Evaluation of a Multiprotocol Agnostic Distributed System for Natural Disasters ⋆

Samuel R. S. Oliveira * **Saulo N. Matos** *,**,***
Caetano M. Ranieri **** **Alan Kardek Rego Segundo** †
Marco A. Spohn ‡ **Vitor N. Vian** * **Jó Ueyama** *

* *ICMC - Instituto de Ciências Matemáticas e de Computação Universidade de São Paulo (USP), São Carlos – SP, Brazil*
** *School of Computing and Communications, Lancaster University, Lancaster, United Kingdom*
*** *Instituto Tecnológico Vale - ITV-MI, MG*
**** *Instituto de Geociências e Ciências Exatas (IGCE), Universidade Estadual Paulista (UNESP), Rio Claro – SP, Brazil*
† *Universidade Federal de Ouro Preto (UFOP), Ouro Preto – MG, Brazil*
‡ *Universidade Federal da Fronteira Sul (UFFS), Chapecó – SC, Brazil*

Abstract: In landslide scenarios, robust communication systems are essential to deliver timely warnings and support disaster response. Recent studies have investigated sensor networks and single-protocol solutions, highlighting the need for more flexible and resilient approaches. However, many existing methods do not fully address the trade-off between throughput, latency, and range in environments with limited infrastructure. This work proposes a multiprotocol communication system integrating LoRa, WiFi, and 4G to provide adaptable data transmission in adverse conditions. A series of controlled laboratory tests were conducted – using iperf, ping, Wireshark, and custom scripts – to measure throughput, latency, packet loss, and maximum range. The results show: (i) 4G provides the highest throughput but depends on cellular coverage, (ii) WiFi offers moderate throughput and low latency within shorter distances, and (iii) LoRa achieves extensive range but at the cost of reduced data rates and higher latency. These findings confirm that a flexible combination of protocols can improve situational awareness in landslide monitoring. Future work will include real-world field tests to refine energy usage further and validate the system's performance under practical deployment conditions.

Keywords: Disaster monitoring, multi-protocol communication, IoT, redundant systems, energy efficiency.

1. INTRODUCTION

Natural disasters, including urban floods and landslides, have caused extensive social, economic, and infrastructural damage in numerous regions of Brazil. Rapid urbanization – characterized by the construction of roads, buildings, and impermeable surfaces – reduces water infiltration and increases surface runoff, which can exceed the capacity of local drainage systems. As a result, floods and landslides cause widespread disruption and pose serious threats to public safety (de Moraes, 2023).

A recent example of the severity of these events is the climate-related disaster that affected Petrópolis, a mountainous region in Rio de Janeiro, in February and March 2022. More than 240 people lost their lives, and some areas experienced a staggering 258.6 mm of rainfall in just three hours—equivalent to the expected precipitation for the entire month. In Morro da Oficina alone, a single landslide resulted in 93 fatalities. Despite the city's ongoing containment efforts and over BRL 100 million invested in slope reinforcements, many residents remain vulnerable during alerts for heavy rainfall (de Fato , Redação).

Various municipalities have implemented localized solutions for disaster monitoring. For instance, Santo André's municipal environmental service (SEMASA) tracks water levels in real-time to guide commuters on flooded routes (Serviço Municipal de Saneamento Ambiental de Santo André (SEMASA), 2023). Similarly, the e-NOÉ system in São Carlos employs Wireless Sensor Networks (WSN) and the Internet of Things (IoT) to collect and transmit hydrological data via a 4G network, providing timely updates to emergency services (Ueyama et al., 2017).

This work introduces a hardware and protocol-agnostic communication system with built-in redundancy and resilience. The system employs a publish-subscribe messag-

⋆ This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

ing model with an MQTT [1] broker running on a low-cost Raspberry Pi acting as a coordinator. Upon detecting a potential hazard, it activates ESP32 boards through an analog transceiver signal, allowing data transmission over the most suitable protocol without risking excessive battery drain during periods of inactivity. This architecture has been successfully demonstrated in a lab environment, allowing extensive evaluations to characterize the relative performance of each protocol, particularly concerning data reliability, coverage range, throughput, and power consumption.

To achieve the objective of this study, which is to evaluate and compare the protocols integrated into our multiprotocol communication system for natural disaster monitoring, we formulated three research questions to guide our research:

- **RQ1** What are the protocols' packet delivery rates and latency characteristics under varied channel conditions and congestion levels in the multiprotocol communication system for natural disaster monitoring?
- **RQ2** How does each communication protocol perform under differing connectivity constraints, and to what extent could selecting the most suitable protocol in each scenario optimize the overall performance of a multiprotocol disaster-monitoring system?
- **RQ3** What are the limitations in deploying the multiprotocol communication system for disaster monitoring, and what best practices can be derived to optimize future implementations in disaster-prone areas?

2. RELATED WORKS

Fraile et al. (2020) analyzed the implementation of IoT networks in indoor environments, comparing LoRa (863 - 870MHz) and IEEE 802.15.4 (2.4GHz). The research highlights LoRa's stable long-range communication with low costs, making it advantageous when penetration through walls and floors is critical. This study supports the choice of LoRa in the present project, as it ensures reliable operation under various environmental conditions, mitigates communication failures, and provides long-range coverage. Morin et al. (2017) compared the expected lifespan of IoT devices across various wireless networks, focusing on energy consumption.

The comparison of communication technologies, such as IEEE 802.15.4, Bluetooth Low Energy (BLE), 802.11 Power Save Mode (PSM), 802.11ah, LoRa, and SigFox, provides valuable insights into the most energy-efficient protocols, aiding in the selection of technologies suitable for different operational conditions. The research highlights that LoRa is ideal for low-traffic scenarios, while BLE excels at short distances with high energy efficiency, supporting the project's multiprotocol approach. Hasan et al. (2022) further complements this discussion by demonstrating BLE's effectiveness in emergency alerts within urban environments, showcasing its value in creating a resilient and energy-efficient disaster monitoring system.

[1] MQTT (*Message Queuing Telemetry Transport*) is a lightweight protocol designed for IoT communication over resource-constrained networks.

Butler et al. (2019) presented an IoT system for monitoring landslide disasters using technologies such as SigFox and MQTT. While the study demonstrates the effectiveness of IoT systems in collecting and transmitting high-precision, real-time data with low maintenance and reduced operational costs, it is essential to note that SigFox is a paid service with a cost per service unit. This financial limitation can represent an obstacle to the implementation on a large scale or to projects with budget constraints. Furthermore, reliance on a paid service may reduce system flexibility and autonomy, which are critical factors in natural disaster monitoring scenarios where resilience and technological independence are essential.

Moreover, Szmeja et al. (2023) discusses implementing a modular IoT architecture incorporating advanced traffic management and fault tolerance mechanisms. Applying such approaches in the present project can ensure that communication is maintained even under adverse conditions, guaranteeing the delivery of critical data for the continuous operation of the disaster monitoring system.

From a comparative perspective, each protocol has clearly defined use cases tailored to specific application scenarios, reinforcing the idea of a multiprotocol system applied to a critical environment. The versatility to harness the benefits of all protocols without allowing limitations to become hindrances is essential for ensuring the proposed system's efficiency by utilizing the best aspects of each communication technology. Similarly, avoiding dependence on paid services (e.g., SigFox) is fundamental for maintaining financial sustainability and operational independence.

None of the previous approaches addressed creating an integrated system that combines multiple protocols efficiently in a unified disaster monitoring system, mitigating the individual limitations of each technology and avoiding additional costs. This project proposes to fill this gap by developing a communication system that guarantees continuous monitoring operation in critical situations, efficiently integrating technologies such as LoRa, WiFi, and 4G to create a robust, versatile, and economically viable solution.

3. METHODOLOGY

This section outlines the experimental setup and procedures used to evaluate the relative performance of LoRa, WiFi, and 4G under controlled laboratory conditions. The goal was to compare each protocol's throughput, latency, packet loss, and maximum communication range.

3.1 Hardware and Network Setup

We developed an experimental setup featuring a Raspberry Pi 5, powered by a dedicated source, that functions as the coordinator element. It hosts the internal MQTT broker and acquires sensor measurements published directly to the broker with Quality of Service (QoS) level 2 via a TTL-to-USB converter Segundo et al. (2017). Although hosting the broker on a low-power microcontroller could save energy, the Raspberry Pi was chosen for its ability to run a complete Linux environment, simplifying the development of local services (e.g., data logging) and integrating additional peripherals without overloading an

MCU's limited resources. This decision comes with trade-offs – notably higher power consumption and cost – yet strikes a balance between ease of development, networking flexibility, and performance for coordinating the system's broker.

As Communication Units (CUs), we employ ESP32 LILYGO T-Beam boards for WiFi and LoRa communication and ESP32 LILYGO T-SIM7000G modules for WiFi and 4G connectivity. Each CU subscribes to the internal MQTT broker and then publishes sensor measurements to an external broker. This design ensures reliable and persistent data delivery, as summarized in Table 1 and illustrated in Figure 1. While the CUs handle multiprotocol communication, the comparatively higher-capacity Raspberry Pi offers a stable platform for tasks like caching messages or managing local logic, facilitating a more flexible and robust system architecture.

Table 1. Summary of the System Architecture Components.

Component	Function
Sensors	Collect environmental data (e.g., soil moisture) and publish it directly to the coordinator's MQTT broker.
Coordination	Hosts the internal MQTT broker and facilitates internal communication between the sensors and the Communication Units (CUs). It does not communicate with the external world.
Communication Units (CUs)	Receive data from the coordinator's MQTT broker and publish it to the external MQTT broker. They are responsible for communicating with the external world using protocols such as LoRa, WiFi, and 4G.
External MQTT Broker	Stores the data received from the CUs with a persistence of one day. Acts as an intermediary between the CUs and the backend server.
Backend Server	Subscribed to the external MQTT broker, receives data from the CUs, stores it in a relational database, and provides a public API for data access.

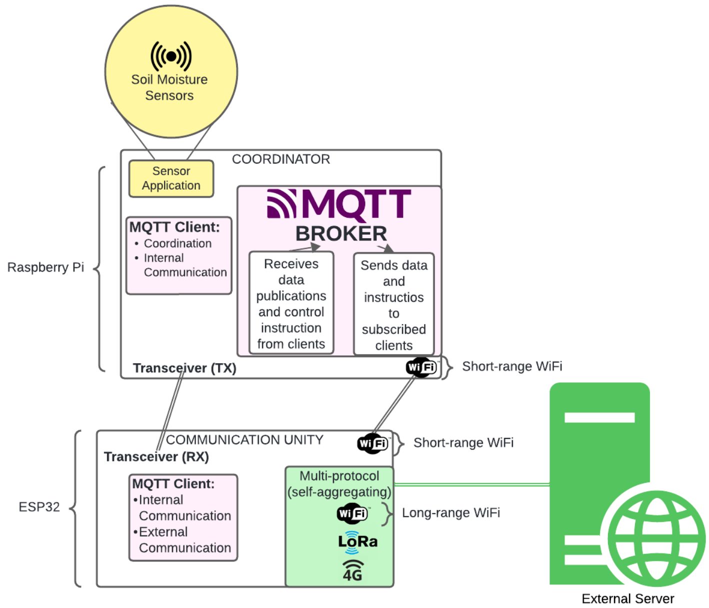

Figure 1. The proposed architecture comprises (i) a coordination unit (Raspberry Pi), (ii) multiple communication units (ESP32 boards, such as ESP32 T-Beam and NodeMCU), and (iii) on-demand transceivers for activation. The Raspberry Pi, powered continuously, hosts the internal MQTT broker and functions as the transmitter, sending signals that activate the communication units. Each ESP32 board includes a corresponding receiver and remains in low-power mode until awakened by the coordinator. Once active, the communication units support external multiprotocol communication (LoRa, WiFi, and 4G), ensuring robust data transmission under adverse conditions. The system architecture is self-aggregating, allowing new protocols to be added without additional wiring, while the on-demand transceivers maximize energy efficiency and resilience for continuous natural disaster monitoring.

Battery-operated ESP32 boards, configured as Communication Units, remain off under normal conditions. When the coordinator detects a hazard or initiates a measurement cycle, an analog transceiver signal activates these ESP32 devices, each running firmware to communicate through LoRa, WiFi, or 4G. The data flow of this arrangement is illustrated in Figure 2, highlighting how sensor information moves from the internal broker to the UCs and ultimately reaches an external MQTT broker and backend server for storage and analysis. During laboratory trials, a 2.4 GHz WiFi network was used to evaluate WiFi performance, while other protocols were tested under similar controlled conditions.

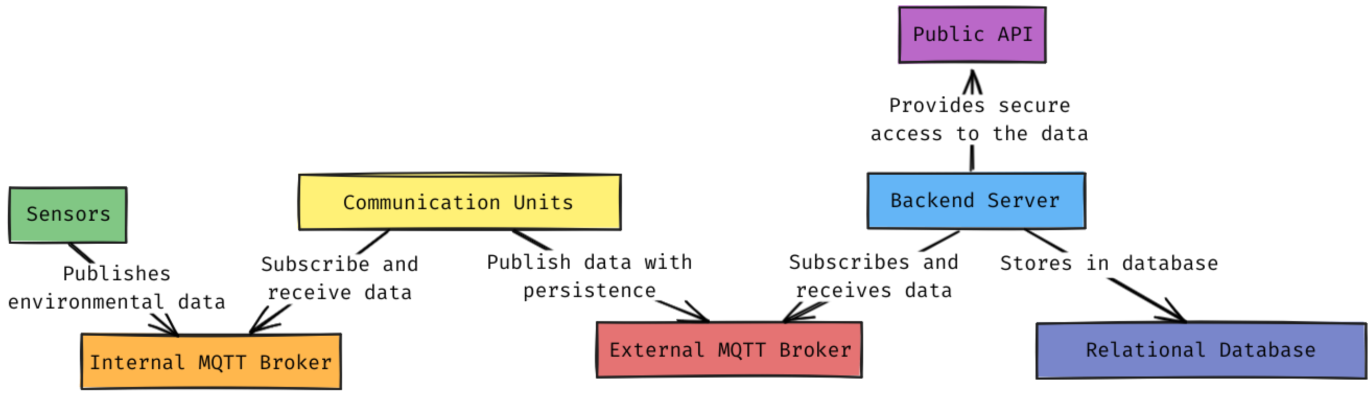

Figure 2. Project Data Flow Diagram.

Figure 3 shows the physical prototype, demonstrating the current stage of development and providing a clear view of how the Raspberry Pi, sensors, and ESP32 boards are integrated.

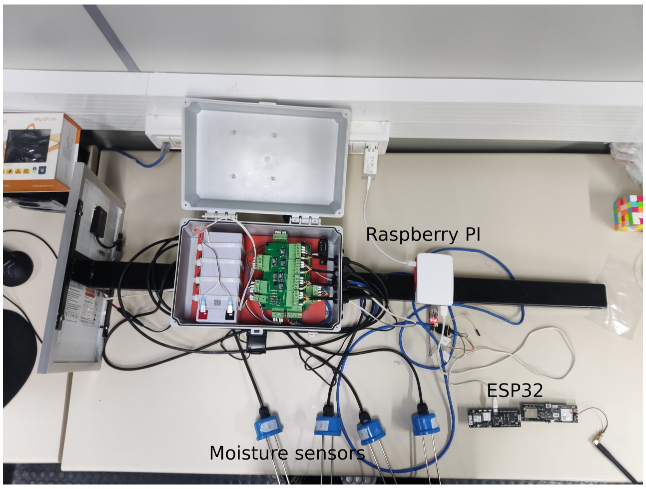

Figure 3. Initial Prototype of the Complete System in Operation

3.2 Software Tools and Scripts

Throughput measurements under WiFi and 4G were obtained using the open-source application `iperf`, which established a client-server connection and recorded transmission rates over 60-second test intervals. Latency data for WiFi and 4G were gathered via the `ping` command, sending a specified number of ICMP echo requests and logging the average round-trip time. For LoRa, we developed custom C++ scripts and deployed them to ESP32 boards using the PlatformIO ecosystem; these scripts enabled the ESP32 boards to send fixed-size packets, receive acknowledgments from a gateway, and calculate throughput and packet loss. In parallel, we employ Wireshark to monitor WiFi and 4G network traffic, providing a detailed analysis of packet retransmissions and drop rates.

The complete source code for the coordinator, sensors, and communication units is available in the project's GitHub repository Oliveira (2024). LoRa-specific measurement scripts for throughput, latency, and packet loss are documented in a public GitHub repository Oliveira (2025). These scripts operate at a frequency of 915 MHz, using the default configurations of the sandeepmistry/LoRa library Mistry (2017). Expressly, the spreading factor (SF) is set to 7, the bandwidth is 125 kHz, and the maximum payload size is limited to 255 bytes per packet at the physical layer. For larger payloads, such as the 1 kB packets used in these tests, the library automatically fragments the data into multiple frames. We optimize these default settings for shorter-range, higher-throughput applications Mistry (2017). By leveraging these configurations and tools, the experimental setup ensured accurate and reproducible measurements of LoRa's performance under controlled conditions while allowing for future adjustments to suit varying application needs.

3.3 Test Procedure

Established standards from the IETF and ITU guide our measurement procedures. For example, *RFC 2679* defines "one-way delay" as the time from the source's wire-time transmission to the destination's wire-time reception; the methodologies in Section 3.6 of that document closely match our use of `ping` to capture round-trip times for WiFi and 4G Almes et al. (1999). Meanwhile, *RFC 6349* specifically recommends tools like `iperf` for TCP throughput evaluations, indicating that "a control connection is used to set up and manage the test data transfers", which we mirrored by running continuous 60-second transmissions Constantine et al. (2011). Additionally, *ITU-T Y.1540* and *Y.1541* emphasize measuring loss ratios and quality parameters across standardized intervals, aligning with our use of Wireshark to capture packet retransmissions and calculate loss rates itu (2019a,b). By following these guidelines—focusing on defined packet sizes, time windows, and stable TCP/ICMP flows—we ensured that our lab-based tests conform to recognized best practices for network performance assessment. Each experiment began by flashing the ESP32 boards with firmware enabling only one protocol at a time (LoRa, WiFi, or 4G). Baseline runs ensured reliable connectivity and proper configuration (for instance, WiFi channel assignments or LoRa spreading factors). After confirming these initial settings, we collect the following metrics:

3.3.0.1. Packet Loss Packet loss was inferred by comparing the total number of packets sent to the number successfully received. The packet loss rate (%) was calculated according to Equation 1.

$$\text{Packet Loss (\%)} = \left(1 - \frac{\text{Packets Received}}{\text{Packets Sent}}\right) \times 100. \quad (1)$$

To operationalize this:

(1) A total of 100 packets was sent for each test.
(2) The number of lost packets was recorded and subtracted from the total sent to determine how many were received.
(3) The uncertainty was derived as the standard error of the mean (SEM) across repeated runs.

3.3.0.2. Latency Latency was measured using repeated echo requests (e.g., `ping` for WiFi and 4G) or by logging the round-trip time of packets in LoRa tests. The average latency (in milliseconds) was computed using Equation 2.

$$\text{Average Latency (ms)} = \frac{\sum_{i=1}^{n} \text{Individual Latency (ms)}}{n}. \quad (2)$$

The specific procedure involved:

(1) Sending 50 packets consecutively for each protocol test.
(2) Summing the individual latencies to obtain a total latency value.
(3) Dividing by the number of packets ($n = 50$) to find the mean latency.
(4) Computing the standard error of the mean to quantify uncertainty.

3.3.0.3. Throughput Throughput (in kilobits per second) was captured by sending a known amount of data within a fixed time interval. For LoRa, custom C++ scripts on the ESP32 boards handled packet transmissions and acknowledgments, while for WiFi and 4G, tools such as `iperf` were used. The throughput calculation followed Equation 3.

$$\text{Throughput (kbps)} = \frac{\text{Packet Size (bits)} \times \text{Number of Packet}}{\text{Total Time (s)} \times 1000} \quad (3)$$

The key steps were:

(1) Setting the packet size to 1024 bytes (8192 bits).
(2) Transmitting 100 packets per run.
(3) Measuring the total transmission time in seconds, then averaging over multiple runs.
(4) Computing the standard error of the mean (SEM) to express variability across trials. [2]

3.3.0.4. Maximum Communication Range To estimate the maximum communication range, the transmitter and receiver moved progressively farther apart until the packet loss rate rose above 90%.

[2] The SEM was calculated as SEM $= \sigma/\sqrt{n}$, where σ is the standard deviation and n is the number of samples.

4. RESULTS

This section presents the results obtained for throughput, latency, packet loss, and maximum communication range under controlled laboratory conditions. While these data offer valuable insights, real-world factors—such as electromagnetic interference, terrain, or the distance from cellular towers—can significantly influence performance. Table 2 summarizes the measured metrics for WiFi, LoRa, and 4G, highlighting each protocol's unique speed, reliability, and coverage trade-offs.

4.1 Throughput

Figure 4 illustrates that 4G recorded the highest average throughput, at 131 ± 9 kbps. WiFi followed closely with 122 ± 35 kbps, indicating its capability to handle moderate data volumes. LoRa, designed for low-bandwidth transmissions, exhibited an average throughput of 62 ± 7 kbps.

The wider variance observed in WiFi's throughput reflects the protocol's susceptibility to interference in laboratory conditions. Despite this variability, the measured WiFi throughput remains adequate for real-time monitoring scenarios with moderate bandwidth requirements.

For WiFi and 4G, throughput measurements were taken using `iperf`, setting up a server-client pair on the devices and running continuous transmissions for 60-second intervals iperf developers (2024). In contrast, LoRa measurements involved custom C++ code running on an ESP32 T-Beam (communication unit) and a LoRaMesh gateway from Radioenge, interfaced with a NodeMCU ESP8266. The ESP32 transmitted 1,kB data packets, which the gateway acknowledged. The total throughput was computed on the ESP32 by tracking the transmission time for 100 consecutive packets.

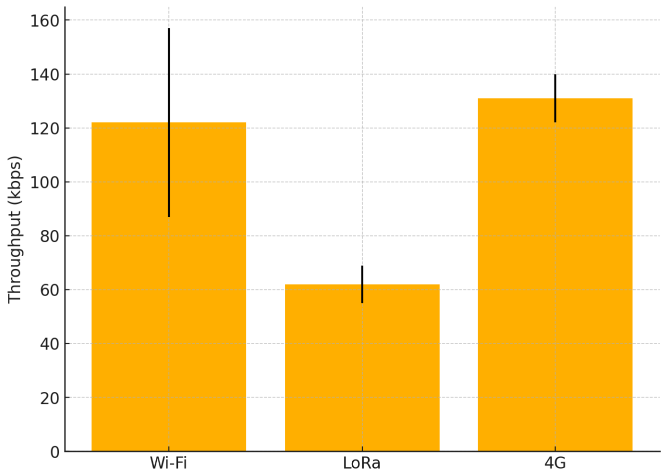

Figure 4. Comparison of Throughput for LoRa, WiFi, and 4G.

Table 2. Comparison of Metrics between Protocols.

Protocol	Throughput (kbps) ± Error	Latency (ms) ± Error	Packet Loss (%) ± Error	Maximum Range (m)
Wi-Fi	122 ± 35	53 ± 1	2.0 ± 0.4	25
LoRa	62 ± 7	1072 ± 22	4.8 ± 1.2	1500
4G	131 ± 9	88 ± 1	1.6 ± 0.4	Variable*

Table 2. * 4G depends on cellular coverage, which can vary or be unavailable in isolated areas.

4.2 Latency

Latency is a critical factor for real-time monitoring systems. As shown in Figure 5, Wi-Fi exhibited the lowest average latency at 53 ± 1 ms, followed by 4G at 88 ± 1 ms. LoRa, on the other hand, presented a significantly higher latency of 1072 ± 22 ms, limiting its applicability in scenarios requiring strict, low-latency communication.

The higher latency values for LoRa stem from its long-range, low-bandwidth design. WiFi and 4G remained relatively stable in controlled conditions, yet real-world factors such as physical obstructions and wireless interference could substantially increase these values.

WiFi and 4G latencies were measured via the `ping` command, sending 50 ICMP packets and calculating the average round-trip time Microchip Developer (2024). For LoRa, the ESP32 T-Beam issued "Ping" messages that the NodeMCU acknowledged, again using the scripts referenced in Oliveira (2025). This process was repeated 50 times, and the average round-trip time (RTT) was determined along with its associated uncertainty, represented by the standard error of the mean (SEM). Combining these procedures allowed for the evaluation of each protocol's performance under consistent test conditions while reflecting real-world constraints such as packet acknowledgment and limited bandwidth.

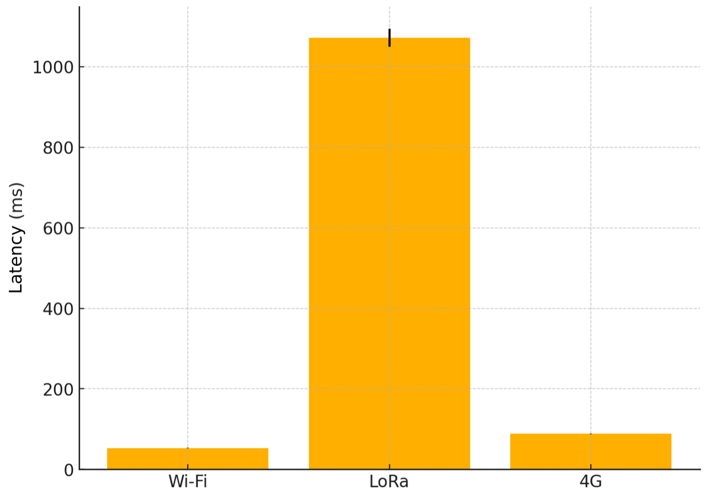

Figure 5. Comparison of Latency for LoRa, WiFi, and 4G.

4.3 Packet Loss

As indicated in Figure 6, the packet loss rate was relatively low for all protocols, with 4G at $1.6 \pm 0.4\%$, WiFi at $2.0 \pm 0.4\%$, and LoRa at $4.8 \pm 1.2\%$. The slightly higher rate for LoRa is not unexpected, given its focus on long-range transmissions and susceptibility to environmental noise. Although the lab environment minimized such interference, LoRa's loss rate may increase in real-world conditions due to more pronounced external factors.

WiFi and 4G loss rates were logged via Wireshark, which captured packet traffic and retransmissions Wireshark Foundation (2024). For LoRa, the sending ESP32 counted the total packets transmitted and waited for corresponding acknowledgments. A 2-second timeout was used to classify a packet as lost if no "ACK" was received.

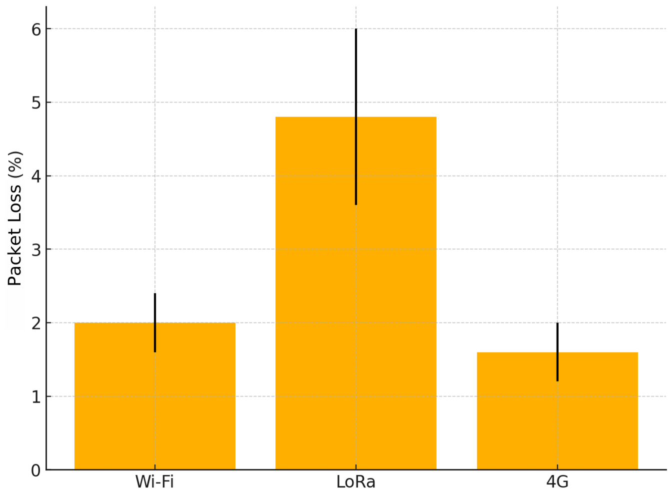

Figure 6. Packet Loss Rates for LoRa, WiFi, and 4G.

4.4 Maximum Communication Range

Under our test conditions at USP São Carlos, the LoRa gateway remained indoors in the laboratory. At the same time, measurements were taken outdoors at progressively greater distances until the packet success rate dropped to around 10%. This approach yielded an effective range of approximately 1.5 km for LoRa, in contrast to WiFi's coverage of roughly 25 m indoors (with potential improvement in open spaces) and the varying range of 4G, which can extend from tens to hundreds of meters in urban environments. These findings are broadly consistent with those of Wang *et al.* Wang et al. (2017), who reported that LoRa packet loss rates could spike to 60–90% at indoor distances near 1,200 m—highlighting that structural obstructions and device placement significantly affect performance. Moreover, Wang *et al.* observed no strict correlation between payload length and packet loss rate, suggesting that larger payloads (with forward-error correction) need not necessarily degrade performance. Weather conditions, particularly rainfall, can also elevate packet loss Wang et al. (2017), emphasizing the need for robust higher-layer protocols or network redundancies in mission-critical IoT applications. Although LoRa provides longer-range coverage compared to WiFi and can function under infrastructure-limited conditions, its reliability in complex indoor or adverse-weather scenarios calls for careful system design and, if necessary, supplementation by other protocols to meet stricter low-latency or high-delivery requirements.

4.5 Test Conditions and Limitations

The laboratory environment offered consistency for benchmarking but did not replicate external factors such as severe weather or unstable power grids. Additionally, 4G performance relies heavily on local cellular coverage, which may be scarce in remote regions. These constraints suggest that in-field measurements are necessary to refine the system for practical disaster monitoring.

Finally, while WiFi naturally supports the TCP/IP stack used by the MQTT protocol, LoRa does not. This discrepancy explains why throughput and packet loss measurements for LoRa required custom firmware for packet creation and acknowledgment. By contrast, WiFi and 4G could natively leverage tools like `iperf` and Wireshark. As a result, LoRa's real-world performance will depend on maintaining reliable acknowledgment schemes and may vary more than WiFi or 4G when deployed across large or obstructed areas.

5. DISCUSSION

We can answer the research questions based on the results presented in Section 4.

RQ1: What are the protocols' packet delivery rates and latency characteristics under varied channel conditions and congestion levels in the multiprotocol communication system for monitoring natural disasters?

Our controlled laboratory tests show that 4G and WiFi generally yield higher throughput and lower latency, with 4G slightly surpassing WiFi in average throughput (131 kbps vs. 122 kbps). However, both protocols rely on stable infrastructure, limiting their utility in remote or less-developed regions. LoRa, by contrast, exhibits significantly higher latency (on the order of one second) and lower throughput (62 kbps), yet delivers the advantage of a much longer communication range and improved resilience in sparse-network environments.

RQ2: How does each communication protocol perform under differing connectivity constraints, and to what extent could selecting the most suitable protocol in each scenario optimize the overall performance of a multiprotocol disaster-monitoring system?

Selecting the most suitable protocol depends on local connectivity constraints. Where dependable cellular coverage or a local network is available, 4G or WiFi provides rapid data transfer and low latency, making them ideal for near-real-time monitoring. Conversely, in scenarios lacking reliable power or network infrastructure, LoRa's extended range and lower power requirements make it the preferred solution despite its reduced data rates. Adapting protocol usage to each site's conditions ensures robust data collection, minimizing downtime or coverage gaps in disaster monitoring.

RQ3: What are the limitations in deploying the multi-protocol communication system for disaster monitoring, and what best practices can be derived to optimize future implementations in disaster-prone areas?

Key challenges include the high energy demands of continuous 4G and WiFi operation, the infrastructure dependence of these protocols, and the limited data rate of LoRa. Best practices involve employing a hybrid, on-demand model in which communication units remain powered down until activated and performing thorough field tests to account for variable terrain, adverse weather, and interference. A flexible, multiprotocol architecture allows LoRa to cover broad, low-infrastructure areas while 4G or WiFi handles higher-bandwidth tasks. By dynamically switching protocols based on coverage, power availability, and data volume, operators can achieve energy efficiency and reliability in disaster scenarios.

6. CONCLUSION

The results obtained in this study confirm the feasibility and importance of a multiprotocol communication system for natural disaster monitoring, with an initial focus on landslide detection. Evaluations under controlled laboratory conditions demonstrate that each protocol—LoRa, WiFi, and 4G—presents distinct advantages and limitations, which can be strategically combined to ensure resilient, energy-efficient operation even in adverse environments. LoRa offers long-range and low energy consumption but suffers from reduced throughput and higher latency, making it suited for transmitting small data packets over remote areas. In contrast, WiFi delivers high throughput and low latency but within a limited range and at the cost of increased power draw, favoring use in locations with existing infrastructure. Meanwhile, 4G provides broader data rates and flexible coverage, constrained by cellular availability and elevated energy requirements.

Despite these constraints, a redundant communication framework seamlessly switching between protocols can leverage each technology's strengths to optimize performance and reliability in different deployment scenarios. In addition, ad hoc mesh networks (e.g., based on IEEE 802.11s) represent a promising approach to address coverage gaps in remote environments, allowing nodes to extend wireless reach back to a gateway without relying on preexisting infrastructure. Real-world field tests remain an essential next step for validating the laboratory findings and adapting the system to specific deployment conditions, where factors like terrain, physical obstacles, and weather can significantly influence performance.

Future work also includes integrating advanced power-saving strategies—such as selective activation of Communication Units—thus extending device longevity in battery-powered deployments. Additional emerging technologies could further expand the system's capabilities. Combining low-cost hardware, multiple communication protocols, and a flexible, publish-subscribe architecture provides a robust and versatile approach to monitoring landslides and other natural disasters. This multiprotocol design enables timely data collection, rapid decision-making, and improved protection of lives and property in areas prone to catastrophic events.

REFERENCES

(2019a). ITU-T Recommendation Y.1540 (Network Performance Parameters for IP-based Services). International Telecommunication Union. https://www.itu.int/rec/T-REC-Y.1540, Accessed: 2025-01-29.

(2019b). ITU-T Recommendation Y.1541 (Network Performance Objectives for IP-based Services). International Telecommunication Union. https://www.itu.int/rec/T-REC-Y.1541, Accessed: 2025-01-29.

Almes, G., Kalidindi, S., and Zekauskas, M. (1999). RFC 2679: A One-way Delay Metric for IPPM. Internet Engineering Task Force. https://www.rfc-editor.org/rfc/rfc2679, Accessed: 2025-01-29.

Butler, M., Angelopoulos, M., and Mahy, D. (2019). Efficient iot-enabled landslide monitoring. In *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 171–176.

Constantine, M., Forget, G., Geib, R., and Mortensen, A. (2011). RFC 6349: Framework for TCP Throughput Testing. Internet Engineering Task Force. https://www.rfc-editor.org/rfc/rfc6349, Accessed: 2025-01-29.

de Fato (Redação), B. (2024). Tragédia das chuvas em petrópolis (rj) completa dois anos. https://www.brasildefato.com.br/2024/02/15/tragedia-das-chuvas-em-petropolis-rj-completa-\\dois-anos. Accessed: 2025-01-25.

de Moraes, R.B.F. (2023). *Development of the Urban Hydrological Disasters Susceptibility Index*. Master's thesis, Fundação Universidade Federal de Mato Grosso do Sul.

Fraile, L.P., Tsampas, S., Mylonas, G., and Amaxilatis, D. (2020). A comparative study of lora and ieee 802.15.4-based iot deployments inside school buildings. *IEEE Access*, 8, 160957–160981. doi:10.1109/ACCESS.2020.3020685.

Hasan, R., Hasan, R., and Islam, T. (2022). Smart city technology for disaster management: Demonstrating the use of bluetooth low energy (ble) beacons for emergency alert dissemination. In *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*, 931–932.

iperf developers (2024). *How to Use iPerf: A network testing tool*. Disponível em: https://iperf.fr/iperf-doc.php. Acesso em: 10 nov. 2024.

Microchip Developer (2024). *How to Use Ping*. Disponível em: https://developerhelp.microchip.com/xwiki/bin/view/applications/tcp-ip/network-tools/. Acesso em: 10 nov. 2024.

Mistry, S. (2017). arduino-lora library api documentation. https://github.com/sandeepmistry/arduino-LoRa/blob/master/API.md. Accessed: 2025-01-19.

Morin, É., Maman, M., Guizzetti, R., and Duda, A. (2017). Comparison of the device lifetime in wireless networks for the internet of things. *IEEE Access*, 5, 7097–7114. doi:10.1109/ACCESS.2017.2688279.

Oliveira, S.R.S. (2024). Mondesa project repository. https://github.com/MonDesa. Accessed: 2025-01-19.

Oliveira, S.R.S. (2025). Lora data measurements gist. https://gist.github.com/rubenszinho/2e50a1f3cd96b4069ace136cc1c415cb. Accessed: 2025-01-19.

Segundo, A.K.R., Pinto, E.S., de Barros Monteiro, P.M., and Martins, J.H. (2017). Sensor for measuring electrical parameters of soil based on auto-balancing bridge circuit. In *2017 IEEE SENSORS*, 1–3. IEEE.

Serviço Municipal de Saneamento Ambiental de Santo André (SEMASA) (2023). Monitoramento de rios e córregos. https://portais.santoandre.sp.gov.br/defesacivil/monitoramento-rios-e-corregos/. Acesso em: 05 nov. 2024.

Szmeja, P., Tsampas, S., Mylonas, G., and Amaxilatis, D. (2023). Assist-iot: A modular implementation of a reference architecture for the next generation internet of things. *Electronics*, 12(4), 854. doi:10.3390/electronics12040854.

Ueyama, J., Faiçal, B.S., Mano, L.Y., Bayer, G., Pessin, G., and Gomes, P.H. (2017). Enhancing reliability in wireless sensor networks for adaptive river monitoring systems: Reflections on their long-term deployment in brazil. *Computers, Environment and Urban Systems*, 65, 41–52.

Wang, S.Y., Chen, Y.R., Chen, T.Y., Chang, C.H., Cheng, Y.H., Hsu, C.C., and Lin, Y.B. (2017). Performance of lora-based iot applications on campus. In *Proc. of the 2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, 1–5. IEEE. doi:10.1109/VTCFall.2017.8288427. https://ieeexplore.ieee.org/document/8288427, Accessed: 2025-01-30.

Wireshark Foundation (2024). *How to Use Wireshark: Packet capture explained*. Disponível em: https://www.wireshark.org/docs/wsug_html_chunked/. Acesso em: 10 nov. 2024.