ResQ Mesh: An Internet-Independent Emergency Messaging System Using ESP32 and Multi-Hop LoRa Relaying

Drishya P[1], Dinesh D[2], Kathiresan P[3], Megaprasanth V[4], Pradeesh S[5], and Sriman M[6]
[1,2,3,4,5,6]*Department of Computer Science and Engineering, SNS College of Technology, Coimbatore, Tamil Nadu, India*

***Abstract*—Cyclones, floods and landslides often damage cellular towers and backhaul links, and the few towers that survive are overwhelmed by traffic, leaving trapped people unable to report where they are or what they need. This paper presents ResQ Mesh, a low-cost prototype that carries short emergency messages over a multi-hop LoRa mesh without any dependence on the Internet or on cellular infrastructure. Each node combines an ESP32 microcontroller, an SX1278 LoRa transceiver operating at 433 MHz and a NEO-6M Global Positioning System (GPS) receiver, and is powered by rechargeable 18650 cells. A sender node packs the victim's position, the number of people present and the kind of help required into a compact packet. Neighbouring nodes rebroadcast the packet under a hop limit and a duplicate-suppression rule until it reaches a gateway attached to a laptop. Because no fixed route is stored, the mesh can keep operating when individual nodes fail. A dashboard plots every alert on a map and sorts alerts into critical, high and normal levels using keyword rules, so that rescuers can see who needs help first, where they are and what they need. We describe the hardware, packet format, relay protocol and dashboard, and analyse the airtime and relaying overhead. ResQ Mesh is an educational prototype and a supplementary channel, not a replacement for cellular emergency services.**

***Index Terms*—Disaster communication, ESP32, Internet of Things, LoRa, mesh networking, multi-hop relaying.**

I. INTRODUCTION

In a major disaster the communication network is usually among the first things to fail. Towers lose mains power and run out of backup batteries, fibre and microwave backhaul links are cut by floodwater or landslides, and the towers that remain in service are congested by the sudden surge of calls and messages. People trapped in the affected area then cannot call for help or say where they are, and rescue teams lose time because they do not know who needs help, where they are, or which need is most urgent.

Several alternatives exist, but each has limits. Satellite phones are expensive and rarely available to ordinary residents, and amateur radio depends on trained operators. Low-power wide-area network (LPWAN) technologies are attractive because LoRa links can span kilometres in open terrain at very low power and operate in licence-exempt sub-GHz bands [1], [2]. The difficulty is that the standard LoRaWAN architecture is a star of stars: end devices talk to gateways that must themselves be connected to a network server, so the architecture still relies on backhaul that a disaster may destroy [1]. Low-cost personal safety devices built on Arduino boards take a simpler approach, combining GPS positioning with GSM text messages to send a user's location to preset contacts [3]. Such devices depend on cellular coverage, and the authors of one design note that its GSM link is limited by network availability in remote areas [3].

This paper describes ResQ Mesh, a temporary, low-cost network that avoids that dependency. Nodes built from common hobby-grade parts form a multi-hop mesh in which every node can relay another node's message, so that a person beyond direct radio range of the rescue point can still be heard. The main contributions are as follows.

1) A compact emergency packet that carries location, head-count and the type of help needed in a single LoRa frame.

2) A stateless relaying rule based on a hop limit, a duplicate cache and a randomised relay delay, which

needs no routing tables and therefore tolerates node failure.

3) A gateway that removes duplicates and passes each alert to a laptop over USB, with location and signal strength attached.

4) An emergency prioritization dashboard that triages alerts into critical, high and normal levels and presents them on a map and in a ranked list.

We position ResQ Mesh as an educational prototype and as a supplementary communication path that can operate when conventional infrastructure cannot be relied upon. It is not intended to replace cellular emergency services. The remainder of the paper is organised as follows: Section II reviews related work, Section III describes the node design and relay protocol, Section IV describes the emergency prioritization dashboard, Section V discusses limitations and future work, and Section VI concludes.

II. RELATED WORK

LoRa uses chirp spread spectrum modulation to trade data rate for link budget, which gives long range and robustness to interference at low power [1], [2]. Measurement and modelling studies also show that a single LoRa gateway serving many devices is limited by collisions, so capacity falls quickly as node density grows [4]. These findings matter for a mesh, because every relay adds to channel occupancy.

Classical mobile ad hoc routing protocols such as AODV [5] and OLSR [6] discover or maintain routes by exchanging control messages and storing routing tables. That is reasonable for laptops or phones but is heavy for a low-cost microcontroller that must also respect the low duty cycle and tiny payloads of LoRa. Simple flooding avoids route state altogether, yet it is known to cause redundant rebroadcasts, contention and collisions, the so-called broadcast storm [7]. Suppression schemes, including the timer-based approach of the Trickle algorithm [8], reduce this overhead by letting a node wait and stay silent when neighbours have already sent the same data. More recent work applies clustering, reinforcement learning and trust mechanisms to route selection in MANET-IoT networks [9]; ResQ Mesh instead avoids route computation altogether.

ResQ Mesh takes a deliberately simple position between these extremes. It uses controlled flooding, with a hop limit to bound propagation, a cache of recently seen messages to stop duplicates and a short random delay before relaying to reduce simultaneous transmissions. The aim is a protocol small enough to run on an ESP32 and to demonstrate clearly in a classroom setting, and not an optimal routing algorithm.

III. SYSTEM DESIGN

A. System Overview

The system has three parts, shown in Fig. 1. Sender nodes carry an SOS button and a GPS receiver so that a person can transmit an emergency message with a position. Relay nodes forward packets they overhear. A gateway node collects packets at the rescue point and passes them to a laptop that runs the dashboard described in Section IV. As an optional access path, a smartphone can connect to the nearest node over Bluetooth Low Energy (BLE) to compose a free-text message, but the phone never needs any network access of its own. A drone carrying a node could serve as an elevated relay for larger areas; this extension is outside the scope of the present design.

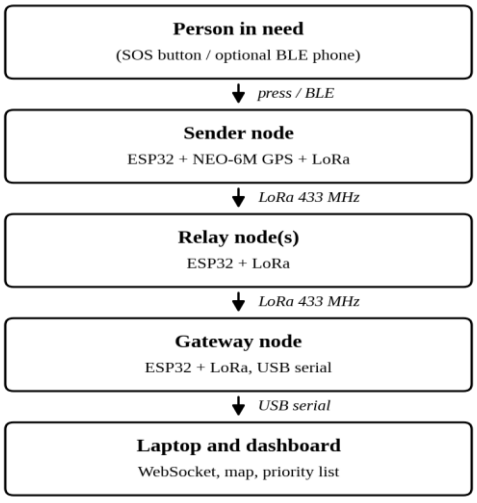

Fig. 1. Block diagram of the ResQ Mesh message path. No Internet connection or cellular tower is used at any stage.

B. Node Hardware

Every node uses an ESP32 microcontroller [10] as its controller. Radio communication is provided by an Ra-02 module based on the Semtech SX1278 transceiver [11], which operates in the 433 MHz band.

This band was chosen because it is licence-exempt in many regions and propagates around obstacles better than higher frequencies. Sender nodes add a NEO-6M GPS receiver [12] that reports position in National Marine Electronics Association (NMEA) 0183 sentences [13]; the firmware parses the latitude and longitude fields from these sentences. Power comes from 18650 lithium cells charged through a TP4056 module, so that a node runs independently of the mains supply. Because the ESP32 also has Wi-Fi, a node can serve a small local web page for configuration. Radio power limits and duty-cycle rules for 433 MHz differ between countries and must be checked before any real deployment.

C. Packet Format

Emergency messages are short, so each packet is built to fit comfortably inside one LoRa frame. Table I lists the fields. The pair of source identifier and sequence number forms the message identifier used for duplicate detection.

For example, a sender can report "I am here, there are 5 people with me, we need food, water and medical help" as a position, a head-count of five and three need flags, with any extra words in the text field.

TABLE I EMERGENCY PACKET FIELDS

Field	Bytes	Purpose
Source ID	1	Node that created the alert
Sequence no.	2	With Source ID, forms the message ID
TTL	1	Remaining hops; decremented at each relay
Flags	1	Message type and priority hint
Latitude	4	From NMEA sentence
Longitude	4	From NMEA sentence
People count	1	Number of people with the sender
Needs	1	Bit flags: food, water, medical, other
Free text	0–40	Short message, for example "trapped on roof"

D. Relay Protocol

Each node follows the same rules when it receives a packet.

1) Look up the message ID in a small circular cache of recently seen messages. If it is present, discard the packet.

2) Otherwise store the ID in the cache. If the node is the gateway, hand the packet to the dashboard path and stop.

3) If the TTL is greater than one, decrement it, wait for a random delay drawn uniformly from a short interval, and retransmit the packet once.

4) If the TTL has reached its last value, do not forward.

The TTL bounds how far a packet can travel, so it cannot circulate indefinitely, and the cache ensures a node relays each message at most once. The random delay reduces the chance that two neighbours transmit at the same instant. Since nodes keep no routes, there is nothing to repair when a node fails: the packet simply reaches the gateway through whichever neighbours are still alive. For instance, if the path Node 1 → Node 2 → Node 3 → Gateway loses Node 2, a packet from Node 1 can still arrive through Node 4 and Node 3.

E. Airtime and Relaying Overhead

The cost of the relay rule can be bounded without any experiment. Because each node relays a given message at most once, a message that reaches a network of n nodes causes at most n transmissions, and the TTL limits how far the message spreads from its source. The time each transmission occupies the channel follows from the LoRa time-on-air formula given by the transceiver manufacturer [11]. With a 125 kHz bandwidth, coding rate 4/5, eight preamble symbols, an explicit header and a payload check, a packet of the maximum 55 bytes of Table I lasts about 108 ms at spreading factor 7 and about 657 ms at spreading factor 10. For a network of 20 nodes the worst-case channel time consumed by one message is therefore roughly 2.2 s at spreading factor 7 and 13 s at spreading factor 10. A higher spreading factor extends range but multiplies this cost, so the choice is a trade-off between reach and airtime, to be made from the expected node spacing. The random relay delay of Section III-D spreads these transmissions in time, and a short free-text field keeps every packet well below the limit of the LoRa frame.

IV. EMERGENCY PRIORITIZATION DASHBOARD

A. Data Path

The gateway has its own LoRa module, discards duplicate packets using the same cache as the relay nodes, and writes each received message to the laptop as one line over a USB serial link. A laptop program parses each line, appends the alert to a log file, assigns a priority level, and pushes the new alert to a browser page over a WebSocket connection, so that the display updates as messages arrive without a page reload. The whole path from sender to screen therefore uses only radio links and a local USB cable.

B. Dashboard Interface

The dashboard presents every alert in two linked views, sketched in Fig. 2. The map view shows one marker per alert at the GPS position carried in the packet, together with the gateway location. The alert list shows the same alerts in priority order. For each alert the rescue team can see the message text, the number of people, the type of help required, the sender location, the received signal strength and the arrival time. Marker shapes distinguish the three priority levels so that the display remains readable in black and white.

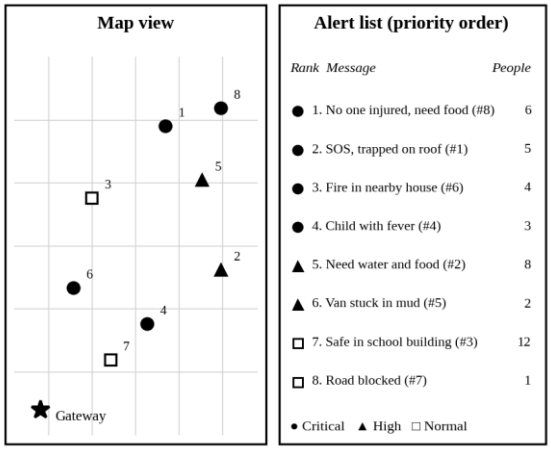

Fig. 2. Layout of the rescue dashboard, drawn as a schematic with the eight sample alerts of Table III. Marker numbers refer to alert numbers.

C. Priority Rules

Alerts are triaged by simple keyword and flag rules, summarised in Table II. A message is critical if its text contains SOS, trapped, injured, medical or fire, or if the medical need flag of Table I is set. It is high if it mentions water, food, stuck or child, and normal otherwise. When a message matches several levels, the most urgent level wins. Keywords are matched as whole words and without regard to letter case.

TABLE II PRIORITY LEVELS

Level	Trigger words	Dashboard action
Critical	SOS, trapped, injured, medical, fire	Listed first
High	water, food, stuck, child	Listed second
Normal	All other messages	Listed last

D. Ordering and Worked Example

Within a level, alerts with more people are shown first, and a tie is broken by earlier arrival. To illustrate the rules, we ran the triage procedure on eight short sample messages written for this purpose. Table III lists the level assigned to each message and its resulting rank.

TABLE III TRIAGE OF SAMPLE MESSAGES

#	Message	People	Level	Rank
1	SOS, trapped on roof, one injured	5	Critical	2
2	Need drinking water and food	8	High	5
3	We are safe, staying in school building	12	Normal	7
4	Child with fever, need medical help	3	Critical	4
5	Van stuck in mud near the bridge	2	High	6
6	Fire in nearby house, SOS	4	Critical	3
7	Road blocked near the market	1	Normal	8
8	Nobody is injured, we need food	6	Critical	1

The order of the list matches the intended reading in most cases: trapped, injured and fire reports precede the request for water and food, and the safe group in the school building is last despite being the largest.

The example also exposes a weakness of keyword matching. Message 8 states that nobody is injured, yet the word "injured" makes it critical, so it is ranked first. Messages with misspellings or local-language words would be missed in the opposite direction. These cases are discussed in Section V.

E. Logging

Every alert is saved to a log file together with its priority, position, signal strength and arrival time. The log lets the team review the sequence of reports after an operation and recover the alert history if the browser page is closed.

V. DISCUSSION

The design favours simplicity and robustness over efficiency. A stateless relay rule has nothing to repair when a node fails, because a packet simply continues through whichever neighbours are still alive, and every node runs identical firmware. The price is airtime: as Section III-E shows, one message can cost up to one transmission per node, and the TTL and spreading factor are the main tuning knobs between reach and channel occupancy. Redundant rebroadcasts also raise the probability of collisions, which the random relay delay reduces but cannot remove [7].

Several limitations should be kept in mind. First, this paper presents the design of a prototype and an analysis of its overhead, but not measured performance. The range of the radio links in flooded or hilly terrain, the delivery ratio and the end-to-end delay still have to be measured with real hardware in field trials. Second, GPS fixes can be slow or unavailable indoors, so a message may need to carry a last-known position. Third, keyword triage is brittle. It cannot read negation, as the example of Table III shows, and it depends on a fixed English vocabulary, so it should be treated as a decision aid and not as a substitute for a human operator reading the alerts. Fourth, the prototype has no authentication or encryption, so false alerts and eavesdropping are not addressed. Fifth, the regulations for 433 MHz transmission differ between countries and must be checked before real use. Finally, a hobby-grade prototype of this kind cannot replace professional emergency systems.

The planned validation consists of range tests between two nodes at different spreading factors, delivery and delay tests over three or more hops, and removal of a relay node during transmission to confirm that messages arrive by an alternative path. Two further directions are left as future enhancements. The first is AI-based emergency prioritisation, replacing the keyword rules with a model that understands negation and local-language wording and weighs message content, head-count and time since the last report. The second is scaling to large mesh networks, where smarter forwarding, such as probabilistic or neighbour-aware relaying, would be needed to control overhead.

VI. CONCLUSION

ResQ Mesh shows how inexpensive ESP32 and LoRa modules can form a temporary multi-hop channel that is intended to carry a victim's location, head-count and needs to a rescue point when towers and the Internet are unavailable. The relay rule of hop limit, duplicate cache and random delay is simple enough for small microcontrollers and needs no routing state. The dashboard turns raw messages into a ranked list that rescuers can act on, although its keyword rules need improvement. The next steps are field measurements of range and delivery with real hardware and the two enhancements outlined above.

DECLARATION

A generative AI tool was used solely for grammar checking of the manuscript text. The authors reviewed all content and take full responsibility for the manuscript.

REFERENCES

[1] J. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A study of LoRa: Long range & low power networks for the Internet of Things," *Sensors*, vol. 16, no. 9, Art. no. 1466, 2016.

[2] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi, "Long-range communications in unlicensed bands: The rising stars in the IoT and smart city scenarios," *IEEE Wireless Commun.*, vol. 23, no. 5, pp. 60–67, Oct. 2016.

[3] W. RajanBabu, N. Pushpalatha, R. Senthil Kumar, V. Gowsalya Devi, A. Mohamed Harrish, S. Supriya, and M. Vinothkumar, "Arduino based women safety device: An experimental study,"

Premier J. Sci., vol. 15, Art. no. 100241, 2025, doi: 10.70389/PJS.100241.

[4] M. C. Bor, U. Roedig, T. Voigt, and J. M. Alonso, "Do LoRa low-power wide-area networks scale?" in *Proc. 19th ACM Int. Conf. Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, Malta, 2016, pp. 59–67.

[5] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proc. 2nd IEEE Workshop Mobile Computing Systems and Applications*, New Orleans, LA, USA, 1999, pp. 90–100.

[6] T. Clausen and P. Jacquet, Eds., "Optimized link state routing protocol (OLSR)," IETF RFC 3626, Oct. 2003.

[7] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proc. 5th Annu. ACM/IEEE Int. Conf. Mobile Computing and Networking (MobiCom)*, Seattle, WA, USA, 1999, pp. 151–162.

[8] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko, "The Trickle algorithm," IETF RFC 6206, Mar. 2011.

[9] V. Nivedhitha, L. Mubaraali, S. Sathish Kumar, and S. Vijayalakshmi, "Optimized MANET for routing in clustered Internet of Things," Research Square, preprint, 2025, doi: 10.21203/rs.3.rs-6793997/v1.

[10] Espressif Systems, *ESP32 Series Datasheet*, Shanghai, China.

[11] Semtech Corp., *SX1276/77/78/79: 137 MHz to 1020 MHz Low Power Long Range Transceiver*, datasheet, Camarillo, CA, USA.

[12] u-blox AG, *NEO-6 u-blox 6 GPS Modules Data Sheet*, Thalwil, Switzerland.

[13] National Marine Electronics Association, *NMEA 0183 Interface Standard*, Severna Park, MD, USA.