ENHANCING RELIABILITY IN IOT BORDER NETWORKS: A WIRELESS MESH APPROACH WITH QOS GUARANTEES FOR GROUP ACTUATION

Victor Côrtes and Markus Endler

LAC, Department of Informatics, Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Rio de Janeiro, Brazil

ABSTRACT

In many IoT applications involving synchronized sensing and actuation at multiple nodes, enactment by the system, must be reliable, requiring enhanced protocols to ensure correct physical outcomes. This paper introduces 'Group Actuation Quality of Service (GA-QoS)' as a concept for coordinated, nearly synchronous actuation and presents a framework for its implementation in IoT mesh networks, particularly under dynamic conditions with communication failures. A representative scenario, pesticide spraying in farmland, was used to validate the framework using off-the-shelf hardware and open-source code. The main contribution is a practical methodology for achieving reliable parallel actuation across distributed devices. Performance evaluation confirmed feasibility, demonstrating reliable parallel actuation on distributed physical devices under challenging conditions, including network partitioning and interference.

KEYWORDS

Internet of Things, Mesh network, Group Actuation, Reliability, IoT Actuation Devices, QoS

1. INTRODUCTION

The Internet of Things (IoT) is frequently employed in control applications that automate processes, where actuation mechanisms alter the state of physical objects in the environment in response to sensed variables. In safety-critical applications, this sensing-and-actuation process demands stringent reliability, necessitating specialized and robust protocols to ensure accurate and effective actuation, as incorrect or delayed actions may result in operational failures or safety hazards [1]. Many IoT applications, such as the synchronized operation of valves in industrial production plants or coordinated control of locks in livestock farms and concert venues, require coordinated actuation among multiple devices. Ensuring the reliability and near-synchronous effectiveness of such group actuations can be challenging due to factors including interference, wireless jamming, and device mobility.

The importance of reliability in IoT extends beyond technical performance. Errors in actuation, often caused by anomalous data or communication failures, can compromise user safety and lead to irreversible environmental or operational consequences [2]. Quantifiable reliability metrics are therefore essential for applications such as remote monitoring, autonomous vehicles, and smart infrastructure [3]. For instance, distributed IoT systems play a critical role in power grid management, healthcare monitoring, and transportation control [4]. Policymakers and regulatory bodies have expressed growing concern over these issues, particularly in safety-critical domains where system dependability is a prerequisite for public trust and regulatory compliance [5]. Addressing these reliability challenges is essential for the continued advancement and responsible

deployment of IoT technologies, especially when taking into consideration IoT frequent volatile context of implementation.

IoT environments are frequently characterized by intermittency, network partitioning, and other disruptive factors, particularly prevalent in mobile and infrastructure-limited scenarios. These conditions pose significant challenges to the reliability of IoT systems. This research addresses these challenges by investigating the provision of guarantees to system architects and programmers, thereby aiming to ensure request fulfillment under such demanding conditions. The study identifies factors compromising reliability and evaluates strategies to mitigate these challenges. Errors in IoT applications can pose significant risks to user safety[2], as anomalous data generation and transmission may lead to incorrect actuations, potentially endangering lives. Therefore, quantifiable reliability measures are essential to address these concerns. Reliable connectivity is critical for diverse applications, including remote monitoring, control systems, autonomous vehicles, and smart city infrastructure [3]. For example, the dependable operation of distributed IoT devices is vital for managing power grids, ensuring information flow, and maintaining the correct operation of critical services[4]. Moreover, policymakers and regulatory bodies have a vested interest in these issues, particularly for safety-critical applications such as healthcare monitoring, transportation systems, and smart home devices[5]. The imperative to address these reliability challenges is fundamental for the future development and success of IoT technology [3].

This paper introduces the concept of Group Actuation Quality of Service (GA-QoS), specifically applied to coordinated actuation, and presents a methodology for designing and implementing IoT solutions within mesh networks, particularly in dynamic environments characterized by unpredictable communication failures. The developed protocol is implemented at the application layer, requiring only minor adjustments to other network stack layers. Experimental validation involved groups of nodes creating a Wireless Mesh Network (WMN) that connected sensors and actuators as client-end devices, with capabilities for middleware-mediated cloud interaction.

This paper contributes to the conceptual design and implementation of a general-purpose IoT architecture and the performance analysis of the toolset when ideally implemented. The proposed architecture allows for user interaction through the cloud, facilitated by mobile devices using an Android application.

The paper is organized as follows: Section 2 summarizes related work, Section 3 presents the proposed system architecture and its envisioned application scenario, Section 4 settles the main terms to be utilized across the study, Section 5 describes the proposed protocol, defines with greater detail the tools used in the creation of the experiments, and defines the configuration of each test, Section 6 provides practical results from the implementation, and Section 7 concludes and envisions future works.

2. BACKGROUND AND RELATED WORK

In this section, we present background information on research aimed at Reliability in IoT, the scope and boundaries of this study, and an overview of the most relevant developments.

2.1. Scope

As discussed in Section 1, this study investigates mechanisms for achieving dependable actuation in Internet of Things (IoT) environments subject to intermittent connectivity, network partitioning, and other structural disruptions. The focus lies in identifying and formalizing

reliability guarantees, such as time-bounded execution and quorum-based confirmation, that system designers can apply to support request fulfillment under adverse conditions. These guarantees are examined within the context of real-world IoT deployments, emphasizing scenarios where group coordination and environmental constraints necessitate robust reliability strategies.

2.2. Comparative Analysis of Related Works

This section examines existing research concerning Internet of Things (IoT) reliability, particularly focusing on aspects such as device behavior, actuator operations, and system-level implementation strategies. While findings indicate that prior investigations contribute significantly to the understanding of reliability across various environments, many primarily focus on concerns such as routing efficiency or low-level scheduling predictability. In contrast, the current study uniquely addresses the specific and complex challenges of ensuring reliable coordinated group actuation within dynamic IoT mesh networks.

A significant area of divergence among existing works lies in their approaches to network dynamics and node mobility. For instance, [6] and [7] primarily address the complexities of dynamic, latency-sensitive applications by focusing on real-time adaptation to topology changes, often relying on compatible node operations to maintain responsiveness in rapidly evolving environments. This contrasts with the present study's emphasis on flexible configurations and middleware-mediated communication between both stationary and mobile nodes. This design choice inherently minimizes the reliance on strict, system-wide synchronism, proving advantageous in scenarios with unpredictable topological variability. Furthermore, while [8] and[9] explore alternative strategies, such as remote mobile control and 5G integration for specific home and building management contexts, our research is distinct in its deployment of a comprehensive middleware-driven architecture tailored for dynamic reconfiguration and robust communication across diverse, decentralized settings

The role and functionality of actuator nodes also reveal important distinctions among the reviewed literature. While [6] and [7] utilize actuator nodes for real-time operational adjustments within dynamic feedback loops based on sensor inputs, other works like [8] employ actuators for more basic on/off operations in home automation settings. Similarly,[9] applies actuators for lighting and HVAC control in building management systems. The present study, while also applicable to scenarios such as lighting control, significantly extends beyond these functionalities. Our work distinguishes itself by its emphasis on broader applicability for various group actuation scenarios and, crucially, by integrating a robust request confirmation mechanism. This explicit focus on ensuring request fulfillment through confirmations is a central aspect of our study and is often unaddressed in existing research that primarily concentrates on real-time individual actuator control or basic command execution.

Regarding implementation strategies, the surveyed literature presents a spectrum from simulation-based evaluations, as demonstrated by [7], to real-world prototypes, as seen in [8] and [9]. The current study adopts a practical approach, utilizing ESP-32 nodes in a WMN with cloud connectivity via mobile gateway nodes. This configuration aligns with the real-world deployment focus of some of the analyzed work, but differentiates itself by the inclusion of cloud connectivity and a strong presence of the middleware layer, which facilitates enhanced flexibility and scalability.Implementation strategies across the reviewed studies range from simulation-based evaluations, as demonstrated by [7], to real-world prototypes, as seen in [8] and [9]. The current study adopts a practical approach, utilizing ESP-32 nodes in a WMN with cloud connectivity via mobile gateway nodes. This configuration aligns with the real-world deployment focus of some of the analyzed work, but differentiates itself by the inclusion of cloud

connectivity and a strong presence of the middleware layer, which facilitates enhanced flexibility and scalability.

In summary, while existing research has advanced various aspects of IoT reliability, including low-level scheduling, routing efficiency, and specific actuator controls, our work offers a comprehensive solution for reliable group actuation in dynamic IoT border networks. This is achieved through the introduction of quantifiable Group Actuation Quality of Service (GA-QoS) guarantees, a middleware-based coordination mechanism, and practical validation under simulated real-world conditions. This approach provides a distinct application-layer reliability model that is not comprehensively present in similar existing methodologies

2.3. Evaluation Metrics and QoS

The findings from the selected articles indicate that evaluation parameters for addressing reliability challenges in IoT systems under conditions of mobility and intermittent connectivity vary depending on specific applications and fields, but several core metrics consistently emerged across studies. These metrics include latency, throughput, packet delivery and loss rates, ping ratios, node/service availability, convergence time, energy levels, and memory usage.

Latency, measured as mean latency, maximum latency, or latency in specific hops, is crucial for understanding the timing performance of IoT systems. Similarly, throughput assesses the network's data handling capacity. Packet delivery and loss rates evaluate the efficiency of data transmission, while ping ratios gauge network responsiveness. Monitoring node/service availability ensures continuous operation, and convergence time assesses the time required for network stabilization. Given IoT devices' resource constraints, monitoring energy consumption is also important for network longevity, and memory usage helps evaluate the resource efficiency of IoT systems.

Additionally, the study identified several Quality of Service (QoS) metrics present in few works that differed from those typically found in studies related to the subject. These include service cost, service time, service load, reliability based on historical performance, information accuracy, energy consumption, coverage, and the correctness of received data blocks [10].

3. SCENARIOS AND APPLICATIONS

A review of the literature, presented in Section 2, identified a lack of studies addressing IoT applications that utilize Wireless Mesh Networks (WMNs) in conjunction with actuator nodes, coordinated group actuation, node mobility, network partitioning, and individual node failure. Few works consider scenarios where wireless infrastructure does not fully cover all nodes, requiring communication to rely on multi-hop routing within the mesh. To address this gap, this section introduces a plausible scenario applicable to IoT systems operating under such constraints. The scenario includes use cases such as habitat monitoring, emergency response, and agricultural automation, as illustrated in Figure 1. These systems may incorporate crowdsourced data collection and distributed actuation across ground nodes, supported by WMNs and complemented by Unmanned Aerial Vehicles (UAVs) and Mobile Devices (MDs) for coordination and decision-making.

The scenario focuses on IoT applications designed to automate distributed actuation on physical entities or devices, operating in geographically remote areas with limited infrastructure, such as habitat preservation, emergency situations in rural areas, and agriculture as illustrated in the figure 1. These applications may benefit from crowdsourcing approaches, where data collection

and node actuation are distributed across groups of ground nodes. These nodes are interconnected through WMNs, and Unmanned Aerial Vehicles (UAVs) and Mobile Devices (MDs) are employed to unify data collection and enable decision-making.

The scenario centers on IoT applications operating in geographically remote areas with limited infrastructure, such as habitat preservation, emergency situations in rural areas, and agriculture as illustrated in the figure 1. These applications may benefit from crowdsourcing approaches, where data collection and node actuation are distributed across groups of ground nodes. These nodes are interconnected through WMNs, and Unmanned Aerial Vehicles (UAVs) and Mobile Devices (MDs) are employed to unify data collection and enable decision-making.

In this architecture, ground nodes are organized into localized WMNs, with each group covering a specific area. The UAV or MD with mobile-hub, would be responsible for connecting with at least one ground node from each group as it traverses the designated area. By doing so, the system would be able collect and relay data, or send actuation, scheduling requests to the entire network via a single ground node. This design allows the middleware to communicate efficiently with all nodes, simplifying data transmission and decision-making.

The positions of the ground nodes are assumed to be fixed and predefined for the duration of the application, which simplifies network management and routing within the WMN. However, node mobility remains a consideration for the UAVs and MDs as they move between different parts of the WMN.

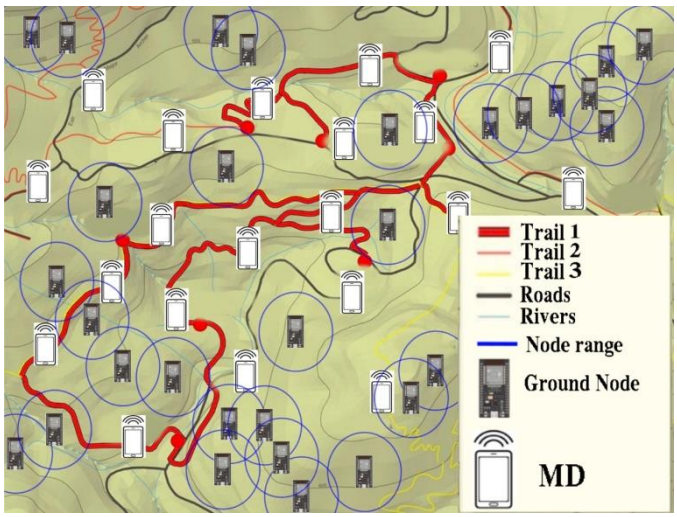

Figure 1. Scenario Overview, containing multiple MDs and ground nodes

3.1. Challenges of the Architecture

There are certain downsides to this particular architecture. The presence of a mobile node in each WMN for sufficient time to enable communication is necessary, and knowing the locations of the nodes is essential. Physical obstacles can hinder the movement of mobile nodes to more isolated areas. Nodes will inevitably fail and need to be replaced or recharged, including those in remote and hard to access locations.

In such application, a group actuation QoS(GA-QoS), as described in sections 4.2 and 5, could be useful when dealing with weather forecasting, as it could be requested that all nodes in a group take pressure and humidity measurements within a time interval of half a second. If these

requirements are not met, all measurements taken should be discarded and the request attempted once more.

4. FUNDAMENTAL CONCEPTS

This section focuses on concept definitions and terms that are instrumental for understanding the meaning of reliability in IoT operations.

4.1. Reliability in IoT

Reliability in IoT is critical, mainly because actuation changes the environment state, which may have a long-term, harmful and/or non-reversible effect. It focuses on understanding possible failures and their causes. Reliability can be analyzed across the different "layers" of IoT and multiplicity of actuators:

4.1.1. Device Reliability

IoT devices, particularly sensors and actuators, often operate under strict constraints related to battery life, memory, and processing power [11]. Battery limitations are significant, especially in inaccessible locations. Restricted memory and computational capacity limit encryption, affecting data security and device functionality [12]. Devices in harsh environments face shortened lifespans, leading to inconsistent performance. That includes the "fail-dirty" phenomenon, where failed sensors continue sending inaccurate data, poses serious reliability issues [13].

4.1.2. Communication and Network Reliability

Mobility in IoT allows nodes to change the connection topology, complicating consistent device identification across networks [14], especially when a single node may appear in different partitions of a network given a wide enough timeframe. IPv4's limited address space is inadequate for IoT, while IPv6 offers a solution but is too large for many IoT devices. The 6LoWPAN protocol compresses IPv6 headers for IoT compatibility [11]. The fragmented landscape of emerging standards complicates ensuring reliable network connections, especially for mission-critical applications [13].

4.1.3. Application Layer Reliability

The application layer's reliability depends on lower layers, as erroneous data from devices can compromise the application. Effective anomaly detection is essential, especially given the diverse data formats transmitted by heterogeneous IoT devices [15]. IoT application reliability varies based on classification methods and data preparation [3].

4.1.4. Group Reliability

Reliable group actuation by Internet of Things (IoT) standards refers to the consistent and time-bound performance of operations by nodes within a determined system. This concept emphasizes not only the correctness of individual actions at a given moment (quality), but also the sustained ability of the system to perform as intended over a specified period (reliability). In the context of IoT, reliable group actuation entails that a defined group of devices can collectively perform a requested actuation in specific constrains, such as near-synchronous operation, despite potential disruptions such as device failure, network partitioning, or environmental interference.

4.2. Group Actuation Quality of Service (GA-QoS)

Group Actuation Quality of Service (GA-QoS) refers to a service-level guarantee that an actuation request directed at a group of nodes results in successful execution, as defined by a minimum threshold of devices completing the action within an acceptable time window. Unlike traditional Quality of Service (QoS), which typically evaluates metrics related to individual device reliability or overall network throughput, GA-QoS is concerned with verifying whether the group operation achieves its intended effect. GA-QoS thereby governs whether the group actuation is considered successful, enabling performance assessments and protocol design tailored to applications where timing, coordination, and partial fault tolerance are operationally critical. The operational environments of IoT systems are frequently characterized by intermittency, network partitioning, and other disruptive factors, especially in mobile and infrastructure-constrained scenarios. In these settings, ensuring reliable actuation given a specific time interval, particularly in critical applications, requires protocols tailored to these specific needs. GA-QoS addresses these challenges by guaranteeing a minimum number of nodes that effectively perform the requested actuation operation, even in the presence of failures. This approach provides quantifiable reliability metrics for performance assessment and compliance with safety standards.

4.2.1. Formal Definition of GA-QoS

Given a set of N actuator nodes, the Group Actuation Quality of Service (GA-QoS) can be formally defined as follows:

Let:

- A be the common actuation requested to the actuator nodes.

- A_i be the action performed by actuator node i.

- $E_f(A_i, t_i)$ denote that actuation action A_i was effectively performed by node i at time instant t_i

- ΔT be the maximum allowed time interval for any action to be effected, ensuring near-synchronous operation.

- min be the minimum number of nodes required to effectively execute the action, where min $min \leq N$.

GA-QoS is satisfied if there exists a subset of min actuator nodes, denoted as $i_1, i_2, \ldots, i_{min} \subseteq 1, 2, \ldots, N$, such that:

1. For all $k \in 1, \ldots, E_f$, the action A_{i_k} is effectively performed at time t_{i_k}.

2. The time difference between the latest and earliest completion among these min nodes is within ΔT:

$$\max(t_{i_k}) - \min(t_{i_k}) \leq \Delta T \qquad (1)$$

4.2.2. Reversibility of Actuation

The reversibility of actuation refers to the system's ability to undo or counteract a previously executed group actuation, restoring the environment to its original state. This capability is critical in scenarios involving temporary interventions, anomaly recovery, or rollback operations.

If fewer than min actuator nodes successfully complete their actions within the time window $[T_o, T_o + \Delta T)$, or if an anomaly is detected, the system may initiate a reversal process. Formally, the reversal mechanism is available if:

$$\left(\sum_{i \mid Ef(A_i,T_i) \wedge T_o \leq T_i < T_o + \Delta T} 1 < k \right) \Rightarrow \text{Initiate } REf(A_y, T_r) \text{ for each } y \text{ where } Ef(A_y, T_y) \text{ holds, with } T_r > T_y > T_o \tag{2}$$

The Reversal Effectiveness Function, denoted $REf(A_y, T_r)$, is defined as:

$$REf(A_y, T_r) \equiv \text{Actuator node } y \text{ effectively performs the reversal of its part of action } A \text{ at time } T_r \tag{3}$$

This function ensures that each participating actuator node y, which previously executed its part of the original action A, can perform a corresponding reversal at a later time T_r, satisfying the condition $T_r > T_y > T_o$

4.2.3. Interaction with Environmental States

The execution of an effective action A leads to a transition in the environmental state from an initial condition S_0 to a modified condition S_1. This change reflects the intended impact of the action on the physical environment. Conversely, the effective reversal of this action, A', induces a transition of the environment back from the modified state S_1 toward the original state S_0. Ideally, a successful reversal operation restores the environment to its pre-action state, which can be formally represented as:

- Action:

$$S_0 \xrightarrow{A} S_1 \tag{4}$$

- Reversal:

$$S_1 \xrightarrow{A'} S_0 \tag{5}$$

Initially, the implemented system may have considered actions exhibiting reversibility and a near-instantaneous nature. However, it is crucial to acknowledge that not all actions align with this description, as easily illustrated by the example of pesticide application introduced in section 1. In this situation, the action of applying the pesticide has consequences that are not immediately reversible and whose effects persist for a considerable duration. Consequently, $REf(A_y, T_r)$ and the temporal constraints associated with actions, cannot always be defined with the precision assumed in the presented formal definition, given the complexity and gradual nature of their effects on the environment.

4.3. Guarantees

To address these challenges, various reliability mechanisms had to be implemented in GRAM:

4.3.1. Synchronous Response

Synchronous Response guarantees that required nodes in the network receive and process requests within a specified timeframe as the defined ΔT in section 4.2 , ensuring near-simultaneous action. For instance, in a lighting network, multiple nodes must illuminate an area within a 2-second timeframe.

4.3.2. Time-Constrained Response

Time-Constrained Response guarantees that nodes will process and respond to requests within a specific time window. For example, in a monitoring network, sensor nodes must transmit data within 4 seconds of establishing a connection.

4.3.3. Node Availability

Node Availability ensures that a specified percentage of nodes in the network will be active and capable of fulfilling a request. In a lighting network, at least 50% of nodes must be operational for a request to be successful. As the defined min in section 4.2.

5. EXPERIMENTS

To address the scenarios outlined in Section 3 and aligned with the IoT reliability definitions of Section 4, we developed a prototype Mesh network. This prototype, named "Group Reliable Actuation in Mesh Networks" (GRAM), was designed to meet the unique challenges of IoT environments by leveraging off-the-shelf hardware and existing open-source libraries. The GRAM system aims to ensure reliability guarantees such as Synchronous Response and high availability of nodes, that are critical for both Communication and Network Reliability as well as Application Layer Reliability.

To achieve these guarantees of GA-QoS, a two-phase commitment protocol was implemented using the ContextNet [16] and the Mobile Hub [17] middleware systems as basis as it provides a more efficient and balanced solution for applications with similar purposes compared to the three-phase commitment protocol, as highlighted in [18].

The Mobile Hub-2 (M-Hub2) is a software that runs on smartphones and portable devices. In conjunction with the ConetxtNet IoT backend services (ContextNet Core), any Mobile Hub-2 can act as an opportunistic locator of nearby WMNs nodes, and act as the intermediary relay of collected sensor data from the WMN to the Core, and of actuation commands from the Core services to the WMN actuator nodes (cf. Figure 2).

5.1. GRAM

GRAM provides a framework for IoT applications requiring reliable group actuation, as defined in section 4.

The project focuses on delivering Quality of Service (QoS) metrics, particularly in terms of reliability for group actuation commands in a WMN. The framework was implemented and tested as shown in figure 2.

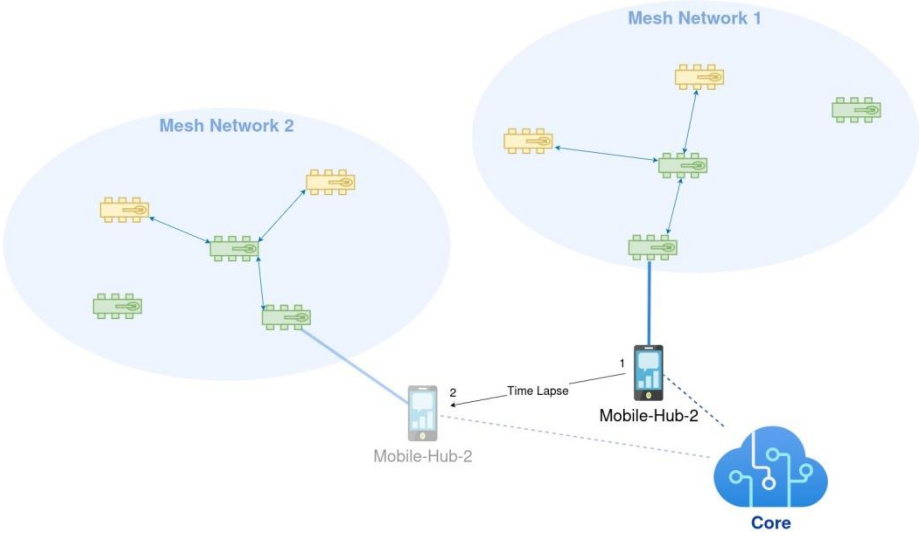

Figure 2. Example of a Mobile Hub-2 moving from one WMN to another and issuing actuation commands to the current WMN depending on the "actuation logic" defined by services executing in the ContextNet Core.(actuator nodes in yellow)

Algorithm Details:

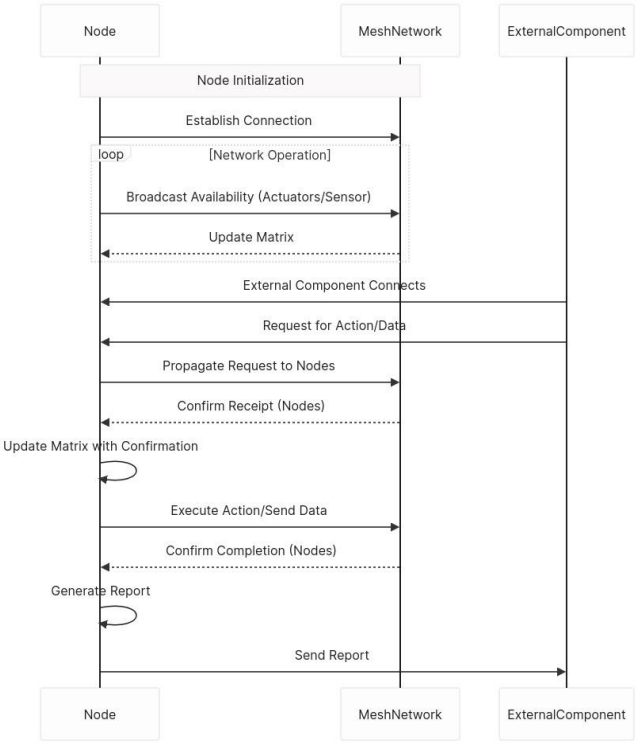

Figure 3. Framework Diagram

Nodes are classified as actuators or non-actuators before initialization. Upon startup, all nodes dynamically and locally create an matrix that is updated whenever changes occur in the WMN. When a node joins the network, all connected nodes identify themselves. Actuator nodes broadcast messages indicating their ability to perform actions, along with their unique identifiers. Upon receiving these messages, all nodes update the matrix accordingly.

When an external component, such as portable device/smartphone running the Mobile-Hub-2, connects to a node of the WMN, it can request actuation or sensor data updates through that node e.g. the connected WMN node (connNode).

To ensure reliability, the system enforces the following guarantees: node availability, time-constrained responses, and synchronous responses. If these conditions are not met, the operation is canceled and reported as a failure.

The actuation request handed down by Mobile-Hub to the connNode is propagated to the WMN nodes registered in the allocation matrix.

The available WMN nodes capable of performing the requested type of actuation confirm receipt of the request to the connNode, which then updates the matrix with this information.

If the prerequisites for the actuation execution are satisfied, the connNode disseminates the actuation request to the available nodes. The actuator or sensor nodes perform the requested operation or send the required data, subsequently confirming the completion of the task. If all or a sufficient number of nodes successfully execute the request as specified, and within the time constrain, the operation is considered successful. Otherwise, the operation is marked as failed, and a rollback request is sent to any nodes that executed the action to restore the previous state.

Finally, the middleware-connected node generates a report detailing the percentage of available nodes relative to known nodes, the percentage of confirmed nodes relative to available nodes, the time elapsed between message transmissions and responses, and whether the operation succeeded or failed. This report is transmitted to Mobile-Hub-2 for further analysis or record-keeping, as displayed in figure 3.

5.2. Metrics and Data Collection

The experimental results were created based on analyzed divergences, if any, between the video recordings, system reports, and real-time communication data, including delays. Metrics were evaluated for each scenario, considering factors such as distances, obstructions, mobility, and network topology. Key findings included the maximum delay, synchronization window, and the number of nodes needed to achieve a 100% success rate per request.

5.3. Individual Experiment Scenarios

Scenario 1: All nodes were placed on a single workbench with no interruptions.

This scenario serves as a baseline test to evaluate the system's basic functionality under ideal conditions. It allows for validation of core communication protocols, data transmission, and actuation without external disruptions, providing a reference point for performance metrics like latency and synchronization.

Scenario 2: Nodes remained on a single workbench, but one node was continuously interrupted with a 30-second interval.

This scenario simulates a controlled node failure to analyze how the system handles intermittent node disconnections. It tests the network's self-healing capabilities and evaluates the reliability of message propagation and actuation commands when a single node experiences disruptions.

Scenario 3: Two nodes were placed on a workbench, two at a passageway, and one in another room, with no interruptions.

This scenario introduces spatial separation between nodes to simulate a more realistic deployment. It tests the system's ability to maintain communication over varying distances and evaluates how the network topology adapts, when forced to create relay nodes to ensure message delivery in a multi-environment setup.

Scenario 4: Similar to Scenario 3, but one node in the passageway was continuously interrupted with a 30-second interval.

This scenario examines the network's ability to maintain functionality when a critical relay node is intermittently disconnected. By targeting a node in the passageway, the test evaluates the system's multi-hop routing capabilities and its ability to reroute messages through alternative paths.

Scenario 5: Two nodes were placed on a workbench, two at a passageway, and one in another room, with both passageway nodes continuously interrupted with a 30-second interval.

This scenario intensifies the challenge of the previous one, by simulating multiple simultaneous node failures in relay positions. It assesses the network's robustness under significant disruptions and its ability to provide fallback mechanisms for message delivery.

Scenario 6: Two nodes were placed on a workbench, two in a passageway, and one in another room. Pedestrians passed close to the passageway nodes as a source of interference.

This scenario introduces dynamic interference to evaluate the impact of real-world environmental factors on the network. It assesses the system's robustness against physical obstructions and external disruptions, testing how such interference affects communication stability and latency.

Scenario 7: One node was placed on a workbench, one at a passageway, and the remaining nodes in another room, with no interruptions.

This scenario tests the network's capacity to handle communication when a single node is responsible for relaying a larger number of messages per request and aims to evaluate the performance of multi-hop routing in a more challenging setup without disruptions, as to provide a baseline for future tests.

Scenario 8: Similar to Scenario 7, but the passageway node was continuously interrupted with a 30-second interval.

This scenario explores the impact of an essential relay node, failing. It was designed to evaluate the network's capability to dynamically reroute traffic and maintain reliable actuation despite disruptions in a key node.

Scenario 9: One node was placed on a workbench, one at a passageway, and the remaining nodes in another room, with pedestrian interference near the passageway node.

This scenario combines the single node relay situation and the dynamic interference to mimic real-world deployment challenges. It assesses the system's performance under simultaneous environmental and topological stressors, providing insights into its ability to maintain reliability in highly dynamic and unpredictable conditions.

Table 1. Scenario Configuration.

Scenario	**Nr. Nodes in the Spaces**			**Nr. Nodes Interrupted**	**Human Body Interference**
	1	**2**	**3**		
1	5	0	0	0	no
2	5	0	0	1	no
3	2	2	1	0	no
4	2	2	1	1	no
5	2	2	1	2	no
6	2	2	1	0	yes
7	1	1	3	0	no
8	1	1	3	1	no
9	1	1	3	0	yes

6. RESULTS

6.1. General Description

In all experimental scenarios, priority was given to maximizing the number of participating actuator nodes, followed by minimizing the synchronization time required for task execution. All requests during the testing period were successfully completed, adhering to the guarantees of "Node Availability" and "Synchronous Response" as defined in section 4. Video analysis confirmed that the system performed within the defined parameters outlined in section 5. When the system could not meet the requirements, it reported failures as expected. The middleware components, ContextNet and Mobile Hub-2 (M-Hub2), were not identified as bottlenecks in any scenario. This is a crucial insight as it highlights the middleware's effectiveness in "facilitating dynamic reconfiguration and asynchronous communication between heterogeneous nodes". The middleware's ability to smoothly coordinate communication between fixed Wireless Mesh Network (WMN) nodes and mobile devices (acting as gateways) is essential for maintaining system responsiveness and reliability without introducing processing or communication delays. The performance evaluation, therefore, primarily focused on the WMN's behavior, including its latency, fulfillment of GA-QoS guarantees, and resilience to interference and network partitioning.

Table 2. Latency and Node Metrics.

Metrics \ Scenario	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Average Latency in the Same Environment	64 ms	65 ms	86 ms	14 ms	75 ms	69 ms	17 ms	-	19 ms
Average Latency at 5.4m	-	-	205 ms	197 ms	217 ms	270 ms	88 ms	-	114 ms
Average Latency at 10.6m	-	-	270 ms	204 ms	301 ms	434 ms	-	-	-
Maximum Latency	261 ms	272 ms	509 ms	1217 ms	453ms	593 ms	128 ms	-	317 ms
Average Number of Participating Nodes	4	3.5	4	3.625	2.5	4	4	-	4
Minimum Number of Nodes Required	4	3	4	3	2	4	4	-	4
Synchronization Window (ms)	300	300	300	300	400	500	300	-	200

Table 2 and figures 4 and 5 present latency comparisons across distances and synchronization times, offering a clear view to be further analyzed at the breakdown of metrics for individual scenarios.

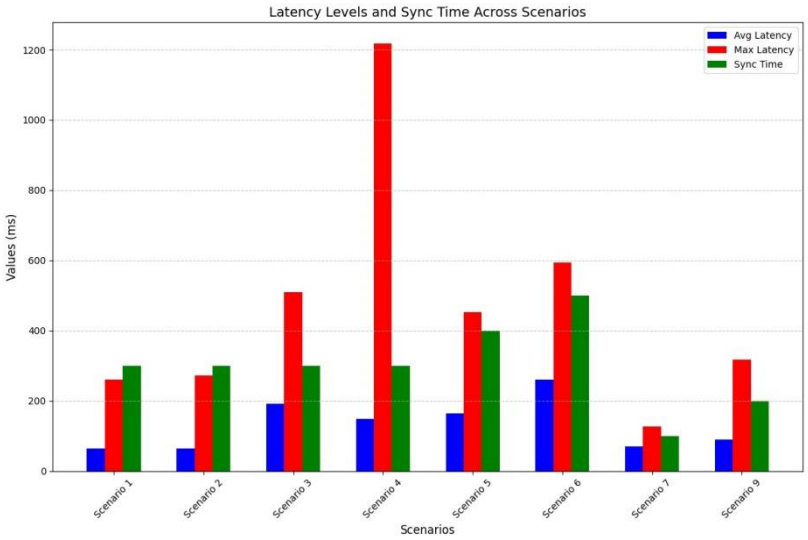

Figure 4. Latency and Sync time Comparison across Scenarios

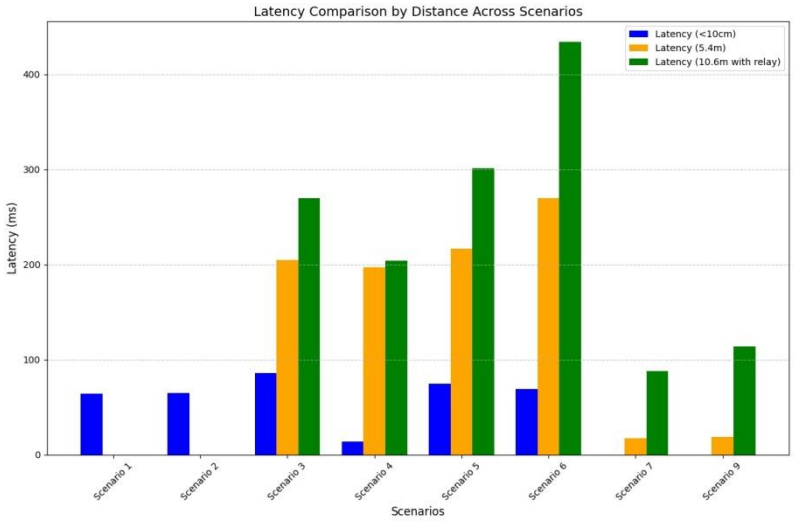

Figure 5. Latency-Ditance Comparison across Scenarios

6.2. Breakdown of Individual Scenarios

6.2.1. Scenario 1

Scenario 1 served as a baseline to validate GRAM's communication and actuation mechanisms under ideal conditions. With all nodes positioned on a single workbench and no external disruptions, the system achieved an average latency of 64 ms and maintained a synchronization window of 300 ms. The low latency is attributed to the proximity of nodes, which minimized multi-hop routing and enabled efficient direct communication via the PainlessMesh protocol implemented on ESP-based devices.

6.2.2. Scenario 2

Scenario 2 evaluated GRAM's resilience under conditions of continuous disruption affecting a single node. Despite intermittent disconnection, the system maintained a low average latency of 65 ms, while the average number of participating nodes decreased to 3.5. To preserve operational integrity, the minimum required nodes for successful actuation were adjusted to three, demonstrating GRAM's capacity to uphold GA-QoS guarantees related to node availability and fault tolerance. This adaptability was enabled by the self-healing properties of the wireless mesh network and the implementation of a two-phase commitment protocol, which ensured reliable participation and completion of requests by the remaining active nodes.

6.2.3. Scenario 3

Scenario 3 assessed GRAM's performance under spatial separation by placing nodes at distances of 5.4 and 10.6 meters, which necessitated the use of relay nodes to maintain network connectivity. While the synchronization window remained stable at 300 ms and all four required nodes successfully participated, average latency increased to 205 ms and 270 ms respectively, reflecting the impact of multi-hop communication. This scenario underscores the inherent trade-off in wireless mesh networks between extended coverage and elevated latency, as each additional relay introduces transmission and processing delays. Nonetheless, GRAM effectively sustained coordination and fulfilled operational guarantees related to node availability and synchronous response across the expanded physical range.

6.2.4. Scenario 4

Scenario 4 examined GRAM's capacity to maintain operational integrity under interference at a critical relay node, or choke point. The synchronization window remained stable at 300 ms, with the lower average of participating nodes to 3.625. These results affirm GRAM's resilience to message delays and omissions, demonstrating its ability to reroute communications and recover through alternative paths within the mesh. The robustness of the two-phase commitment protocol further ensured reliable message delivery and request completion, despite the adverse conditions and associated latency trade-offs.

6.2.5. Scenario 5

Scenario 5 evaluated GRAM's performance under intensified intermittent connectivity caused by multiple simultaneous node failures at relay positions. Despite the severity of disruptions, maximum latency remained relatively close to baseline levels at 453 ms, though average latency increased to 217 ms at 5.4 meters and 301 ms at 10.6 meters, accompanied by a broader synchronization window of 400 ms. The system was able to adapt with a reduced average number of participating nodes to 2.5 and the minimum required to 2, thereby preserving core functionality. These results highlight GRAM's capacity for graceful degradation, maintaining quorum and fulfilling operational thresholds through dynamic rerouting and flexible participation, albeit with trade-offs in latency and synchronization precision.

6.2.6. Scenario 6

Scenario 6 investigated GRAM's performance under dynamic environmental interference caused by human presence, which introduced the most substantial impact on synchronization and latency metrics. The synchronization window expanded to 500 ms, and latency reached 434 ms at 10.6 meters. Nevertheless, all four nodes successfully participated in the request, marking an punctual improvement over the previous two scenarios. This outcome underscores GRAM's resilience in maintaining node availability and synchronous response, likely achieved through compensatory retransmissions facilitated by the two-phase commitment protocol.

6.2.7. Scenario 7

Scenario 7 assessed GRAM's performance with a wider spatial distribution of nodes and a single relay, without external interruptions. Although average latency increased to 88 ms at 5.4 meters compared to the baseline, the system successfully maintained a stable synchronization window of 300 ms, with full participation from all four nodes. These results demonstrate GRAM's capacity to manage multi-hop routing efficiently, suggesting that strategic grouping of high-latency nodes and robust orchestration mechanisms can preserve synchronization precision and ensure consistent network performance despite a simple increase in transmission distances.

6.2.8. Scenario 8

Scenario 8 examined GRAM's resilience under a critical failure involving an essential relay node, resulting in a network partition that prevented request processing from one segment of the system. Consequently, only the local behavior of the connected nodes was evaluated, and detailed performance metrics were excluded due to limitations in serial data verification. Despite this constraint, the system successfully executed actions via direct connections in the unaffected environment, with performance comparable to Scenario 1. This test underscores GRAM's capacity for graceful degradation under partition conditions, while also revealing the

vulnerability posed by complete isolation by interruption in key relay nodes and the practical challenges in validating such severe disruptions.

6.2.9. Scenario 9

Scenario 9 evaluated GRAM's performance under combined challenges of a single relay node and dynamic pedestrian interference, simulating complex real-world conditions. The system maintained full node participation, with an average latency of 114 ms at 5.4 meters and a synchronization window of 200 ms. These results mirrored the adaptive behavior observed in previous scenarios, particularly Scenario 6, and demonstrated consistent performance under varying levels of environmental and topological stress. The findings reinforce GRAM's robustness, highlighting its capacity to sustain reliable operation through mesh routing, retry mechanisms, and flexible quorum thresholds defined by GA-QoS.

6.3. Failure Models and Actuation Reliability Provided by GRAM

6.3.1. Failures GRAM Can Handle (or Mitigate)

6.3.1.1. Message Omission / Dropped Messages

GRAM is designed to operate reliably despite messages being sent but not received, which often occurs due to network congestion or corruption. The system's robustness in "challenging conditions such as network partitioning and interference at relay nodes in critical choke points" demonstrates its ability to overcome dropped messages by rerouting or retransmitting, even in circumstances that the rerouting and retransmission of messages is not enough to contact all nodes, if the number of participants does not respond the request is not completed based on the implementation of the two-phase commitment protocol, Which inherently provides a mechanism to confirm receipt and eventual execution, ensuring that missing messages are detected and operations either retry or fail safely.

6.3.1.2. Network Partition

This is a core focus of the research. GRAM is explicitly designed to handle network partitions, where connectivity between parts of the network is lost. Tests included scenarios, such as Scenario 8, where a "network partition" occurred due to the interruption of a critical relay node, preventing communication between parts of the network. The Mobile Hub-2 (M-Hub2) middleware also plays a role by acting as an "opportunistic locator of nearby WMNs nodes" and relaying commands, enabling it to connect with different partitions over time.

6.3.1.3. Pause/Crash-Recover / Node Failures

GRAM is built to sustain functionality despite individual node failures. The system ensures "Minimal Node Availability," which guarantees that "a specified percentage of nodes in the network is needed to fulfill a request. Experiments, such as Scenario 2, 4, 5, and 8, explicitly simulate "continuously interrupted" nodes, demonstrating the network's "self-healing capabilities" and ability to adapt to "intermittent node disconnections". If a node fails, the system can still achieve group actuation if the "minimum number of nodes required" condition is met.

6.3.1.4. Message Delay / Latency

While "high network latencies can be interpreted as a fault", GRAM is designed to operate within and report on latency constraints. It defines "Time-Constrained Response," ensuring nodes

"process and respond to requests within a specific time window". The "Synchronization Window (ms)" is a key metric, showing the system's ability to maintain near-simultaneous actions despite varying latencies caused by distance or interference. The system ensures that if the required time constraints are not met, the operation is canceled and reported as a failure, or a rollback is initiated.

6.3.1.5. Clock Drift / Clock Skew (causing logical inconsistencies)

While GRAM aims for "near-simultaneous action" within a "Synchronization Window", it doesn't describe explicit clock synchronization protocols (e.g., NTP) to prevent clock drift or skew from accumulating, it only implements the native PainlessMesh Library solution. Even if significant clock drift were to occur, it could not impact the precise timing required for the synchronization window or the rollback mechanism's time-based conditions, as the measurements are taken in a single node, and all windows tested, did not require prolonged clock accuracy.

6.3.1.6. Message Duplication / Message Reordering

GRAM's custom application-layer protocol does handle these network layer phenomena in specific cases, as all requests have an unique ID, a single node will only update the current actuation number once it has received the ID once, further messages containing the same ID are ignored. (not solicit in the testing)

6.3.2. Failures GRAM does not directly handle

6.3.2.1. Storage Faults (Bit Rot, Latent Sector Error, Lost Write, Torn Write, Misdirected Write/Read, Storage Corruption)

The article does not address low-level storage errors within the ESP-32 nodes themselves. While "Device Reliability" is mentioned as a concern, particularly regarding "restricted memory and computational capacity" and devices in "harsh environments", the implemented protocols (like the two-phase commitment) primarily operate at the application layer and rely on the underlying hardware/firmware to manage data integrity on persistent storage. The system doesn't specify mechanisms to detect or recover from silent data corruption on disk or flash memory. But it does in fact maintains the current state of the network on all nodes partaking in the system. Once a node has failed, but reconnected to the WMN it will be on the same request number as the rest, and will receive/map the entire network once again, as observed in Scenarios 2, 4 and 5.

6.3.2.2. Memory Corruption

Similar to storage faults, memory corruption (where data read from memory differs from what was written) is a hardware or low-level software issue that is not explicitly addressed by GRAM's application-layer protocols. The assumption is that the underlying ESP-32 hardware and its SDK handle these issues at a lower level. But has the same workaround.

6.3.3. Failures GRAM Does Not Handle

- "Fail-Dirty" Phenomenon (Incorrect Actuation): Although not explicitly named "fail-dirty" in the context of handling, the GA-QoS definition implicitly addresses the consequences of incorrect or incomplete actuation.
- Byzantine Fault: The current work does not describe mechanisms to handle Byzantine faults, where a node behaves maliciously, sending incorrect or deceptive messages.

7. CONCLUSION

This study presented a comprehensive analysis of reliability in IoT networks utilizing Wireless Mesh Networks (WMN), with a focus on practical scenarios involving actuator nodes, group actuation, mobility, network partitioning, and node failures. Our findings emphasized the significant impact of factors such as network topology and environmental conditions on system reliability. Although various strategies, such as fault-tolerant mechanisms and adaptive algorithms, have been proposed in related works, this research tailored a novel solution for rural or remote environments with limited infrastructure, offering guarantees for user-driven action requests. The practical tests validated the proposed system modifications, demonstrating its capacity to ensure reliable communication and actuation under challenging conditions. These experiments highlighted the strengths of the prototype, particularly in adapting to network disruptions and maintaining performance during real-world constraints. Physical constraints limited scalability testing, therefore scalability was tested only via network partitioning simulation. A node was isolated to test system operation under partition. Partitioning test indicates consistent operation across WMNs with increased nodes. Subgroup division requires increased M-Hub2 visits for data collection, and actuation requests.

However, the study's limitations must be acknowledged. The hardware used for validation, including a 30-fps camera and redundant data collection methods (e.g., serial communication and system reports of the nodes themselves), constrained the precision of certain measurements. Additionally, testing was restricted to a specific WMN implementation, keeping out of reach the broader applicability of other network technologies, such as Bluetooth and LoRa. Further testing will be necessary to evaluate energy consumption across various communication technologies to strengthen the generalizability of the results.

Future research should focus on expanding the scope of this work by comparing the proposed prototype with commonly used communication protocols under similar conditions. Testing additional technologies like Bluetooth Mesh and LoRa will enable a deeper understanding of reliability in diverse scenarios. Furthermore, real-world deployments would be a prime objective to be pursued in order to refine the solution and validate its performance in practical applications.

REFERENCES

[1] Daniel Aikhuele, Herold Nwosu, and Desmond Ighravwe. Data-driven model for the evaluation of the reliability of sensors and actuators used in iot system architecture. Journal of Reliable Intelligent Environments, 9:135–145, 2023.

[2] Sato H., Kanai A., Tanimoto S., and Kobayashi T. Establishing trust in the emerging era of iot. Sympo-sium on Service-Oriented System Engineering, page 398–406, 2016.

[3] Samuel J Moore, Chris D Nugent, Shuai Zhang, and Ian Cleland. Iot reliability: a review leading to 5 key research directions. CCF Transactions on Pervasive Computing and Interaction, 2:147–163, 2020.

[4] X. You, P. Zhang, S. Li, F. Wang, G. Su, and S. Zhang. Fahp-based reliability evaluation of distributed iot devices in a distribution power grid. Wireless Communications and Mobile Computing, 2022:1–11, 2022.

[5] Media Department for Digital, Culture, Sport (DCMS), and National Cyber Security Centre (NCSC). Code of practice for consumer iot security, 2018. Accessed: 2024-09-09.

[6] Venkata P. Modekurthy, Abusayeed Saifullah, and Sanjay Madria. A distributed real-time scheduling system for industrial wireless networks. ACM Trans. Embed. Comput. Syst., 2021.

[7] S. Harihara Gopalan, V. Vignesh, D. Udaya Suriya Rajkumar, A.K. Velmurugan, D. Deepa, and R. Dhanapal. Fuzzified swarm intelligence framework using fpsor algorithm for high-speed manet - internet of things (iot). Measurement: Sensors, 2024.

[8] E. Munoz-Abad; R. Suquinagua-Otavalo; F. Astudillo-Salinas; L. I. Minchala; A. Vazquez-Rodas. Home automation architecture: design and implementation using esp8266. 2020 7th International Conference on Control, Decision and Information Technologies (CoDIT), 2020.

[9] F. Hoo; F. S. Lim Tan; R. Ching Bon Chan; P. Waszecki; S. L. Keoh; C. Kiat Seow; M. Li; Q. Cao; C. S. Sum. 5g-wi-sun for building management system. 2023 6th International Conference on Applied Computational Intelligence in Information Systems (ACIIS), 2023.

[10] Gangyong Jia, Yujie Zhu, Youhuizi Li, Zongwei Zhu, and Li Zhou. Analysis of the Effect of the Reliability of the NB-IoT Network on the Intelligent System. IEEE Access, 2019.

[11] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. IEEE Communications Surveys & Tutorials, 17(4):2347–2376, 2015.

[12] A. Rayes and S. Salam. Internet of Things: From Hype to Reality: The Road to Digitization. Springer, Cham, 1st edition, 2016.

[13] A. Karkouch, H. Mousannif, H. Al Moatassime, and T. Noel. Data quality in internet of things: A state-of-the-art survey. Journal of Network and Computer Applications, 73:57–81, 2016.

[14] J. Rafferty, J. Synnott, C. D. Nugent, A. Ennis, P. A. Catherwood, I. McChesney, I. Cleland, and S. McClean. A scalable, research oriented, generic, sensor data platform. IEEE Access, 6:45473–45484, 2018.

[15] A. Abeshu and N. Chilamkurti. Deep learning: The frontier for distributed attack detection in fog-to-things computing. IEEE Communications Magazine, 56(2):169–175, 2018.

[16] Markus Endler and Francisco Silva e Silva. Past, present and future of the contextnet iomt middleware. Open Journal of Internet Of Things (OJIOT), 4(1):7–23, 2018. Special Issue: Proceedings of the International Workshop on Very Large Internet of Things (VLIoT 2018) in conjunction with the VLDB 2018 Conference in Rio de Janeiro, Brazil.

[17] Tatiana Reimer, Luis E. Talavera R´ıos, Millena Cavalcanti, La´ercio Lucchesi, and Markus Endler. Design and implementation of a flexible mobile edge middleware for multi-protocol wireless connectivity. In 2024 IEEE Conference on Pervasive and Intelligent Computing (PI-Com), pages 40–46, 2024.

[18] D. K. Gautam G. U. Mali. A model application of two phase commit protocol in wireless dtn. International Journal of Computer Applications, 162(5):23–28, Mar 2017.

AUTHORS

Markus Endler is Doctor rer. nat. from the Technical University of Berlin (1992) and M.Sc. from PUC-Rio, both in Computer Science. In 2001 he became Professor Livre-docente (Habilitation) from the University of São Paulo (2001). Before joining the Department of Informatics of PUC-Rio in 2001 where he is currently Associate Professor III, he was Assistant Professor at the Institute of Mathematics and Statistics of the University of São Paulo (USP). His main research interests include Mobile and Pervasive Computing, Internet of Things, Middleware systems, Distributed Algorithms, flying networks (i.e. swams of drones) and Data Stream Processing, He has supervised 17 Doctoral and 41 (strictu sensu) M.Sc. dissertations, and 36 undergraduate final course projects. He has published more than 220 scientific papers in journals and refereed conferences and 12 book chapters and 2 books. He participates in several Program Committees of international and Brazilian Conferences.

Victor de Barros Barreto Pamplona Cortes - currently a Master's student at the Department of Informatics, PUC-Rio. Prior to his current academic pursuit, he completed his Bachelor of Science in Electrical Engineering at PUC-Rio(2022). In his professional capacity, he is an Electrical Engineer at Aurea Medic in Rio de Janeiro, where he has been actively involved in the development of IoT projects. His journey has encompassed a range of roles, including Engineering Intern at Aurea Medic, where he worked on developing new products and services(2021), as well as Scientific Initiations at PUC-Rio, where he engaged in the creation of educational objects(2020) and implementing IoT systems in the medical field(2021).