txt']`
	`File written with: Test Data - Testing write functionality`
	`Read from file: Test Data - Testing write functionality`
	`File content read successfully:`
	`Test Data - Testing write functionality`

Test Results:	The MCU successfully initialised the MicroSD Card Reader and data was successfully written to and read from the mounted filesystem. Hence, within the limits of the test conducted, the module has met the minimum pass criteria.

M.2.9.2 MicroSD Card Reader Test Code

```
from machine import SPI, Pin
import os
from sdcard import SDCard

def sdtest():
    # Initialize SPI with the correct pins
    spi = SPI(1, baudrate=4000000, polarity=0, phase=0, sck=Pin(36),
              mosi=Pin(35), miso=Pin(37))
    cs = Pin(47, Pin.OUT)  # Chip Select pin

    # Create SD card object
    sd = SDCard(spi, cs)

    # Try to mount the SD card and handle errors
    try:
        # Deactivate and then activate the SD card chip select
        cs.value(1)  # Deactivate SD card
        cs.value(0)  # Activate SD card

        # Create a FAT filesystem
        vfs = os.VfsFat(sd)

        # Mount the filesystem
        os.mount(vfs, "/fc")
        print("Filesystem mounted successfully")
        # List all files on the SD card
        print("SD Card files:", os.listdir("/fc"))

        # Write to a file
        filename = "/fc/test.txt"
        with open(filename, "w") as f:
            written_content = "Test Data - Testing write functionality"
            f.write(written_content)
        print("File written with:", written_content)

        # Read from the file
        with open(filename, "r") as f:
            read_content = f.read()
        print("Read from file:", read_content)
        if read_content == written_content:
            print("File content read successfully:")

    except OSError as e:
        print("Error: SD card not detected or initialization failed.")
        print("Error details:", e)

# Call the test function
sdtest()
```

M.2.10 IMU Module Test

M.2.10.1 IMU Test Report

Table 66: Unit Test Report: QMI8658 IMU Communication Test

Test Type:	Unit
Tested Module:	QMI8658 IMU
Test Description:	The test validates the communication between the microprocessor unit (MCU) (ESP32-S3) and the QMI8658 IMU. It ensures that the MCU successfully initializes the onboard IMU module and reads valid sensor values.
Pass Criteria:	The MCU successfully initializes the QMI8658 IMU and valid sensor values are read.
Fail Criteria:	The tested module fails to meet the pass criteria.
Test Method:	1. Connect TTGO T-BEAM SUPREME to the development system (laptop) via USB-to-USB-C cable. 2. Open Thonny IDE and configure the interpreter by selecting "MicroPython (ESP32) •Espressif CDC Device @ COMX" where X is the COM Port that the device is plugged into. 3. Write and upload the test code (shown below). 4. Upload and run the code on the MCU. 5. Monitor the response and record the results based on the terminal outputs.
Test Code Output:	`WHO_AM_I register: 0x3e` `Incorrect device ID or communication error.`
Test Results:	The QMI8658 did not get successfully initialized and therefore no valid sensor values were read. Hence, within the limits of the test conducted, the module has failed to meet the minimum pass criteria.

M.2.10.2 IMU Test Code

```
from machine import Pin, SPI
from qmi8658c import *
import time

# WHO_AM_I device control register for QMI8658 IMU
EXPECTED_DEVICE_ID = 0x05

# Setup SPI connection in line with the manufacturer pinmap
SPI = SPI(1, baudrate=1000000, polarity=0, phase=0, sck=Pin(36),
          mosi=Pin(35), miso=Pin(37))

# Chip Select pin for QMI8658
CS = Pin(34, Pin.OUT)

# Initialise the Qmi8658 object
IMU = Qmi8658(SPI, CS)

def MAIN():
    # Read the WHO_AM_I control register from the device
    WHO_AM_I = IMU.qmi8658_read(QMI8658_WHO_AM_I)
    # Print register value
    print(f"WHO_AM_I register: {hex(WHO_AM_I)}")
    # If this does not match the register on the sensor datasheet
    # (0x05), then raise error.
    if WHO_AM_I != EXPECTED_DEVICE_ID:
        print("Incorrect device ID or communication error.")

MAIN()
```

M.3 Subsystem Testing