AUTONOMOUS DATA FERRYING FROM A SELF-ORGANIZING/HEALING WSN IN DISCONNECTED ZONES

25-26J-010

Project Proposal Report

Thamasha W.Y.M.G.

B.Sc. (Hons) Degree in Information Technology Specializing in Computer Systems & Network Engineering

Department of Computer Systems Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

August 2025

AUTONOMOUS DATA FERRYING FROM A SELF-ORGANIZING/HEALING WSN IN DISCONNECTED ZONES – CLUSTER DATA COMMUNICATION

25-26J-010

Project Proposal Report

B.Sc. (Hons) Degree in Information Technology Specializing in Computer Systems & Network Engineering

Department of Computer Systems Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

August 2025

Declaration

I declare that this is my own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	Student ID	Signature
Thamasha W.Y.M.G.	IT22088000	

The above candidate is carrying out research for the undergraduate Dissertation under my supervision.

Signature of the supervisor: Date: 28/08/2025

Abstract

Environmental monitoring in remote and disconnected regions remains a major challenge due to the absence of permanent communication infrastructure, limited energy availability, and the risk of data loss during transmission. Traditional Wireless Sensor Networks (WSNs) often struggle under these conditions, particularly in sustaining long-term operations with minimal human intervention. To address this, the proposed research introduces a self-organizing, cluster-based WSN supported by a mobile data ferry using an Unmanned Aerial Vehicle (UAV) to enable reliable and energy-efficient data collection. Main Set (MS) nodes equipped with environmental sensors locally store data while conserving solar-harvested energy through deep sleep cycles. A Cluster Head (CH) is dynamically elected based on residual energy and link quality, after which it assigns time slots to MS nodes for collision-free data transfer using ESP-NOW. Received data is persistently stored in JSON format within SPIFFS, ensuring resilience against CH re-election or failures. UAV arrival is detected via BLE beaconing, and once acknowledged, the UAV activates its Wi-Fi access point and MQTT broker to receive buffered data from the CH with acknowledgement-based reliability. Collected data is then relayed to a central server for visualization through a monitoring dashboard. Anticipated outcomes include improved energy efficiency, robustness against data loss, and scalable applicability for environmental monitoring in hard-to-reach locations.

Keywords: Wireless Sensor Network (WSN), UAV Data Ferrying, ESP-NOW, BLE Beaconing, MQTT

Table of Contents

List Of Figures

List of Tables

1 Introduction

1.1 Background and Literature Survey

Environmental monitoring in disconnected regions such as biodiversity preserves, fire-prone forests, and disaster zones faces chronic constraints due to the lack of permanent backhaul, harsh terrain, and the need for unattended, long-lived deployments. Wireless Sensor Networks (WSNs) have long been used to collect such data, but classic designs assume at least intermittent infrastructure or multi-hop relays to a sink, which can quickly drain node batteries. To mitigate this, UAV-assisted data ferrying has emerged, where a drone periodically visits clusters of sensors and offloads their buffered readings, thereby avoiding the energy cost of long-range or continuous networking. Comprehensive surveys highlight how UAVs act as mobile sinks to extend coverage and network lifetime in such settings [1].

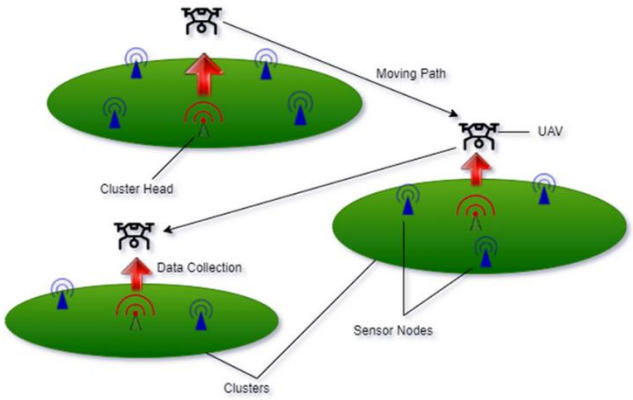

Figure 1.1: UAV acting as a mobile data ferry to collect buffered data from cluster heads in a WSN

Source: https://www.mdpi.com/2079-9292/10/21/2603

To further reduce energy consumption and improve scalability, cluster-based WSNs appoint a Cluster Head (CH), which aggregates data from Main Set (MS) nodes and handles upstream transfer. Foundational work such as LEACH established rotating CHs and local aggregation to reduce transmissions, with later analyses consolidating clustering as a core WSN strategy [2][3][4]. Modern CH selection often incorporates multi-criteria trust (e.g., residual energy, link quality) to keep leadership stable and efficient, while time-slot scheduling (TDMA) minimizes collisions and idle listening

by allowing each MS to wake only in its assigned slot. Recent surveys and studies reaffirm TDMA's role in energy conservation for dense, contention-prone WSNs [5].

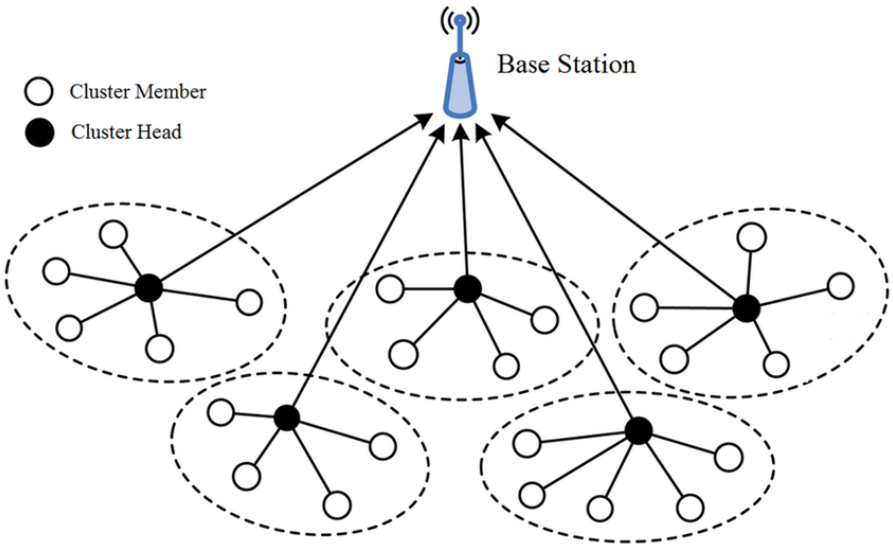

Figure 1.2: Cluster based WSN structure

Source: https://www.researchgate.net/figure/Cluster-based-WSN-Architecture_fig1_49600415

For short-range intra-cluster exchange, ESP-NOW is attractive because it is connectionless (no Wi-Fi association required), fast, and provides low overhead—ideal when MS nodes must wake briefly, transmit, and return to sleep. Official documentation and embedded stacks (ESP-IDF, Arduino, MicroPython) describe robust peer-to-peer messaging suitable for sensor bursts in constrained nodes [6][7].

When a UAV arrives to collect data, the system must detect it with minimal power overhead. Bluetooth Low Energy (BLE) beaconing satisfies this requirement: the UAV periodically advertises, the CH listens, and then responds with an acknowledgement, enabling proximity confirmation without costly Wi-Fi scans. Recent studies detail BLE-based proximity or positioning (often RSSI-assisted) for UAV and mobile contexts, demonstrating both practicality and low energy cost [8][9].

Once proximity is confirmed, MQTT provides a lightweight, reliable publish–subscribe channel over Wi-Fi for bulk transfer of buffered readings from CH to UAV, with QoS levels and acknowledgements that fit intermittent contacts [10][11]. Finally, using SPIFFS with JSON at the CH enables persistent, structured storage, ensuring that data survives resets and CH re-elections. This practice, widely adopted in

embedded ESP32 deployments, supports durability until data is successfully handed over [6].

1.2 Research Gap

While clustering, low-power intra-cluster communication, UAV-based data ferrying, and BLE-based proximity detection have each been investigated individually, there is a notable lack of systems that integrate these elements into a cohesive, fault-tolerant architecture tailored for disconnected environmental monitoring.

Most current research efforts still exhibit one or more of the following shortcomings:

- Energy inefficiency at MS nodes: Many implementations still use continuous polling or idle listening for data requests, forcing MS nodes to remain active for extended periods. This not only drains batteries faster but also reduces the effectiveness of solar-powered systems in low-light or shaded environments [4].
- High-power UAV detection methods: Several UAV-assisted WSNs detect UAV presence using Wi-Fi scanning or high-duty-cycle radios, which consume far more energy than BLE advertising [1][8].

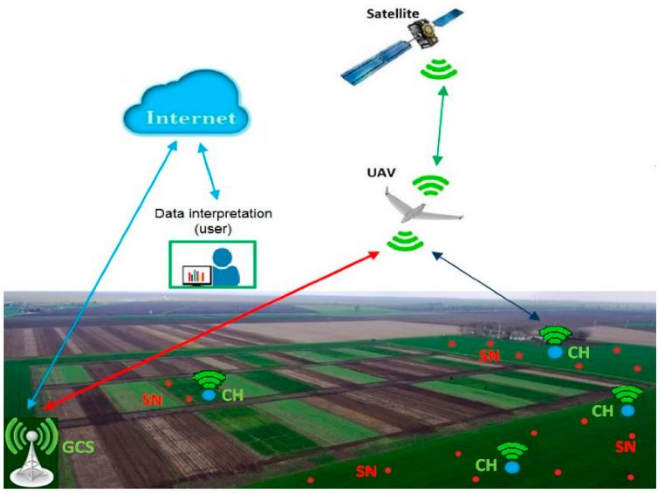

Figure 1.3: UAV-WSN collaboration in remote monitoring applications, highlighting real world relevance of BLE based UAV detection

Source: https://www.mdpi.com/1424-8220/19/21/4690

- Vulnerability to CH failure: In many designs, if the CH fails mid-collection cycle, there is no persistent handover of stored data to a newly elected CH, leading to partial or complete data loss [3].
- Volatile data buffering: Even when intra-cluster communication is optimized, the CH often stores aggregated data in volatile RAM. Power resets, unexpected brownouts, or reboots during UAV approach erase this information permanently [5].
- Inefficient transfer initiation: Many ferry-based systems begin bulk data transfer immediately after UAV detection, without first ensuring stable communication. This can cause packet loss if the Wi-Fi link is activated prematurely or under unstable conditions [8].

The research gap therefore lies in creating a unified, low-power, resilient communication and storage strategy that:

- Minimizes active time for MS nodes while ensuring timely delivery to the CH.
- Uses ultra-low-power proximity detection (BLE beaconing with ACK) for UAV arrival to conserve CH energy.
- Implements persistent, nonvolatile data storage at CHs to survive CH changes and reboots.
- Initiates data transfer only once UAV–CH communication has been stably established, ensuring reliable delivery during short UAV contacts.

1.3 Research Problem

The core research problem can be stated as:

How can a UAV-assisted WSN operating in fully disconnected environments be designed to minimize energy consumption at sensor nodes, ensure robust detection of short UAV contact opportunities, and guarantee lossless data delivery even in the presence of node failures or role changes?

This problem encompasses several interlinked technical challenges:

- Scheduling energy-efficient intra-cluster communication: The network must ensure that only a subset of MS nodes is active at any time, using assigned TDMA slots over ESP-NOW to avoid collisions and allow others to remain in deep sleep.
- Ensuring reliable UAV detection without excessive power draw: BLE beaconing from the UAV, with CH acknowledgements, must replace high-power Wi-Fi scans while still offering adequate detection range and responsiveness.
- Maintaining data integrity through CH changes: CHs must persist data in SPIFFS (JSON format) to enable seamless transfer to a newly elected CH if the current one fails mid-cycle.
- Optimizing contact window utilization: Once UAV arrival is confirmed via BLE beaconing and CH acknowledgement, buffered data must be transmitted over MQTT within the short contact period, with acknowledgement checks before clearing the buffer.
- Hardening against environmental unpredictability: The architecture must function reliably under variable solar charging conditions, intermittent UAV visits, and potential RF interference, without risking data loss.

By directly addressing these aspects, this research aims to create an end-to-end operational framework that can be deployed in real-world disconnected scenarios such as forest reserves, agricultural monitoring, or disaster response zones, where missing even a small window of data could compromise decision-making.

2 Objectives

2.1 Main Objective

To design and implement an energy-efficient and fault-tolerant cluster data communication method for Wireless Sensor Networks (WSNs), focusing on intra-cluster (MS–CH) scheduling, persistent CH data storage, UAV detection through BLE beaconing, and reliable handover of data from the CH to the UAV.

2.2 Specific Objectives

Develop a Time-Sliced MS to CH Communication Method

- Implement a scheduling mechanism where the CH assigns specific time slots to each Main Set (MS) node.
- Ensure synchronized wake-up of MS nodes only during their assigned slots, while others remain in deep sleep to conserve solar-harvested energy.
- Achieve collision-free data transmission using ESP-NOW, minimizing retransmissions and idle listening energy waste.

Implement Persistent and Fault-Tolerant Data Storage at the CH

- Design a data buffering system at the CH using SPIFFS-based flash storage in JSON format.
- Ensure safe and durable storage of MS data even during CH role changes or unexpected power interruptions.
- Develop recovery logic where a newly elected CH can continue data handling from the last acknowledged point, preventing data loss.
- Incorporate MS battery status into JSON payloads to assist UAV in adaptive path prioritization and ensure timely data retrieval from nodes at risk of power depletion.

Integrate BLE-Based UAV Detection at the CH

- Program the CH to listen for BLE beacon signals broadcast by the UAV.

- Enable the CH to send an acknowledgement (ACK) upon receiving the beacon, confirming its availability to the UAV.
- Maintain a lightweight and low-power detection routine to reduce CH energy consumption compared to Wi-Fi-based scans.

Enable CH to UAV Data Transfer via MQTT

- Configure the CH as an MQTT client capable of publishing buffered JSON data once UAV proximity is confirmed through BLE.
- Ensure data reliability by using acknowledgement mechanisms (QoS levels) before clearing stored data at the CH.
- Support seamless handover of data without requiring CH-side reconfiguration of UAV hardware.

3 Methodology

The methodology for this research component focuses on the design and implementation of the cluster data communication framework, covering the communication between Main Set (MS) nodes and the Cluster Head (CH), as well as the reliable handover of data from the CH to the UAV. This section outlines the system design, implementation tasks, required materials, data handling, scheduling, and anticipated outcomes.

3.1 System Design

The proposed design connects Main Set (MS) nodes with a dynamically elected Cluster Head (CH) and subsequently enables the CH to securely hand over aggregated data to a mobile data ferry (UAV). Each stage of the methodology is carefully structured to address the specific limitations of existing WSN deployments in disconnected environments, namely excessive energy waste, vulnerability to collisions, and risk of data loss during node failures.

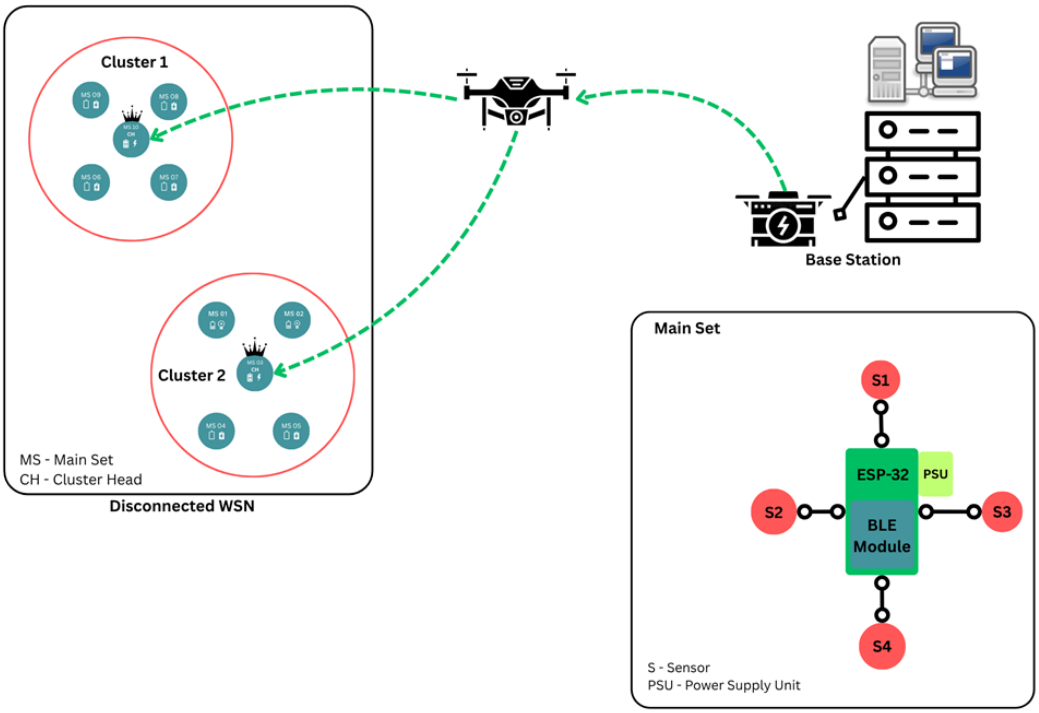

Figure 3.1: Proposed UAV-assisted cluster-based WSN architecture

3.2 Implementation Tasks

- MS↔CH Communication

The first stage of the methodology concentrates on intra-cluster communication between MS nodes and the CH. Unlike in traditional polling-based approaches, which require MS nodes to remain awake and continuously listen for CH requests, this work adopts a time-slot–based scheduling method. Once a CH is elected, it broadcasts a schedule to all MS nodes, assigning a unique transmission slot for each node. During its allocated slot, an MS wakes up, transmits its buffered sensor data using ESP-NOW, and returns immediately to deep sleep. All other MS nodes remain inactive until their slots arrive, thus reducing idle listening and extending battery life. This scheme not only improves energy efficiency but also eliminates collisions, as no two MS nodes transmit simultaneously.

- CH Local Storage

The second stage focuses on local data management at the CH. As MS nodes deliver their data, the CH aggregates and stores the information in JSON format within its SPIFFS flash memory. This ensures data persistence even if the CH undergoes re-election or a power interruption. In the event of a CH change, the old CH forwards its locally stored data to the newly elected CH, allowing the system to resume from the last confirmed transmission rather than restarting the entire collection process. This persistence mechanism is crucial to prevent data loss, which is one of the major weaknesses of conventional cluster-based systems.

- UAV Detection

The third stage involves UAV detection and communication initiation. To minimize the continuous energy drain associated with active Wi-Fi scanning, the CH relies on BLE beaconing. The UAV periodically broadcasts beacon packets, and the CH listens for these signals. Upon detection, the CH responds with an acknowledgement (ACK), confirming its availability. This beacon–ACK handshake ensures reliable UAV

detection without the need for high-power scans or signal strength estimation, thus conserving CH energy.

- CH↔UAV Data Transfer

The final stage of the methodology concerns CH-to-UAV data transfer. Once UAV proximity is confirmed through BLE acknowledgement exchange, the CH switches to Wi-Fi mode and operates as an MQTT client. The UAV, configured as a mobile broker, receives the published data. The CH publishes its buffered JSON files to designated MQTT topics and only clears its local storage once the UAV has sent acknowledgement of successful reception. This ensures guaranteed delivery, even in cases of temporary link disruptions..

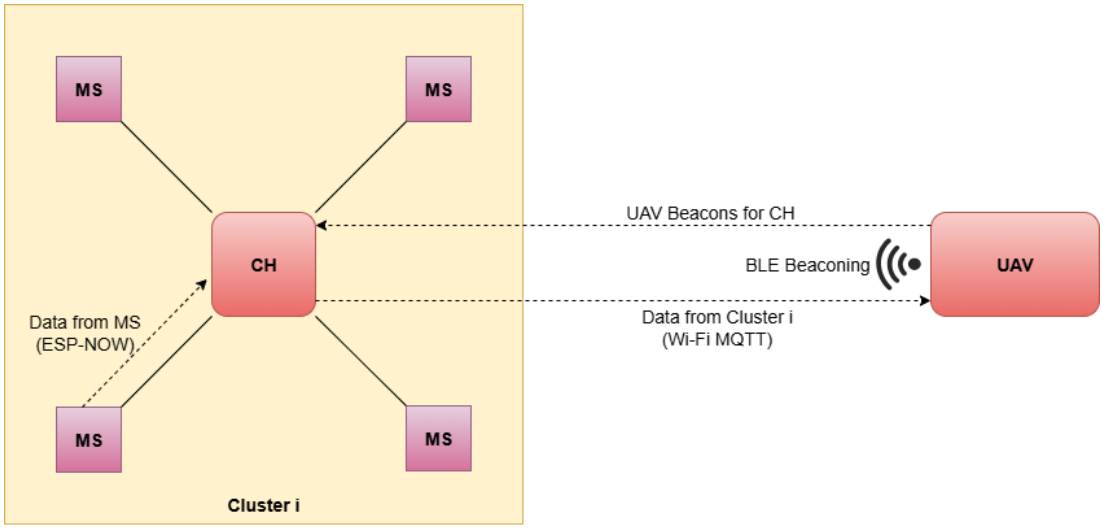

Figure 3.2: Intra-cluster communication and UAV handover process

3.3 Required Materials

To implement these stages, the following materials will be used:

- Hardware: ESP32 microcontrollers for MS and CH nodes, solar charging modules for energy harvesting, environmental sensors (temperature, humidity, soil moisture, air quality), and a UAV platform capable of BLE beaconing and Wi-Fi-based data collection.
- Software & Tools: Arduino IDE and PlatformIO for firmware development, ESP-IDF libraries for ESP-NOW, BLE, SPIFFS, and MQTT integration. Serial

monitoring will be used for debugging. Workflow diagrams will be created with Draw.io, and version control will be maintained using GitHub.

3.4 Data Handling & Testing

No human surveys or interviews are conducted. Instead, experimental data will be collected by deploying MS nodes with test sensors. Data handling will involve:

- Generating structured JSON files from MS transmissions at the CH.
- Storing and retrieving data from SPIFFS.
- Simulating CH failures to test recovery.
- Testing BLE beacon reception from UAV simulators.
- Conducting Wi-Fi MQTT transfers to UAV-side logs for validation.

The implementation will follow a staged timeline. Initially, MS-to-CH communication will be tested with 2–3 MS nodes to validate slot synchronization and ESP-NOW reliability. CH storage will then be tested for JSON integrity and recovery after simulated failures. Next, BLE-based UAV detection will be validated using controlled beacon broadcasts. Finally, MQTT-based CH-to-UAV handover will be tested for acknowledgement-based delivery reliability.

3.5 Anticipated Outcomes

The methodology is expected to demonstrate:

- Reduced energy consumption by minimizing idle listening.
- High delivery reliability ensured by MQTT acknowledgements.
- Robustness against CH failure due to persistent storage in SPIFFS.
- Feasibility of BLE beacon–ACK handshakes for low-power UAV detection. Overall, this approach will prove its applicability for environmental monitoring in disconnected regions, enabling scalable and sustainable deployments.

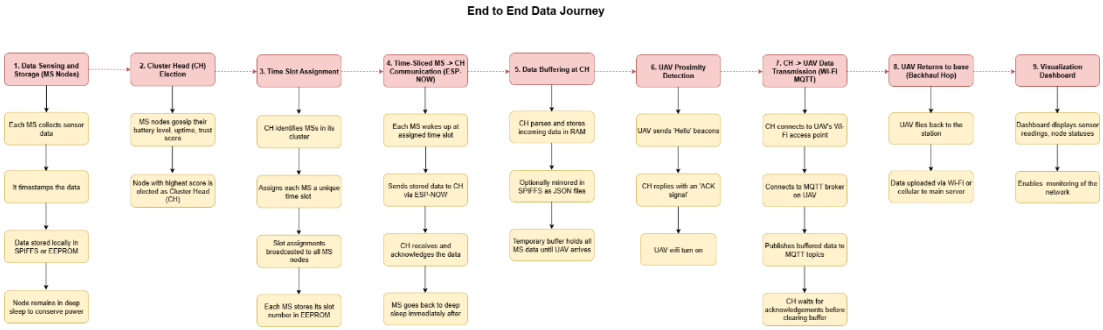

Figure 3.3: End-to-end workflow of data collection and transfer in the proposed system

4 Project Requirements

4.1 Functional requirements

- Time Sliced data Transmission

The system must ensure that each Main Set (MS) transmits its stored sensor data only during its assigned time slot, as scheduled by the Cluster Head (CH). This avoids collisions and minimizes energy wasted in idle listening. A synchronization broadcast from the CH ensures all MS nodes remain aligned with the schedule.

- Persistent Data Storage at CH

The CH must be capable of receiving, formatting, and storing sensor data reliably in SPIFFS using JSON structures. This ensures that data is preserved even during unexpected resets or CH re-elections, guaranteeing continuity of environmental records. Battery level metadata from MS nodes must also be stored alongside sensor readings to assist UAV in adaptive path prioritization and ensure timely data retrieval from nodes at risk of power depletion.

- Cluster Head Re-election Handover

In the event of CH failure or battery depletion, the old CH must securely hand over its buffered data to the newly elected CH. This guarantees no data loss during leadership transitions, ensuring robustness in disconnected environments.

- UAV Detection via BLE

The system must allow the UAV to detect the presence of the CH using BLE beaconing. The UAV broadcasts a beacon, and the CH responds with an acknowledgement, establishing proximity before initiating data transfer.

- Data Offload to UAV (MQTT)

The CH must reliably publish buffered sensor data to the UAV via MQTT once proximity is confirmed. Each transfer must include acknowledgement verification to ensure all data packets are successfully received before the CH clears its buffer.

4.2 Non-Functional Requirements

- Reliability

The system must provide dependable data delivery across all communication stages. This is achieved by incorporating persistent SPIFFS storage at the CH, handover protocols during CH re-election, and acknowledgement-based clearing mechanisms during UAV transfer, ensuring that no environmental data is lost even under unexpected failures.

- Energy Efficiency

Since MS nodes are powered by solar energy, the system must minimize unnecessary energy usage. This is achieved through deep sleep cycles, time-slot scheduling that avoids idle listening, and lightweight BLE-based UAV detection that consumes significantly less power than continuous Wi-Fi scanning.

- Scalability

The architecture must support expansion to multiple clusters and larger numbers of MS nodes without introducing excessive communication collisions or storage overhead. Time-sliced communication and lightweight protocols such as ESP-NOW provide the necessary foundation for handling more nodes efficiently.

- Latency and Responsiveness

The system should ensure rapid response during UAV contact, as the data ferry window is limited. By detecting UAV presence through BLE and transferring pre-buffered JSON files over MQTT, the system minimizes delays and maximizes data offload within the available contact time.

- Persistence and Robustness

All collected data must remain safely stored in SPIFFS until successfully transferred to the UAV. Even in the event of resets, power failures, or CH handovers, data integrity and recoverability must be guaranteed.

4.3 System Requirements

- Hardware Requirements

The system relies primarily on ESP32 microcontrollers for both Main Set (MS) and Cluster Head (CH) nodes. Each MS is equipped with a set of environmental sensors (e.g., temperature, humidity, soil moisture, and air quality), a solar power module for energy harvesting, and onboard flash storage compatible with SPIFFS for persistent data saving. The CH requires slightly more processing power and memory capacity, as it is responsible for receiving data from multiple MS nodes, managing time-slot schedules, and performing temporary data storage. On the UAV side, a Raspberry Pi or equivalent onboard computer is needed to act as a Wi-Fi access point and MQTT broker, facilitating reliable CH-to-UAV communication. Both MS and CH nodes use integrated BLE radios for UAV detection and Wi-Fi for data offloading once proximity is confirmed.

- Software Requirements

The firmware for MS and CH nodes will be developed using the Arduino IDE or PlatformIO, with specific libraries for ESP-NOW (intra-cluster communication), BLE scanning and acknowledgement (UAV detection), SPIFFS and ArduinoJSON (data formatting and persistence), and PubSubClient or an equivalent MQTT library (data offload). The UAV will run a lightweight MQTT broker (such as Mosquitto) on the Raspberry Pi to receive published data. Data visualization will be enabled through a dashboard hosted at the base station, connected to the UAV's backhaul network.

- Networking Requirements

Communication is handled through a hybrid multi-layered approach. Intra-cluster communication between MS and CH is achieved using ESP-NOW, enabling low-latency, low-power packet exchanges without Wi-Fi association overhead. BLE is reserved for UAV detection, providing a lightweight method to establish proximity and trigger data transfer readiness. Finally, Wi-Fi MQTT is employed for CH-to-UAV communication, allowing structured, reliable, and acknowledgement-based data delivery.

- Energy Requirements

Given that MS nodes are powered by solar energy, the system must maintain extremely low idle power consumption. This is achieved through deep sleep modes on the ESP32, activated outside of sensing or transmission windows. CH nodes, while more active, must also manage their energy efficiently, particularly when handling re-elections or large data buffers.

4.4 Test Cases

- Time Slot Scheduling Validation

A test will be conducted to ensure that each MS node transmits its data only during its assigned time slot. Multiple MS nodes (e.g., 2–4) will be configured with distinct schedules, and logs at the CH will be analyzed to verify that no collisions occur and that idle listening is minimized.

- Persistent Storage Integrity

To test SPIFFS-based data storage at the CH, sensor readings will be collected and stored as JSON files. The CH will then be intentionally reset to simulate a failure. Upon reboot, the CH's storage will be examined to confirm that all data remains intact and retrievable.

- Cluster Head Re-election Robustness

This test will simulate a CH failure (e.g., forced shutdown or low battery scenario) to trigger re-election. Once a new CH is elected, the old CH will hand over its buffered data. Verification will be carried out to ensure that no data packets are lost during the transition and that the new CH resumes operations seamlessly.

- UAV Detection Accuracy

To validate BLE-based UAV detection, the UAV will periodically broadcast beacon signals while approaching the cluster. The CH will be monitored to confirm that it responds promptly with acknowledgements, and that detection is consistent within the designed range.

- Data Offload Reliability (MQTT)

For this test, the CH will attempt to publish buffered data to the UAV's MQTT broker. The test will verify that all JSON packets are delivered successfully, and acknowledgements are received. The CH's buffer will only be cleared after these acknowledgements, ensuring data reliability.

- Energy Efficiency Assessment

The power consumption of MS and CH nodes will be measured during active transmission, idle listening, and deep sleep states. Results will be compared to confirm that the scheduling mechanism and deep sleep usage contribute significantly to energy savings.

5 Description of Personal and Facilities

5.1 Work Breakdown Structure

Table 5.0.1: Work Breakdown Structure

Level	WBS	Task Description
1	1	System Design
2	1.1	Define Architecture
3	1.1.1	Identify role of MS, CH,UAV
3	1.1.2	Map communication links (ESP-NOW, BLE, Wi-Fi MQTT)
3	1.1.3	Create initial system diagram
2	1.2	Specify Communication Layers
3	1.2.1	Select protocols per layer (intra cluster, detection, offload)
3	1.2.2	Define packet structures (JSON message, MQTT topic etc.)
2	1.3	Design Time-Sliced Scheduling Model
3	1.3.1	Choose TDMA based slot allocation method
3	1.3.2	Define slot length (ms/sec) based on sensor data size
3	1.3.3	Decide synchronization methos (CH broadcast time sync)
2	1.4	Design data Persistence Model
3	1.4.1	Define file naming structure in SPIFFS (per MS/per slot)
3	1.4.2	Select JSON schema (timestamp, node ID, sensor values)
3	1.4.3	Plan CH-to-CH transfer protocol during re-election
2	1.5	UAV Detection Workflow
3	1.5.1	Specify CH scan cycle (scan interval, scan window)
3	1.5.2	Define ACK format (short packet confirmation)
3	1.5.3	Map state transitions: *Idle* → *Beacon Received* → *ACK* → *Wi-Fi ON*
2	1.6	CH Handover Protocol
3	1.6.1	Create procedure for data transfer (SPIFFS → new CH)
3	1.6.2	Validate continuity with test data
1	2	MS ↔ CH Communication (ESP-NOW with Time Slots)
2	2.1	CH Generates Schedule

3	2.1.1	Assign unique slot per MS
3	2.1.2	Create schedule table
3	2.1.3	Store in CH SPIFFS for persistence
2	2.2	MS Receives and Stores Schedule
3	2.2.1	MS parses broadcast schedule
3	2.2.2	Save slot timing in EEPROM (for recovery)
3	2.2.3	Synchronize internal clock with CH
2	2.3	Data Transmission from MS
3	2.3.1	MS wakes up at scheduled time
3	2.3.2	Collects and formats sensor data
3	2.3.3	Transmits packet to CH via ESP-NOW
3	2.3.4	Includes timestamp & MS ID
2	2.4	CH Reception & Acknowledgement
3	2.4.1	CH listens during each slot
3	2.4.2	Verifies packet integrity (CRC)
3	2.4.3	Sends ACK packet to MS
3	2.4.4	Logs data into temporary buffer
2	2.5	MS Sleep Cycle
3	2.5.1	On ACK receipt, MS enters deep sleep
3	2.5.2	Remains inactive until next assigned slot
2	2.6	Testing Phase
3	2.6.1	Simulate multiple MS transmissions
3	2.6.2	Measure collision probability
3	2.6.3	Analyze power consumption logs
1	3	Persistent Storage at CH (SPIFFS + JSON)
2	3.1	SPIFFS Setup
3	3.1.2	Configure SPIFFS partition size in firmware
3	3.1.2	Initialize file system during boot
3	3.1.3	Test read/write operations
2	3.2	JSON Formatting

3	3.2.1	Define schema:{ "MS_ID": x, "Timestamp": y, "Sensor_Data": {…} }
3	3.2.2	Validate schema with ArduinoJSON library
3	3.2.3	Ensure compact size to save memory
2	3.3	Data Storage Routine
3	3.3.1	On each received MS packet → format JSON
3	3.3.2	Append entry to SPIFFS file
3	3.3.3	Maintain file per session/day
2	3.4	Recovery Testing
3	3.4.1	Simulate CH reset → reboot system
3	3.4.2	Check file system integrity
3	3.4.2	Reload JSON logs to RAM
2	3.5	CH Re-Election Transfer
3	3.5.1	Old CH packages stored SPIFFS data
3	3.5.2	Transfers files via ESP-NOW to new CH
3	3.5.3	New CH imports and continues operations
2	3.6	Validation
3	3.6.1	Test recovery after power loss
3	3.6.2	Validate no duplicate/lost entries
1	4	UAV Detection (BLE Beaconing)
2	4.1	CH Beacon Scanning
3	4.1.1	Configure CH BLE module in scan mode
3	4.1.2	Set scan interval/window to minimize energy use
3	4.1.3	Parse incoming beacon packets
2	4.2	ACK Mechanism
3	4.2.1	On beacon reception → CH sends BLE ACK
3	4.2.2	UAV listens for ACK to confirm CH presence
2	4.3	Wi-Fi Activation Trigger
3	4.3.1	Once ACK confirmed → CH activates Wi-Fi
3	4.3.2	UAV switches Wi-Fi on simultaneously
2	4.4	Energy Optimization

3	4.4.1	Tune scan duty cycle (intermittent scans instead of continuous)
3	4.4.2	Compare energy use BLE vs Wi-Fi scans
2	4.5	Validation
3	4.5.1	Test UAV detection time (latency)
3	4.5.2	Measure detection reliability at 5m, 10m, 20m distances
1	5	CH ↔ UAV Data Transfer (Wi-Fi MQTT)
2	5.1	CH MQTT Client Setup
3	5.1.1	Install PubSubClient on ESP32
3	5.1.2	Connect to UAV's Wi-Fi
3	5.1.3	Establish MQTT session
2	5.2	Data Publishing
3	5.2.1	CH reads SPIFFS buffer
3	5.2.2	Publishes JSON files to MQTT topic
3	5.2.3	Supports batch uploads
2	5.3	Acknowledgement Handling
3	5.3.1	MQTT QoS1/QoS2 ensures delivery guarantee
3	5.3.2	UAV broker sends PUBACK
3	5.3.3	CH marks data as delivered
2	5.4	Buffer Clearing
3	5.4.1	On full ACK confirmation → delete data from SPIFFS
3	5.4.2	If no ACK → retry mechanism
2	5.5	Stress Testing
3	5.5.1	Simulate short UAV contact window (1–2 minutes)
3	5.5.2	Test max buffer size transmission
3	5.5.3	Measure packet loss rate
1	6	Testing & Validation
2	6.1	Time Slot Energy Test
3	6.1.1	Measure MS energy use (deep sleep vs idle listening)
3	6.1.2	Compare baseline vs time-slot scheduling
2	6.2	Storage Robustness Test

3	6.2.1	Simulate resets, brownouts
3	6.2.2	Verify stored data remains intact
2	6.3	CH Re-Election Test
3	6.3.1	Force CH shutdown
3	6.3.2	Verify new CH receives stored data
3	6.3.3	Validate no data loss
2	6.4	UAV Detection Test
3	6.4.1	Vary UAV distance & speed
3	6.4.2	Measure detection latency and reliability
2	6.5	End-to-End Data Journey Test
3	6.5.1	Collect → Store → Detect → Transfer → UAV → Base station
3	6.5.2	Verify 100% delivery success

5.2 Timeline

Figure 5.1: Gantt Chart

5.3 Estimated Budget

Table 5.2: Estimated Budget

Component	Prize (Rs.)
UAV (Estimated)	60,000.00
ESP32-S3-DevKitC-1U (8MB Flash, 8MB PSRAM) x 7	1,850.00
18650 Li-ion 3500 mAh battery (Samsung)	780.00
3x18650 Battery Holder (parallel wiring)	120.00
CN3065 Solar Li-ion Charger module	350.00
INA219 Current sensor (I²C)	350.00
Mini Solar Panel 6V 1W (110x60mm)	350.00
SMA antenna + IPEX to SMA cable	690.00
ENS160 + AHT21 air quality sensor module x 6	7,100.00
INMP441 I²S MEMS microphone module x 6	5,040.00
HMC5883L 3-axis magnetometer module x 6	2,700.00
BME280 pressure/temperature/humidity sensor x 6	5,700.00
Total	**85,030.00**

References

[1] *Mdpi.com*. [Online]. Available: https://www.mdpi.com/1424-8220/19/21/4690. [Accessed: 19-Aug-2025].

[2] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, p. 10pp.vol.2-, 2000.

[3] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wirel. Commun.*, vol. 1, no. 4, pp. 660–670, 2002.

[4] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Comput. Commun.*, vol. 30, no. 14–15, pp. 2826–2841, 2007.

[5] A. N. Pathak and A. R. Yadav, "Scheduling based on residual energy of sensors to extend the lifetime of network in wireless sensor network," *J. Eng. Appl. Sci.*, vol. 71, no. 1, 2024.

[6] "ESP-NOW - ESP32 - — ESP-IDF Programming Guide v5.5 documentation," *Espressif.com*. [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/network/esp_now.html. [Accessed: 19-Aug-2025].

[7] "espnow — support for the ESP-NOW wireless protocol — MicroPython latest documentation," *Micropython.org*. [Online]. Available: https://docs.micropython.org/en/latest/library/espnow.html. [Accessed: 19-Aug-2025].

[8] *Mdpi.com*. [Online]. Available: https://www.mdpi.com/1424-8220/24/22/7170. [Accessed: 19-Aug-2025].

[9] C. Hernández-Goya, R. Aguasca-Colomo, and C. Caballero-Gil, “BLE-based secure tracking system proposal,” *Wirel. Netw.*, vol. 30, no. 6, pp. 5759–5770, 2024.

[10] “MQTT - the standard for IoT messaging,” *Mqtt.org*. [Online]. Available: https://mqtt.org/. [Accessed: 19-Aug-2025].

[11] “MQTT V3.1 protocol specification,” *Ibm.com*. [Online]. Available: https://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html. [Accessed: 19-Aug-2025].