UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

GUILHERME ROTTH ZIBETTI

# Improving Network Resilience through an Environmental Context-aware System

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Advisor: Prof. Dr. Juliano Araújo Wickboldt

Porto Alegre
September 2023

**CIP — CATALOGING-IN-PUBLICATION**

Zibetti, Guilherme Rotth

Improving Network Resilience through an Environmental Context-aware System / Guilherme Rotth Zibetti. – Porto Alegre: PPGC da UFRGS, 2023.

81 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2023. Advisor: Juliano Araújo Wickboldt.

1. Context-aware. 2. IoT. 3. LPWAN. 4. Tactical networks. I. Wickboldt, Juliano Araújo. II. Título.

*“What to do to philosophize? To throw away self-concept.*
*Is impossible for a man to begin to learn what he thinks he knows.”*

— EPICTETUS

## ACKNOWLEDGEMENTS

Firstly, I would like to thank my advisor, who guided me in this project we call a Master's degree. Teaching me how to think about a scientific problem, how to develop solutions to the problem, and finally, how to discuss the results of this problem. I would also like to thank Professor Luciano Gaspary, who, in addition to having taught me a lot in the disciplines as a special student, introduced me to an advisor who knew how to meet my needs well in the scientific field. Finally, I would like to thank Professor Marcio Pohlmann, who guided me from my undergraduate until the ingress of my master's degree.

In addition to the people who contributed to my intellectual development in the research area, I would also like to thank my friends, colleagues at the UFRGS Institute of Informatics, and co-workers who gave me all the necessary support during the master's degree.

I want to thank my family, especially my parents and my future wife, who, in addition to sponsoring my study hours for the master's degree, gave me the necessary emotional support to complete this work. Finally, I would like to thank my dog, Bento, who made me stop studying for a few minutes to rest my mind playing ball with him.

## ABSTRACT

Networks are a crucial resource in the digital age, enabling applications in domains such as agriculture, smart cities, and military systems. As technology advances, some applications require a more reliable and predictable communication channel. This is especially important for real-time applications like industrial automation, smart cities, healthcare, and military systems. However, despite technological advancements and the growing significance of networks in society, it is known that current levels of resilience, predictability, and survivability need improvement. Towards improving the communication network, this work presents an context-aware environmental monitoring system that uses real-time context information to increase networks' resilience and survivability. A proof-of-concept based on simulation and experimentation validates the proposed approach. The results show that implementing this system can improve communication even when the network is exposed to unfavorable climatic factors.

**Keywords:** Context-aware. IoT. LPWAN. Tactical networks.

**Melhorando a Resiliência da Rede através de um Sistema com Reconhecimento de Contexto Ambiental**

## RESUMO

As redes são um recurso crucial na era digital, possibilitando aplicações em domínios como agricultura, cidades inteligentes e sistemas militares. Conforme a tecnologia avança, algumas aplicações requerem um canal de comunicação mais confiável e previsível. Isso é especialmente importante para aplicações em tempo real, como automação industrial, cidades inteligentes, saúde e sistemas militares. No entanto, apesar dos avanços tecnológicos e da crescente importância das redes na sociedade, é sabido que os níveis atuais de resiliência, previsibilidade e capacidade de sobrevivência precisam ser aprimorados. Visando melhorar a rede de comunicação, este trabalho apresenta um sistema de monitoramento ambiental consciente do contexto, que utiliza informações de contexto em tempo real para aumentar a resiliência e a capacidade de sobrevivência das redes. Uma prova de conceito baseada em simulação e experimentação valida a abordagem proposta. Os resultados mostram que a implementação desse sistema pode melhorar a comunicação mesmo quando a rede está exposta a fatores climáticos desfavoráveis.

**Palavras-chave:** *Context-aware*. IoT. LPWAN. Redes Táticas..

# LIST OF FIGURES

## LIST OF TABLES

## LIST OF ABBREVIATIONS AND ACRONYMS

| | |
|---|---|
| ACK | Acknowledgment |
| ADR | Adaptive Data Rate |
| AS | Application Server |
| CayenneLPP | Cayenne Low Power Payload |
| CSS | Chirp Spread Spectrum |
| C4ISR | Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance |
| CZ | Control Zones |
| CC | Communication Center |
| DP | Data Processing |
| DR | Data Rate |
| dBi | Decibels Relative to Isotropic |
| DDoS | Distributed Denial of Service Attacks |
| ECAS | Environment Context-Aware System |
| FOMS | Field Object Monitoring Sensor |
| fCnt | Frame Counter |
| ICN | Information Centric Networking |
| IoT | Internet of Things |
| IoBT | Internet of Battle Things |
| LQI | Link Quality Indicator |
| LPWAN | Low-Power Wide-Area Networks |
| LoRa | Long Range |
| ML | Machine Learning |
| MSE | Mean Square Error |
| MCC | Mobile Communication Center |

| | |
|---|---|
| MS | Monitoring Sensor |
| NS | Network Server |
| PDR | Packet Delivered Rate |
| POV | Point of View |
| RFR | Random Forest Regression |
| RSS | Received Signal Strength |
| RSSI | Received Signal Strength Indicator |
| RP | Reverse-polarity |
| SN | Sensor Network |
| SNR | Signal to Noise Ratio |
| SCC | Static Communication Center |
| SD | Standard Deviation |
| SDN | Software Defined Networks |
| SF | Spreading Factor |
| SMA | SubMiniature version A |
| TEN | Tactical Edge Network |

# CONTENTS

# 1 INTRODUCTION

Networks have played an essential role in several areas, being considered a fundamental resource in the current digital age(CROWCROFT; WOLISZ; SATHIASEELAN, 2015). Besides, networks come as an enabler in the development of applications in several fields such as agriculture (DOBRESCU; MEREZEANU; MOCANU, 2019), smart cities (KAMIENSKI et al., 2017), and Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) systems (ZHENG; CARTER, 2015). Thus, the Internet has become a critical infrastructure that almost every aspect of our lives depends on (FOUNDATIONS, 1997).

As network technologies advance, various applications with different requirements emerge. Some applications can tolerate significant network instability and latency, while others need more reliable and predictable communication channels. For instance, in industrial process automation and control (SURIYACHAI; BROWN; ROEDIG, 2010), smart cities (KAMIENSKI et al., 2017), healthcare (AZZAWI; HASSAN; BAKAR, 2016), and mission-critical military systems (ZACARIAS et al., 2017), there are applications with real-time requirements (i.e., network behavior must be predictable and deterministic). Hence, these applications require a resilient and reliable network to support them. However, despite its technological advances and the growing importance of networks in society, it is known that current levels of resilience, predictability, and survivability need improvement (STERBENZ et al., 2013).

In order to create more resilient networks, some strategies and frameworks developed for building resilient networks discuss the factors that affect the correct behavior of networked systems, and some of them are network external factors (STERBENZ et al., 2010; CETINKAYA; STERBENZ, 2013). High temperatures (BOANO; CATTANI; RÖMER, 2018), rain (BOANO et al., 2009), and vegetation (MARFIEVICI et al., 2013) are factors that can directly affect the performance and availability of communications. Exposure of communication devices to climatic and environmental factors is usual in many scenarios, such as the Internet of Things (IoT) and military networks. Therefore, monitoring context information, which affects communication performance and availability, helps explain and anticipate network behavior.

This proposal suggests a context-aware monitoring system to improve network resiliency, survivability, and efficiency. It is motivated by military tactical networks, which already have a network of sensors for surveillance, reconnaissance, and logistics. The system can rely on these sensors or suggest installing a few more on battlefield objects to collect relevant information, allowing the prediction of network performance degradation or link unavailability. The main contributions of this work are the following:

1. The proposal of a context-aware monitoring system that – unlike current proposals found in the literature – uses information collected from the physical environment to improve overall communication efficiency in terms of packet delivered rate (PDR) and power consumption;

2. In order to perform the proposed system evaluation was necessary to collect a dataset to give as input in the prediction model. The collected dataset provides relevant data to support other works that aim to develop an attenuation model for Low-Power Wide-Area Networks (LPWAN) technologies and create prediction models with machine learning techniques.

3. The proposed system was designed to be incorporated into military tactical networks with minimal impact on the existing infrastructure and communication architecture. Lastly, the proposed system is technology agnostic, nevertheless insights on the implementation are provided based on a state-of-the-art open source IoT framework and related technologies.

The remainder of this work is organized as follows. **Chapter 2** reviews related studies in the field of resilience and survivability of networks, environmental impact on wireless communication technologies, and systems that use the concept of context-awareness to adapt to the context. **Chapter 3** presents a battlefield scenario and the proposed solution architecture. **Chapter 4** presents a proof-of-concept experiment of the proposed solution on a simulated environment and a partial conclusion of the obtained results. **Chapter 5** presents a system prototype and a proof-of-concept experiment of the prototype on a real environment, followed by the **Chapter 6** that presents the prototype evaluation results. Finally, **Chapter 7** presents conclusions and future work.

# 2 RELATED WORK

Constructing a resilient network has been a longstanding topic of discourse, with numerous studies offering insights into effective approaches for tackling this challenge. Diverse threats manifest within both the logical and the physical environment. As such, formulating strategies that comprehensively target both domains becomes critical.

Aiming at a resilient network able to address distributed denial of service attacks (DDoS), (XIE et al., 2009) proposed a mechanism that uses the application and network-level information to detect anomalies in the network behavior and mitigate possible threats. In more recent work, (GONZÁLEZ et al., 2021) too proposed an approach to address DDoS problems; however, different from (XIE et al., 2009), in this work, the authors proposed a mechanism that uses the programmable data plane to identify and mitigate those attacks. The programmable data plane is part of the Software Defined Networks (SDN) paradigm and has been widely adopted to address network security problems(DONG; ABBAS; JAIN, 2019). Some similarities of the works above are the use of context information, the threats addressed, which is DDoS, and the context of the threats, the logical context.

In the work presented by (XIE et al., 2014), the authors proposed a system that aims to enhance the network's resilience in natural disasters, power outages, and other threats to the communication network. The authors use the SDN approach to create a backup topology for disaster cases. Similar to the mechanism proposed by (GONZÁLEZ et al., 2021), this system also uses the SDN paradigm to improve network resilience. However, the threats treated in (GONZÁLEZ et al., 2021) work come from the own network, while the work proposed by (XIE et al., 2014) includes threats from external environments to the network. This factor could be crucial to reach a more resilient network.

Creating frameworks and guides with best practices is present in several research areas; it is no different in the context of resilient networks. Towards resilient networks, (STERBENZ et al., 2013) proposed a framework consisting of strategies, metrics, and techniques to evaluate and quantify the resilience of an existing network and a new proposed network. In this framework, one of the prerequisites is context-awareness, which refers to the system's ability to collect and process context information to determine its behavior (SCHILIT; ADAMS; WANT, 1994). The authors state that, for a given network to be resilient, it is necessary to monitor the channel's conditions, the link state, and other events that may impact its correct functioning.

Besides gathering and processing context information, many approaches rely on adjusting configuration parameters to improve communication. Several studies suggest exploiting the network configuration parameters to increase signal strength, and the most commonly used parameter is transmission power control using different approaches. The first approach is at the network level (PARK; SIVAKUMAR, 2002; NARAYANASWAMY et al., 2002), which is based on applying the same transmission power to all nodes in the network. The second approach is at the node level (KUBISCH et al., 2003) by applying different transmission power to each node in the network. The third approach is based on neighborhood relation (XUE; KUMAR, 2004), in which nodes use a different transmission power for each neighbor. A fourth approach can apply different transmission power at a packet level (LIN et al., 2016). A common limitation among all those approaches is that the network configuration parameters were adjusted using only the information collected from the network itself, such as the Received Signal Strength Indicator (RSSI) and Link Quality Indicator (LQI). While this information is helpful, currently, other factors can be considered to determine network configuration parameters. In this work, we aim to explore additional information collected from the physical environment where the nodes are installed in addition to the commonly used quality indicators.

The work developed by (BOANO; CATTANI; RÖMER, 2018) presents the impact of temperature variation, a possible climatic factor, on Long Range (LoRa) communication links. The results show that higher temperatures compromise the network operation and, in extreme cases, cause total link rupture. (LUOMALA; HAKALA, 2015) presents the climatic effects of temperature and humidity on communication links using ZigBee wireless technology. Like on the LoRa technology, the results show that temperature significantly negatively impacts signal strength on ZigBee technology. The authors used the RSSI as a metric, showing its oscillation according to the climatic variations observed from the external environment during the experiments. In both studies by (BOANO; CATTANI; RÖMER, 2018) and by (LUOMALA; HAKALA, 2015), modifications in configuration parameters were explored to assert the resilience and survivability of these networks when exposed to climatic factors. These studies help to emphasize how critical the monitoring of external factors is to increase network resiliency, although they are quite theoretical and do not integrate into an actual network framework or solution.

Towards designing context-awareness systems that adapt to external environmental factors, (DOBRESCU; MEREZEANU; MOCANU, 2019) suggests implementing a context-aware system for precision agriculture environments. The authors' proposal aims to integrate three emerging technologies (IoT, cloud computing, and context-awareness) to provide an irrigation system and monitor parameters related to soil properties. In this system, the environmental changes collected by the sensors determine how the system will manage water, nutrients, and pesticides. In another work, developed by (KAMIENSKI et al., 2017), the authors propose using IoT-related technologies to improve the process of energy consumption in public buildings and universities. In this work, the authors suggest using presence sensors and smart plugs to control room lights and monitor the energy consumption of buildings. These are examples of context-aware systems that use information collected from the surrounding physical environment to determine their behavior. Although both works use information gathered from the context, neither uses it to determine its functioning, but rather the functioning of their application.

Another example that uses information collected from the context to define what action to take is the work proposed by (RAK, 2016). The author proposes adapting the topology of a wireless mesh network using the information on predicted attenuation of links based on radar measurements. More specifically, the author uses real echo rain maps to calculate attenuation in communication links and modify the network topology to improve network throughput during rainstorm periods. Differently from (DOBRESCU; MEREZEANU; MOCANU, 2019) context-aware precision agriculture system and the intelligent buildings of (KAMIENSKI et al., 2017), in (RAK, 2016) proposal, a third-party system provides the weather maps that serve as context information. In contrast, the other systems collected context information through sensors to decide the system's action.

In the context of critical networks, it is possible to highlight three works implementing context-awareness to tackle different issues. The first work developed by (PAPAKOSTAS et al., 2018) proposes an algorithm for building a backbone in military networks with energy awareness to deal with power efficiency in constructing the network topology. In the second work, (POULARAKIS et al., 2019) developed a system that uses the SDN paradigm to address changes in the network topology from nodes distributed across the battlefield. Lastly, (MISHRA et al., 2017) proposes a context-oriented proactive decision support framework to accelerate decision-making. The developed system uses data related to the mission, environment, assets, threats, time, workload, and other factors related to the decision-making process. Based on the acquired data, the system suggests to the decision-maker some possible decisions that could be taken to solve a given problem. In addition to the works already mentioned, other studies also address using context-aware applications to support decision-making on the battlefield (LIN et al., 2019; CASTIGLIONE et al., 2017; MOON; JUNG; JEONG, 2010). However, as in the works developed by (DOBRESCU; MEREZEANU; MOCANU, 2019), and (KAMIENSKI et al., 2017), the studies mentioned above use context data to determine the application behavior. A step ahead of works in this direction is presented in (LEAL et al., 2019), which gives an approach that combines SDN and Information-Centric Networking (ICN) to support context-awareness in IoBT systems.

The difference between the proposal presented in this work and the works mentioned above is that, although some used information collected from the physical environment, this information determined the application's behavior rather than the network's. In contrast, in work developed by (RAK, 2016), the information that determines the system's behavior was not collected directly from the system context but provided by a third-party system. Relying only on an external entity to provide information relevant to the system's context can cause problems because it may not meet the time constraints of critical network environments. Among critical network studies, it is possible to highlight the works developed by (PAPAKOSTAS et al., 2018) and (POULARAKIS et al., 2019), which propose solutions for context-aware networking. However, none of these works consider the effect of environmental factors on networks. Other works, like the developed by (MISHRA et al., 2017) and (LEAL et al., 2019) in the critical networks' context, consider context-awareness applications to assist in the decision-making process of Command and Control (C2) systems, leaving out of scope the network management itself.

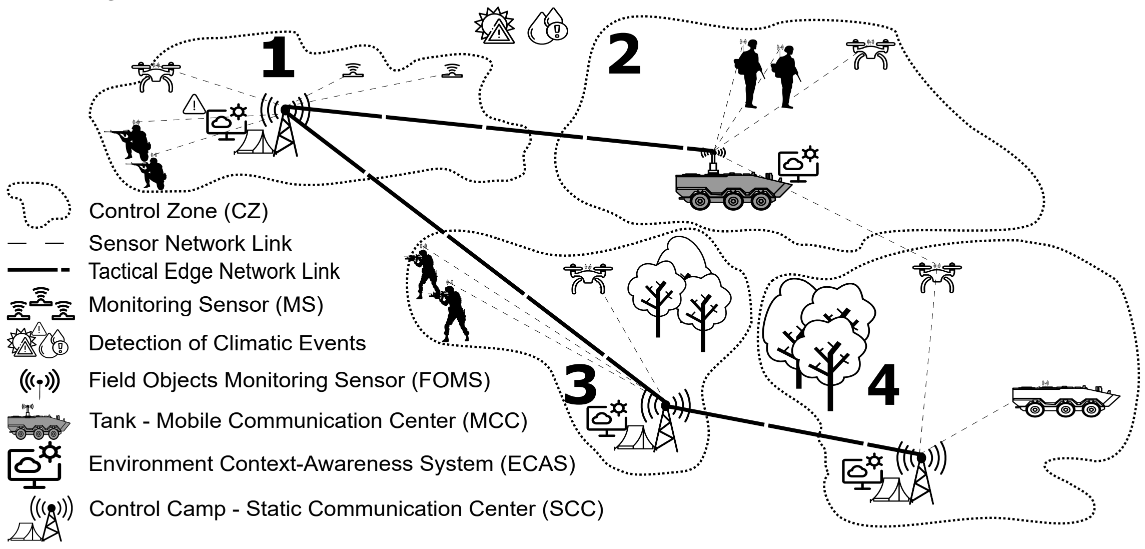
# 3 SYSTEM DESIGN

This chapter is organized into two sections. First, **Section** 3.1 presents a battlefield scenario, a critical networks' scenario example, with the necessary elements to perform a proof-of-concept. Then, **Section** 3.2 describes the proposed system and its architecture.

## 3.1 Application Scenario

The battlefield scenario is an example of critical networks where the resources are constrained, and the applications present real-time requirements. Thus, to support these applications, it is necessary to implement techniques that increase the reliability and resilience of networks. **Figure 3.1** presents an example of this kind of scenario. The dotted line in **Figure 3.1** delimits the Control Zones (CZ), identified by numbers from one to four. Each CZ has a Communication Center (CC), which can be Static (SCC) or Mobile (MCC), represented by control camps and a combat vehicle with a high-power antenna attached to the top (located at CZ-2). The connections between the control camps and the CZ-2 combat vehicle represent a Tactical Edge Network (TEN).

Figure 3.1: Battlefield scenario. (ZIBETTI; WICKBOLDT; FREITAS, 2022).


1
2
3
4
Control Zone (CZ)
Sensor Network Link
Tactical Edge Network Link
Monitoring Sensor (MS)
Detection of Climatic Events
Field Objects Monitoring Sensor (FOMS)
Tank - Mobile Communication Center (MCC)
Environment Context-Awareness System (ECAS)
Control Camp - Static Communication Center (SCC)

Source: Author

In addition to the TEN, the battlefield has a Sensor Network (SN). This network is responsible for transmitting the data from the sensors spread across the CZs to the Environment Context-Aware System (ECAS) servers in the CC of each CZ. The data traffic of the TEN has no relation to the data traffic carried out by the SN. The sensors distributed on the battlefield can communicate with any CC within range to deliver data collected from the battlefield. For instance, it is possible to observe this type of communication in CZ-4, whose flying drone is connected with the CCs of zones two and four. Therefore, it can deliver the collected data to both CCs.

The battlefield has two types of sensors distributed: the Monitoring Sensor (MS) and the Field Object Monitoring Sensor (FOMS). MSs are distributed on the battlefield, collecting information from a specific location, whereas the objects moving across the battlefield carry attached one FOMS. Both sensors can collect temperature and humidity information. However, FOMSs can collect additional information, such as vital signs, location, weaponry, and other necessary information, from the objects to which they are attached.

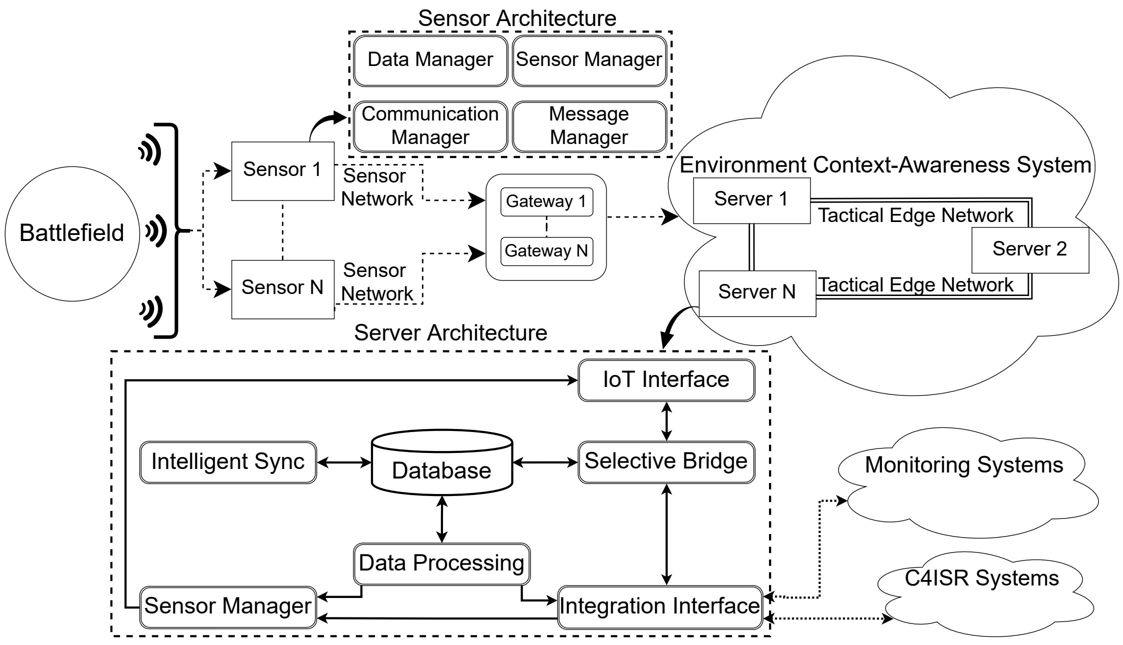
The ability of MSs and FOMSs to communicate with any server makes ECAS a distributed system. Besides receiving information from multiple MS and FOMS, the ECAS servers distributed across the battlefield also share the information collected in their respective CZ through the TEN. Sharing the information collected in each CZ increases the coverage area of the ECAS. Therefore, this information can support decision-making and mitigate problems in any CZs with a CC.

In **Figure 3.1**, two icons at the top exemplify the system's operation, representing the detection of climatic events related to temperature and humidity. The sensor closest to these events is the MS located in the upper right corner of the CZ-1. This sensor will notify the ECAS server of its respective CZ that it will process the received information and perform the necessary actions in response to these events. After processing the received data, the ECAS server of the CZ-1 will share the processed data with the other ECAS servers, which may need the data. After receiving the data, the other ECAS servers will process the information to verify whether to act in their respective CZ. Thus, the environment covered by the ECAS can adapt to the climatic conditions using as inputs the data collected from the battlefield through the MS and FOMS sensors.

## 3.2 Proposed System Architecture

**Figure 3.2** illustrates the system architecture. The circle, named battlefield, refers to the monitored environment. The sensors (1 to Sensor N) represent the MSs and FOMSs distributed across the battlefield. The box with Gateway 1 and Gateway N represents the Gateways distributed by the battlefield, being a Gateway for each CZ. As it is a system that acts in a distributed way, it is necessary to have a Gateway for each ECAS server distributed by the battlefield. The dashed box on the top of the figure highlights the internal components of each sensor (i.e., the sensor architecture).

Figure 3.2: System Architecture. (ZIBETTI; WICKBOLDT; FREITAS, 2022).



Source: Author

ECAS is composed of servers that act independently and collaboratively. The distribution of these servers across the battlefield is in the same number as the existing CZ for a given military operation. Each CZ has a Gateway and an ECAS server to handle locally collected data. The collaborative network of servers is represented in **Figure 3.2** within the cloud shape, indicating that the servers' communication flows over the TEN. In the bottom part of **Figure 3.2** there is a highlight with the internal components of each server (i.e., the server architecture).

The following flow describes how the data collected on the battlefield is transmitted, processed, and shared. The SN is responsible for transmitting the data collected by the sensors on the battlefield to the ECAS servers distributed across the CZs. When receiving the collected data, an ECAS server must process and share it with the other servers that comprise the system. This sharing takes place through the TEN. Therefore, the SN serves exclusively for sensory data traffic and reconfiguration of sensors distributed across the battlefield. The modules that compose the ECAS servers and sensors are detailed in the following sub-sessions.

### 3.2.1 Sensor architecture

In **Figure 3.2**, the dashed box indicated by an arrow on Sensor 1 represents the internal architecture of each sensor. Each sensor has four modules that determine its operation. The modules are Sensor Manager, Message Manager, Communication Manager, and Data Manager.

The Sensor Manager module is responsible for overseeing the operating mode of the sensor as a whole. Each MS and FOMS sensor can read environment data (e.g., temperature, humidity, location, vital signs) and send this data over the SN to a gateway. The Sensor Manager module manages the process of collecting, storing, sending, and receiving data, which takes place through the components of each sensor. This module manages the frequency of data collection and local storage, as well as the frequency of transmission (in bulk). This process follows two basic criteria, which are defined as Periodic and Threshold-based. Periodic criteria mean configurable time intervals will guide the collection, local storage, and transmission of measured data. For example, under normal conditions, temperature and humidity could be measured every few seconds but transmitted only once per minute to save resources. Threshold-based criterion intends to reduce the frequency of data collection and transmission under unsafe operating conditions. For instance, when the measured temperature reaches a harmful threshold, it can be set for immediate transmission instead of waiting until the next periodic cycle.

The Message Manager module manages the message format sent to the ECAS server. Two types of messages were created: Periodic messages and Trigger messages. Periodic messages are related to the Periodic criteria, while Trigger messages are related to the Threshold-based, both defined by the Sensor Manager module. Suppose the data for a region monitored by a particular MS/FOMS sensor is outside safe operational limits. In this case, this sensor should classify the data contained in the message as urgent and send it immediately. On the other hand, if the data is within safe operational limits, the sensor can store it locally and wait for an opportune moment to send it, as determined by the periodic criterion. Differentiating messages is a way to prioritize the communication and processing resources of the system for the most critical messages.

Defined the frequency of data collection and transmission and the types of messages, it remains to specify how to present the data to the application. The Data Manager module performs this function. ECAS aims to increase the resilience of networks as well as support the applications of C4ISR systems. Therefore, the sensor that collects and sends relevant data to ECAS does the same for C4ISR systems. The Periodic criterion determines that all sensors present in an MS/FOMS read and send the collected data to the ECAS server at a specific time. This collection criterion can be applied when the sensor is in a region where operational limits are safe; however, this is not always true. When a sensor is in a region whose collected data is outside a safe operating threshold, the messages must contain only relevant data (i.e., data exceeding the threshold). This approach reduces the overhead of the collection and transmission process by prioritizing processing and communication resources for the most relevant data. For example, a sensor in a specific region detects that the local humidity and temperature exceed the specified threshold. The Data Manager module gathers the collected data, assembles a package with the relevant data for this specific case (e.g., sensor id, collection timestamp, temperature, humidity, GPS location), and delivers this package to the Message Manager module. The Message Manager module classifies the message as Trigger and delivers it to the Communication Manager module, which immediately sends it over the SN to the ECAS server.

The Communication Manager module manages the communication resources of MS and FOMS. The Sensor Manager module specifies the criteria for collecting and sending data to the ECAS server. Therefore, data collection and transmission work asynchronously, allowing the Communication Manager to set communication resources into stand-by or power-saving mode when unnecessary. The Message Manager module defines two types of messages: Periodic and Trigger and each type has specific requirements for using communication resources. Periodic messages should be delivered to the ECAS server as quickly as possible; however, if necessary, they can wait longer to access the medium as the delay will not significantly affect the system. On the other hand, a Trigger type message contains sensitive data (i.e., data directly related to the good system functioning); therefore, these messages must take priority above others in accessing the medium and guaranteeing the earliest delivery. Thus, the Communication Manager module manages the communication resources to fulfill the message's requirements to be transmitted.

### 3.2.2 Server architecture

Represented in the dashed box pointed out by Server N in **Figure 3.2** is the ECAS server's architecture. Each ECAS server has the same architecture: six processing modules and one database.

The first module of the ECAS server architecture to be discussed is the IoT Interface. IoT communication technologies provide specific protocols to meet the needs of these networks. This module is responsible for interpreting the data received from the sensors and translating them into a format consumed by ECAS and other systems that use this data. In addition to interpreting and translating sensor data, this module does the reverse process for messages sent from servers to sensors. Therefore, the IoT Interface module is the communication interface between IoT technologies and ECAS servers. The function performed by this module simplifies the implementation of heterogeneous IoT technologies on the sensor side, as it centralizes the communication process between different technologies in a single module.

ECAS uses data collected by sensors to increase the reliability of the network. In addition to data collected for ECAS, the sensors collect data relevant to C4ISR and other systems. Therefore, after the IoT Interface receives and interprets the data from sensors, it is necessary to forward the collected data to the respective systems of interest. The Selective Bridge module performs this function. It selectively distributes data received from the IoT Interface to be stored and consumed locally by ECAS or forwarded to external systems. This module works like a bridge between the IoT network and C4ISR or other systems, introducing only minimal overhead as data travels through ECAS. As depicted by the arrows in the ECAS Server Architecture at **Figure 3.2**, the data coming from the IoT Interface passes through the Selective Bridge module that, after separating the useful data that needs to be stored locally in the ECAS Database, it forwards the rest to the respective systems of interest.

Once the data has been collected and separated, it is necessary to forward the data that is not useful to ECAS and process the ones that are. However, ECAS needs the information from other systems available to combine with the sensors' data and perform the processing. Information such as RSSI, signal to noise ratio (SNR), and PDR are useful for ECAS to perform its role, and the providers of this information are network monitoring systems. The Integration Interface module aims to provide a single communication interface among network monitoring systems, C4ISR systems, and ECAS servers, as represented in **Figure 3.2** by the arrows connecting the Integration Interface module and the clouds that represent external systems. The first function of the Integration Interface is to distribute the data received from the Selective Bridge module to the respective systems of interest and to search other monitoring systems or C4ISR for information relevant to ECAS. The second function of this module is to generate alarms for other systems (i.e., when ECAS detects an anomaly, it can generate alarms for other systems to assist in the decision-making process of C4ISR systems). Finally, the Integration Interface can reconfigure the parameters of the sensors of interest of the C4ISR systems (e.g., modify thresholds of vital signs, weapons, etc.), as represented in **Figure 3.2** by the arrow that goes from the Integration Interface module to the Sensor Manager module (discussed later). This last function enables ECAS to serve as a central point for the dynamic reconfiguration of parameters in all sensors on the battlefield, regardless of whether that parameter is relevant to ECAS or external systems.

The first three modules focus on collecting data from IoT sensors or external sources and storing it in a local database. The Data Processing (DP) module is responsible for querying the database for context information received from the IoT network (periodic and trigger messages), combining this data with information from other monitoring systems (e.g., link quality indicators), and computing potential actions that could be taken to adjust the network according to the currently assessed situation. Some possible outputs of this processing are generating alarms for C4ISR systems (through the Integration Interface), reconfiguration the parameters of one or a set of sensors (through the Sensor Manager), updating the quality indicators in the database, as well as storing the processed data for later use as a dataset in the process of reasoning, which aims to anticipate the behavior of the network.

As mentioned above, two types of sensors may collect data in the field: MS and FOMS. An MS/FOMS collects data (e.g., temperature, humidity, vital signs, weaponry) for the ECAS and other C4ISR systems. Each system determines the thresholds and time intervals for reading measurements. The Sensor Manager module aims at abstracting the differences between possibly heterogeneous MS/FOMS in modifying the configuration parameters. For instance, suppose a C4ISR system needs to modify the critical threshold values of a parameter such as vital signs. In that case, this system communicates the parameters through the Integration Interface to the Sensor Manager module, which assembles and sends a package in the format expected by the sensor. The process for ECAS is the same. Suppose that a particular context situation causes the Data

Processing module to calculate a decision that requires changing the temperature or humidity thresholds of the sensor. In this case, the Data Processing module instructs the Sensor Manager module to set the new configuration parameters in the sensors. The Sensor Manager, in turn, relies on the IoT Interface to communicate with the sensors on the battlefield (represented in Figure 3.2 by the arrow that connects the Sensor Manager and IoT Interface modules).

After collecting, distributing, processing, and storing locally the outputs of the previously described modules inside an ECAS server instance, there is still the need to verify if these outputs can be helpful for other ECAS servers distributed on the battlefield. The output generated by a server in one CZ may or may not be useful for servers in other CZs. The Intelligent Sync module checks which ECAS server needs each piece of data and shares it. As mentioned, the ECAS system relies on servers distributed throughout the battlefield collecting, processing, and performing actions on the devices of their respective CZs. Misuse of resources can overload the network, potentially compromising network reliability. Thus, synchronizing data across servers should not overwhelm available network resources. The Intelligent Sync module checks the outputs of the Data Processing module and decides whether to share information with other ECAS servers based on context data, such as other servers' location data, to prevent network overload. Thus, if the generated output is useful to another CZ, Intelligent Sync sends it to this specific server over the Tactical Edge Network. For instance, the server located at CZ-1 may generate an output pertinent to the server located at CZ-2 (e.g., rain detected at CZ-1 and is moving towards CZ-2). Since the information related to this event is pertinent to the server located in CZ-2, this information should be sent as soon as possible. Nevertheless, forwarding the same data to the servers in CZs 3 and 4 is unnecessary since those will likely be unaffected by the phenomenon.

# 4 SETUP AND EVALUATION BY SIMULATION

In order to demonstrate the efficiency of the proposed system, simulations have been performed as proof-of-concept. The network simulator used was the *"ns-3 - discrete-event network simulator"*(RILEY; HENDERSON, 2010). This simulator offers a LoRaWAN module, which allows the simulation of a sensor network based on LoRa technology. Also, the NYUSIM simulator (SUN, 2017) was employed to calculate the attenuation in the wireless communication links by varying weather factors, as explained in the following sections.

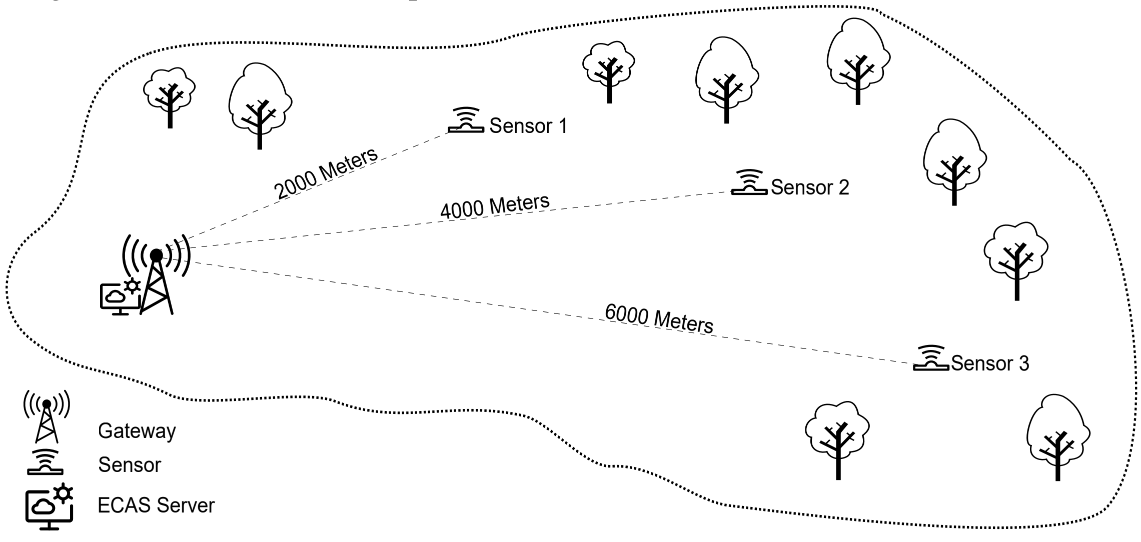
## 4.1 Simulation Setup

The implemented ECAS architecture uses LPWAN technology to communicate sensors distributed on the battlefield with servers. In the proof-of-concept simulation, LoRa was selected as the LPWAN communication technology. LoRa is a proprietary spread spectrum modulation technique, derivative of chirp spread spectrum modulation (CSS); military applications have used this technique due to its long-range coverage and interference robustness (SINHA; WEI; HWANG, 2017). Although LoRa's characteristics make it suitable for the operation of the proposed system, factors such as support for applications with real-time requirements (LEONARDI; BATTAGLIA; BELLO, 2019)(KHUTSOANE et al., 2019), interoperability with command and tactical control systems (JALAIAN et al., 2018) were also considered for the choice of this technology.

LoRa radio has four configuration parameters: carrier frequency, spreading factor (SF), bandwidth, and coding rate. Setting these parameters determines signal robustness, power consumption, and transmission range (BOR; VIDLER; ROEDIG, 2016). This simulation explored one of the main LoRa features, the Data Rate (DR). The DR consists of the combination of two LoRa configuration parameters: the bandwidth and the spreading factor. The highest DR in LoRa communication provides more payload data transmitted over shorter distances. Although the amount of data transmitted is less in the lower DR, the signal strength is grander, allowing communication between the Sensor and the Gateway over long distances and unfavorable weather conditions. These factors are important to consider regarding the application scenario under concern.

An experiment with the following parameters was carried out. A LoRa gateway operating on three channels of 868MHz, one channel for each sensor, was installed at 15 meters height. Three sensors with a fixed location (without mobility) at 2,000, 4,000, and 6,000 meters from the gateway and at 1.7 meters in height. Each round of the experiment represented 24 hours of simulation, and each node, every 60 seconds, sent a packet to the gateway. **figure 4.1** illustrates the scenario of the experiment just described.

Figure 4.1: Scenario of the Experiment. (ZIBETTI; WICKBOLDT; FREITAS, 2022).



Source: Author

At each experiment round, was applied a different DR in the sensors. For each DR, an attenuation representing 1mm/h of rain was applied to the signal, then run one round. Therefore, 151 rounds of the experiment were executed for each DR. Starting with 0mm/h of rain, representing a channel free of attenuation due to rain, and ending with 150mm/h, representing a heavy attenuation scenario.

In the first stage of the experiment, the same DR was configured in the three sensors, ranging from DR-0 to DR-5. The DR-5 has the highest packet rate and the lowest signal reach, and the DR-0 has the lowest packet rate and a greater signal reach. **table 4.1** presents the combinations of SF and bandwidth for the DRs applied during the experiment. The DR-5 represents the configuration that can transmit the most massive volume of data. While the DR-0 represents a more robust signal to attenuation, therefore it reaches greater distances. Although the robustness of the signal experienced by DR-0 is better than the other DRs, the volume of data transmitted by it is lower than the higher DRs. Therefore, applying the appropriate DR to the sensors distributed across the battlefield, considering the physical and environmental context, increases the network's reliability by offering an adaptable context communication infrastructure.

In the second stage of the experiment, the goal was to observe how ECAS acts in the monitored environment. In this approach, ECAS collects the battlefield information

Table 4.1: Data Rates of the Experiment.

| DR | SF | Bandwidth |
|---|---|---|
| 0 | 12 | 125kHz |
| 1 | 11 | 125kHz |
| 2 | 10 | 125kHz |
| 3 | 9 | 125kHz |
| 4 | 8 | 125kHz |
| 5 | 7 | 125kHz |

and adapts the DR of each sensor to the best DR according to the observed weather conditions. This adaptation is made using two approaches called Conservative and Aggressive. In the Conservative approach, ECAS receives the rain rate in mm/h from each sensor, calculates the attenuation of this factor in the communication link, and, if necessary, reconfigures the DR of a given sensor to increase signal robustness or improve transmission rate. In this approach, the system modifies the sensor's DR before reaching a threshold. In contrast, in the Aggressive approach, ECAS expects a given sensor to decrease in the PDR, and only then it reconfigures that sensor's DR.

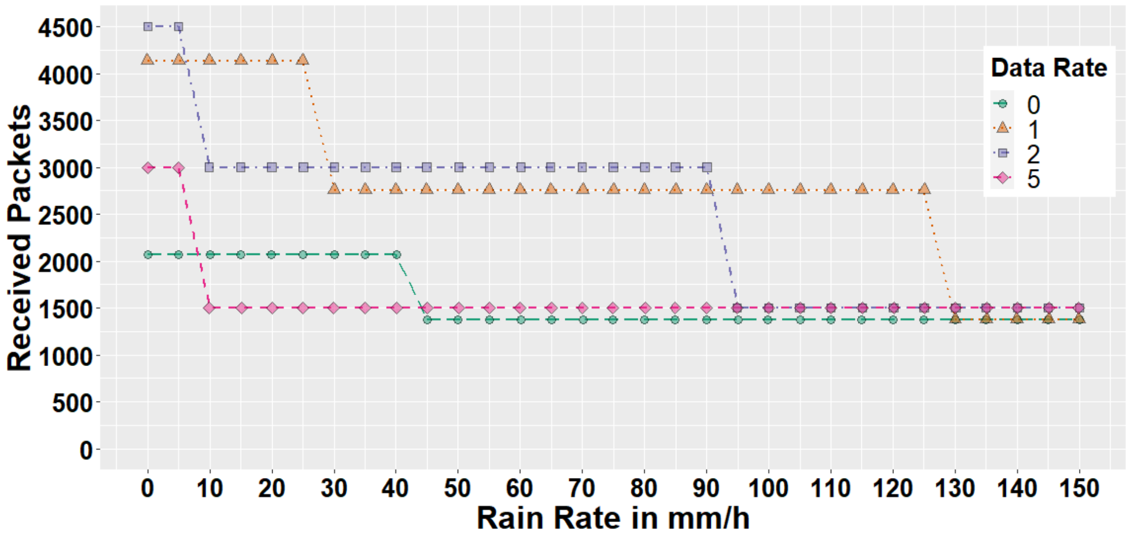
The idea of naming the approaches with the terms Conservative and Aggressive refers to whether the system tolerates losses or not. In the Conservative approach, the system aims to avoid losses; therefore, changing the sensor parameters even if it incurs performance degradation. On the other hand, the Aggressive approach aims to achieve the highest possible data rate, even if it causes some losses in the communication process.

The Conservative approach has been configured to reduce the DR of the sensor, which can be affected by attenuation, by one 5mm/h of rain before the attenuation causes any impact on the PDR. Therefore, before any sensor has a reduction in PDR, its DR is reduced to increase signal robustness, although the packet sending rate also reduces. In the Aggressive approach, the highest DR is used until a reduction in PDR from a given sensor is observed. Only then are adjustments made to the DR of the respective sensor. This approach allows better use of higher data rates in scenarios with minor climate variations.

## 4.2 Simulation Results

**Figure 4.2** shows the results of the first stage of the experiment, in which a fixed DR on the three sensors was set. The Y-axis of **figure 4.2** represents the received packets by the gateway per round of the experiment; the X-axis represents the increase of rain in mm/h that goes from 0mm/h to 150mm/h. The DR-3 and DR-4 have been hidden from the **figure 4.2** to visualize the results better. However, the information of these DRs can be found in **table 4.2**, which shows the number of packets sent, received, lost, and PDR in percentage for each DR applied in the experiment. During the experiment, it was observed that the packet sending rate experienced by the three sensors from DR-5 to DR-2 was the same, with 1500 packages per experimental round. While for DR-1 and DR-0, all sensors suffered a fall in the packet sending rate. It reached approximately 1380 packets sent per round with DR-1 and 690 with DR-0 per sensor.

Figure 4.2: Delivered Packets Per Data Rate.



Source: Author

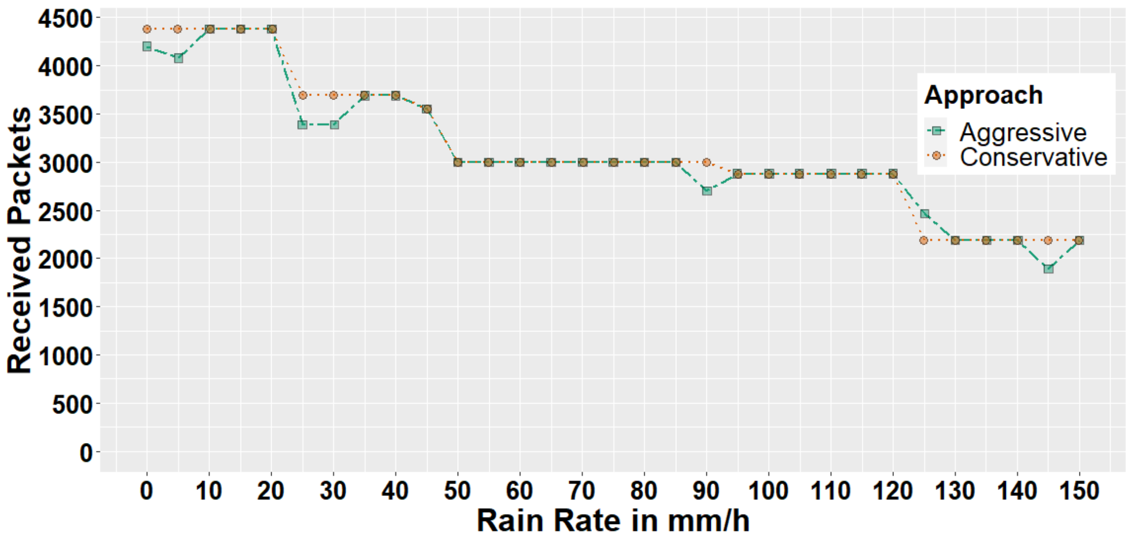
Table 4.2: Sent, Received, Lost Packets and PDR% per Data Rate.

| DR | Sent | Received | Lost | PDR% |
|---|---|---|---|---|
| 0 | 64,139 | 48,959 | 15,180 | 76% |
| 1 | 128,216 | 86,851 | 41,365 | 68% |
| 2 | 139,500 | 78,000 | 61,500 | 56% |
| 3 | 139,500 | 66,000 | 73,500 | 47% |
| 4 | 139,500 | 57,000 | 82,500 | 41% |
| 5 | 139,500 | 49,500 | 90,000 | 35% |

DR-0 and DR-1 achieved a better PDR, although the number of packets sent is lower than higher DRs. In DR-1, the packet sent rate per node approaches 1,380 packets, while in DR-0, the rate drops to approximately 690. **Figure 4.2** shows that the DR that performed better in contrast to the others was DR-1. In DR-1, the packets sent rate comes near 1,500 packets, which is experienced by the higher DRs, and the signal's robustness makes all three sensors connect with the gateway at higher rainfall rates. Configured with DR-1, the furthest sensor can connect with the gateway up to 25mm/h of rain. At DR-2 and above, the same sensor's connection does not exceed 5mm/h of rain. Sensor 2 can connect to the gateway up to 125mm/h of rain with DR-1, contrasting with the 90mm/h of DR-2 and lower rainfall rates in the higher DRs. In the DR-0, which represents the most robust signal, but with the lowest packet sent rate, the farthest node had a connection with the gateway up to 40mm/h of rain when it lost connection. At this DR configuration, sensors 1 and 2 remain connected to the gateway up to 150mm/h of rain.

**Figure 4.3** presents the experiment's results considering a scenario with the implementation of the ECAS. The DRs adapted in this scenario consider the rain rate and the sensor's distance to the gateway. The Y-axis in the chart of **figure 4.3** represents the received packets' rate, while the X-axis represents the rain rate in mm/h. For both the Aggressive adaptive approach and the Conservative adaptive approach, the results obtained were higher than fixed DRs.

Figure 4.3: Delivered Packets Adaptive Aggressive X Conservative Approach.



Source: Author

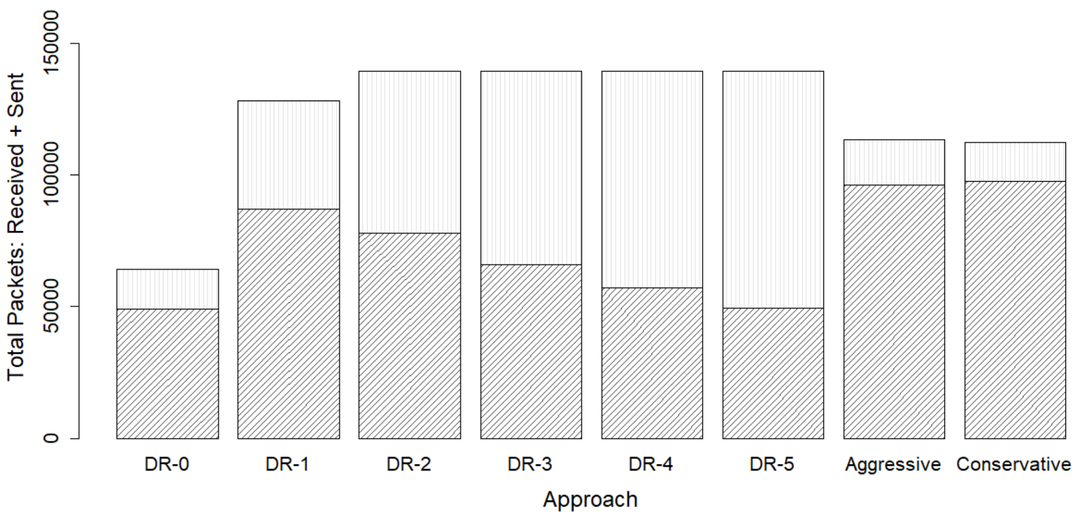
In the Aggressive approach, the sensor DR was reduced by one for each loss detection to increase signal robustness and improve the connection with the gateway. This approach allows configuring the sensor with a DR higher, which has a higher data rate, still considering the climatic conditions at the time of transmission. The Conservative approach behaves differently. When the system observes a given rain level, the DR of the sensor is decremented by one 5mm/h of rain before the signal can be affected. Therefore, the sensor should not lose connection to the gateway whatsoever. However, suppose the rain does not exceed this 5mm/h threshold. In that case, the sensor will have stopped using the DR with a higher packet sending rate until the system sees a reduction in the rain rate, which would allow an increase in DR considering the new rain scenario.

In **figure 4.3**, the dotted line represents the Aggressive approach. At the beginning of the experiment, the Aggressive approach sometimes presented a received packet rate lower than the Conservative approach. This behavior occurs because the Aggressive approach tries to use the largest possible DR from the beginning of the experiment, then starts the experiment with losses until it finds the ideal DR. From the 10mm/h of rain, the Aggressive approach achieves the same number of packet received as the Conservative approach. During the experiment, sometimes, the Aggressive approach achieves a lower number of packets received than the Conservative approach. It happens due to the method used to determine when to adjust the DR in the Aggressive approach. At 120mm/h, the Aggressive approach performs better than the Conservative approach. It was expected that this would happen at some moment. While the Conservative approach prevents losses, the Aggressive approach uses losses as a factor in determining when to adjust the DR. At this moment of the experiment, the loss occurred in the 124mm/h of rain, which ensured the number of packets received was higher for the Aggressive approach in the range of 120mm/h to 124mm/h of rain.

The experiment, which served as a proof-of-concept to exemplify the functioning of the ECAS, adopted the premise that the rain rate only increases. Thus, the experiment started with a 0mm/h rain rate and ended at 150mm/h. This premise demonstrates the system's behavior if adopted for the battlefield scenario where the rain only increases. However, in a real scenario, rain rates do not increase linearly. The rain can increase, stabilize, or even reduce before stopping. It is not possible to determine when one of these events will occur. Thus, two approaches have been proposed to demonstrate the system's behavior, considering the performance and reliability requirements.

**Figure 4.4** compares the experiment performed with fixed DR and the Aggressive and Conservative approaches of the ECAS. On the Y-axis of **Figure 4.4**, the Total Packets Received plus the Total Packets Sent is displayed. Each bar represents a setting applied in the experiment. Adopting any of the approaches offered by ECAS makes it possible to obtain a higher PDR than fixed DR approaches. The PDR experienced by DR-1 and DR-2 is closer to the PDR experienced by the approaches offered in the ECAS. **Table 4.3** presents the employed approach, the number of packets sent, and the PDR for each experiment round. ECAS approaches offer a packet-sending rate that approximates the approaches with the best fixed DR performance. However, the amount of received packets is much higher when ECAS is in place to monitor and adapt to the environment.

Figure 4.4: Total Packets Received plus Sent per Approach - Fixed Data Rate, Adaptive Aggressive, and Conservative.



Source: Author

Table 4.3: Result per Experiment Setup.

| Approach | Packets | PDR |
|---|---|---|
| Fixed DR-0 | 64139 | 76% |
| Fixed DR-1 | 128216 | 68% |
| Fixed DR-2 | 139500 | 56% |
| Fixed DR-3 | 139500 | 47% |
| Fixed DR-4 | 139500 | 41% |
| Fixed DR-5 | 139500 | 35% |
| ECAS-Aggressive | 113309 | 85% |
| ECAS-Conservative | 112231 | 87% |

## 4.3 Partial Conclusions

The simulation results showed that the dynamic adaptation of network parameters could make the communication process more efficient. Compared with the fixed approach, the adaptive results achieved an 11% increase in the total packets delivered while reducing the total packets sent by 12%. From these results, it is possible to infer that the use of the adaptive approach, in addition to making the communication process more efficient, also has the potential to reduce resource/energy consumption.

# 5 PROTOTYPE SETUP AND EVALUATION

This chapter aims to present a system prototype of the architecture defined at **Figure 3.2**. First, the **Section 5.1** describes the hardware and software components that compose the system prototype. Then, the **Section 5.2** presents the scenario and method of evaluation of the deployed system prototype.

## 5.1 Prototype Setup

The proposed system depicted at **Figure 3.2** is divided into sensor and server architecture, which are discussed in **Subsection 3.2.1** and **Subsection 3.2.2**, respectively. Therefore, this section follows the same organization to describe the system prototype. **Subsection 5.1.1** presents the sensor prototype's hardware and software components. Then, **Subsection 5.1.2** presents the ECAS server prototype's hardware and software components.

### 5.1.1 Sensor prototype

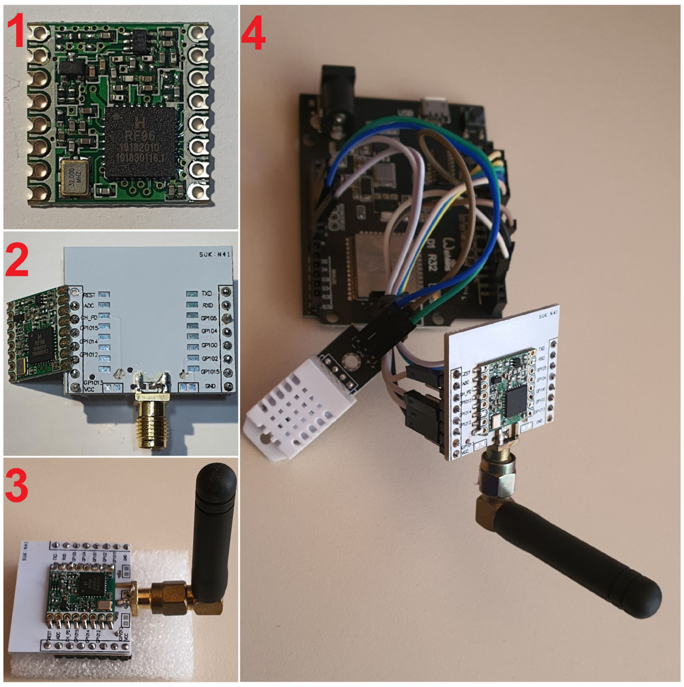
The works presented at **Chapter 2** concluded that environmental factors impact wireless communication technologies. However, neither of the cited works, and as far as our knowledge has reached, no dataset showing environmental factors' effect on wireless technologies were found. Therefore, to deploy a prototype and evaluate its performance in the real world, it was necessary to collect temperature and humidity from a real environment and measure its impact on wireless communication. In order to create a dataset, a communication module with the LoRa technology was developed.

**Figure 5.1** shows the communication module and hardware components that compose the sensor prototype. The number from one to four in the upper left corner at **Figure 5.1** represents the developed sensor's step, which goes from the transceiver chip to the sensor's prototype.

The number one at **Figure 5.1** shows the LoRa transceiver RFM95W by (HOPERF ELETRONIC, 2019). One of the configuration parameters of the simulated experiment presented in **Chapter 4** was the frequency band 868MHz. Thus, to deploy and evaluate a system closest to the simulation setup, the RFM95W was chosen to compose the sensor prototype, once the frequency bands that it operates are 868MHz to 915MGHz through a Semtech SX1276 chip (SEMTECH, 2020).

An ESP8266 module adapter plate, number two at **Figure 5.1**, was used to attach the SubMiniature version A (SMA) Reverse-polarity (RP) connector and give a pinout to connect the communication module with a development board. The number three at **Figure 5.1** shows the communication module with an SMA antenna of +2 decibels relative to isotropic (dBi) attached to it.

Figure 5.1: Communication Module and Sensor Prototype.



Source: Author

The number four at **Figure 5.1** shows the communication module connected at a Wemos D1 R32 board. The choice of the Wemos board is due to its design and configuration. The Wemos D1 R32 is a proprietary development board that has a microcontroller ESP32 WROOM-32, 512KB of SRAM, 4MB of flash memory, and pinout design by Arduino uno boards (AZ-DELIVERY, 2019). Lastly, the last component of the sensor prototype, also presented in **Figure 5.1** number four, is the DHT22 temperature and humidity reading sensor (AOSONG ELECTRONICS, 2017).

The sensor architecture depicted at **Figure 3.2** presents four sensor modules: Data Manager, Sensor Manager, Communication Manager, and Message Manager. However, some sensor modules were partially deployed due to the lack of a dataset to evaluate the proposed system and the time spent to create that. Thus, the modules implemented and the libraries that compose them are described below.

The **Sensor Manager** module is for those configurations which determine the basic sensor behavior, such as data measure frequency, if sending measure data or storing in the sensor to send in the next moment, if sleep or stay wake-up due to unsafe operations conditions, and so on. Therefore, this module is essential to the sensor's basic operation. The sensor manager module functions deployed at the system prototype were: intervals of collection and sent, whether the sensor should stay awake or sleep, and data persistence when the sensor sleep.

After receiving and processing a message from the sensor, the ECAS server can respond to the sensor with some sensor reconfiguration parameters. If this happens, the sensor processes the received message, performs reconfiguration, and responds to the server. For instance, if this message contains an order for the sensor to change its send interval and not go to sleep due to unsafe conditions, the sensor will reconfigure itself and respond to the server with its new configuration. The **Message Manager** module of the sensor prototype performs this task.

The **Data Manager** module specifies how the data will be represented to the applications. In order to perform it, the sensor prototype uses two libraries, Cayenne Low Power Payload (CayenneLPP) and ArduinoJSON. The payload size available at LoRaWAN packet is quite limited; hence, optimizing the packet size is necessary. The myDevices company created CayenneLPP, intending to provide a convenient and easy way to send data over LPWAN technologies, considering the payload size limitations of these technologies (MYDEVICES, 2017). The EletronicCats developed a library that implements CayenneLPP in Arduino projects (CATS, 2021), and the sensor prototype uses this library to send data to the ECAS server (i.e., uplink message).

In contrast, the downlink message (i.e., the data sent by the server to the sensor) is not size significant. These are the send thresholds and configuration parameters to set at the sensors. Thus, the sensor prototype uses the ArduinoJson library (BLANCHON, 2023) to receive downlink messages from the server.

The **Communication Manager** module configures the transceiver module to the data transmit and receive routines. As mentioned in **Section 4.1**, LoRa refers to the proprietary spread spectrum modulation technique. However, the network protocol widely implemented on LoRa is named LoRaWAN. LoRaWAN is a network protocol developed by LoRa Alliance and optimized for battery-powered end devices (LORA ALLIANCE®, 2016). The library that implements the LoRaWAN protocol in the sensor prototype is Beelan-LoRaWAN (BEELAN, 2022), and some resources of the LoRaWAN protocol deployed by this library and management by the **Communication Manager** module are Device Classes A or C, frequency band, TX power, and DR.

### 5.1.2 Server prototype

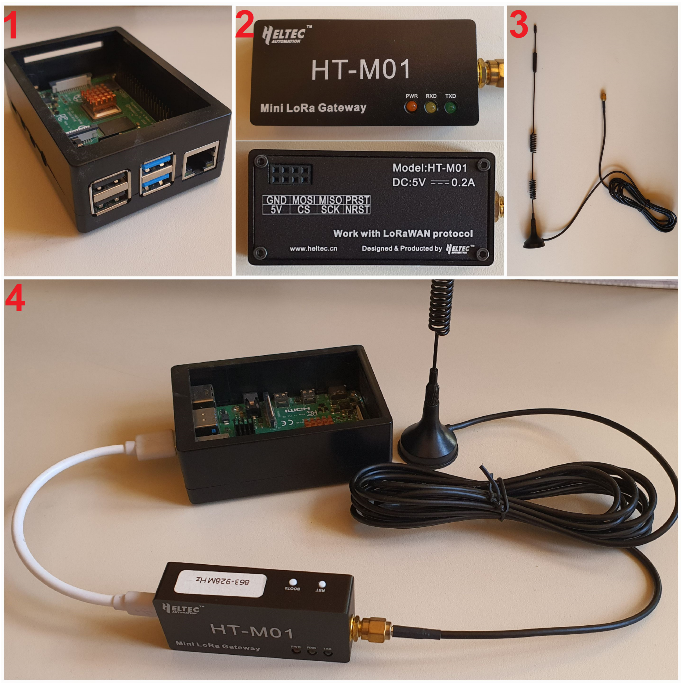
In (ZIBETTI; WICKBOLDT; FREITAS, 2022), the authors highlighted some technologies to demonstrate the proposed system development feasibility. Thus, this section presents a system prototype implemented from some technologies suggested by the authors, as well as other mechanisms that showed themselves as a better option for this prototype implementation.**Figure 5.2** presents the hardware which composes the ECAS server prototype. The numbers one to three in the upper left corner of **Figure 5.2** identify the components of the server, and the number four shows the server prototype.

The battlefield scenario has limited resources like power supply and communication networks; therefore, to deploy systems in this context, it is necessary to consider these factors. Highlighted with number one in **Figure 5.2**, it is a Raspberry Pi 4 model B with 8GB RAM. The Raspberry Pi 4 model b has a quad-core cortex-A72 (ARM v8) 64-bit processor, four USB ports, two micro HDMI ports, one micro SD card slot, one RJ45 Gigabit Ethernet LAN connector, and wireless 802.11 2.4 GHz and 5.0 GHz (RASPBERRY PI TRADING LTD., 2021). These configurations make the Raspberry Pi 4 a great candidate to perform the server role.

The number two at **Figure 5.2** shows the front and back part of the mini LoRa gateway HT-M01 developed by Heltec. The main physical characteristics that make the HT-M01 a suitable candidate for the gateway role are a resistant aluminum case, a compact size, and two communication interfaces, one SPI, and one micro USB. Beyond the physical characteristics, the HT-M01 is compatible with the LoRaWAN protocol and Linux OS distributions. Lastly, number three at **Figure 5.2** shows the antenna, with +15 dBi gain, used by the HT-M01 gateway.

Since ECAS works like a distributed system, sharing information to support decision-making from all its ecosystems, the first premise that directed its development was the easy scalability. Furthermore, another premise that directed the system prototype development was prioritizing free and open-source software. Therefore, the server prototype executes a Linux Ubuntu Server, 20.04.6 version installed at 64GB micro SD attached to Raspberry Pi, and deploys the ECAS modules on top of Docker Containers.

Figure 5.2: Server Prototype.



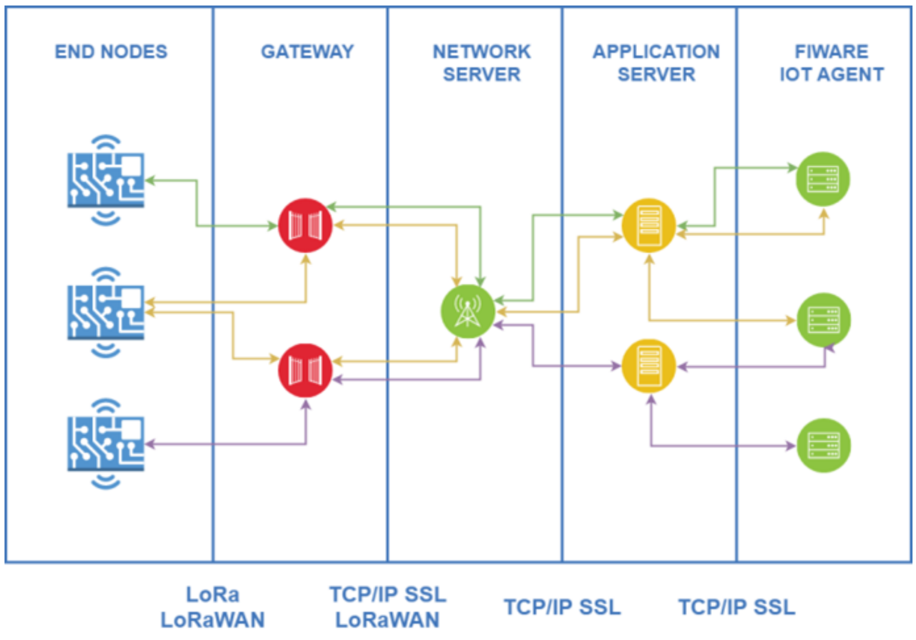
Source: Author

A container is a standard unit of software that gathers a code and all its dependencies to run a particular application quickly and reliably independent of adjacent computing infrastructure. This software unit runs on the host machine as a sandbox process; thus, each container runs its processes isolated from the host machine and other containers (DOCKER INC., 2023).

Section 5 of (ZIBETTI; WICKBOLDT; FREITAS, 2022) suggests deploying the ECAS on top of FIWARE. FIWARE is a framework that aims to accelerate the development of IoT-related solutions (FIWARE, 2021a). In the system prototype, FIWARE provides, through NGSI API (FIWARE, 2021b), a data model based on entities and an interface to support message and command exchanges between sensors and servers. Another task that a FIWARE component performs on the system prototype: storing data in the system database. Therefore, the system data model, exchange of messages, and data storage are tasks implemented on top of the FIWARE framework components in the system prototype.

The first described module is the **IoT Interface**, whose main task is to provide a communication interface between servers and sensors. In (ZIBETTI; WICKBOLDT; FREITAS, 2022), the authors suggest deploying the **IoT Interface** module with an IoT Agent based on JSON or Ultralight encoding provided by FIWARE. Also, in the work conclusion, the authors reported that security-related issues were outside the scope; however, they claim that the LoRaWAN protocol addresses such questions. Despite IoT Agents cited by authors being suitable candidates, FIWARE also has an IoT Agent, ready to use, based on the LoRaWAN protocol, which could change data and commands between servers and sensors (CALVO et al., 2022). Hence, to perform the message exchange between servers and sensors and address security questions covered by the LoRaWAN protocol, the **IoT Interface** module was deployed through the FIWARE IoT Agent based on the LoRaWAN protocol and its adjacent component architecture. **Figure 5.3** shows the necessary stack of components to deploy a FIWARE IoT Agent based on LoRaWAN protocol, composed of a Gateway, Network Server (NS), and Application Server (AS).

Figure 5.3: Network architecture for LoRaWAN protocol-based systems and FIWARE IoT Agent.



Source: (CALVO et al., 2022)

The Gateway, depicted between end nodes and the Network Server at **Figure 5.3**, makes the message exchange between sensors and the NS. The Gateway and NS connect over TCP/IP protocol with SSL, while the connection of the sensor with the Gateway is over LoRaWAN protocol, as presented at **Figure 5.3**.

LoRaWAN relies on a star network topology, and the NS stays at the center of the topology. Some functions of NS are end-device address check, frame authentication check, frame counter check, acknowledgments, and data rate adaptation. Therefore, the NS represents the last mile of LoRaWAN protocol processing (LORA ALLIANCE®, 2020). From the NS to the AS, the connection is established over the TCP/IP protocol with SSL, and only the payload is forwarded (LORA ALLIANCE®, 2020).

The last component of LoRaWAN architecture, the Application Server, handles the application layer payload from end nodes to the target application, in this case, the ECAS, through the FIWARE IoT Agent. Application Server also generates the application layer downlink payloads (*e.g.*, the JSON messages forwarded from the ECAS server) towards the end nodes (LORA ALLIANCE®, 2020).

The FIWARE IoT Agent based on the LoRaWAN protocol has compatibility with the above component stack, Gateway, NS, and AS deployed by The Things Network and Chirpstack technologies (CALVO et al., 2022). Due to hardware and licensing requirements, the IoT Interface module was deployed with the components stack developed by the Chirpstack open source project.

Initially, the **Selective Bridge** module had two tasks: store received data on the ECAS database and delivers the interest data to the **Integration Interface** module to forward to third-party systems. The objective of the second task was to reduce the overhead inside the ECAS, making the received data available to other systems as soon as possible. However, although initially proposed on the **Selective Bridge** module at **Section 3.2.2**, the **Integration Interface** module performs the second task in the system prototype.

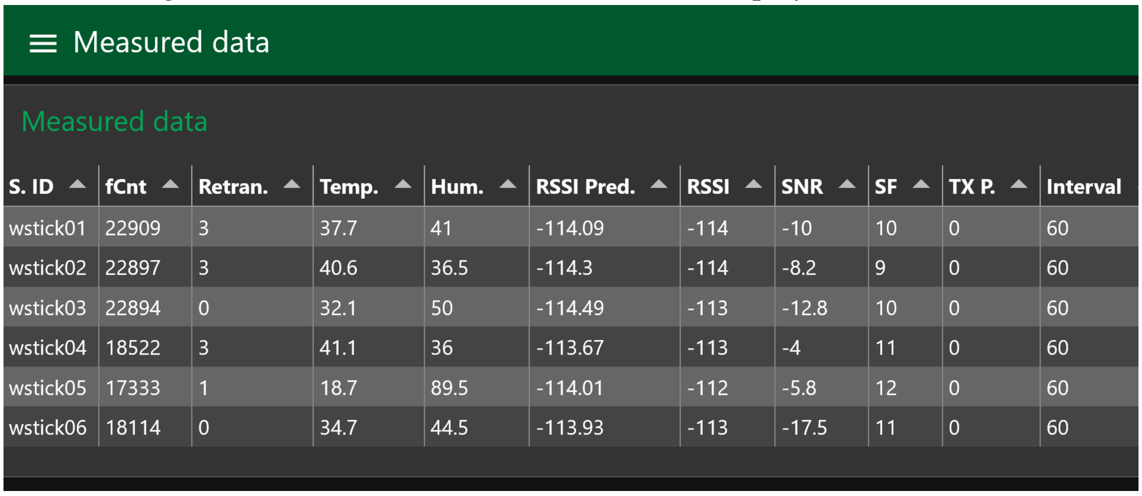
In order to store received data in the ECAS database, the **Selective Bridge** module uses the Cygnus connector. The Cygnus is a connector based on Apache Flume developed to persist context data into third-party database systems (BUENO et al., 2022). In the ECAS case, the database in which the Cygnus connector performs the data persist task is the open-source relational database MariaDB (MARIADB FOUNDATION, 2022).

The third deployed module, **Integration Interface**, serves as an interface between the ECAS and other systems. It performs this task using the MQTT Broker Eclipse Mosquitto and the NodeRed tool. Eclipse Mosquitto is a Broker that implements the MQTT protocol to exchange messages (ECLIPSE MOSQUITTO™, 2022). In the ECAS, the technologies which compose the system modules, such as the FIWARE IoT Agent, Gateway, NS, AS, and NodeRed, perform message exchanges through the MQTT Broker. Therefore, any third-party system compatible with MQTT can use it to exchange messages with the ECAS.

The NodeRed is a flow-based programming tool developed over Node.js to connect hardware devices and APIs (OPENJS FOUNDATION AND NODE-RED CONTRIBUTORS, 2023). **Figure 5.4** presents a dashboard created through NodeRed, which shows the last data received by the sensors in real-time. Some helpful information depicted at **Figure 5.4** is the last received packet's retransmissions number, the RSSI predicted by the sensor used to determine the parameters of the transceiver module, the RSSI and SNR measured at the last received packet, TX power, and SF set at the sensors.

Although the NodeRed consumes locally the data to show on a dashboard at the system prototype, this tool has many plugins, named pallets, that make an interface with third-party systems. Therefore, the dashboard depicted at **Figure 5.4** aims to demonstrate the viability of sharing the collected data from the ECAS with external systems (e.g., as the NodeRed dashboard plugin).

Figure 5.4: Real-time Measured Data Pannel Deployed at NodeRed.



Measured data

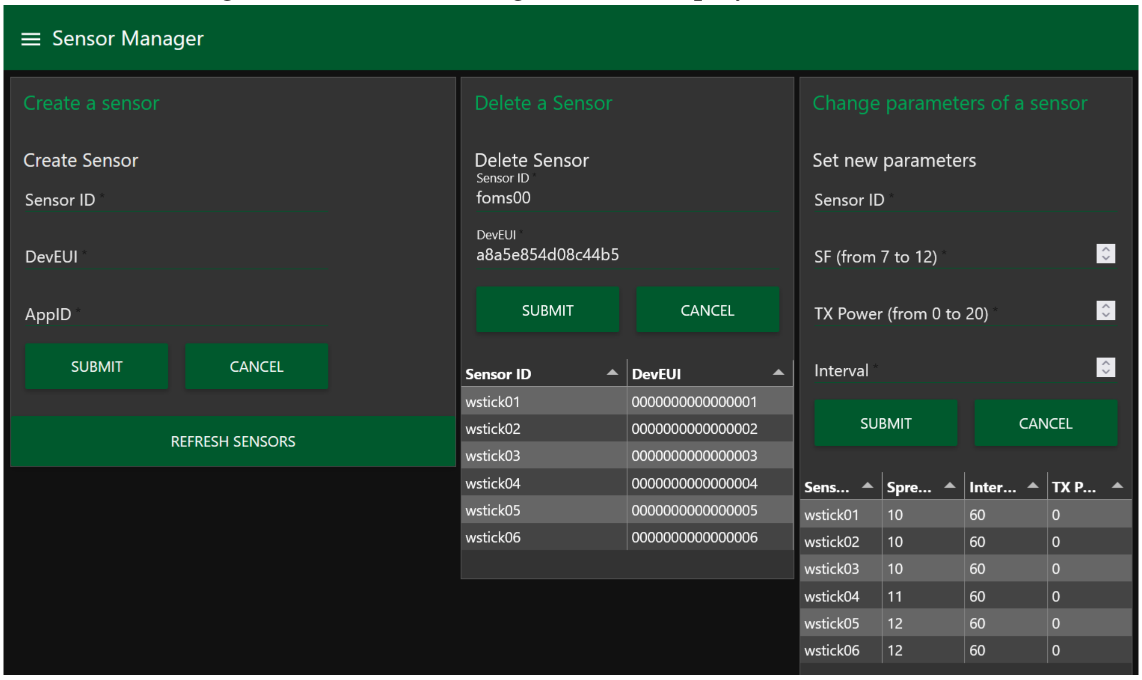
| S. ID | fCnt | Retran. | Temp. | Hum. | RSSI Pred. | RSSI | SNR | SF | TX P. | Interval |
|---|---|---|---|---|---|---|---|---|---|---|
| wstick01 | 22909 | 3 | 37.7 | 41 | -114.09 | -114 | -10 | 10 | 0 | 60 |
| wstick02 | 22897 | 3 | 40.6 | 36.5 | -114.3 | -114 | -8.2 | 9 | 0 | 60 |
| wstick03 | 22894 | 0 | 32.1 | 50 | -114.49 | -113 | -12.8 | 10 | 0 | 60 |
| wstick04 | 18522 | 3 | 41.1 | 36 | -113.67 | -113 | -4 | 11 | 0 | 60 |
| wstick05 | 17333 | 1 | 18.7 | 89.5 | -114.01 | -112 | -5.8 | 12 | 0 | 60 |
| wstick06 | 18114 | 0 | 34.7 | 44.5 | -113.93 | -113 | -17.5 | 11 | 0 | 60 |

Source: Author

The next module described, **Sensor Manager**, is the sensors' reconfiguration interface on the server's side. Initially, only the server would configure the sensors' transceivers parameters by applying the proposed approaches, Aggressive or Conservative. However, if only the server did this task, the reconfiguration process could be inaccurate because the server would use a condition prior to the current condition of the sensor to support decision-making. Thus, the system prototype proposes two ways to configure the sensors' transceivers, one on the server side and the other on the sensor side.

From the sensor side, the transceiver configuration process is as follows, the sensor performs a temperature and humidity read, predicts the RSSI value for the next communication, and configures its transceiver module to guarantee that the next message reaches the server; how this is done will describe at the end of this subsection, in **Data Processing** module. On the other hand, if the server wants to send a reconfiguration command to a sensor, this is done through of developed interface on the NodeRed, depicted at **Figure 5.5**. In addition to the sensors reconfiguration menu, the Sensor Manager interface provides two more menus, one to create the sensors that compose the system and the other to delete an existing sensor, according to represented at **Figure 5.5**.

Figure 5.5: Sensor Manager module deployed at NodeRed.



Sensor ID: foms00

DevEUI: a8a5e854d08c44b5

| Sensor ID | DevEUI |
|---|---|
| wstick01 | 0000000000000001 |
| wstick02 | 0000000000000002 |
| wstick03 | 0000000000000003 |
| wstick04 | 0000000000000004 |
| wstick05 | 0000000000000005 |
| wstick06 | 0000000000000006 |

| Sens... | Spre... | Inter... | TX P... |
|---|---|---|---|
| wstick01 | 10 | 60 | 0 |
| wstick02 | 10 | 60 | 0 |
| wstick03 | 10 | 60 | 0 |
| wstick04 | 11 | 60 | 0 |
| wstick05 | 12 | 60 | 0 |
| wstick06 | 12 | 60 | 0 |

Source: Author

Until here were described four system modules, **IoT Interface**, **Selective Bridge**, **Integration Interface**, and **Sensor Manager** (i.e., remains to describe two modules, **Data Processing** and **Intelligent Sync**). Despite its importance to the system, due to time constraints, the system prototype does not have the **Intelligent Sync** module implemented. Thus, the remainder of this subsection will describe the **Data Processing** module, which defines the system's intelligence that supports decision-making.

In **Section 3.2.2**, the description of the **Data Processing** module includes three essential tasks: combine environmental and network context data to produce actions to make on network configuration parameters, generate alarms to other systems, as well update the quality indicators to improve the following decisions. To perform these tasks, the ECAS uses the data stored in its database, depicted in **Table 5.1** and described below. Nevertheless, before describing the **Table 5.1** data, it is worth mentioning that not all these data support decision-make at the system prototype. Thus, a summary of data that supports decision-making in the system prototype follows, and a description of the remaining data will be available in the project repository on GitHub (ZIBETTI, 2023).

The first column of **Table 5.1** presents the data label; the second column contains the data unit of measurement, and the third column specifies who provides the data. The first line, a top-down, contains the Timestamp, which represents the instant that collected data arrives in the Chirpstack AS. As the name suggests, Send Interval is the time interval

Table 5.1: Stored Data at the ECAS Database.

| Label | Unit of Measurement | Provide by |
|---|---|---|
| Timestamp | UTC | Chirpstack AS |
| Bandwidth | kHz | Gateway |
| Frequency | MHz | Gateway |
| Spreading Factor | n of CSS | Gateway |
| Frame Counter | Int. | Gateway |
| Measured RSSI | dBm | Gateway |
| Measured SNR | dB | Gateway |
| Predicted RSSI | dBm | Sensor |
| TX Power | dB | Sensor |
| Measured Temperature | $^{\circ}C$ | Sensor |
| Measured Humidity | RH | Sensor |
| Send Interval | Seconds | Sensor |
| Frame Retransmission Attempt Counter | Int. | Sensor |

in seconds that a sensor sends an uplink to the server. Frame Counter (fCnt) is a counter deployed by LoRaWAN protocol to control the process of retransmission and acknowledgment of the uplink and downlink messages. Frame Retransmission Attempt Counter is a counter deployed by the ECAS that counts the number of times a sensor sends the same frame (*i.e.*, the frame with the same fCnt) to the server until the sensor receives an acknowledged. TX Power is a unit in dB set at the sensor transceiver module to increase the power of transmission or reception. As mentioned at **Section 4.1**, the Spreading Factor configuration parameter composes the LoRa technology. Measured Temperature and Measured Humidity collected by the sensor at its physical context through the DHT22 sensor. The Predicted RSSI is a value generated by the sensor trying to predict the Measured RSSI at the next transmission before of perform it. Measured RSSI and SNR represent the values that Gateway measures when a sensor message arrives.

In order to perform the tasks that support the sensors transceivers' reconfiguring process, the DP module uses the Random Forest Regression algorithm and the Micromlgen Python library. For the alarms generate task, the DP module uses Grafana, which in addition to alarms generate to other systems, provides a monitoring interface that can improve the data visualization.

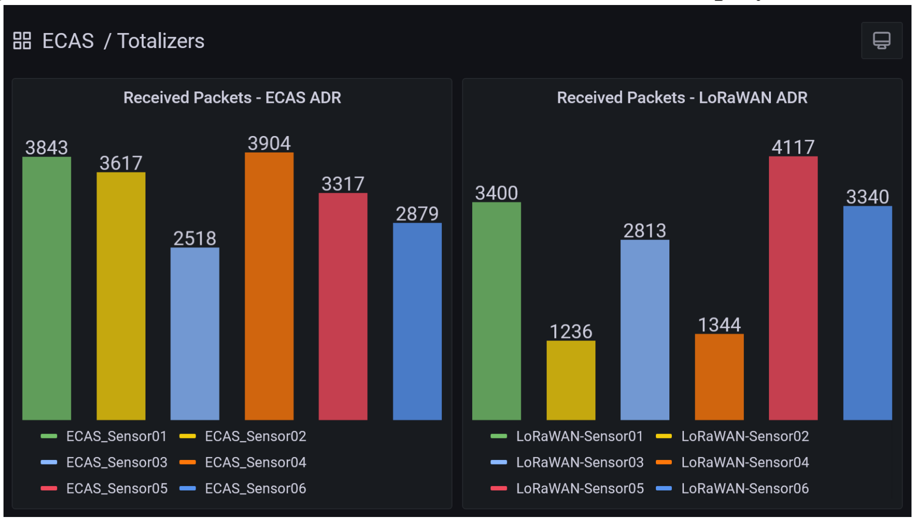
Random Forest Regression (RFR) is a supervised machine learning (ML) algorithm, which belongs to the ensemble learning methods, to solve regression problems (RUSSELL; NORVIG, 2020). In summary, the RFR operates as follows, given an input dataset whose shape comprises one predictor sets and one class, give one prediction model to predict the class-related value. In other words, in the ECAS context, given the humidity, temperature, TX power, and SF as predictors; and the measured RSSI related to these predictors as the class, give one prediction model to predict the RSSI to the next communication.

In the system prototype, the RFR algorithm deployed by the sklearn python library (BUITINCK et al., 2013) give the prediction model, and the micromlgen python library provides the source code that deploys the model at the sensor. Once the sensor has deployed the prediction model, it provides the current temperature and humidity reading, TX power, and SF set at the transceiver module to get a predicted RSSI for the next communication. Thus, from the predicted RSSI, to increase the probability of a packet arriving at the receiver, the sensor can change the transceiver module configuration, such as TX power and SF, before sending.

In order to generate alarms for other systems, the DP module uses the Grafana tool. Grafana is an open-source tool that provides time-series data queries, data visualization at dashboards and tables, and a generate alarms service (CHAKRABORTY; KUNDAN, 2021). Some available resources at Grafana alarms service include alert rules, labels, notification policies, and contact points. Alert Rules specify the criteria that determine when to fire an alarm (e.g., a condition to fire, the frequency of evaluation of this condition, and the duration that a condition can stay until necessary to fire an alarm). Labels aid in matching alert rules with notification policies, creating a group of rules by severity. Notification policies assemble information about where, when, and how an alert must be routed. Finally, contact points refer to each alarm type's message template and the medium used to send the message (e.g., E-mail, Discord, Telegram, and others)(MEY, 2022).

Another resource the DP module provides through Grafana at the system prototype is some dashboards to view system operation. **Figure 5.6** shows the first dashboard deployed in the system prototype, which helped in accounting for the experiment results. The dashboard depicts two panels with six bars each, where each panel represents the approach set at the experiment, and each bar represents a sensor of this approach. The ECAS ADR approach and the sensors employed in this approach are in the left panel, and on the right panel are the sensors employed in the LoRaWAN ADR approach at the system evaluation. The y-axis displays the total packets received, and the x-axis displays the sensors, whose legends identify each sensor from one to six to each approach, ECAS ADR or LoRaWAN ADR.

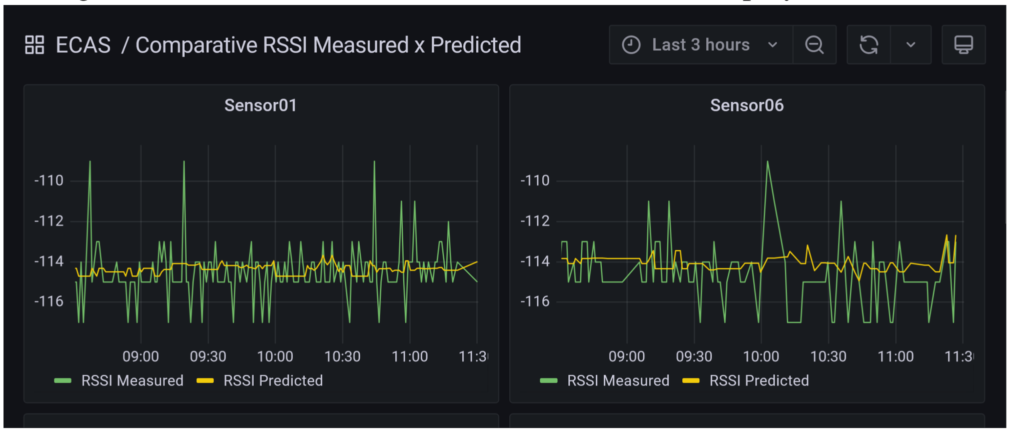
Figure 5.6: Real-time Measured Totalizers Data Pannel Deployed at Grafana.



Source: Author

The second dashboard created in the system prototype, depicted at **Figure 5.7**, aims to accompany the measured RSSI with the predicted RSSI value by the RFR model deployed at the sensors. The left panel of **Figure 5.7** compares measured and predicted RSSI values at Sensor01, and the right panel shows the same comparison at Sensor06. The y-axis depicts the RSSI value, which varies from $-116$ to $-110$ at the two panels, and the x-axis depicts the time, which this case, is shown as the last three hours according to the box in the upper right corner. Lastly, below the x-axis legend, each panel shows the y-axis legend, which identifies the measured, and predicted RSSI values lines.

Figure 5.7: Real-time Measured RSSI Data Pannel Deployed at Grafana.



Source: Author

## 5.2 Prototype Evaluation Method

This section aims to describe the prototype system evaluation method, starting with a description of the environment and conditions for dataset creation. Then, the experiment scenario will be presented, including the distance between sensors and gateway and some topographical information. Lastly, the method and metrics used to evaluate the system prototype will be described.

In order to create a dataset, seven sensors were deployed, six for data production and one to control. The six data production sensors had the same configuration presented at **Subsection 5.1.1**, in contrast to the control sensor, which was attached to an antenna of +5 dBi. As per the definition, data production sensors aim to produce the dataset that the ML algorithm will use to generate the prediction model, differently from the control sensor, which aims to deliver a more significant number of temperature and humidity reads even in unfavorable weather conditions to provide a validation of the reads performed by the other sensors (i.e., a read temperature and humidity to compare with the reads performs by the data production sensors).

The following assumptions and care guided the collection process to produce a more reliable dataset:

1. With aims to ensure one sensor's communication process does not interfere with another; each sensor uses one different DR from another at a time, and each sensor uses one different channel from another during all collection time. Except for the control sensor, which uses only the DR-2; however, it also uses a different channel from the other sensors.

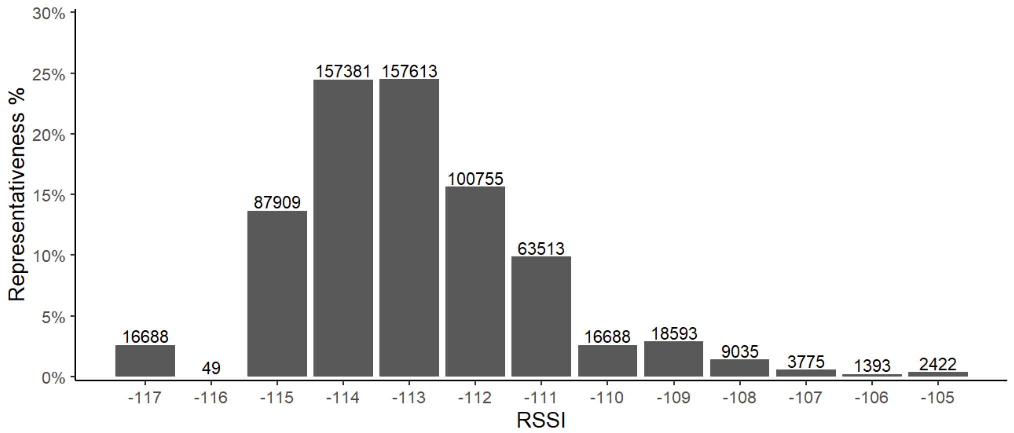
2. In order to reduce a possible bias due to the communication module building quality of one sensor to another, the DR of the six data production sensors was alternated periodically every 24 hours (e.g., the sensor01 started with DR-5, in 24 hours was changed to DR-4, in a more 24 hours to DR-3, and so on until return to DR-5). Thus, each sensor worked with a different DR for 24 hours and returned to its initial DR every seven days.

3. Since the sensors would be exposed to the weather, each sensor was conditioned in a plastic container with the antenna and DHT22 sensor attached to the outside, aiming to collect the environment's temperature and humidity measures.

The period and frequency of collections were as follows: The dataset collection started on 11 October, 2022, and stopped on 27 February, 2023 (i.e., it began in the spring and ended in the summer of Brazil), totaling around 140 days of collection. During this period, every 45 seconds, each sensor performed a temperature and humidity reading and sent it to the server. At the end of the collection time, the dataset had 643,223 samples summing the readings of the six sensors.

Before describing the experiment scenario, it is necessary to remark on the dataset. Since the sensors performed the collection from a fixed place, the measured RSSI did not show a substantial variation, getting between $-121$ and $-93$. **Figure 5.8** presents the samples total for each measured RSSI in the range of $-117$ to $-105$, totaling 635,814 samples (i.e., 98.85% of samples total); in other words, this means the measures from -121 to -118 and from -104 to -96 represent only 1.15% of the samples total. Hence, to reduce the less representative number of samples and thus avoid noise in the dataset, the measured RSSI $-121$, $-95$, $-94$, and $-93$, whose samples total only 22, were removed from the training dataset.

Figure 5.8: Main measures that compose the dataset and its representativeness of the total samples.

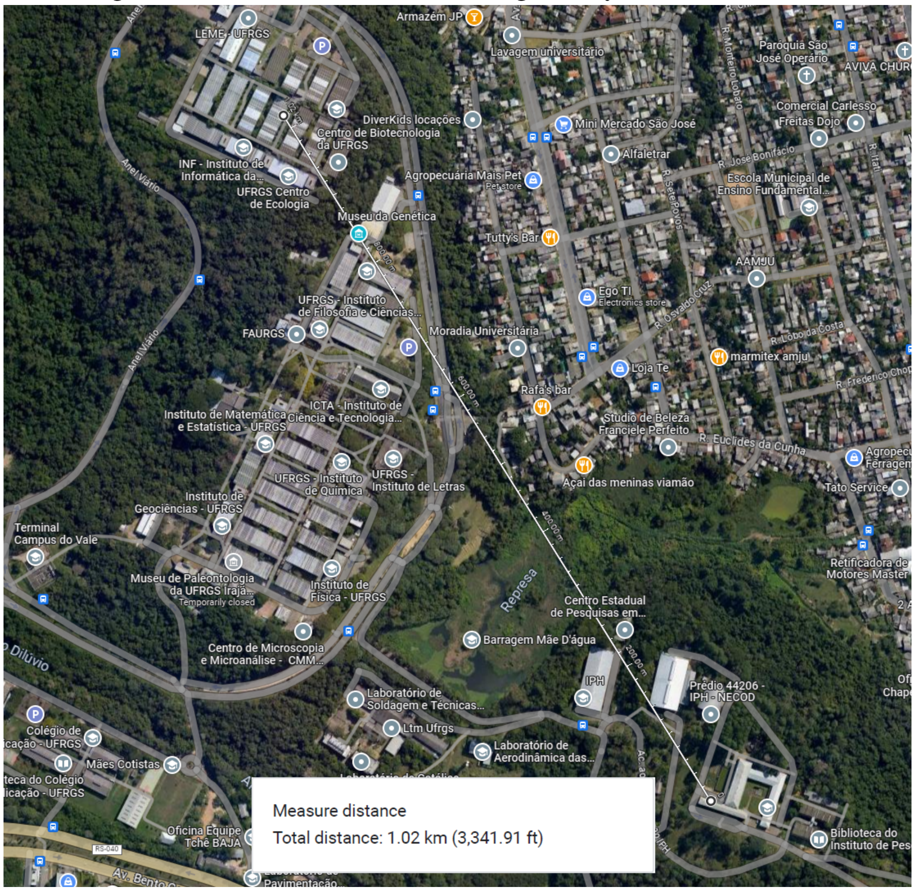


Source: Author

**Figure 5.9** shows the scenario of dataset collection and system prototype evaluation. The white line connecting the two points on the **Figure 5.9** measures the distance from the sensors to the gateway, approximately 1000 meters. The point in the upper left corner represents the gateway and its altitude concerning sea level is 94m. The sensors are in the lower right corner, and their altitude concerning sea level is 74m (i.e., 20 meters below the gateway concerning sea level). The server and the gateway have been placed in a building beside a window with its antenna installed to the outside, and the sensors have been attached to the outside of a window with their antennas directed to the gateway side.

Due to vegetation and buildings between the gateway and the sensors, neither of them is in the other's line of sight.

Figure 5.9: Distance between the gateway and the sensors.


LEME - UFRGS
Armazém JP
Lavagem universitário
Paroquia São José Operário
DiverKids locações
Centro de Biotecnologia da UFRGS
Mini Mercado São José
Comercial Carlesso
INF - Instituto de Informática da...
UFRGS Centro de Ecologia
Agropecuária Mais Pet
Alfaletrar
Escola Municipal de Ensino Fundamental...
Museu da Genética
Tutty's Bar
AAMJU
UFRGS - Instituto de Filosofia e Ciências...
FAURGS
Ego TI
Moradia Universitária
marmitex amju
Loja Te
Rafa's bar
ICTA - Instituto de Ciência e Tecnologia...
Instituto de Matemática e Estatística - UFRGS
Studio de Beleza Franciele Perfeito
Açai das meninas viamão
UFRGS - Instituto de Química
UFRGS - Instituto de Letras
Instituto de Geociências - UFRGS
Terminal Campus do Vale
Tato Service
Museu de Paleontologia da UFRGS Irajá...
Instituto de Física - UFRGS
Represa
Centro Estadual de Pesquisas em...
Retificadora de Motores Master
Centro de Microscopia e Microanálise - CMM...
Barragem Mãe D'água
IPH
Prédio 44206 - IPH - NECOD
Laboratório de Soldagem e Técnicas...
Ltm Ufrgs
Laboratório de Aerodinâmica das...
Colégio de Aplicação - UFRGS
Mães Cotistas
Oficina Equipe Tchê BAJA
Biblioteca do Instituto de Pes...
Av. Bento...

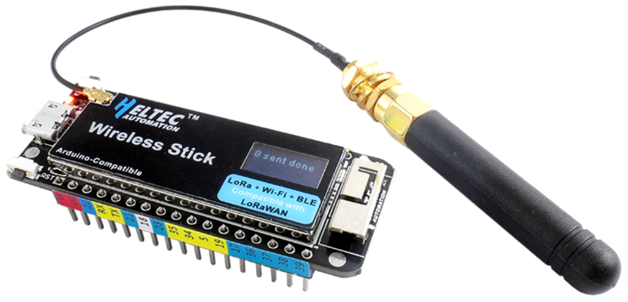
Measure distance
Total distance: 1.02 km (3,341.91 ft)

Source: Google

Although the sensor prototype presented at **Subsection 5.1.1** has fulfilled its role well in collecting data for building the training dataset, in order to evaluate the system prototype, a different sensor was deployed. The objective of implanting a different sensor in the system prototype evaluation is to create a scenario with devices designed and built under industrial quality control. Hence, a similar configuration board was chosen, the Wireless Stick board, depicted at **Figure 5.10**, that is a LoRa development kit ready to use with the following configuration: a microprocessor ESP32, LoRa chip SX1276, 520KB of SRAM, and 4MB of flash memory. Moreover, to evaluate the system prototype, six boards were conditioned in a plastic container with a DHT22 sensor and +2dBi antenna fixed to the outside, similar to the sensor that collected the dataset.

As mentioned in **Chapter 1**, this work proposes a system that aims to improve the network resiliency from environmental data. In a simulated experiment, **Chapter 4** presented a comparison between the fixed DR approach and the Aggressive and Conservative

Figure 5.10: Heltec Wireless Stick.



Source: (HELTEC AUTOMATION, 2019)

adaptive approaches. However, although this experiment only compares the proposed approach and fixed DR, the LoRaWAN protocol has another approach, the LoRaWAN Adaptive Data Rate (ADR). Thus, once validated the hypothesis that it is possible to improve the network resilience from environmental data in a simulated scenario, to evaluate the system prototype in a real scenario performed a comparison between the following approaches: Fixed DR, LoRaWAN ADR, and the proposed approach at this work, which we named ECAS ADR.

In the fixed DR approach, the administrator sets the sensor DR. On the other hand, in the LoRaWAN ADR and ECAS ADR, an algorithm supports the sensor configuration decision-making process. In the LoRaWAN ADR, the sensor configuration process is as follows: the sensor tries to send two times with the DR with higher performance, the DR-5; if the sensor does not receive an acknowledgment (ACK) message in these two tries, it changes the DR to the next DR with minor performance that the DR-5 but higher than the rest DRs, in this case, the DR-4. Thus, the LoRaWAN ADR will reduce the sensor' DR successively every two tries until the most robust DR, with less performance (i.e., LoRaWAN ADR algorithm uses only ACK messages to support its decision-making and adopts a performance reduction strategy to improve signal robustness).

Like the LoRaWAN ADR, the ECAS ADR also uses the ACK messages to support its reconfiguration decision-making process; however, in addition to the ACK messages, the ECAS ADR uses the temperature, humidity, TX power, and DR. Furthermore, in contrast to the LoRaWAN ADR, which adapts just the DR, the ECAS ADR can adapt the sensors' DR and TX power. Lastly, although we believed that the proposed approaches in **Chapter 4** would present a different result, both presented a similar PDR in the simulated experiment. Therefore, an approach was used in the system prototype that combines some characteristics of the Conservative and Aggressive approaches.

The decision-making process in the ECAS ADR occurs as follows: the sensor performs a temperature and humidity measurement, and it executes the prediction of an RSSI value to the next transmission based on measured data, TX power, and DR set at the sensor. Then, to determine the DR, the ECAS ADR verifies the predicted RSSI on a scale from 0 to 5, where 0 represents simultaneously -115 RSSI and DR-0, and 5 represents -96 RSSI, and DR-5 (i.e., a value inside the range of dataset RSSI values combined with the available DRs). In order to determine the TX power, the ECAS ADR performs the same process as to determine the DR; however, rather than scale from 0 to 5, the scale for the TX power goes from 10 to 0, where 10 represents -115 RSSI and maximum TX power value, and 0 represents -96 RSSI and minimum TX power value. Thus, the ECAS ADR adapts the TX Power and DR before performing a send, aiming to increase the probability of delivering the package, similar to the Conservative approach, as well as using the most robust TX Power and DR possible, seeking to obtain the best performance as in the Aggressive approach.

Once described how the ECAS ADR obtains the TX power and DR values, the next step is to describe the reconfiguration transceiver module process. Thus, the reconfiguration process is as follows: the sensor performs a prediction, determines the TX power and DR based on the process above, sets the TX power and DR, and tries to send two times with this configuration. If the sensor does not receive an ACK message after two tries, it reduces one DR, sets TX power to 10, and tries to send one more time. If the sensor does not receive an ACK message, it sets the DR-1 and tries to send it one more time. If the sensor does not receive an ACK message, it sets the DR-0 and tries to send it one more time. After five tries, the sensor discards this packet and starts the process from the beginning.

The LoRa technology has at least six DR combinations, according to depicted at **Table 4.1** and, as mentioned above, to evaluate the system prototype, we had just six sensors. Hence, to create an evaluation fairer scenario, where the environmental factors of one approach were similar to the other, the system prototype evaluation was performed in three rounds, as follows:

- In the first round, three sensors were configured with the LoRaWAN ADR and three with the ECAS ADR. Therefore, the six sensors were exposed to the same environmental factors.

- In the second round, the six sensors were configured with the fixed DR.

- Finally, in the third round, the sensors were configured with the same configuration as the first round (i.e., three with LoRaWAN ADR and three with ECAS ADR).

Although the fixed DR approach evaluation has been done in a different period of the LoRaWAN ADR and ECAS ADR, all the experiment was performed in the range of 12 days, from April 04, 2023, to April 16, 2023 (i.e., the whole experiment occur at the same station, on the Brazilian fall, in the range of 12 days, and during this period there was no significant variation in climate). Lastly, each experiment round had a duration of 96 hours (i.e., only the packets transmitted and received within this interval were counted in the results).

In order to evaluate the system prototype performance, were selected the following metrics: Sent Packets, Received Packets, Packet Delivered Rate (PDR), Packet Retransmission Attempts, Prediction Total Hits, and Mean Square Error (MSE). Below is a summary of each metric used.

- Sent Packets represents the total packets sent by the sensors, and the smaller or closer to the total packets received, the better the performance.

- Received Packets represents the total packets received by the server, and the higher or closer to the total packets sent, the better the performance.

- Packet Delivered Rate (PDR) represents the received total packets by the server in contrast to the send packets total by the sensors in percent scale.

- Packet Retransmission Attempts represents the number of tries a sensor performs to send the same packet (i.e., retransmission number of the same packet) until it receives an ACK message or discards the packet. It is accounted for as follows: before performing a send routine, the sensor verifies if the packet fCnt is the same as the previous packet; if false, the sensor resets the counter; if true, the sensor counts one (i.e., each sends try of the same fCnt represents one attempt).

- Lastly, the Prediction Total Hits and the Mean Square Error (MSE) from the RFR algorithm were used to evaluate the prediction model hit rate. During the model tuning to find the better hit rate, the dataset was split into a training dataframe and a test dataframe, where the test dataframe represented 5% of the dataset. The hit score obtained during the tuning was 30.84%, with a MSE of 3,24. Thus, to assess the accuracy of the system prototype's prediction model, the hit score and MSE were from the deployed prediction model and the predicted values obtained during the experiment.

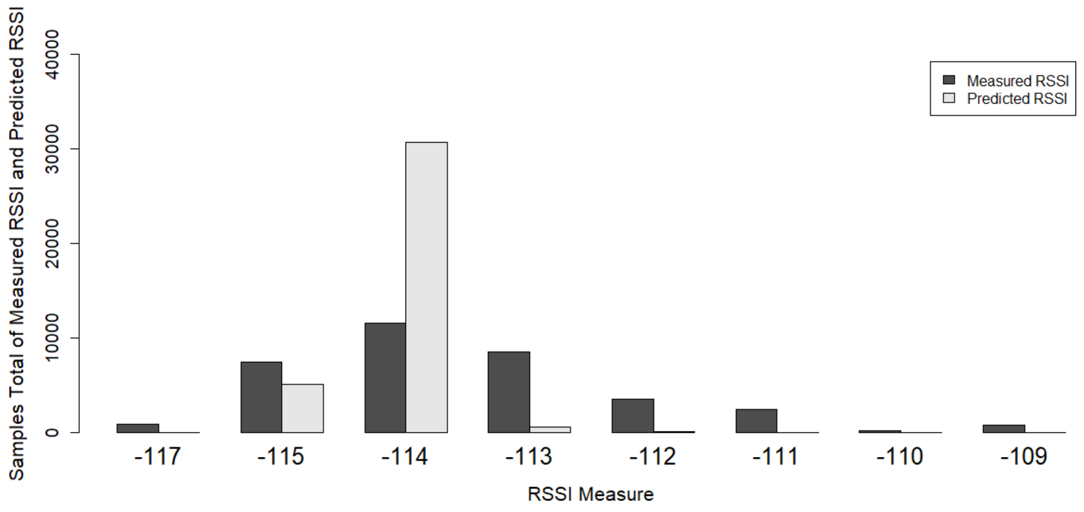
# 6 RESULTS

This chapter presents the experiment results, beginning with a discussion about the prediction model performance and ending with a discussion comparing the three approaches' performance: Fixed DR, LoRaWAN ADR, and ECAS ADR.

Before discussing the prediction model's performance, we have a note about the presented results. The prediction model uses TX Power and DR to predict an RSSI value. Thus, the collected data during the Fixed DR approach was removed to avoid distortions in the prediction model performance results once the Fixed DR sensors always used the same TX Power and DR.

**Figure 6.1** presents the Measured RSSI and Predicted RSSI. The $Y$-axis in **Figure 6.1** represents the Samples Total of Measured RSSI and Predicted RSSI, and the $X$-axis represents the RSSI value. The Measured RSSI varies from $-119$ to $-96$ in the experiment; however, once the Predicted RSSI values vary only from $-115$ to $-112$, the other values have been suppressed from the **Figure 6.1** to a better visualization. Although the training dataset has samples with values from $-121$ to $-93$, the largest volume of samples is between the values $-115$ and $-112$, as presented in **Figure 5.8**. Therefore, the result presented at **Figure 6.1** was expected since the values with the highest number of predictions are contained in the values with the highest number of samples in the training dataset.

Figure 6.1: Distribution of Measured and Predicted RSSI.



Source: Author

In order to analyze the prediction model, one confusion matrix were generated. The **Figure 6.2** $Y$-axis presents the predicted values, and $X$-axis presents the measured values. The highlighted dark gray diagonal presents the point where $Y$-axis and $X$-axis intersect, and the highlighted light gray shows the prediction model total hits. Analyzing the rate hits for each value in the **Figure 6.2**; the model was right at 1,223 from a total of 5,082 to the $-115$ value, 9,578 of 30,720 to the $-114$ value, 141 of 513 to the $-113$ value, and one of eight to the $-112$ value, which represents 24.07%, 31.18%, 27,49%, and 12.5% of correct predictions, respectively. Lastly, the prediction model was right at 10,943 of 36,323, totalizing 30.13% of accuracy.

Figure 6.2: Confusion Matrix to Measured RSSI and Predicted RSSI.

**Predicted Value (Accuracy 30.13%)**

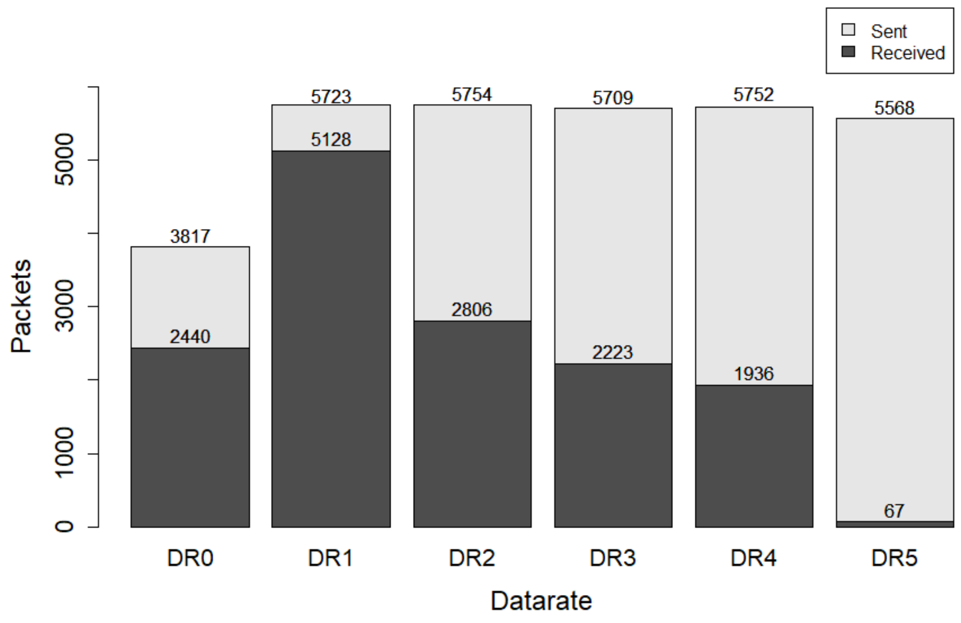
| | **Measured** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Predicted** | -119 | -118 | -117 | -115 | -114 | -113 | -112 | -111 | -110 |
| -119 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -118 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -117 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -115 | 0 | 0 | 142 | 1223 | 1873 | 1090 | 386 | 192 | 8 |
| -114 | 7 | 3 | 708 | 6177 | 9578 | 7240 | 3060 | 2166 | 175 |
| -113 | 0 | 0 | 13 | 62 | 121 | 141 | 77 | 55 | 4 |
| -112 | 0 | 0 | 0 | 1 | 3 | 2 | 1 | 1 | 0 |
| -111 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -110 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Source: Author

**Figure 6.3** presents sent and received packets per DR in the Fixed DR approach, where the $Y$-axis represents the total packets sent and received, and the $X$-axis represents the DR. According to shown at **Figure 6.3**, the DRs from one to five presents a similar sent packets number, approximately 5,700 packets, whereas the DR0 presents a significant reduction, totalize 3,817 sent packets. This behavior was expected since the DR0 has more time in the air than the other DR. Similar to the results obtained in the simulated experiment presented at **Figure 4.4**, the system prototype evaluation results present a received total packets greater in the DR1.

According to presented at **Figure 6.3**, the DR1 delivered 5,128 packets of 5,723, representing a PDR of 89.6%. The DR2 was the second DR that delivered more packets, totaling 2,806 of 5,754, representing the third highest PDR with 48.76%. The third DR that delivered more packets obtained the second better PDR, the DR0, which delivered 2,440 packets (i.e., 366 packets less than DR2) but, in contrast to the other DRs, sent only 3,817 packets, obtaining a PDR of 63.92%. The DR3 delivered 2,223 packets, and the DR4 delivered 1,936, presenting a PDR of 38.95% and 33.65%, respectively. Finally, the DR5 presented the worst PDR, 1.19%, with just 67 delivered packets of 5,568 sent packets.

Figure 6.3: Comparative of Sent and Received Packets per DR with Fixed DR Approach.



Source: Author

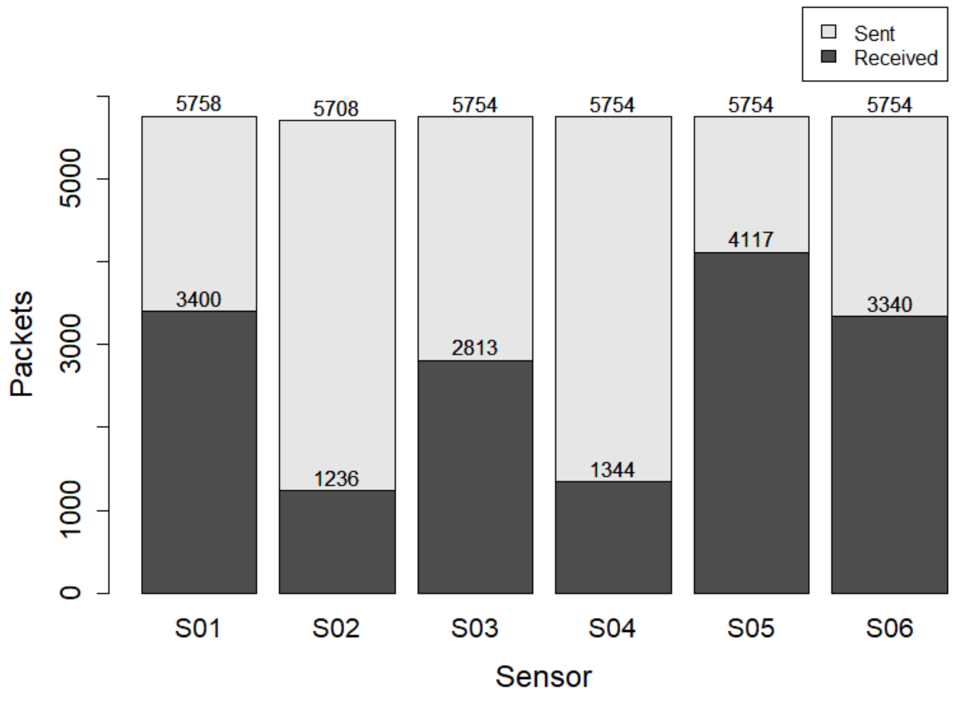
**Figure 6.4** presents the sent and received total packets per sensor with the LoRaWAN ADR approach. The **Figure 6.4** $Y$-axis presents the total packets sent and received, and the $X$-axis presents each sensor, from one to six, deployed at the system prototype evaluation. According to **Figure 6.4**, the LoRaWAN ADR approach sent almost the same packet number per sensor, approximately 5,750. On the other hand, the number of delivered packets presents greater variation.

According to shown the **Figure 6.4**, the sensor that delivered more packets in the LoRaWAN ADR approach, S05, presented a PDR of 71.55%, with 4,117 delivered packets from 5,754 sent packets. After, with 3,400 of 5,758 and 3,340 of 5,754, the sensors S01 and S06, whose PDRs presented are 59.04% and 58.04%, respectively. With a small reduction in the delivered total packets compared to the S01 and S06 came the S03, which obtained a 48.88% PDR, with 2,813 delivered packets of 5,754 sent packets. Finally, as presented at **Figure 6.4**, the sensors with a minor number of delivered packets, S04 and S02, presented a PDR of 23.35% and 21.65%, with 1,344 of 5,754 and 1,236 of 5,708 delivered and sent packets, respectively.

As described in **Section 5.2**, the LoRaWAN ADR executes the adaptation of the DR based on ACK messages. Thus, the sensor starts each sending process with the same DR as the previous communication and executes an adaptation when not receiving an

ACK message. As the results presented in the **Figure 6.3** and **Figure 6.4**, the LoRaWAN adaptive approach can supply a greater PDR compared with the Fixed approach; however, the LoRaWAN approach does not seem to deliver this improvement optimally, this is because, to deliver 1,650 packets more than the Fixed DR, the Lorawan ADR sent 2,159 packets more. Hence, from a performance perspective, the LoRaWAN ADR delivered more packets at a more resource cost.

Figure 6.4: Comparative of Sent and Received Packets per Sensor with LoRaWAN ADR Approach.

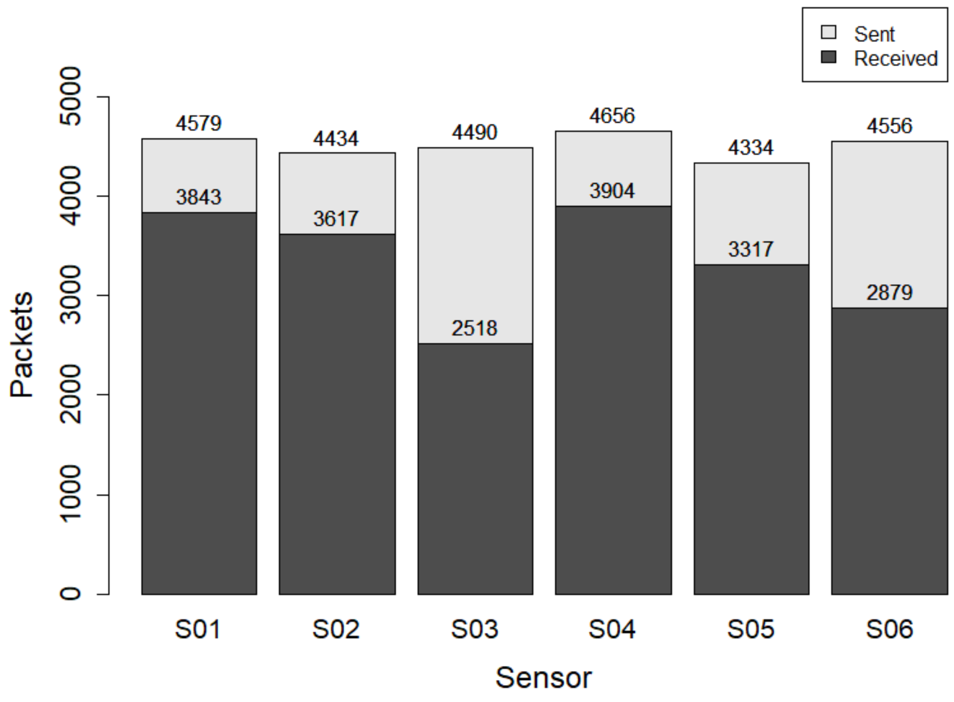


Source: Author

**Figure 6.5** presents the sent and received total packets per sensor to the ECAS ADR approach. The **Figure 6.5** $Y$-axis presents the total packets sent and received, and de $X$-axis presents the sensors deployed at the system prototype evaluation. In contrast to the Fixed DR and LoRaWAN ADR, the ECAS ADR showed a greater variation in the sent packets number per sensor. We believe that the variation in the sent packets number occurs because the ECAS ADR can adapt the TX Power and DR before each send process, thus responding to the sensors' context variations.

The S04, as shown in **Figure 6.5**, delivered 3,904 of 4,656 total packets, being the sensor that presented the highest number of packets delivered in the ECAS ADR approach. Following the S04, the S03 delivered 3,843 of 4,579, and the S02 delivered 3,617 of 4,434 packets. These three sensors presented the major PDR per sensor in the ECAS ADR approach, and although the S04 has delivered more packets than others, it presented the second-highest PDR. Thus, the S01 presented the highest PDR with 83.93%, followed by the S04 with 83.85% and the S02 with 81.57%. The S05, as shown at **Figure 6.5**, delivered 3,317 of 4,334 packets, presenting a PDR of 76.53%. The S06 and S03 delivered the least number of packages from the six sensors, being 2,879 of 4,556 by the S06 and 2,518 of 4,490 by the S03, presenting a PDR of 63.19% and 56.08%, respectively.

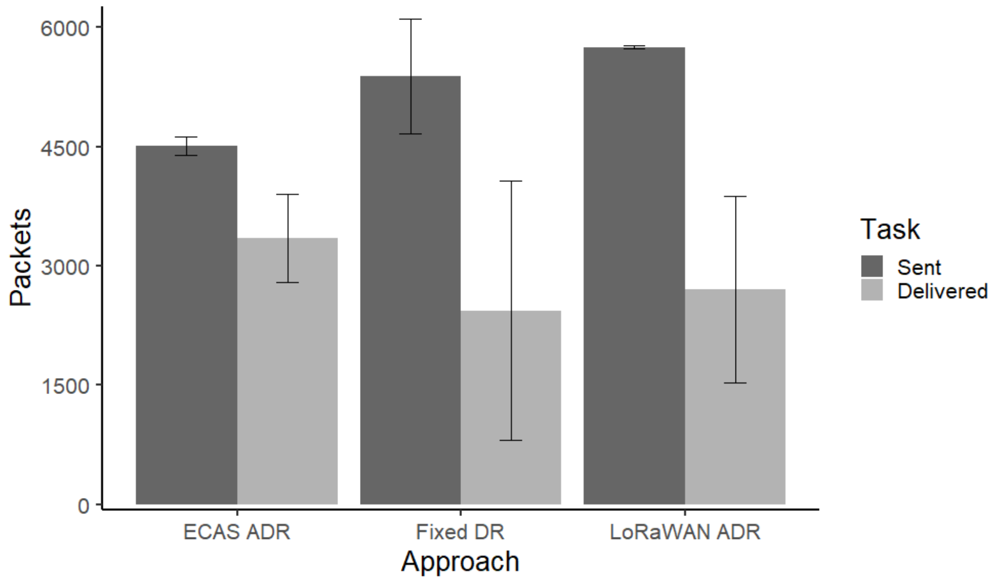
Figure 6.5: Comparative of Sent and Received Packets per Sensor with ECAS ADR Approach.



Source: Author

The **Figure 6.6** shows the mean of sent and delivered packets per approach. The **Figure 6.6** $Y$-axis presents the packet rate, and the $X$-axis presents the approaches, ECAS ADR, Fixed DR, and LoRaWAN ADR. As shown in **Figure 6.6**, the ECAS ADR approach sent, on average, 4,508 packets with a Standard Deviation (SD) of 114 and delivered, on average, 3,346 with a SD of 554. The Fixed DR presents a sent and received packet mean of 5,387 with a SD of 722 and 2,433 with a SD of 1,630, respectively. Finally, the LoRaWAN ADR presents the greater of sent packets with a lower SD, 5,747 and 19, respectively. However, just like the Fixed DR, the LoRaWAN ADR presents an average delivered rate of less than half of the sent total packets, being 2,708 with a SD of 1,174. Based on results presented at **Figure 6.6**, we concluded that the ECAS ADR approach offers a more stable average delivery rate, and although the send rate presents a greater variation than the LoRaWAN ADR approach, the ECAS approach provides a better approximation between the average packet sending and receiving rate.

Figure 6.6: Average Rate of Packets Sent and Received per Approach with Standard Deviation.



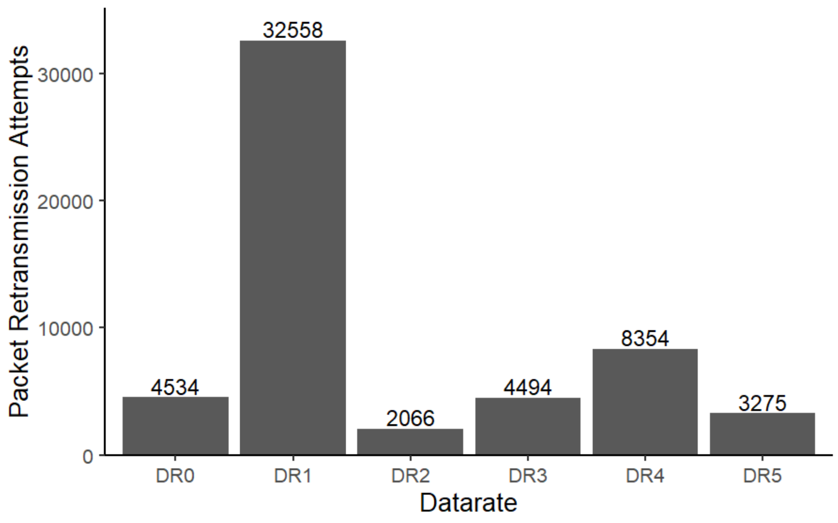
Source: Author

In order to evaluate the approaches' efficiency, we count the attempts number a sensor performs to retransmit a packet. As described at **Subsection 5.1.2**, for each attempt to send a packet with the same fCnt, we count as one attempt. Thus, we can compare the number of attempts for each delivered packet and estimate the approach efficiency. Lastly, we presented the results per sensor instead of by average due to some outliers in the send attempts count.

**Figure 6.7** presents the send attempts total per DR to the Fixed DR approach, where the $Y$-axis represents the packet retransmission attempts total, and the $X$-axis represents the DR. As presented in **Figure 6.7**, the DR1 presents a most significant send attempts number, totaling 32,558 send attempts. In contrast, on average, the other five DRs present approximately 4,500 send attempts. This result could represent that the sensor with the DR1 had a problem. However, as shown in **Figure 6.3**, the sensor with DR1 delivered the most significant amount of packets in the Fixed DR approach. We believe that this sensor did not receive an ACK message in the first send, and according to the LoRaWAN specification, the ACK messages are only sent to the latest message and never retransmitted (LORA ALLIANCE®, 2016). Thus, this sensor continuously sent the same packet until it discards it because it did not receive an ACK message, resulting in a high send attempts number. The DR5 presents another result that we consider an outlier in the Fixed DR approach because the send attempt total to its DR was 3,275 of 67 delivered packets (i.e., to each packet, the sensor performed 49 send tries). Therefore, although the

DR5 presents a value closer to the DRs zero, two, three, and four than the DR1, the DR5 presents the major number of attempts per packet. Lastly, the DRs zero, two, three, and four presented a retransmission number per delivered packet between one and four.

Figure 6.7: Comparative of Packet Retransmission Attempts per Sensor with Fixed DR Approach.

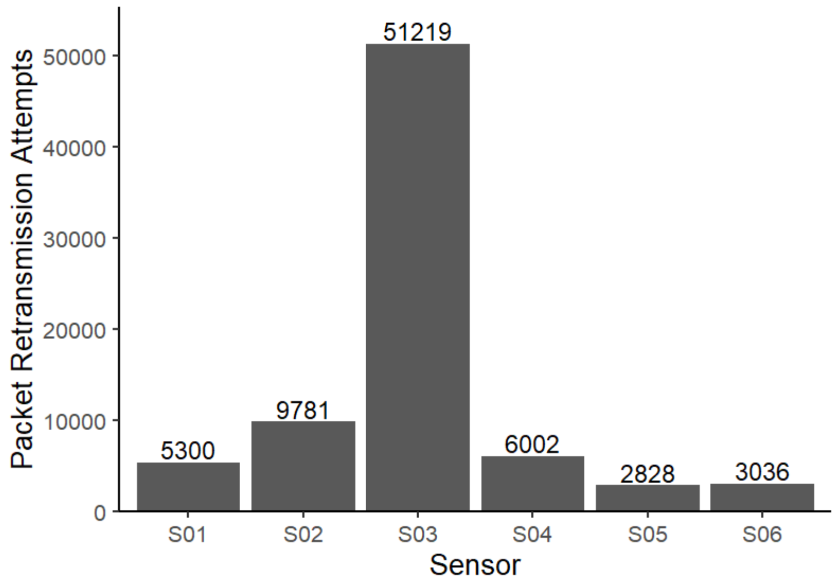


Source: Author

**Figure 6.8** presents the send attempts total to the LoRaWAN ADR approach. The **Figure 6.8** $Y$-axis represents the packet retransmission attempts total, and the $X$-axis represents the sensors deployed in the system prototype evaluation to the LoRaWAN ADR approach. Just like the Fixed DR, the LoRaWAN ADR presents two sensors that we consider outliers, the S03 and the S02. The S03 accounted for 51,219 send attempts, representing, on average, 19 tries to send one packet (i.e., the first try representing zero, more additional 18 tries to the same packet). The second sensor we consider an outlier, the S02, accounted for 9,789 retransmission attempts for 1,236 delivered packets, representing, on average, eight retransmission tries per packet. Finally, sensors one, four, five, and six present a retransmission total per delivered packet between one and four.

**Figure 6.9** presents the total retransmission attempts per sensor to the ECAS ADR approach. The **Figure 6.9** $Y$-axis represents the packet retransmission attempts total, and the $X$-axis represents the sensors deployed in the ECAS ADR approach evaluation. Like the other approaches, the ECAS ADR presents two sensors we consider outliers, S01 and S05. The S01, which presented the major retransmission attempts number, presents a total of 20,896, representing, on average, five retransmissions per delivered packet. In contrast to the Fixed DR and LoRaWAN ADR, five attempts may seem few; however, the sensor that presented more retransmissions per delivered packet in the ECAS ADR was the S05,

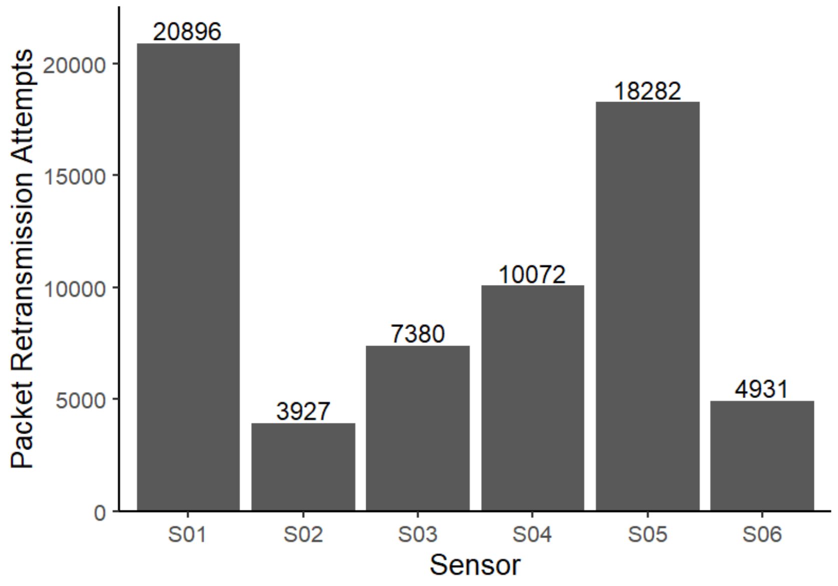
Figure 6.8: Comparative of Packet Retransmission Attempts per Sensor with LoRaWAN ADR Approach.



Source: Author

which presented, on average, six retransmissions per packet. Therefore, for the average retransmission number of the other four sensors, which stayed between one and three for each delivered packet, we consider the six retransmission tries by the S05 and the five retransmission tries by the S01 as outliers.

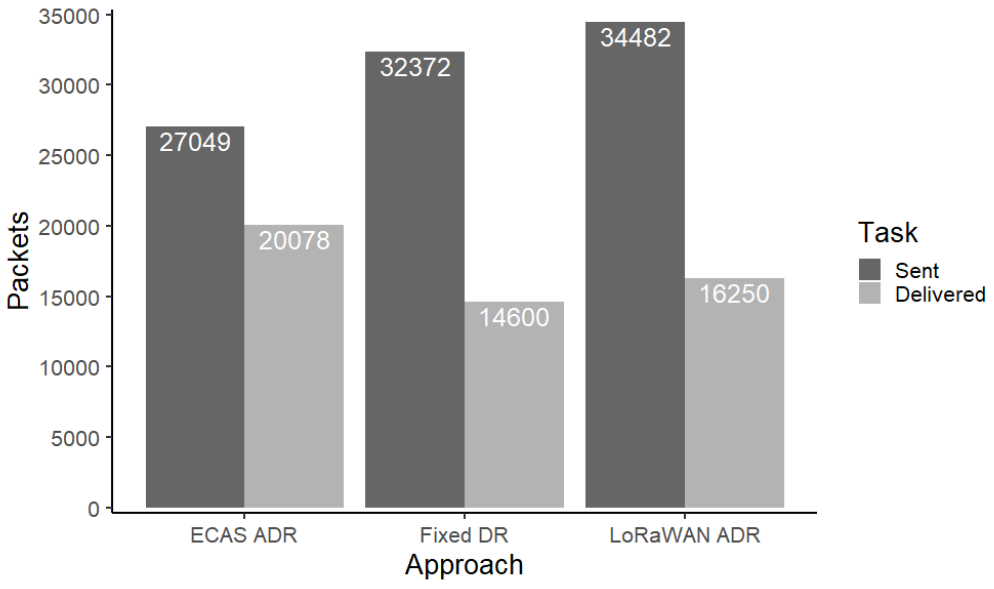
Figure 6.9: Comparative of Packet Retransmission Attempts per Sensor with ECAS ADR Approach.



Source: Author

**Figure 6.10** presents the sent and delivered packets total per approach, where the $Y$-axis represents the packets' total, and the $X$-axis represents the approach. According to **Figure 6.10**, the ECAS ADR sent 27,049 packets total and delivered 20,078 packets. Out of 32,372 packets, the Fixed DR approach delivered 14,600 packets to the server. Lastly, as **Figure 6.10**, the LoRaWAN ADR approach sent 34,482 packets to the server and delivered 16,250 packets.

Figure 6.10: Comparative of Sent and Received Packets between Approaches.



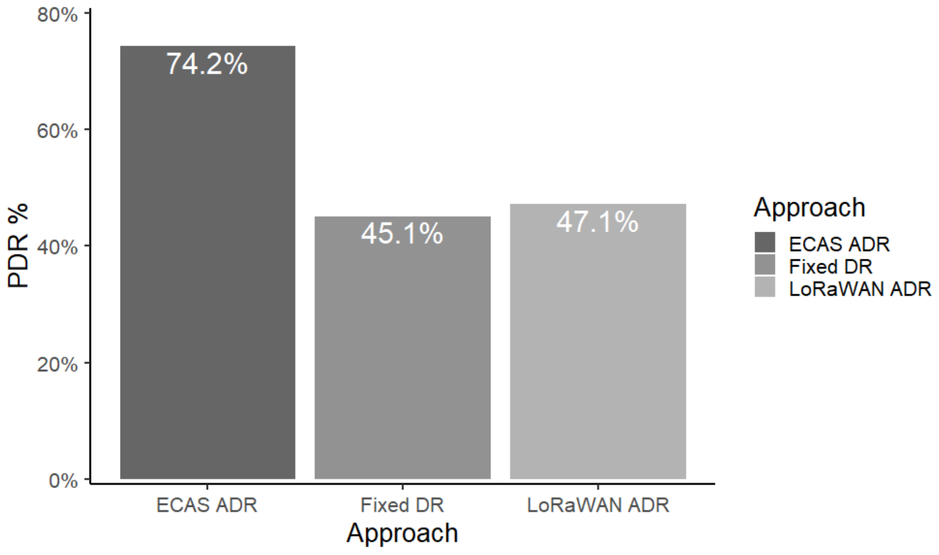
Source: Author

**Figure 6.11** presents the percentile score between sent and delivered packets per approach. The **Figure 6.11** $Y$-axis presents the PDR, and the $X$-axis presents the approaches. As presented at **Figure 6.11**, The ECAS ADR presents the highest PDR, 74.2%. With 47.1%, the LoRaWAN ADR presents the second-highest PDR, followed by the Fixed DR with 45.1%.

Although the ECAS ADR presents the smaller number of sent packets between the three approaches, it presents the highest delivered rate. In contrast, although the Fixed DR and LoRaWAN ADR approaches sent more than double the received packets, both had a lower PDR than the ECAS ADR approach. Therefore, based on results presented at **Figure 6.10** and **Figure 6.11**, we can verify that sending an enormous volume of packets does not guarantee a higher delivery volume and still congests the means of communication, being able to reduce the system performance.

In the context where we tested the three approaches, the PDR presented by the Fixed DR and LoRaWAN ADR are very similar. However, as shown at **Figure 6.6**, we concluded that although the Fixed DR has delivered a significant packet number, its performance varies greatly from one DR to another. Furthermore, this experiment was per-

Figure 6.11: Percentile Score of Packet Delivered Rate per Approach.



Source: Author

formed in a reduced and controlled environment. Thus, we can not conclude that the Fixed DR would perform similarly in a large-scale environment.

On the other hand, although more stable than the Fixed DR in this experiment, as presented at **Figure 6.6**, the LoRaWAN ADR presented a significant variability in the delivered packets number compared to the ECAS ADR. The variability reduction in the delivered packets number in the ECAS ADR is due to its capacity to reconfigure the transceiver module before each sends packet process, in contrast to the LoRaWAN ADR that uses the last send setting to perform the next send and performs adaptations if not receive ACK messages.

# 7 CONCLUSION AND FUTURE WORK

The unpredictability of the battlefield context makes it necessary to implement systems that improve the decision-making process time-effectively. The present work proposed a context-aware monitoring system to adapt network configuration parameters to changes in the climatic factors observed on the battlefield. A simulation and experimental were developed as proof-of-concept to demonstrate how the proposed approach performs under varied weather conditions.

The simulation results showed that the dynamic adaptation of network parameters can make the communication process more efficient. In contrast with the Fixed DR approach, the adaptive approach achieved an 11% increase in the total packets delivered while simultaneously reducing the total packets sent by 12%. From these results, using the context-aware adaptive approach can make the communication process more efficient, presenting the potential to optimize the available resources.

In order to evaluate the proposed system in a real context, a prototype was created. The system prototype evaluation results present a similar behavior to the simulation results, demonstrating that improving the communication process based on environmental context information is possible. The results present a reduction in the receive packet number variability, proving more stable than the other approaches. For real-time systems, instability and unpredictability in the communication process represent significant challenges since the objective of these systems is the search for determinism. Thus, the work results demonstrate that environmental information can help anticipate the network behavior and provide inputs to improve the network configuration decision-making process.

Knowing that the physical medium reliability is another challenge to improving the communication process, it is necessary to take precautions for the system not to misuse it by flooding it with retransmissions. In this aspect, the system prototype results demonstrate that the proposed approach occupies the physical medium for a shorter time during the communication process in contrast to the other approaches. On average, including or not the outliers discussed in the **Chapter 6**, the proposed approach performed the minor packet retransmission number during the experiment.

Finally, like on the battlefield, energy is a limited resource in other environments, such as sensor networks (PATEL; WON, 2017). The proposed system, as presented at **Figure 6.11**, presented a packet delivered rate quite significant in contrast to the other approaches, reaching almost a quartile higher than the second highest rate. In addition to the highest PDR, as discussed in **Chapter 6**, the proposed approach presented fewer packet retransmission attempts than the other approaches. Therefore, although we did not monitor energy consumption in the system prototype evaluation, based on these results, we can infer that the proposed system consumes less energy to send and receive packets than other approaches.

Several works report the issue of interference caused by environmental factors in wireless network technologies (ANASTASI et al., 2004)(FEDERICI; MA; MOELLER, 2016)(RANGARAJAN; BASKARAN, 2015)(BRUZGIENE et al., 2020). Although the LoRa technology was used to develop a proof-of-concept in this work, the proposed system is technology agnostic. Therefore, in addition to the feasibility of implementing ECAS in an existing sensor network, it is possible to use the system to assist in configuring the parameters of various wireless network technologies.

In the system prototype, we used the prediction model to predict an RSSI and, based on it, set a TX Power and DR that provide the most proximity between the predicted RSSI and the Measured RSSI. This approach complies with the Aggressive Approach presented at **Section 4.1**. However, we could use the prediction model to provide TX Power and DR values and, based on these values, predict an RSSI and verify if the predicted RSSI is satisfactory to guarantee that the packet will arrive at the server in the following send.

Due to financial restrictions, the system prototype evaluation scenario was deployed with minimal sensors to simultaneously evaluate the six DRs LoRa under the same environmental conditions. According to results presented at **Chapter 6**, the Fixed DR and LoRaWAN ADR presented a similar result when evaluated from a PDR perspective. However, in a large-scale scenario, the result can be different. Therefore, creating a broad, scalable, and more complex scenario is necessary to perform a performance evaluation of the proposed approach with other approaches.

As future work, the plan is to improve the prediction model to predict values for TX Power and DR, aiming to implement the conservative approach proposed in **Section 4.1**. The work presented by (AIMI et al., 2023) proposes an emulation tool based on *ns-3* (RILEY; HENDERSON, 2010), a simulator used on the experiment presented at **Chapter 4**, to help in evaluate LoRaWAN systems in a large-scale scenario, which is another factor that must be evaluated in the proposed system. Finally, to evaluate the system performance from an energy consumption view, we plan to deploy some battery-powered sensors to measure the power consumption.

## REFERENCES

AIMI, A. et al. Elora: End-to-end emulation of massive iot lorawan infrastructures. In: **2023 IEEE/IFIP Network Operations and Management Symposium (NOMS)**. [S.l.: s.n.], 2023.

ANASTASI, G. et al. Performance measurements of motes sensor networks. In: **Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems**. [S.l.: s.n.], 2004. p. 174–181.

AOSONG ELECTRONICS. **Digital-output relative humidity & temperature sensormodule DHT22 Datasheet**. [S.l.], 2017. Available from Internet: <https://files.seeedstudio.com/wiki/Grove-Temperature_and_Humidity_Sensor_Pro/res/AM2302-EN.pdf>.

AZ-DELIVERY. **D1 R32 ESP32**. [S.l.], 2019. Available from Internet: <https://www.halloweenfreak.de/arduino/pdfs/D1_R32_ENG.pdf>.

AZZAWI, M. A.; HASSAN, R.; BAKAR, K. A. A. A review on internet of things (iot) in healthcare. **International Journal of Applied Engineering Research**, v. 11, n. 20, p. 10216–10221, 2016.

BEELAN. **Device library for LoRaWAN network US, EU and AS. Support SX1276/72 or RFM95**. 2022. Available from Internet: <https://github.com/ElectronicCats/Beelan-LoRaWAN.git>.

BLANCHON, B. **A simple and efficient JSON library for embedded C++.** 2023. Available from Internet: <https://github.com/bblanchon/ArduinoJson.git>.

BOANO, C. A. et al. Low-power radio communication in industrial outdoor deployments: The impact of weather conditions and atex-compliance. In: SPRINGER. **International Conference on Sensor Applications, Experimentation and Logistics**. [S.l.], 2009. p. 159–176.

BOANO, C. A.; CATTANI, M.; RÖMER, K. Impact of temperature variations on the reliability of lora. In: SCITEPRESS-SCIENCE AND TECHNOLOGY PUBLICATIONS, LDA. **Proceedings of the 7th International Conference on Sensor Networks**. [S.l.], 2018. p. 39–50.

BOR, M.; VIDLER, J.; ROEDIG, U. Lora for the internet of things. In: **Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks**. USA: Junction Publishing, 2016. (EWSN ’16), p. 361–366. ISBN 9780994988607.

BRUZGIENE, R. et al. Quality-driven schemes enhancing resilience of wireless networks under weather disruptions. In: **Guide to Disaster-Resilient Communication Networks**. [S.l.]: Springer, 2020. p. 299–326.

BUENO, F. R. et al. **Cygnus.** 2022. Available from Internet: <https://github.com/telefonicaid/fiware-cygnus.git>.

BUITINCK, L. et al. API design for machine learning software: experiences from the scikit-learn project. In: **ECML PKDD Workshop: Languages for Data Mining and Machine Learning**. [S.l.: s.n.], 2013. p. 108–122.

CALVO, D. et al. **FIWARE IoT Agent for the LoRaWaN Protocol.** 2022. Available from Internet: <https://github.com/Atos-Research-and-Innovation/IoTagent-LoRaWAN.git>.

CASTIGLIONE, A. et al. Context aware ubiquitous biometrics in edge of military things. **IEEE Cloud Computing**, IEEE, v. 4, n. 6, p. 16–20, 2017.

CATS, E. **CayenneLPP Arduino Library - Compatible with Cayenne Low Power Payload.** 2021. Available from Internet: <https://github.com/ElectronicCats/CayenneLPP.git>.

CETINKAYA, E. K.; STERBENZ, J. P. A taxonomy of network challenges. In: IEEE. **2013 9th International Conference on the Design of Reliable Communication Networks (DRCN)**. [S.l.], 2013. p. 322–330.

CHAKRABORTY, M.; KUNDAN, A. P. Grafana. In: **Monitoring Cloud-Native Applications: Lead Agile Operations Confidently Using Open Source Software**. [S.l.]: Springer, 2021. p. 187–240.

CROWCROFT, J.; WOLISZ, A.; SATHIASEELAN, A. Towards an Affordable Internet Access for Everyone: The Quest for Enabling Universal Service Commitment (Dagstuhl Seminar 14471). **Dagstuhl Reports**, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, v. 4, n. 11, p. 78–137, 2015. ISSN 2192-5283. Available from Internet: <http://drops.dagstuhl.de/opus/volltexte/2015/4971>.

DOBRESCU, R.; MEREZEANU, D.; MOCANU, S. Context-aware control and monitoring system with iot and cloud support. **Computers and Electronics in Agriculture**, v. 160, p. 91–99, 2019. ISSN 0168-1699. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0168169918315862>.

DOCKER INC. **What is a Container?** [S.l.], 2023. Available from Internet: <https://docs.docker.com/get-started/>.

DONG, S.; ABBAS, K.; JAIN, R. A survey on distributed denial of service (ddos) attacks in sdn and cloud computing environments. **IEEE Access**, IEEE, v. 7, p. 80813–80828, 2019.

ECLIPSE MOSQUITTO™. **An open source MQTT broker.** [S.l.], 2022. Available from Internet: <https://mosquitto.org/>.

FEDERICI, J. F.; MA, J.; MOELLER, L. Review of weather impact on outdoor terahertz wireless communication links. **Nano Communication Networks**, Elsevier, v. 10, p. 13–26, 2016.

FIWARE. **About-Us**. 2021. Available from Internet: <https://www.fiware.org/about-us/>.

FIWARE. **Developers**. 2021. Available from Internet: <https://www.fiware.org/developers/>.

FOUNDATIONS, C. **Protecting America's Infrastructures: The Report of the President's Commission on Critical Infrastructure Protection**. [S.l.]: Washington, DC: The President's Commission on Critical Infrastructure Protection, 1997.

GONZÁLEZ, L. A. Q. et al. Bungee: An adaptive pushback mechanism for ddos detection and mitigation in p4 data planes. In: IEEE. **2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)**. [S.l.], 2021. p. 393–401.

HELTEC AUTOMATION. **Wireless Stick**. [S.l.], 2019. Available from Internet: <https://heltec.org/project/wireless-stick/>.

HOPERF ELETRONIC. **RFM95W transceiver 868/915 MHz Datasheet**. [S.l.], 2019. Rev. 2.

JALAIAN, B. et al. Evaluating lorawan-based iot devices for the tactical military environment. In: IEEE. **2018 IEEE 4th World Forum on Internet of Things (WF-IoT)**. [S.l.], 2018. p. 124–128.

KAMIENSKI, C. et al. Application development for the internet of things: A context-aware mixed criticality systems development platform. **Computer Communications**, Elsevier, v. 104, p. 1–16, 2017.

KHUTSOANE, O. et al. Watergrid-sense: A lora-based sensor node for industrial iot applications. **IEEE Sensors Journal**, IEEE, v. 20, n. 5, p. 2722–2729, 2019.

KUBISCH, M. et al. Distributed algorithms for transmission power control in wireless sensor networks. In: IEEE. **2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003.** [S.l.], 2003. v. 1, p. 558–563.

LEAL, G. M. et al. Empowering command and control through a combination of information-centric networking and software defined networking. **IEEE Communications Magazine**, v. 57, n. 8, p. 48–55, 2019.

LEONARDI, L.; BATTAGLIA, F.; BELLO, L. L. Rt-lora: A medium access strategy to support real-time flows over lora-based networks for industrial iot applications. **IEEE Internet of Things Journal**, IEEE, v. 6, n. 6, p. 10812–10823, 2019.

LIN, K. et al. Emotion-aware system design for the battlefield environment. **Information Fusion**, Elsevier, v. 47, p. 102–110, 2019.

LIN, S. et al. Atpc: adaptive transmission power control for wireless sensor networks. **ACM Transactions on Sensor Networks (TOSN)**, ACM New York, NY, USA, v. 12, n. 1, p. 1–31, 2016.

LORA ALLIANCE®. **LoRaWAN® Specification v1.0.2**. [S.l.], 2016.

LORA ALLIANCE®. **LoRaWAN Backend Interfaces Specification**. [S.l.], 2020. Available from Internet: <https://hz137b.p3cdn1.secureserver.net/wp-content/uploads/2020/11/TS002-1.1.0_LoRaWAN_Backend_Interfaces.pdf?time=1681361080>.

LUOMALA, J.; HAKALA, I. Effects of temperature and humidity on radio signal strength in outdoor wireless sensor networks. In: IEEE. **2015 Federated Conference on Computer Science and Information Systems (FedCSIS)**. [S.l.], 2015. p. 1247–1255.

MARFIEVICI, R. et al. How environmental factors impact outdoor wireless sensor networks: a case study. In: IEEE. **2013 IEEE 10th international conference on mobile ad-hoc and sensor systems**. [S.l.], 2013. p. 565–573.

MARIADB FOUNDATION. **From the MariaDB Knowledge Base.** [S.l.], 2022. Available from Internet: <https://mariadb.org/wp-content/uploads/2023/03/MariaDBServerKnowledgeBase.pdf>.

MEY, G. D. **Alerting in Grafana.** 2022. Available from Internet: <https://grafana.com/docs/grafana/latest/alerting/?src=grafana_footer>.

MISHRA, M. et al. A context-driven framework for proactive decision support with applications. **IEEE Access**, IEEE, v. 5, p. 12475–12495, 2017.

MOON, Y.-W.; JUNG, H.-S.; JEONG, C.-S. Context-awareness in battlefield using ubiquitous computing: Network centric warfare. In: IEEE. **2010 10th IEEE International Conference on Computer and Information Technology**. [S.l.], 2010. p. 2873–2877.

MYDEVICES. **Cayenne Low Power Payload**. [S.l.], 2017. Available from Internet: <https://docs.mydevices.com/docs/lorawan/cayenne-lpp>.

NARAYANASWAMY, S. et al. Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the compow protocol. In: FLORENCE, ITALY. **European wireless conference**. [S.l.], 2002. v. 2002, p. 156–162.

OPENJS FOUNDATION AND NODE-RED CONTRIBUTORS. **Node-RED - Low-code programming for event-driven applications.** [S.l.], 2023. Available from Internet: <https://nodered.org/>.

PAPAKOSTAS, D. et al. Energy-aware backbone formation in military multilayer ad hoc networks. **Ad Hoc Networks**, Elsevier, v. 81, p. 17–44, 2018.

PARK, S.-J.; SIVAKUMAR, R. Quantitative analysis of transmission power control in wireless ad-hoc networks. In: IEEE. **Proceedings. International Conference on Parallel Processing Workshop**. [S.l.], 2002. p. 56–63.

PATEL, D.; WON, M. Experimental study on low power wide area networks (lpwan) for mobile internet of things. In: IEEE. **2017 IEEE 85th Vehicular Technology Conference (VTC Spring)**. [S.l.], 2017. p. 1–5.

POULARAKIS, K. et al. Flexible sdn control in tactical ad hoc networks. **Ad Hoc Networks**, Elsevier, v. 85, p. 71–80, 2019.

RAK, J. A new approach to design of weather disruption-tolerant wireless mesh networks. **Telecommunication Systems**, Springer, v. 61, n. 2, p. 311–323, 2016.

RANGARAJAN, J.; BASKARAN, K. Evaluating the impact of weather condition on manet routing protocols. **International Journal on Electrical Engineering and Informatics**, School of Electrical Engineering and Informatics, Bandung Institute of . . . , v. 7, n. 3, p. 454, 2015.

RASPBERRY PI TRADING LTD. **Raspberry Pi 4 Computer Model B**. [S.l.], 2021. Available from Internet: <https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf>.

RILEY, G. F.; HENDERSON, T. R. The ns-3 network simulator. **Modeling and tools for network simulation**, Springer, p. 15–34, 2010.

RUSSELL, S.; NORVIG, P. The theory of learning. In: **Artificial intelligence: A Modern Approach.** [S.l.]: Pearson, 2020. p. 672–713.

SCHILIT, B.; ADAMS, N.; WANT, R. Context-aware computing applications. In: IEEE. **1994 First Workshop on Mobile Computing Systems and Applications**. [S.l.], 1994. p. 85–90.

SEMTECH. **SX1276/77/78/79 - 137MHz to 1020 MHz Low Power Long Range Transceiver Datasheet**. [S.l.], 2020. Rev. 7.

SINHA, R. S.; WEI, Y.; HWANG, S.-H. A survey on lpwa technology: Lora and nb-iot. **Ict Express**, Elsevier, v. 3, n. 1, p. 14–21, 2017.

STERBENZ, J. P. et al. Evaluation of network resilience, survivability, and disruption tolerance: analysis, topology generation, simulation, and experimentation. **Telecommunication systems**, Springer, v. 52, n. 2, p. 705–736, 2013.

STERBENZ, J. P. et al. Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines. **Computer networks**, Elsevier, v. 54, n. 8, p. 1245–1265, 2010.

SUN, S. Nyusim user manual. **New York University and NYU WIRELESS**, 2017.

SURIYACHAI, P.; BROWN, J.; ROEDIG, U. Time-critical data delivery in wireless sensor networks. In: SPRINGER. **International Conference on Distributed Computing in Sensor Systems**. [S.l.], 2010. p. 216–229.

XIE, A. et al. Designing a disaster-resilient network with software defined networking. In: IEEE. **2014 IEEE 22nd International Symposium of Quality of Service (IWQoS)**. [S.l.], 2014. p. 135–140.

XIE, L. et al. Towards resilient networks using programmable networking technologies. In: SPRINGER. **Active and Programmable Networks: IFIP TC6 7th International Working Conference, IWAN 2005, Sophia Antipolis, France, November 21-23, 2005. Revised Papers 7**. [S.l.], 2009. p. 83–95.

XUE, F.; KUMAR, P. R. The number of neighbors needed for connectivity of wireless networks. **Wireless networks**, Springer, v. 10, n. 2, p. 169–181, 2004.

ZACARIAS, I. et al. Combining software-defined and delay-tolerant approaches in last-mile tactical edge networking. **IEEE Communications Magazine**, v. 55, n. 10, p. 22–29, 2017.

ZHENG, D. E.; CARTER, W. A. **Leveraging the internet of things for a more efficient and effective military**. [S.l.]: Rowman & Littlefield, 2015.

ZIBETTI, G. R. **Context-aware environment monitoring to support LPWAN-based.** 2023. Available from Internet: <https://github.com/Guilherme-Zibetti/ECAS.git>.

ZIBETTI, G. R.; WICKBOLDT, J. A.; FREITAS, E. P. de. Context-aware environment monitoring to support lpwan-based battlefield applications. **Computer Communications**, Elsevier, v. 189, p. 18–27, 2022.

# APPENDIX A — RESUMO EXPANDIDO

As redes desempenham um papel essencial em várias áreas, sendo consideradas um recurso fundamental na atual era digital(CROWCROFT; WOLISZ; SATHIASEELAN, 2015). Além disso, as redes surgem como facilitadoras no desenvolvimento de aplicações em diversos campos, como agricultura(DOBRESCU; MEREZEANU; MOCANU, 2019), cidades inteligentes(KAMIENSKI et al., 2017) e sistemas de Comando, Controle, Comunicações, Computadores, Inteligência, Vigilância e Reconhecimento (C4ISR)(ZHENG; CARTER, 2015). Dessa forma, a Internet se tornou uma infraestrutura crítica na qual quase todos os aspectos de nossas vidas dependem(FOUNDATIONS, 1997).

À medida que as tecnologias de rede avançam, surgem várias aplicações com diferentes requisitos. Algumas aplicações podem tolerar instabilidade significativa na rede e latência, enquanto outras necessitam de canais de comunicação mais confiáveis e previsíveis. Por exemplo, em automação e controle de processos industriais(SURIYACHAI; BROWN; ROEDIG, 2010), cidades inteligentes(KAMIENSKI et al., 2017), saúde(AZZAWI; HASSAN; BAKAR, 2016) e sistemas militares críticos para missões(ZACARIAS et al., 2017), existem aplicações com requisitos de tempo real (ou seja, o comportamento da rede deve ser previsível e determinístico). Portanto, essas aplicações requerem uma rede resiliente e confiável para suportá-las. No entanto, apesar dos avanços tecnológicos e da crescente importância das redes na sociedade, é sabido que os níveis atuais de resiliência, previsibilidade e capacidade de sobrevivência precisam ser aprimorados(STERBENZ et al., 2013).

Para criar redes mais resilientes, algumas estratégias e estruturas desenvolvidas para a construção de redes resilientes discutem os fatores que afetam o comportamento correto de sistemas em rede, sendo alguns deles fatores externos à rede(STERBENZ et al., 2010; CETINKAYA; STERBENZ, 2013). Altas temperaturas(BOANO; CATTANI; RÖMER, 2018), chuva(BOANO et al., 2009) e vegetação(MARFIEVICI et al., 2013) são fatores que podem afetar diretamente o desempenho e a disponibilidade das comunicações. A exposição de dispositivos de comunicação a fatores climáticos e ambientais é comum em muitos cenários, como a Internet das Coisas (IoT) e redes militares. Portanto, monitorar informações de contexto que afetam o desempenho e a disponibilidade da comunicação ajuda a explicar e antecipar o comportamento da rede.

## A.1 Contribuições da Dissertação

As redes de computadores contavam, principalmente, com indicadores de qualidade coletados da própria rede. Entretanto, apenas as informações coletadas da própria rede não eram capazes de explicar alguns comportamentos detectados na rede. Este trabalho propôs a criação de um sistema de monitoramento ambiental que fornecesse informações do contexto físico para auxiliar no monitoramento da rede, e no processo de configuração dos dispositivos de rede. Portanto, a principal contribuição desta dissertação é um sistema de monitoramento ambiental ciente de contexto para dar suporte na configuração dos parâmetros de redes, aspirando uma rede mais resiliente. No sistema proposto, o processo de configuração dos parâmetros de rede dos sensores é feito de forma automática e suportado através de dados de temperatura e umidade, coletados do próprio ambiente do sensor, por meio de técnicas de aprendizagem de máquina. Por fim, a implantação do sistema é feita através de containers Docker, o que facilita o processo de implantação e aumenta a escalabilidade.

## A.2 Principais Resultados Alcançados

O trabalho desenvolvido propôs o uso de dados ambientais para aumentar a resiliência das redes. Por meio de um experimento simulado, foram feitos testes para validar a tese de que: é possível usar os dados ambientais para suportar o processo de configuração das redes, e através deles aumentar a taxa de entrega de pacotes. Após a simulação, foi confirmada a tese de que era possível obter maior taxa de entrega de pacotes através da abordagem proposta. Sendo assim, foi desenvolvido um protótipo de sistema e submetido a um cenário real de avaliação. Os resultados do experimento mostraram que o uso da abordagem proposta pode:

1. Reduzir a variabilidade da quantidade de pacotes entregues;
2. reduzir o número de retransmissões de pacotes;
3. e aumentar a taxa de pacotes entregues.

# APPENDIX B — PUBLISHED PAPER (COMCOM)

**Guilherme Rotth Zibetti**, Juliano Araujo Wickboldt, Edison Pignaton de Freitas. **Context-aware environment monitoring to support LPWAN-based battlefield applications**. Elsevier Computer Communications (ComCom), E-ISSN:0140-3664, pp. 18–27, v. 189, May 2022. DOI: 10.1016/j.comcom.2022.02.020.

- **Title**: Context-aware environment monitoring to support LPWAN-based battlefield applications.
- **Journal**: Elsevier Computer Communications.
- **Qualis / CAPES**: A2.
- **Date**: May 2022.
- **URL**: <https://www.sciencedirect.com/science/article/pii/S0140366422000639>.
- **DOI**: <https://doi.org/10.1016/j.comcom.2022.02.020>.
- **Commentary**:

  The paper presents our initial thesis that it is possible to improve network resilience based on environmental context data. In this work, we develop a simulated experiment and evaluate with two proposed approaches, Aggressive and Conservative. The obtained results presented that our approach could increase the rate of packets delivered by reducing the number of packets sent. Therefore, creating a less congested and more reliable network would be possible.

Computer Communications 189 (2022) 18–27

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

# Context-aware environment monitoring to support LPWAN-based battlefield applications

Guilherme Rotth Zibetti [*], Juliano Araujo Wickboldt, Edison Pignaton de Freitas

*Institute of Informatics, Federal University of Rio Grande do Sul, Porto Alegre, Brazil*

A R T I C L E  I N F O

*Keywords:*
Context-aware
IoT
LPWAN
Tactical networks

A B S T R A C T

The use of IoT-related technologies is growing in several areas. Applications of environmental monitoring, logistics, smart cities are examples of applications that benefit from advances in IoT. In the military context, IoT applications can support the decision-making process by delivering information collected directly from the battlefield to Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance (C4ISR) systems. Taking the benefit of the installed IoT network in the battlefield, the use of the data collected by the IoT nodes is a way to improve resiliency and increase the survivability of networks, as well as to optimize the use of available resources. Towards improving the communication network present on the battlefield, this work presents a context-aware environmental monitoring system that uses real-time battlefield information to increase military networks' resilience and survivability. The proposed approach is validated by a proof-of-concept experiment. The obtained results show that the implementation of this system can improve the communication process even when the network is exposed to unfavorable climatic factors.

## 1. Introduction

The discussion about the convergence of the real and cybernetic world is not recent, and one of the paradigms with the most significant impact in this context is the Internet of Things (IoT) [1,2]. Nowadays, IoT applications are employed to solve problems in several fields such as agriculture [3], smart cities [4], and C4ISR battlefield systems [5], which brought the concept of Internet of Battle Things (IoBT) [6]. In particular, the military context involves mission-critical applications with real-time requirements [7], so it is necessary to rely on resilient and reliable networks to support these applications [6,8].

Among the communication technologies developed to implement IoT applications, it is possible to highlight Low-Power Wide-Area Networks (LPWAN). Attributes, such as low power consumption, low data rate, low implementation cost, and long signal range, drive the efforts to develop these technologies [9]. Although LPWAN technologies have not been developed to support military applications, recent research efforts seek to adapt their use to solve problems in this context [10,11]. Like other widespread wireless networking technologies, LPWANs are exposed to interference-related problems due to climatic and environmental factors. High temperatures [12], rain [13], and vegetation [14] are factors that can directly affect the performance and availability of communications.

Boano et al. [12] conducted a study using Long Range (LoRa) – one of the most widespread LPWAN technologies – showing that exposure of transmission and reception devices to high temperatures jeopardizes network reliability. When exposed to high temperatures, devices present attenuation in the received signal strength (RSS) and reduction in packet delivery rate (PDR) and may, in extreme cases, become subject to total rupture of the communication link. Traditional monitoring systems commonly use metrics, such as signal noise ratio (SNR), RSS, and PDR, collected directly from network devices as communication quality indicators to improve network performance and reliability. Such metrics are undoubtedly important, but they may still not provide enough information to anticipate or explain performance degradation or communication link failures.

Some strategies and frameworks developed for building resilient networks discuss the factors that affect the correct behavior of networked systems [15,16]. External environmental factors can significantly impact on network operations and, therefore, must be considered in the design of dependable communications through context-aware monitoring mechanisms [17]. Context-aware systems paradigm refers to a system's ability to use information collected from its surrounding physical or virtual context to develop mechanisms to respond to events [18]. In military networks, exposure of communication devices to climatic and environmental factors is usual. Therefore, monitoring context information, which affects communication performance and availability, helps to explain, and possibly to anticipate, network behavior.

This paper proposes a context-aware monitoring system to improve the resiliency and increase the survivability of military tactical networks. A modern battlefield scenario already envisions a network of sensors to support applications such as surveillance, reconnaissance,

[*] Corresponding author.
*E-mail addresses:* guilherme@ufrgs.br (G.R. Zibetti), jwickboldt@inf.ufrgs.br (J.A. Wickboldt), epfreitas@inf.ufrgs.br (E.P.d. Freitas).

https://doi.org/10.1016/j.comcom.2022.02.020
Received 13 July 2021; Received in revised form 3 December 2021; Accepted 23 February 2022
Available online 10 March 2022

and logistics. This proposal can either rely on these sensors or suggest installing a few more in the battlefield objects to collect relevant information, which allows the prediction of a possible degradation in network performance or link unavailability. The main contributions of this work are the following:

1. The proposal of a context-aware monitoring system that – unlike current proposals found on the literature – uses information collected from the physical environment to improve battlefield overall communication efficiency in terms of PDR;
2. The proposal of two different approaches to dynamically adapt network configuration parameters based on environmental factors. In contrast with fixed configurations, both proposed approaches result in higher packet reception by the system and lower packet emission by the objects in the battlefield;
3. The proposed system was designed to be incorporated into military tactical networks with minimal impact on the existing infrastructure and communication architecture;
4. The proposed system is technology agnostic, nevertheless insights on the feasibility of implementation are provided based on a state-of-the-art open source IoT framework and related technologies.

The remainder of this article is organized as follows. Session 2 reviews related studies in the field of resilience and survivability of networks, environmental impact on wireless communication technologies, and systems that use the concept of context-awareness to adapt to the context. In Session 3, a battlefield scenario is introduced, and in Session 4 the proposed solution is presented. Session 5 presents insights that demonstrate the feasibility of implementation. Session 6 presents a proof-of-concept experiment of the proposed solution on a simulated environment and a discussion of the obtained results. Finally, Session 7 presents conclusions and future work.

## 2. Related work

Creating frameworks and guides with best practices is present in several research areas; it is no different in the context of resilient networks. Towards resilient networks, Sterbenz et al. [19] proposed a framework consisting of strategies, metrics, and techniques to evaluate and quantify the resilience of the architecture of an already existing and a proposed network. In this framework, one of the prerequisites is context-awareness. The authors state that, for a given network to be resilient, it is necessary to monitor the channel's conditions, the link-state, and other events that may impact its correct functioning.

As aforementioned, a context-aware system can adapt in response to external events that affect its functioning. However, for this to happen, it is necessary to monitor the surrounding environment. Some works present techniques to collect, process, and make useful the data collected from the context to support other systems. In a study developed by Preeja and Krishnamoorthy [20], the authors highlight relevant points in constructing a context-aware system, such as context modeling and organization, and the use of a middleware to simplify the development considering the heterogeneity of technologies. In a recent work developed by Pradeep et al. [21], the authors proposed a generic context model and a context organization method for IoT environments. Context modeling is related to the syntax used to represent the raw data, while the organization provides the semantics for this data and its relationships. Therefore, a system must be modeled in a way that is generic enough to support a heterogeneous environment and well organized so that the collected information makes sense for the systems that need it.

Besides gathering and processing context information, to actually improve the communication process of networks, many authors rely on the adjustment of configuration parameters. Modifying the network configuration parameters to increase the signal strength is already explored in several studies. The most commonly used parameter is transmission power control using different approaches. The first approach is at the network level [22,23], which is based on applying the same transmission power to all nodes in the network. The second approach is at the node level [24] by applying different transmission power to each node in the network. The third approach is based on neighborhood relation [25], in which nodes use a different transmission power for each neighbor. A fourth approach can apply different transmission power at a packet-level [26]. A common limitation among all those approaches is that the network configuration parameters were adjusted using only the information collected from the network itself, such as Received Signal Strength Indicator (RSSI) and Link Quality Indicator (LQI). While this information is useful, currently, other factors can be considered to determine network configuration parameters. In this work, we aim to explore, in addition to the commonly used quality indicators, additional information collected from the physical environment where the nodes are installed.

The work developed by Boano et al. [12] presents the impact of temperature variation, a possible climatic factor, on LoRa communication links. The results show that higher temperatures compromise the network operation and, in extreme cases, cause total link rupture. In the work developed by Luomala and Hakala [27], the authors present the climatic effects of temperature and humidity on communication links using ZigBee wireless networking technology. The results of this experiment show the impact of these factors on communication links. The authors used the RSSI as a metric, showing its oscillation according to the climatic variations observed from the external environment during the experiment. In both studies by Boano et al. [12] and by Luomala and Hakala [27] modifications in configuration parameters were explored to assert the resilience and survivability of these networks when exposed to climatic factors. These studies help to emphasize how critical the monitoring of external factors is to increase network resiliency, although they are quite theoretical and do not integrate into an actual network framework or solution.

Towards the design of context-awareness systems with the ability to adapt to the external environmental factors, Dobrescu et al. [3] suggests the implementation of a context-aware system for the precision agriculture environments. The authors' proposal aims to integrate three emerging technologies (IoT, cloud computing, and context-awareness) to provide an irrigation system and monitor parameters related to soil properties. In this system, the environmental changes collected by the sensors determine how the system will manage water, nutrients, and pesticides. In another work, developed by Kamienski et al. [4], the authors propose the use of IoT-related technologies to improve the process of energy consumption in public buildings and universities. In this work, the authors suggest using presence sensors and smart plugs to control room lights and monitor the energy consumption of buildings. These are examples of context-aware systems that use information collected from the surrounding physical environment to determine their behavior. Although both works use information gathered from the context, neither uses it to determine its functioning, but rather the functioning of the application they are serving.

Another example that uses information collected from the context to define what action to take is the work proposed by Rak [28]. The author proposes adapting the topology of a wireless mesh network using the information on predicted attenuation of links based on radar measurements. More specifically, the author uses real echo rain maps to calculate attenuation in communication links and modify the network topology to improve network throughput during rainstorm periods. Differently from Dobrescu's [3] context-aware precision agriculture system and the intelligent buildings of Kamienski et al. [4], in Rak's proposal, a third-party system provides the weather maps that serve as context information. In contrast, the other systems collected context information through sensors to decide the system's action.

In the context of military networks, it is possible to highlight three works that implement context-awareness tackling different issues. The first work developed by Papakostas et al. [29] proposes an algorithm for building a backbone in military networks with energy-awareness to deal with power efficiency in the construction of the network topology. In the second work, Poularakis et al. [30] developed a system that uses the concept of Software Defined Network (SDN) to address changes in the network topology from nodes distributed across the battlefield. The third work to be highlighted was the work developed by Mishra et al. [31]. The authors propose a context-oriented proactive decision support framework that aims to accelerate the decision-making process. The developed system uses data related to the mission, environment, assets, threats, time, workload, and other factors related to the decision-making process. Based on the acquired data, the system suggests to the decision-maker some possible decisions that could be taken to solve a given problem. In addition to the works already mentioned, other studies also address the use of context-aware applications to support decision-making on the battlefield [32–34]. However, as in the works developed by Dobrescu et al. [3], and Kamienski et al. [4], the aforementioned studies in the military context use the data context to determine the application behavior. A step ahead of work in this direction is presented in [35], which presents an approach that combines SDN and Information-Centric Networking (ICN) to support context awareness in IoBT systems.

The difference of the proposal presented in this current paper from the works mentioned above is that, although some of them used information collected from the physical environment, this information determined the application's behavior and not the network's behavior. In contrast, in the work developed by Rak [28], the information that determined the system's behavior was not collected directly from the system context, but provided by a third-party system. Relying on an external entity to provide information relevant to the context of the system can cause problems due to the fact that it may not meet the time constraints of critical environments such as battlefield networks. Among the studies in the military context, it is possible highlight the works developed by Papakostas et al. [29] and Poularakis et al. [30] , which propose solutions for context-aware networking. However, none of these works consider the effect of environmental factors on the military networks. Other works, like the developed by Mishra et al. [31] and Leal et al. [35] in the military context, consider context-aware applications to assist in the decision-making process of Command and Control (C2) systems, leaving out of scope the network management itself.

## 3. Application scenario

A fictional scenario was created containing some of the objects usually present in the battlefield to make the proposal clearer. Fig. 1 presents this battlefield scenario divided into Control Zones (CZ), identified by numbers from one to four. Each CZ has a Communication Center (CC), that can be Static (SCC) or Mobile (MCC), represented by control camps and a combat vehicle with a high-power antenna attached to the top (located at CZ-2). The connections between the control camps and the CZ-2 combat vehicle represent a Tactical Edge Network (TEN).

In addition to the TEN, the battlefield has a Sensor Network (SN). This network is responsible for transmitting the data from the sensors spread across the CZs to the Environment Context-Aware System (ECAS) servers located in CC of each CZ. The data traffic of the TEN has no relation to the data traffic carried out by the SN. The sensors distributed on the battlefield can communicate with any CC within range to deliver data collected from the battlefield. For instance, it is possible to observe this type of communication in CZ-4, whose flying drone has a connection with the CCs of zones two and four. Therefore, it can deliver the collected data to both CCs.

There are two types of sensors distributed on the battlefield, the Monitoring Sensor (MS) and the Field Object Monitoring Sensor (FOMS). MSs are distributed on the battlefield collecting information from a specific location, whereas FOMSs are installed on objects moving

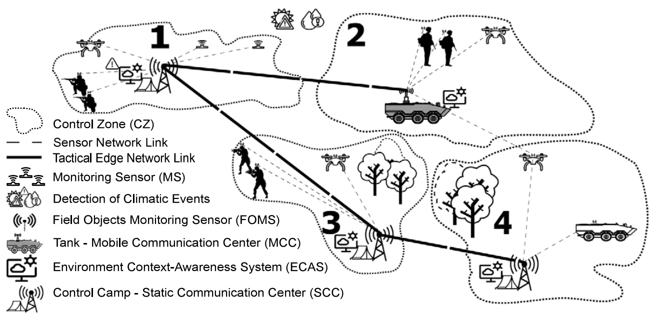

1
2
3
4
Control Zone (CZ)
Sensor Network Link
Tactical Edge Network Link
Monitoring Sensor (MS)
Detection of Climatic Events
Field Objects Monitoring Sensor (FOMS)
Tank - Mobile Communication Center (MCC)
Environment Context-Awareness System (ECAS)
Control Camp - Static Communication Center (SCC)

**Fig. 1.** Battlefield scenario.

across the battlefield. Both sensors can collect temperature and humidity information. However, FOMSs can collect additional information such as vital signs, location, weaponry, and so on, from the objects to which they are attached.

The ability of MSs and FOMSs to communicate with any server makes ECAS a distributed system. Besides receiving information from multiple MS and FOMS, the ECASs servers distributed across the battlefield also share the information collected in their respective CZ through the TEN. Sharing the information collected in each CZ increases the coverage area monitored by ECAS. Therefore, this information can support decision-making and mitigate problems in any CZs that have a CC.

In Fig. 1, there are two icons at the top to exemplify the system's operation, which represent the detection of climatic events related to temperature and humidity. The sensor closest to these events is the MS located in the upper right corner of the CZ-1. This sensor will notify the ECAS server of its respective CZ that it will process the received information and perform the necessary actions in response to these events. After processing the received data, the ECAS server of the CZ-1 will share the processed data with the other ECAS servers, which may need the data. After receiving the data, the other ECAS servers will process the received information to verify whether to perform an action in their respective CZ. Thus, the environment covered by the ECAS can adapt to the climatic conditions using as inputs the data collected from the battlefield through the MS and FOMS sensors.

## 4. Proposed system architecture

Fig. 2 illustrates the system architecture. The circle, named battlefield, refers to the monitored environment. The sensors (1 to Sensor N) represent the MSs and FOMSs distributed across the battlefield. The box with Gateway 1 and Gateway *N* represents the Gateways distributed by the battlefield, being a Gateway for each CZ. As it is a system that acts in a distributed way, it is necessary to have a Gateway for each ECAS server distributed by the battlefield. The dashed box on the top of the figure highlights the internal components of each sensor (i.e., the sensor architecture).

ECAS is composed of servers that act independently and collaboratively. The distribution of these servers across the battlefield is in the same number of the existing CZ for a given military operation. That is, for each CZ, there is a Gateway and an ECAS server to handle the data collected locally. The collaborative network of servers is represented in Fig. 2 within the cloud shape, indicating that the communication among servers flows over the TEN. In the bottom part of Fig. 2 there is a highlight with the internal components of each server (i.e., the server architecture).

The following flow describes how the data collected on the battlefield is transmitted, processed, and shared. The SN is responsible for transmitting the data collected by the sensors on the battlefield to the ECAS servers distributed across the CZs. When receiving the collected data, an ECAS server must process and share it with the other

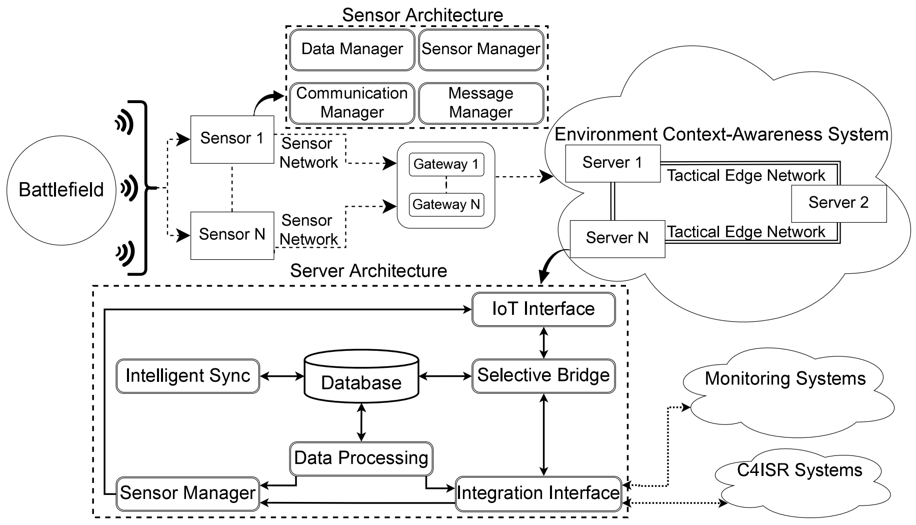


**Fig. 2.** System architecture.

servers that make part of the system. This sharing takes place through the TEN. Therefore, the SN serves exclusively for sensory data traffic and reconfiguration of sensors distributed across the battlefield. The detailing of the modules that compose the ECAS servers and sensors are presented in the following sub-sessions.

### *4.1. Sensor architecture*

In Fig. 2, the dashed box indicated by an arrow on Sensor 1 represents the internal architecture of each sensor. Each sensor has four modules that determine its operation. The modules are Sensor Manager, Message Manager, Communication Manager, and Data Manager.

The Sensor Manager module is responsible for overseeing the operating mode of the sensor as a whole. Each MS and FOMS sensor has the ability to read environment data (e.g., temperature, humidity, location, vital signs) and to send this data over a SN to a gateway. The role of managing the process of collecting, storing, sending, and receiving data, which takes place through the components of each sensor, is performed by the Sensor Manager module. This module manages the frequency of data collection and local storage, as well as the frequency of transmission (in bulk). This process follows two basic criteria, which are defined as Periodic and Threshold-based. Periodic criteria means that configurable time intervals will guide the collection, local storage, and transmission of sensored data. For example, under normal conditions, temperature and humidity could be measured every few seconds, but transmitted only once per minute to save resources. Threshold-based criterion intends to reduce the frequency of data collection and transmission under unsafe operating conditions. For instance, when the measured temperature reaches a harmful threshold, it can be set for immediate transmission instead of waiting until the next periodic cycle.

The Message Manager module manages the format of the message sent to the ECAS server. Two types of messages were created: Periodic messages and Trigger messages. Periodic messages are related to the Periodic criteria, while Trigger messages are related to the Threshold-based, both defined by the Sensor Manager module. Suppose the data for a region monitored by a particular MS/FOMS sensor is outside safe operational limits. In this case, this sensor should classify the data contained in the message as urgent and send it immediately. On the other hand, if the data is within safe operational limits, the sensor can store this data locally and wait for an opportune moment for sending it, determined by the periodic criterion. Differentiating messages is a way to prioritize the communication and processing resources of the system for the most critical messages.

Defined the frequency of data collection and transmission, the types of messages, it remains to specify how to present the data to the application. The Data Manager module performs this function. ECAS aims to increase the resilience of networks as well as support the applications of C4ISR systems. Therefore, the sensor that collects and sends relevant data to ECAS does the same for C4ISR systems. The Periodic criterion determines that all sensors present in an MS/FOMS read and send the collected data to the ECAS server at a specific time. This collection criterion can be useful when the sensor is in a region where operational limits are safe; however, this is not always the case. When a sensor is in a region whose collected data is outside a safe operating threshold; the messages must contain only relevant data (i.e., data exceeding the threshold). This approach reduces the overhead of the collection and transmission process by prioritizing processing and communication resources for the most relevant data. For example, a sensor in a specific region detects that the local humidity and temperature are above the specified threshold. The Data Manager module gathers the collected data, assembles a package with the relevant data for this specific case (e.g., sensor id, collection timestamp, temperature, humidity, GPS location) and delivers this package to the Message Manager module. The Message Manager module classifies the message as Trigger and delivers it to the Communication Manager module, which sends it over the gateway to the ECAS server immediately.

The Communication Manager module is responsible for managing the communication resources of a MS/FOMS. The Sensor Manager module specifies the criteria for collecting and sending data to the ECAS server. Therefore, data collection and transmission work asynchronously, allowing the Communication Manager to set communication resources into stand-by or power-saving mode when they are not needed. The Message Manager module defines two types of messages: Periodic and Trigger, and each type have specific requirements to use the communication resources. Periodic messages must be delivered to the ECAS server as soon as possible, but may wait longer to access the medium as their delay will not significantly impact other systems. A message of type Trigger, on the other hand, contains sensitive data (i.e., data directly related to the good functioning of the system); therefore, these messages must take priority in accessing the medium and guaranteeing earliest delivery. Therefore, the Communication Manager module manages the communication resources to fulfill the message's requirements to be transmitted.

### *4.2. ECAS server architecture*

Represented in the dashed box pointed out by Server $N$ in Fig. 2 is the ECAS server's architecture. Each ECAS server has the same architecture, consisting of six processing modules and one database.

The first module of the ECAS server architecture to be discussed is the IoT Interface. IoT communication technologies provide specific protocols to meet the needs of these networks. This module is responsible for interpreting the data received from the sensors and translating them into a format consumed by ECAS and other systems that use this data. In addition to interpreting and translating sensor data, this module does the reverse process for messages sent from servers to sensors. Therefore, the IoT Interface module is the communication interface between IoT technologies and ECAS servers. The function performed by this module simplifies the implementation of heterogeneous IoT technologies on the sensor side, as it centralizes the communication process between different technologies in a single module.

ECAS uses data collected by sensors to increase the reliability of the network. In addition to data collected for ECAS, the sensors collect data relevant to C4ISR and other systems. Therefore, after the IoT Interface receives and interprets the data coming from sensors, it is necessary to forward the collected data to the respective systems of interest. The Selective Bridge module performs this function. It selectively distributes data received from the IoT Interface to be either stored and consumed locally by ECAS, as well as forwarded to external systems. This module works like a bridge between the IoT network and C4ISR or other systems, introducing only minimal overhead as data travels through ECAS. As depicted by the arrows in the ECAS Server Architecture of Fig. 2, the data coming from the IoT Interface passes through the Selective Bridge module. This module then separates data that is useful to ECAS, which needs to be stored locally it in the Database, and forwards the rest to the respective systems of interest.

Once the data has been collected and separated, it is necessary to forward the data that is not useful to ECAS and process the ones that are. Before doing this processing, ECAS needs to have information from other systems available to combine with the data collected from the sensors. Information such as RSSI, SNR, and PDR are useful for ECAS to perform its role, and the providers of this information are network monitoring systems. The Integration Interface module aims to provide a single communicate interface among network monitoring systems, C4ISR systems, and ECAS servers, as represented in Fig. 2 by the arrows connecting the Integration Interface module and the clouds that represent external systems. The first function of the Integration Interface is to distribute the data received from the Selective Bridge module to the respective systems of interest and to search other monitoring systems or C4ISR for information relevant to ECAS. The second function of this module is to generate alarms for other systems (i.e., when ECAS detects an anomaly, it can generate alarms for other systems to assist in the decision-making process C4ISR systems). Finally, the Integration Interface can reconfigure the parameters of the sensors of interest of the C4ISR systems (e.g., modify thresholds of vital signs, weapons, etc.), as represented in Fig. 2 by the arrow that goes from the Integration Interface module to the Sensor Manager module (discussed later). This last function enables ECAS to serve as a central point for dynamic reconfiguration of parameters in all sensors in the battlefield, regardless of whether that parameter is relevant to ECAS or external systems.

The three modules described so far are mainly focused on gathering information, either from the sensors in the IoT network or from external sources, and selectively storing the gathered information in a local Database. The Data Processing module is responsible for querying the database for context information received from the IoT network (periodic and trigger messages), combining this data with information from other monitoring systems (e.g., LQI), and compute potential actions that could be taken to adjust the network according to the currently assessed situation. Some possible outputs of this processing are: generating alarms for C4ISR systems (through the Integration Interface), reconfiguration of the parameters of one or a set of sensors (through the Sensor Manager), updating the quality indicators in the Database, as well as storing the processed data for later use as a dataset in the process of reasoning, which aims to anticipate the behavior of the network.

As mentioned above, two types of sensors may be collecting data in the field, called MS and FOMS. An MS/FOMS collects data (e.g., temperature, humidity, vital signs, weaponry) for the ECAS and other C4ISR systems. Each system is responsible for determining the thresholds and time intervals for reading measurements. The Sensor Manager module aims at abstracting the differences between possibly heterogeneous MS/FOMS in modifying the configuration parameters. For example, if a C4ISR system needs to modify the critical threshold values of a parameter such as vital signs, this system communicates the parameters through the Integration Interface to the Sensor Manager module, which assembles and sends a package in the format expected by the sensor. The same process applies to ECAS itself. Suppose that a given context situation makes the Data Processing module compute a decision that requires changing the sensor's temperature or humidity thresholds. In this case, the Data Processing module instructs the Sensor Manager module to set the new configuration parameters in the sensors. The Sensor Manager, in turn, relies on the IoT Interface to communicate with the sensors in the battlefield (represented in Fig. 2 by the arrow that connects the Sensor Manager and IoT Interface modules).

After collecting, distributing, processing, and storing locally the outputs of the previously described modules inside an ECAS server instance, there is still the need to verify if these outputs can be useful for other ECAS servers distributed on the battlefield. The output generated by a specific server from one CZ might be useful for servers from another CZs, but it might not be either. The function of checking which ECAS server may need each piece of data, and sharing this data is performed by the Intelligent Sync module. As already mentioned, the ECAS system relies on a set of servers distributed throughout the battlefield collecting, processing, and performing actions on the devices of their respective CZs. Due to the scarcity of resources available on the battlefield, misusing these resources can compromise network reliability. Therefore, the task of synchronizing data across servers on the battlefield should not overwhelm available network resources. The Intelligent Sync module verifies the outputs generated by the Data Processing module and decides based on context information, such as the location data of other ECAS servers and sensors, whether information should be shared or not. If the generated output is considered useful to another CZ, Intelligent Sync sends it over the TEN to this specific server. For example, the server located at CZ-1 may generate an output pertinent to the server located at CZ-2 (e.g., rain detected at CZ-1 and is moving towards CZ-2). Since the information related to this event is pertinent that the server located in CZ-2, this information should be sent to it as soon as possible. Nevertheless, there is no need to forward the same data to the servers in CZs 3 and 4 since the phenomenon is not likely to affect those.

## 5. Implementation feasibility

As previously explained, the system proposed in this work aims to increase the reliability and survivability of networks exposed to harsh environmental and climatic factors on the battlefield. To illustrate the feasibility of the proposed system architecture, this session presents possible implementation choices for some of the components presented in Fig. 2.

FIWARE is an example of framework that aims to accelerate the development of IoT-related smart solutions [36], which could be used to handle both data collection and issuing commands between sensors and ECAS servers. This framework provides a Context Broker (CB) component whose function is to manage context information so that other systems can consume it [37]. The data processed by the CB can come from several sources, including external systems or battlefield sensors. Through an IoT Agent component, the CB can collect context information, process it and make it available to ECAS. The CB can manage data collected from the context and send commands from the ECAS

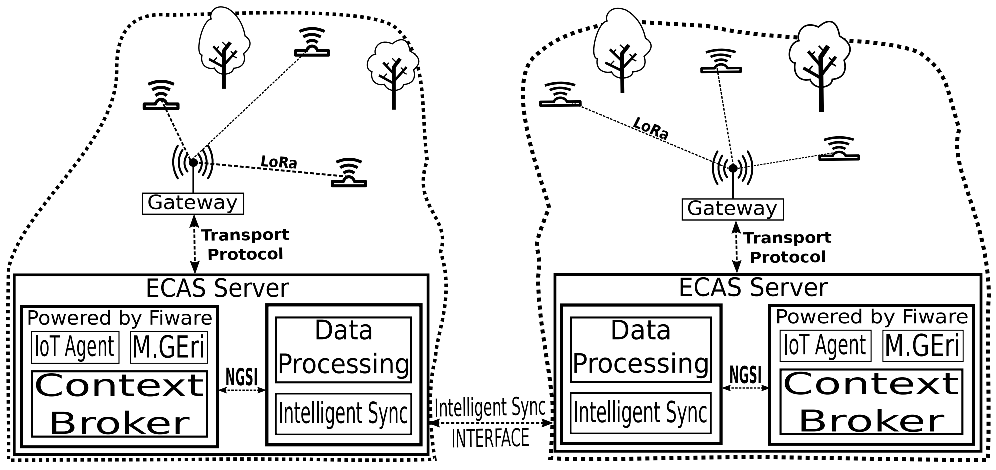


**Fig. 3.** System architecture "Powered by FIWARE".

to sensors on the battlefield through the IoT Agent. There are ready-to-use implementations of IoT Agents based on JSON and Ultralight encoding, transporting data over AMQP, HTTP, and MQTT [38]. All interactions between the CB, IoT Agents, and ECAS components can take place through an open and standardized RESTful API called Next Generation Service Interface (NGSI) [37].

FIWARE's NGSI API defines a data model and two interfaces for the process of exchanging context data. The main elements of the NGSI data model are entities, attributes, and metadata. An entity in FIWARE represents an object, which can be physical or virtual. This entity has attributes and metadata related to it [39]. Considering ECAS elements with FIWARE, an entity can represent an MS sensor, a FOMS sensor, a CZ, a CC. This entity has a unique ID and a Type (e.g., ID: Sensor-01, Type: MS; or Zone-01, Type: Control Zone). As attributes, a sensor entity could have Local Temperature, Local Humidity, Location. A CZ entity could have the location, vegetation as attributes. FIWARE entities can have relationships with each other. Therefore, it is possible to state that Sensor-01, which is of the MS type, is located in CZ-01 (i.e., it is related to CZ-1). Thus, if Sensor-01 collects and sends data, the ECAS server will know that Sensor-01 is in CZ-01.

On top of the FIWARE framework it is possible to implement several components from ECAS architecture. For example, an IoT Agent receives data collected by sensors through a Gateway and transmit commands from ECAS servers to sensors through a Gateway. Sensors in a FIWARE-based solution can represent a sensor or an actuator. A sensor can report the state of the world around it, while an actuator can change the state of the system by responding to control signals coming from the CB [40]. Therefore, with the FIWARE framework, it is possible to implement the four modules present in the sensor architecture since the properties of an actuator include the possibility to set the sensor configuration parameters based on commands received from a CB.

The ECAS server IoT Interface module can be represented by the IoT Agent. In addition to the IoT Agent, FIWARE provides a Monitoring GEri that aims to incorporate monitoring systems with the CB [41]. Thus, it is possible to use FIWARE to implement two other ECAS modules, Selective Bridge and Integration Interface. Since FIWARE can set configuration parameters of the sensors, it also enables the Sensor Manager module's implementation on the ECAS server's side. Any platform or solution built on top of FIWARE has only one mandatory component, a FIWARE Context Broker Generic Enabler [42]. Furthermore, since this solution has this Context Broker, it is named "Powered by FIWARE". Since this section suggests implementing ECAS using a CB, Fig. 3 shows how ECAS would be developed on top of the FIWARE framework.

The modules implemented by FIWARE are highlighted in Fig. 3 inside a box called "Powered by FIWARE". This box shows the three software components that make up the ECAS server, an IoT Agent, Monitoring GEri, and the CB. These components perform the functions of the Sensor Manager, Selective Bridge, Integration Interface, and IoT Interface modules. An NGSI interface handles the communication process between the ECAS modules and the modules implemented by FIWARE.

Fig. 3 represents the communication of sensors with the Gateway using LoRa, but other LPWAN technologies such as NB-IoT, LTE-M, and Weightless can be used for this communication. Likewise, the communication between the Gateway and the IoT Agent can use any of the supported transport protocols. Fig. 3 can represent different IoT Agent implementations. For example, it is possible to have an Ultralight-based IoT Agent in one area, while the other area might have a JSON-based IoT Agent, both running over the HTTP transport protocol. Examples of other supported transport protocols are AMQP and MQTT; in which case, it would be necessary to deploy an additional message broker that is not represented in Fig. 3. Among the ECAS servers, there is an interface called Intelligent Sync Interface, and this interface is responsible for synchronizing the data stored in each server and distributing it to the servers that may need this data.

According to the needs of each ECAS server, the CB processes and classifies the data collected from the context. Therefore, to implement the Intelligent Sync module it is possible to use artificial intelligence techniques to determine what should be shared and with which servers. The Intelligent Sync module is responsible for distributing the data to other ECAS servers on the battlefield. Instead of replicating every piece of data in all servers, some data distribution techniques already used in other works could enable the development of this module. Liu et al. [43] propose a distributed mechanism that works offline and online to determine which data center to place data based on read/write requests. The Intelligent Sync module can benefit from the scalability and distributed architecture introduced by this mechanism. In another work, Liu et al. [44] proposed a framework called DataBot, which uses techniques such as neural networks (NN) and reinforcement learning (RL) to create data placement policies for a system with distributed data centers. The data that needs to be shared by ECAS servers involves information collected from the physical context. Therefore, implementing artificial intelligence and machine learning (ML) techniques to assist in the intelligent distribution of data can increase the system's efficiency.

## 6. Experiments and results

To demonstrate the efficiency of the proposed system, simulations have been performed as a proof-of-concept. The network simulator used was the *"ns-3 - discrete-event network simulator"*. This simulator offers a LoRaWAN module, which allows the simulation of a sensor network based on LoRa technology. Also, the NYUSIM simulator [45] was

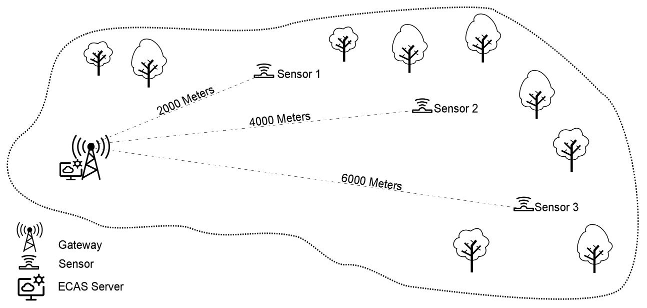


**Fig. 4.** Scenario of the experiment.

employed to calculate the attenuation in the wireless communication links by varying weather factors, as explained in detail in the following sections.

## 6.1. Experimental setup

The implemented ECAS architecture uses LPWAN technology to communicate sensors distributed on the battlefield with servers. In the simulation developed as a proof-of-concept, the chosen LPWAN communication technology was LoRa. LoRa is a proprietary spread spectrum modulation technique that is derivative of chirp spread spectrum modulation (CSS); this technique has been used in military applications because of its long-range coverage and interference robustness [46]. Although LoRa's characteristics make it suitable for the operation of the proposed system, factors such as support for applications with real-time requirements [47,48], interoperability with command and tactical control systems [11] were also considered for the choice of this technology.

LoRa radio has four configuration parameters: carrier frequency, spreading factor (SF), bandwidth, and coding rate. Setting these parameters determines signal robustness, power consumption, and transmission range [49]. An important feature of LoRa that was explored in this simulation was the Data Rate (DR). The DR consists of the combination of two LoRa configuration parameters: the bandwidth and the spreading factor. The highest DR in LoRa communication provides a greater volume of data transmitted over shorter distances. Although the amount of data transmitted is less in the lower DR, the signal strength is greater, allowing communication between Sensor and Gateway over long distances and unfavorable weather conditions. These factors are important to consider regarding the application scenario under concern.

An experiment with the following parameters was carried out. A LoRa gateway operating on three channels of 868 MHz, one channel for each sensor, was installed at 15 meters height. Three sensors with a fixed location (without mobility) at 2,000, 4,000, and 6,000 meters away from the gateway, and at 1.7 meters height. Each round of the experiment represented 24 hours of simulation and each node sent a packet to the gateway periodically every 60 seconds. Fig. 4 illustrates the scenario of the experiment just described.

At each round of the experiment, the DR of the sensors were modified. For each DR, an attenuation representing 1 mm/h of rain was applied to the signal, then run one round. Therefore, 151 rounds of the experiment were executed for each DR. Starting, for each DR, in 0 mm/h of rain, which represented a channel free of attenuation due to rain, and ending with 150 mm/h, representing a heavy attenuation scenario.

In the first stage of the experiment, the same DR was configured in the three sensors, ranging from DR-0 to DR-5. Being DR-5 the highest packet rate and the lowest reach, and the DR-0 the lowest packet rate and greater reach of signal. Table 1 presents the combinations of SF and bandwidth for the DRs applied during the experiment. The DR-5 represents the configuration with the ability to transmit the most massive volume of data. While the DR-0 represents a more robust signal to attenuation, therefore it reaches greater distances. Although the robustness of the signal experienced by DR-0 is better than the other DRs, the volume of data transmitted by it is lower than the higher DRs. Therefore, applying the appropriate DR to the sensors distributed across the battlefield, considering the physical and environmental context, increases the network's reliability by offering a communication infrastructure adaptable to the context.

**Table 1**
Data rates of the experiment.

| DR | SF | Bandwidth |
|---|---|---|
| 5 | 7 | 125 kHz |
| 4 | 8 | 125 kHz |
| 3 | 9 | 125 kHz |
| 2 | 10 | 125 kHz |
| 1 | 11 | 125 kHz |
| 0 | 12 | 125 kHz |

In the second stage of the experiment, the goal was to observe how ECAS acts in the monitored environment. In this approach, ECAS collects the battlefield information and adapts the DR of each sensor to the best DR according to the observed weather conditions. This adaptation is made using two approaches called: Conservative and Aggressive. In the Conservative approach, ECAS receives the rain rate in mm/h from each sensor, calculates the attenuation of this factor in the communication link, and, if necessary, reconfigures the DR of a given sensor to increase signal robustness or improve transmission rate. In this approach, the system modifies the sensor's DR before reaching a threshold. In contrast, in the Aggressive approach, ECAS expects a given sensor to decrease in the PDR, and only then it reconfigures that sensor's DR.

The Conservative approach has been configured to reduce the DR of the sensor by one, which can be affected by attenuation, 5 mm/h of rain before the attenuation causes any impact on the PDR. Therefore, before any sensor has a reduction in PDR, its DR is reduced to increase signal robustness, although the packet sending rate also reduces. In the Aggressive approach, the highest DR is used until a reduction in PDR from a given sensor is observed. Only then adjustments are made to the DR of the respective sensor. This approach allows better use of higher data rates in scenarios with small climate variation.

## 6.2. Results

Fig. 5 shows the results of the first stage of the experiment, in which a fixed DR on the three sensors were set. The $Y$-axis of Fig. 5 represents the received packets by the gateway per round of the experiment; the $X$-axis represents the increase of rain in mm/h that goes from 0 mm/h to 150 mm/h. For better visualization of the results, the DR-3 and DR-4 have been hidden. However, the information of these DRs can be found in Table 2, which shows the number of packets sent, received, lost, and PDR in percentage for each DR applied in the experiment. During the experiment, it was observed that the packet sending rate experienced by the three sensors from DR-5 to DR-2 was the same, 1500 packages per experimental round. While for DR-1 and DR-0, all sensors suffered a fall in the packet sending rate. It reached approximately 1380 packets sent per round with DR-1 and 690 with DR-0 per sensor.

It is possible to observe in Table 2 that, because the packet sent rate experienced by the DRs from 5 to 2 is the same, for the parameters applied to the experiment, the use of DR-5 does not present the best performance compared to DR-4, DR-3, and DR-2. At the beginning of the experiment, sensors 1 and 2, located at 2,000 and 4,000 meters away from the gateway, connect from 0 mm/h to 5 mm/h of rain. At the same time, the third sensor, located at 6,000 m, has no connection since the beginning of the experiment. From 5 mm/h on, sensor 2 loses connection to gateway due to the attenuation caused by the increase in rain rate. For DR-2, whose packet sent rate is the same as that

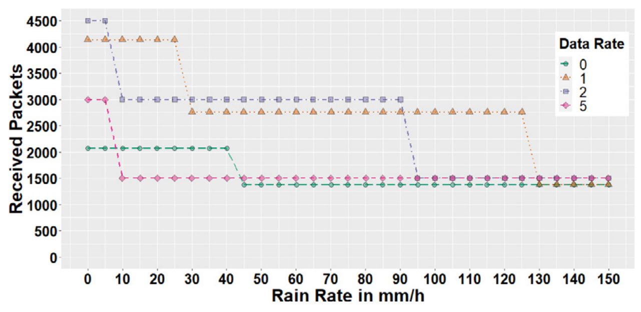


**Fig. 5.** Delivered packets per data rate.

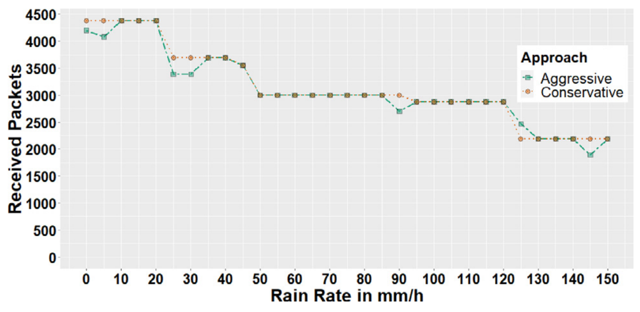


**Fig. 6.** Delivered packets adaptive aggressive X conservative approach.

**Table 2**
Sent, received, lost packets and PDR% per data rate.

| DR | Sent | Received | Lost | PDR% |
|---|---|---|---|---|
| 5 | 139,500 | 49,500 | 90,000 | 35% |
| 4 | 139,500 | 57,000 | 82,500 | 41% |
| 3 | 139,500 | 66,000 | 73,500 | 47% |
| 2 | 139,500 | 78,000 | 61,500 | 56% |
| 1 | 128,216 | 86,851 | 41,365 | 68% |
| 0 | 64,139 | 48,959 | 15,180 | 76% |

experienced by DR-5 and the signal is more robust, at the beginning of the experiment, all nodes have a connection to the gateway, reaching a PDR of 1,500 per node (4,500 in total). This PDR is experienced up to 5 mm/h of rain, when the PDR drops to 3,000 packets per round, 1,500 packets of each of sensors 1 and 2, and remains up to 90 mm/h of rain when the second node loses connection to the gateway.

DR-0 and DR-1 achieved a better PDR, although the number of packets sent is lower than higher DRs. In DR-1, the packet sent rate per node approaches the 1,380 packets, while in DR-0, the rate drops to approximately 690. Fig. 5 shows that the DR that performed better in contrast to the others was DR-1. In DR-1, the packets sent rate comes near 1,500 packets, which is experienced by the higher DRs, and the signal's robustness makes all three sensors connect with gateway at higher rainfall rates. Using DR-1, the furthest sensor can connect with the gateway up to 25 mm/h of rain. Using DR-2 and above, this sensor's connection does not exceed 5 mm/h of rain. Sensor 2 can connect to the gateway up to 125 mm/h of rain with DR-1, contrasting with the 90 mm/h of the DR-2 and lower rainfall rates in the higher DRs. In the DR-0, which represents the most robust signal, but with the lowest packet sent rate, the farthest node had a connection with gateway up to 40 mm/h of rain when it loses connection. At this DR configuration sensors 1 and 2 remain connected to the gateway up to 150 mm/h of rain.

Fig. 6 presents the results of the experiment considering a scenario with the implementation of the ECAS. The DRs adapted in this scenario, consider the rain rate and sensor's distance to the gateway. The $Y$-axis in the chart of Fig. 6 represents the received packets, while the $X$-axis represents the rain rate in mm/h. For both the Aggressive adaptive approach and the Conservative adaptive approach, the results obtained were higher than fixed DRs.

In the Aggressive approach, each loss detection, the sensor DR was reduced by one to increase signal robustness and improve the gateway connection. This approach allows the use of DR with a higher rate still considering the climatic conditions at the time of transmission. The Conservative approach behaves differently. When a given rain level is observed, the DR of the sensor is decremented by one 5 mm/h of rain before the signal can be affected. Therefore, the sensor should not lose connection to the gateway whatsoever. However, suppose the rain does not exceed this threshold of 5 mm/h. In that case, the sensor will have stopped using the DR with a higher packet sending rate until the system sees a reduction in the rain rate, which would allow an increase in DR considering the new rain scenario.

In Fig. 6, the dotted line represents the Aggressive approach. At the beginning of the experiment, this approach presents some received packet lower than the Conservative approach. This behavior occurs because the Aggressive approach tries to use the largest possible DR from the beginning of the experiment, then starts the experiment with losses until it finds the ideal DR. From the 10 mm/h of rain, the Aggressive approach achieves the same number of packet received as the Conservative approach. During the experiment, every now and then, the Aggressive approach achieves a lower number of packet received than the Conservative approach; this is due to the method used to determine when to adjust the DR in the Aggressive approach. At 120 mm/h, the Aggressive approach performs better than the Conservative approach. It was expected that this would happen at some moment. While the Conservative approach prevents losses, the Aggressive approach uses losses as a factor in determining when to adjust the DR. At this moment of the experiment, the loss occurred in the 124 mm/h of rain, which ensured a number of packet received higher for the Aggressive approach in the range of 120 mm/h to 124 mm/h of rain.

The experiment, which served as a proof-of-concept to exemplify the functioning of the ECAS, adopted the premise that the rate of rain only increases. Thus, the experiment started with the rain rate at 0 mm/h and ended in 150 mm/h. This premise demonstrates what the system's behavior would be if adopted for the battlefield scenario where the rain only increases. However, in a real scenario, rain rates do not increase linearly. The rain can increase, stabilize, or even reduce before stopping. It is not possible to determine when one of these events will occur. Thus, two approaches have been proposed to demonstrate the behavior for the system considering the performance and reliability requirements.

Fig. 7 presents a comparison of the experiment performed with fixed DR and the Aggressive and Conservative approaches of the ECAS. On the $Y$-axis of Fig. 7, the Total Packets Received plus the Total Packets Sent is displayed. Each bar represents a setting applied in the experiment. By adopting any of the approaches offered by ECAS, it is possible to obtain a higher PDR than fixed DR approaches. The PDR experienced by DR-1 and DR-2 is one that is closer to the PDR experienced by the approaches offered in the ECAS. Table 3 presents the employed approach, the number of packets sent, and the PDR for each experiment round. Approaches using ECAS offer a packet sending rate that approximates the approaches with the best fixed DR performance. However, the amount of received packets is much higher when ECAS is in place to monitor and adapt to the environment.

## 7. Conclusion

The unpredictability of the battlefield context makes it necessary to implement systems that improve the decision-making process time-effectively. The present work proposed a context-aware monitoring system to adapt network configuration parameters to changes in the climatic factors observed on the battlefield. A simulation was developed as a proof-of-concept to demonstrate how the proposed approach performs under varied weather conditions.

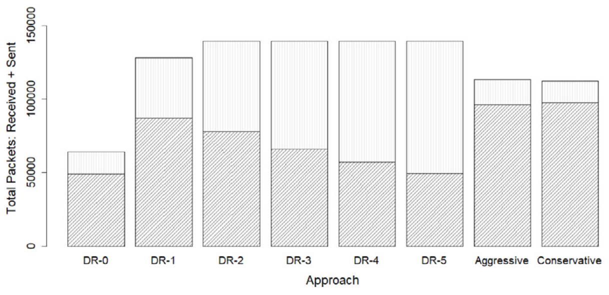


**Fig. 7.** Total packets received plus sent per approach — Fixed Data Rate, Adaptive Aggressive and Conservative.

**Table 3**
Result per experiment setup.

| Approach | Packets sent | PDR% |
|---|---|---|
| Fixed DR-5 | 139500 | 35% |
| Fixed DR-4 | 139500 | 41% |
| Fixed DR-3 | 139500 | 47% |
| Fixed DR-2 | 139500 | 56% |
| Fixed DR-1 | 128216 | 68% |
| Fixed DR-0 | 64139 | 76% |
| ECAS-Aggressive | 113309 | 85% |
| ECAS-Conservative | 112231 | 87% |

The simulation results showed that the dynamic adaptation of network parameters can make the communication process more efficient. Comparing the best results of the fixed and adaptive approaches, it was achieved an 11% increase in the total packets delivered, while simultaneously reducing the total packets sent by 12%. From these results, it is possible to infer that the use of the adaptive approach, in addition to making the communication process more efficient, also has a potential to reduce resource/energy consumption.

Several works report the issue of interference caused by environmental factors in wireless network technologies [50,51][52,53]. Although the LoRa technology was used to develop a proof-of-concept in this work, the proposed system is technology agnostic. Therefore, in addition to the feasibility of implementing ECAS in an existing sensor network, it is possible to use the system to assist in configuring the parameters of various wireless network technologies.

Issues related to security such as confidentiality, integrity, and authenticity, despite their importance in military networks, were left outside the scope of this work. However, the technologies used in this work already address these issues in other layers that can be implemented to provide security for the proposed system [54].

As future work, the plan is to implement ECAS on existing IoT platforms like FIWARE, as presented in **Session** 5. Another factor to be evaluated in the system is the impact of temperature on signal quality. The ECAS should adapt the network configuration parameters analyzing multiple climatic factors, besides the rainfall attenuation considered in this work. Finally, ECAS will be implement using real-world devices to validate in practice the system's efficiency in terms of resilience, network survivability, as well as energy consumption.

## CRediT authorship contribution statement

**Guilherme Rotth Zibetti:** Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Writing – original draft. **Juliano Araujo Wickboldt:** Conception and design of study, Analysis and/or interpretation of data, Writing – review & editing. **Edison Pignaton de Freitas:** Conception and design of study, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This study was partially funded by CAPES, Brazil - Finance Code 001. We also thank the funding of CNPq, Brazil, Research Productivity Scholarship grants ref. 313893/2018-7, ref. 309505/2020-8 and Universal research project ref. 420109/2018-8. We also thank to the Brazilian Army for the support provided via the research project S2C2, ref. 2904/20. This work was also supported by the São Paulo Research Foundation (FAPESP), Brazil grant # 2020/05182-3. Approval of the version of the manuscript to be published (the names of all authors must be listed): G. R. Zibetti, J. A. Wickboldt, E. P. Freitas.

## References

[1] M. Conti, S.K. Das, C. Bisdikian, M. Kumar, L.M. Ni, A. Passarella, G. Roussos, G. Tröster, G. Tsudik, F. Zambonelli, Looking ahead in pervasive computing: Challenges and opportunities in the era of cyber–physical convergence, Pervasive Mob. Comput. 8 (1) (2012) 2–21.

[2] E. Borgia, The internet of things vision: Key features, applications and open issues, Comput. Commun. 54 (2014) 1–31.

[3] R. Dobrescu, D. Merezeanu, S. Mocanu, Context-aware control and monitoring system with IoT and cloud support, Comput. Electron. Agric. 160 (2019) 91–99.

[4] C. Kamienski, M. Jentsch, M. Eisenhauer, J. Kiljander, E. Ferrera, P. Rosengren, J. Thestrup, E. Souto, W.S. Andrade, D. Sadok, Application development for the internet of things: A context-aware mixed criticality systems development platform, Comput. Commun. 104 (2017) 1–16.

[5] D.E. Zheng, W.A. Carter, Leveraging the Internet of Things for a more Efficient and Effective Military, Rowman & Littlefield, 2015.

[6] A. Kott, A. Swami, B.J. West, The internet of battle things, Computer 49 (12) (2016) 70–75, http://dx.doi.org/10.1109/MC.2016.355.

[7] I. Zacarias, L.P. Gaspary, A. Kohl, R.Q.A. Fernandes, J.M. Stocchero, E.P. de Freitas, Combining software-defined and delay-tolerant approaches in last-mile tactical edge networking, IEEE Commun. Mag. 55 (10) (2017) 22–29, http://dx.doi.org/10.1109/MCOM.2017.1700239.

[8] T. Plesse, C. Adjih, P. Minet, A. Laouiti, A. Plakoo, M. Badel, P. Muhlethaler, P. Jacquet, J. Lecomte, OLSR performance measurement in a military mobile ad hoc network, Ad Hoc Netw. 3 (5) (2005) 575–588.

[9] D. Patel, M. Won, Experimental study on low power wide area networks (LPWAN) for mobile internet of things, in: 2017 IEEE 85th Vehicular Technology Conference, VTC Spring, IEEE, 2017, pp. 1–5.

[10] D. Singh, G. Tripathi, A.M. Alberti, A. Jara, Semantic edge computing and IoT architecture for military health services in battlefield, in: 2017 14th IEEE Annual Consumer Communications & Networking Conference, CCNC, IEEE, 2017, pp. 185–190.

[11] B. Jalaian, T. Gregory, N. Suri, S. Russell, L. Sadler, M. Lee, Evaluating lorawan-based IoT devices for the tactical military environment, in: 2018 IEEE 4th World Forum on Internet of Things, WF-IoT, IEEE, 2018, pp. 124–128.

[12] C.A. Boano, M. Cattani, K. Römer, Impact of temperature variations on the reliability of lora, in: Proceedings of the 7th International Conference on Sensor Networks, SCITEPRESS-Science and Technology Publications, Lda, 2018, pp. 39–50.

[13] C.A. Boano, J. Brown, Z. He, U. Roedig, T. Voigt, Low-power radio communication in industrial outdoor deployments: The impact of weather conditions and ATEX-compliance, in: International Conference on Sensor Applications, Experimentation and Logistics, Springer, 2009, pp. 159–176.

[14] R. Marfievici, A.L. Murphy, G.P. Picco, F. Ossi, F. Cagnacci, How environmental factors impact outdoor wireless sensor networks: a case study, in: 2013 IEEE 10th International Conference on Mobile Ad-Hoc and Sensor Systems, IEEE, 2013, pp. 565–573.

[15] J.P. Sterbenz, D. Hutchison, E.K. Çetinkaya, A. Jabbar, J.P. Rohrer, M. Schöller, P. Smith, Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines, Comput. Netw. 54 (8) (2010) 1245–1265.

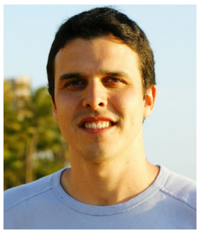
[16] E.K. Cetinkaya, J.P. Sterbenz, A taxonomy of network challenges, in: 2013 9th International Conference on the Design of Reliable Communication Networks, (DRCN), IEEE, 2013, pp. 322–330.

[17] D. Hutchison, J.P. Sterbenz, Architecture and design for resilient networked systems, Comput. Commun. 131 (2018) 13–21.

[18] B. Schilit, N. Adams, R. Want, Context-aware computing applications, in: 1994 First Workshop on Mobile Computing Systems and Applications, IEEE, 1994, pp. 85–90.

[19] J.P. Sterbenz, E.K. Çetinkaya, M.A. Hameed, A. Jabbar, S. Qian, J.P. Rohrer, Evaluation of network resilience, survivability, and disruption tolerance: analysis, topology generation, simulation, and experimentation, Telecommun. Syst. 52 (2) (2013) 705–736.
[20] P. Pradeep, S. Krishnamoorthy, The MOM of context-aware systems: A survey, Comput. Commun. 137 (2019) 44–69.
[21] P. Pradeep, S. Krishnamoorthy, R.K. Pathinarupothi, A.V. Vasilakos, Leveraging context-awareness for internet of things ecosystem: Representation, organization, and management of context, Comput. Commun. (2021).
[22] S.-J. Park, R. Sivakumar, Quantitative analysis of transmission power control in wireless ad-hoc networks, in: Proceedings. International Conference on Parallel Processing Workshop, IEEE, 2002, pp. 56–63.
[23] S. Narayanaswamy, V. Kawadia, R.S. Sreenivas, P. Kumar, Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the COMPOW protocol, in: European Wireless Conference, vol. 2002, Florence, Italy, 2002, pp. 156–162.
[24] M. Kubisch, H. Karl, A. Wolisz, L.C. Zhong, J. Rabaey, Distributed algorithms for transmission power control in wireless sensor networks, in: 2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003, vol. 1, IEEE, 2003, pp. 558–563.
[25] F. Xue, P.R. Kumar, The number of neighbors needed for connectivity of wireless networks, Wirel. Netw. 10 (2) (2004) 169–181.
[26] S. Lin, F. Miao, J. Zhang, G. Zhou, L. Gu, T. He, J.A. Stankovic, S. Son, G.J. Pappas, ATPC: adaptive transmission power control for wireless sensor networks, ACM Trans. Sensor Netw. 12 (1) (2016) 1–31.
[27] J. Luomala, I. Hakala, Effects of temperature and humidity on radio signal strength in outdoor wireless sensor networks, in: 2015 Federated Conference on Computer Science and Information Systems (FedCSIS), IEEE, 2015, pp. 1247–1255.
[28] J. Rak, A new approach to design of weather disruption-tolerant wireless mesh networks, Telecommun. Syst. 61 (2) (2016) 311–323.
[29] D. Papakostas, S. Eshghi, D. Katsaros, L. Tassiulas, Energy-aware backbone formation in military multilayer ad hoc networks, Ad Hoc Netw. 81 (2018) 17–44.
[30] K. Poularakis, Q. Qin, E.M. Nahum, M. Rio, L. Tassiulas, Flexible SDN control in tactical ad hoc networks, Ad Hoc Netw. 85 (2019) 71–80.
[31] M. Mishra, D. Sidoti, G.V. Avvari, P. Mannaru, D.F.M. Ayala, K.R. Pattipati, D.L. Kleinman, A context-driven framework for proactive decision support with applications, IEEE Access 5 (2017) 12475–12495.
[32] K. Lin, F. Xia, C. Li, D. Wang, I. Humar, Emotion-aware system design for the battlefield environment, Inf. Fusion 47 (2019) 102–110.
[33] A. Castiglione, K.-K.R. Choo, M. Nappi, S. Ricciardi, Context aware ubiquitous biometrics in edge of military things, IEEE Cloud Comput. 4 (6) (2017) 16–20.
[34] Y.-W. Moon, H.-S. Jung, C.-S. Jeong, Context-awareness in battlefield using ubiquitous computing: Network centric warfare, in: 2010 10th IEEE International Conference on Computer and Information Technology, IEEE, 2010, pp. 2873–2877.
[35] G.M. Leal, I. Zacarias, J.M. Stocchero, E.P.d. Freitas, Empowering command and control through a combination of information-centric networking and software defined networking, IEEE Commun. Mag. 57 (8) (2019) 48–55, http://dx.doi.org/10.1109/MCOM.2019.1800288.
[36] FIWARE, About-us, 2021, FIWARE URL https://www.fiware.org/about-us/.
[37] FIWARE, Developers, 2021, FIWARE URL https://www.fiware.org/developers/.
[38] FIWARE, FIWARE/IoT-agents, 2021, GitHub URL https://github.com/FIWARE/catalogue/tree/master/iot-agents.
[39] FIWARE, FIWARE-NGSI v2 specification, 2018, FIWARE URL https://fiware.github.io/specifications/ngsiv2/stable/.
[40] FIWARE, FIWARE/tutorials.IoT-sensors, 2021, GitHub URL https://github.com/FIWARE/tutorials.IoT-Sensors/tree/NGSI-v2.
[41] FIWARE, FIWARE-monitoring-geri, 2016, FIWARE URL https://fiware-monitoring.readthedocs.io/en/develop/.
[42] FIWARE, Catalogue, 2021, FIWARE URL https://www.fiware.org/developers/catalogue/.
[43] K. Liu, J. Peng, J. Wang, W. Liu, Z. Huang, J. Pan, Scalable and adaptive data replica placement for geo-distributed cloud storages, IEEE Trans. Parallel Distrib. Syst. 31 (7) (2020) 1575–1587.
[44] K. Liu, J. Wang, Z. Liao, B. Yu, J. Pan, Learning-based adaptive data placement for low latency in data center networks, in: 2018 IEEE 43rd Conference on Local Computer Networks, LCN, IEEE, 2018, pp. 142–149.
[45] S. Sun, NYUSIM User Manual, New York University and NYU WIRELESS, 2017.
[46] R.S. Sinha, Y. Wei, S.-H. Hwang, A survey on LPWA technology: LoRa and NB-IoT, Ict Express 3 (1) (2017) 14–21.
[47] L. Leonardi, F. Battaglia, L.L. Bello, RT-LoRa: A medium access strategy to support real-time flows over LoRa-based networks for industrial IoT applications, IEEE Internet Things J. 6 (6) (2019) 10812–10823.
[48] O. Khutsoane, B. Isong, N. Gasela, M. Abu-Mahfouz, WaterGrid-sense: A loRa-based sensor node for industrial IoT applications, IEEE Sens. J. 20 (5) (2019) 2722–2729.
[49] M. Bor, J. Vidler, U. Roedig, LoRa for the internet of things, in: Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks, in: EWSN '16, Junction Publishing, USA, ISBN: 9780994988607, 2016, pp. 361–366.
[50] G. Anastasi, A. Falchi, A. Passarella, M. Conti, E. Gregori, Performance measurements of motes sensor networks, in: Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2004, pp. 174–181.
[51] J.F. Federici, J. Ma, L. Moeller, Review of weather impact on outdoor terahertz wireless communication links, Nano Commun. Netw. 10 (2016) 13–26.
[52] J. Rangarajan, K. Baskaran, Evaluating the impact of weather condition on MANET routing protocols, Int. J. Electr. Eng. Inf. 7 (3) (2015) 454.
[53] R. Bruzgiene, L. Narbutaite, T. Adomkus, P. Pocta, P. Brida, J. Machaj, E. Leitgeb, P. Pezzei, H. Ivanov, N. Kunicina, et al., Quality-driven schemes enhancing resilience of wireless networks under weather disruptions, in: Guide To Disaster-Resilient Communication Networks, Springer, 2020, pp. 299–326.
[54] L. Alliance, Lorawan 1.1 specification, LoRa Alliance 11 (2017).

**Guilherme Rotth Zibetti** (guilherme@ufrgs.br) is an M.Sc. student in computer networks at the Federal University of Rio Grande do Sul (UFRGS), Brazil. He holds a degree in Computer Networks from Faculdade Senac Porto Alegre, Brazil, 2019. His research interests are currently related to wireless networks, sensor networks, software-defined networks, ad hoc network, and the Internet of Things.



**Juliano Araujo Wickboldt** is an associate professor at the Federal University of Rio Grande do Sul (UFRGS) in Brazil. He holds both M.Sc. (2010) and Ph.D. (2015) degrees in computer science from UFRGS. Juliano was an intern at NEC Labs Europe in Heidelberg, Germany for one year between 2011 and 2012. In 2015, Juliano was a visiting researcher at the Waterford Institute of Technology in Ireland. His research interests include softwarized networking, IoT, and 5G technologies.

**Edison Pignaton de Freitas** is a computer engineer graduated from the Military Institute of Engineering in Rio de Janeiro, Brazil (2003) He received an M.Sc. degree in computer science from the Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil (2007), and the Ph.D. degree in computer science and engineering in the area of wireless sensor networks from Halmstad University, Halmstad, Sweden (2011). Currently, he holds an Associate Professor with UFRGS, affiliated to the Graduate Programs in Computer Science and Electrical Engineering, acting mainly in the following areas: Wireless sensor networks, real-time and embedded systems, avionics, and unmanned vehicles.