1 2 9 0

UNIVERSIDADE D
COIMBRA

Marco André Temprilho Domingues

Interoperable and Secure IoT Architecture for Digital Healthcare: Wireless Monitoring of Untethered Patients in Smart Beds

Dissertation within the scope of the Master's in Electrical and Computer Engineering, specialization in Computers with subspecialization in Computational Learning, supervised by Doctor David Bina Siassipour Portugal, Prof. Doctor Paulo José Monteiro Peixoto and Eng. José Nuno da Cruz Faria and presented to the Faculty of Science and Technology of University of Coimbra at the Department of Electrical and Computer Engineering.

September 2022

· U C ·

FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Interoperable and Secure IoT Architecture for Digital Healthcare: Wireless Monitoring of Untethered Patients in Smart Beds

Marco André Temprilho Domingues

Coimbra, September 2022

Interoperable and Secure IoT Architecture for Digital Healthcare: Wireless Monitoring of Untethered Patients in Smart Beds

Supervisor:

Doctor David B. S. Portugal

Co-Supervisors:

Prof. Doctor Paulo Peixoto
Eng. José Faria

Jury:

Prof. Doctor Jorge Miguel Sá Silva

Prof. Doctor Jorge Nuno de Almeida e Sousa Almada Lobo

Prof. Doctor Paulo José Monteiro Peixoto

Dissertation submitted in partial fulfillment for the degree of Master of Science in Electrical and Computer Engineering.

Coimbra, September 2022

Acknowledgements

I would like to thank the following people, without whom I would not have been able to complete this research, and without whom I would not have been able to complete my Master's degree!

First of all, I would like to thank Dr. David Portugal, who challenged me and gave me the opportunity to address this important topic, and who set high expectations for the work I ended up doing as my dissertation and within the WoW project. Thanks to his guidance and support, I can say that I never felt lost during the development of the project. It was a great pleasure to work with him and I hope he knows that I will never forget the way he helped me. I would also wish to sincerely thank my supervisor, Prof. Dr. Paulo Peixoto, for his feedback on my dissertation and his important and fundamental point of view.

Of course, the completion of this work would have been impossible without the supervision of Eng. José Faria. I will always be grateful to him for his help and guidance with all the technologies used in the project. I would also take this opportunity to wish him the best of success in his career as a researcher.

My special thanks go to my family, my mother who encouraged me to pursue my goals, my father who, although living abroad, has always supported me, and especially my grandmother Maria, who has always been like a mother to me and helped me with everything I needed throughout the years. A special tribute goes to my dear girlfriend, Carolina Neves, who is always there for me, even when she has no idea of what I am talking about.

Many thanks to my colleagues Diogo, Marta, Nuno and Ulisses during these years of the course. Marta for being my teammate during the most difficult projects, and Diogo for being my closest friend who has always supported me during my most difficult times and for making this research period a lot easier.

Finally, I would like to thank my friends Pedro and Fábio for always being emotionally supportive and helping me clear my thoughts after a long day of work. I would also like to thank my best friend from childhood, João Castelhano, for never leaving my side and wish him every success in his career as a future doctor and my friend João Pedro, who is always willing to help and is an expert in everything.

Resumo

O envelhecimento da população, juntamente com o aumento de doenças crónicas, está a colocar uma pressão enorme nos sistemas de saúde atuais, tendo sido observada uma necessidade clara de aliviar esta pressão. Tendo em conta que os dispositivos de Internet das Coisas (IoT) permitem recolher sinais vitais de doentes sem estes saírem de suas casas, a IoT é vista como uma grande promessa para o desenvolvimento de sistemas de monitorização remota de pacientes. Apesardos benefícios potenciais da Internet das Coisas na área da saúde, ainda existem diversos obstáculos para a sua aplicação, tais como a interoperabilidade e a segurança.

Esta dissertação foi desenvolvida no âmbito do projecto WoW, que propõe o desenvolvimento de um sistema de monitorização de pacientes sem fios interligado a um sistema de informação hospitalar (HIS). Este trabalho tem como principal objetivo consolidar e integrar os diversos componentes da arquitetura IoT proposta, com especial ênfase na interoperabilidade e segurança das comunicações.

Inicialmente, a arquitetura IoT é materializada, consolidando os componentes de aquisição Bluetooth Low Energy (BLE) e transmissão de dados por MQTT, bem como o a Interface Gráfica do Utilizador (GUI). Também são implementadas várias funcionalidades para melhorar a componente de transmissão de dados, nomeadamente um mecanismo de redundância, uma funcionalidade keep-alive, e uma de emparelhamento. Além disso, uma camada de tradução usando o protocolo FHIR, um dos mais utilizados para a transmissão de informação de saúde no ambiente de software clínico, é concebida e desenvolvida para promover interoperabilidade com o sistema de informação hospitalar. Por fim, é realizada uma análise de vulnerabilidade de todo o sistema para definir melhorias a serem implementadas.

Devido a circunstâncias externas, testes em cenários reais não puderam ser realizados. No entanto, foi realizado um teste completo do sistema num ambiente controlado, que indicou claras melhorias na fiabilidade e robustez no sistema em consequência dos desenvolvimentos apresentados nesta dissertação.

Palavras-chave: Internet das Coisas; Cuidados de Saúde; Bluetooth Low Energy; MQTT; FHIR; Segurança; Interoperabilidade; Redundância.

Abstract

The aging population, along with the increase in chronic diseases, is putting substantial strain on traditional healthcare systems, and a clear need to relieve this strain has been observed. Considering that Internet of Things (IoT) devices make it possible to collect vital signs from patients without them having to leave their home, IoT is seen as a great promise for the development of remote patient monitoring systems. In spite of the potential benefits of IoT is very beneficial for healthcare, there are still several barriers to its application, including interoperability and security.

This dissertation has been developed in the scope of the WoW project, which proposes the development of a wireless patient monitoring system interconnected to a Hospital Information System (HIS). The main objective of this work is to consolidate and integrate the various components of the proposed IoT architecture, with special emphasis on interoperability and communications security.

Initially, the IoT architecture is materialized, consolidating the Bluetooth Low Energy (BLE) acquisition and Message Queuing Telemetry Transport (MQTT) data transmission components, as well as the Graphical User Interface (GUI). Several features are also implemented to enhance the data transmission component, namely a redundancy mechanism, a keep-alive feature, and a pairing feature. In addition, a translation layer using the Fast Healthcare Interoperability Resources (FHIR) protocol, one of the most widely used for transmitting health information in the clinical software environment, is designed and developed to promote interoperability with the HIS. Finally, a system-wide vulnerability analysis is performed to define improvements to be implemented.

Due to external circumstances, tests in real scenarios could not be performed. However, a full system test in a controlled environment was performed, which demonstrated a clear improvement in the system's reliability and robustness as as result of the developments presented in this dissertation.

Keywords: Internet of Things; Healthcare; Bluetooth Low Energy; MQTT; FHIR; Security; Interoperability; Redundancy.

"If you think that the internet has changed your life, think again. The Internet of Things is about to change it all over again!"

— Brendan O'Brien

Contents

List of Acronyms

AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
AWS	Amazon Web Services
BLE	Bluetooth Low Energy
BLEAK	Bluetooth Low Energy platform Agnostic Klient
CHUC	Centro Hospitalar e Universitário de Coimbra
CoAP	Constrained Application Protocol
DDS	Data Distribution Service
ECG	Electrocardiogram
EHR	Eletronic Health Record
FHIR	Fast Healthcare Interoperability Resources
FTPS	File Transfer Protocol over SSL
GATT	Generic ATTribute Profile
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HIS	Hospital Information System
HL7	Health Level Seven
HTTP	Hypertext Transfer Protocol

HTTPS	Hypertext Transfer Protocol Secure
IAM	Identity and Access Management
ICT	Information Communications Technology
IMU	Inertial Measurement Unit
IoT	Internet of Things
IP	Internet Protocol
JSON	JavaScript Object Notation
LoRa	Long Range
LPWAN	Low Power Wide Area Network
LTE-M	Long Term Evolution for Machines
M2M	Machine To Machine
MPU	Microprocessing Unit
MQTT	Message Queuing Telemetry Transport
NFC	Near Field Communication
PDA	Personal Digital Assistant
QoS	Quality of Service
RBAC	Role Based Access Controls
REST	Representational state transfer
RFID	Radio-Frequency Identification
SSL	Secure Sockets Layer
SQL	Structured Query Language
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UUID	Universally Unique Identifier
Wi-Fi	Wireless Fidelity
WoW	Wireless biOmonitoring stickers and smart bed architecture: toWards Untethered Patients
XML	Extensible Markup Language

List of Figures

List of Tables

1 Introduction

1.1 Context and Motivation

The aging population, along with the rise of chronic illness, is putting substantial strain on traditional healthcare systems, requiring a large increase in resources, from hospital beds to healthcare personnel. According to studies from the World Health Organization [1], it is predicted that there will be 2 billion elderly people by 2050 (22% of the world population). Around 89 % of the elderly are expected to live independently, however, research surveys point that about 80% of these currently suffer from at least one chronic disease [2], and have difficulty in taking care of themselves. Accordingly, ensuring a reasonable quality of life for the elderly has become a significant societal problem. Since traditional health monitoring approaches are difficult and time-consuming for all parties involved [3], there has been a growing development of digital healthcare solutions to relieve the strain on hospital systems and healthcare professionals, enhance treatment quality, and contribute to lower healthcare costs by keeping patients out of hospitals for regular care.

The IoT holds great promise for the development of remote healthcare monitoring systems. IoT devices can gather health data such as heart rate, blood pressure, temperature, and more from remotely located patients, removing the need for patients to travel to doctors or collect and send the measurements themselves. When an IoT device captures patient data, it has the potential to make it available to a software application that healthcare professionals and/or patients may access. Moreover, Artificial Intelligence (AI) algorithms can be used to evaluate data in order to make treatment recommendations or trigger warnings. An IoT sensor, for example, that identifies a patient's exceptionally low heart rate can be used to trigger an alarm so that healthcare professionals may intervene timely. While IoT can be highly beneficial to healthcare, it faces significant challenges currently, such as interoperability and security [4].

Interoperability is described as "*the degree to which two products, programs, etc. can be used together, or the quality of being able to be used together*", according to the Cambridge

Dictionary [5].

It is fundamental in the IoT context, since devices are generally characterized by a high degree of heterogeneity [6], and may generate massive volumes of data from multiple sources [7]. Furthermore, according to the McKinsey Global Institute [8], without interoperability, at least 40% of the possible benefits of IoT cannot be realized. This is evident given that transparent integration and connectivity of various IoT systems and their components would significantly simplify their deployment, maximize performance, and ease their operation alongside other systems.

In the case of medical devices, interoperability enables the remote monitoring of various body vital signals such as heart rate, blood pressure, or breath rate via wearable sensors. System interconnection and integration are required in order to gather, analyze, and utilize large volumes of data from disparate sources.

Security is "*the protection of information against being stolen or utilized incorrectly or illegally*", according to the Cambridge Dictionary [5]. Security is one of the most important factors for obtaining a reliable system, and lack of security can cause serious risks.

When connected, an IoT device can become a potential threat. In fact, 32% of IT leaders consider security as a major obstacle to IoT success [9].

Hacker attacks may expose patients' personal information, and, most importantly, threaten patients' personal safety. According to S. Alder [10], a total of 1,531,855 records were leaked across 39 healthcare data breaches in February 2020 alone. In order to reduce the chances of data breaches, it is necessary to develop efficient security measures to ensure the confidentiality and integrity of the patient's diagnostic data transmitted and received from the IoT environment.

1.2 Objectives

During the first year of the WoW R&D project [11], researchers at the Institute of Systems and Robotics (ISR) developed non-intrusive wearable devices, designated as *Biostickers*, which are electronic patches equipped with sensors that acquire and transmit patient's vital signs to a *Smart box* wirelessly in real time. The Smart box encompasses a single board computer (Raspberry Pi 4 Model B) that communicates with the Biostickers through Bluetooth Low Energy (BLE); and a *Gateway*, that connects the Smart boxes to the Globalcare Hospital Information System (HIS) and is in charge of managing users and data in the context of the project.

The primary goal of this dissertation work is to materialize the WoW system architecture, with a special emphasis on communication interoperability and security, as these are the most significant challenges to IoT as previously stated.

More precisely, the key objectives are:

1. Study the state of the art in IoT and IoT for Digital Healthcare, examining related works and identifying relevant gaps in the literature;
2. Characterize and consolidate the BLE data acquisition from the Biostickers;
3. Implement appropriate sensor data transmission from the Smart box to the Gateway using the MQTT protocol and implement features such as redundancy mechanisms and keep alive features;
4. Redesign/Improve the system's GUI focusing on responsiveness and usability;
5. Improve the security and interoperability of the overall IoT system;
6. Evaluate the performance of the system through extensively lab testing;
7. Support the project prototypes and carry out experimental validation in real scenarios at the Centro Hospitalar e Universitário de Coimbra (CHUC).

1.3 Document outline

This chapter establishes the motivation for this dissertation and the main objectives. The remainder of the document addresses the work performed and is organised as follows: Chapter 2 provides an overview of the state of the art in the Internet of Things, with a particular focus on digital healthcare and includes a survey of related literature, focusing on interoperability and security. The premise of the WoW project and the contributions of this dissertation to the project are described in detail in chapter 3. Chapter 4 describes how interoperability and security are achieved. Chapter 5 provides detailed results and discussion on the experiments and validations performed on the IoT system. Chapter 6 concludes the dissertation by summarising the main results and pointing out limitations and future possibilities for further improvement of the system.

2 Background

In this chapter, we take a deeper look into the concept of IoT and its application in healthcare, focusing on its benefits and current challenges.

According to Oracle [12], the **Internet of Things (IoT)** is a network of physical items ("things") that are equipped with sensors, software, and other technologies in order to connect and exchange data with devices and systems through the internet. These devices range in complexity from common household items to sophisticated industrial equipment. Experts predict that by 2025, there will be 22 billion linked IoT devices [12]. This technology enables automation in a variety of industries, as well as the collection of big data.

Hailed as the driver of the Fourth Industrial Revolution [13], the Internet of Things has already found applications in areas such as smart parking [14], smart agriculture [15], and smart water management [16]. In addition, research into the usage of IoT for building intelligent systems has been done in several fields including traffic management [17], health monitoring [18] and smart grids [19].

2.1 IoT in Digital Healthcare

IoT is changing the domain of device and human interaction in the delivery of healthcare solutions, revolutionizing this sector [20]. It has applications in healthcare that benefit patients, families, doctors and hospitals.

Patients and their families place a high value on privacy, accessibility, and comfort in hospital rooms, according to [21]. Security is also a top concern, and increased patient fulfilment may result in reduced stress and a faster recovery [22].

In hospital rooms, IoT technologies such as smart thermostats and customized lighting controls enable improved satisfaction for patients and control for caregivers [23]. Automated window curtains regulate illumination while allowing patients to obtain the physical and emotional benefits of sunlight [24]. Also, sensors in beds can track sleep patterns and alert personnel in case of unusual situations [25].

Other useful IoT devices include wearable technology [26], which can provide real-time data, trends, and warnings about possible health concerns, such as wearable fitness trackers [27], wearable Electrocardiogram (ECG) monitors [28], smart health watches [29], etc.

IoT devices and systems can also assist **physicians** in making more informed decisions and providing efficient patient care [30, 31]. An IoT system may combine information and create reports by uploading data directly and instantly to a cloud database [32], allowing clinicians to immediately detect trends.

Aside from monitoring patients' health, IoT devices are also beneficial in a variety of other areas in **hospitals**. Wheelchairs, defibrillators, nebulizers, oxygen pumps, various monitoring devices, or medical personnel deployment in several locations may all be tracked in real time utilizing sensor-enabled IoT devices [33].

For hospital patients, infection transmission is a major concern. Hygiene monitoring devices with IoT capabilities can assist in the prevention of infection in patients [34]. Asset management [35], such as pharmaceutical inventory control [36] and environmental monitoring, such as temperature and relative humidity applied to product maintenance in hospitals or pharmaceutical entities [37], can also be aided by IoT devices.

2.1.1 Advantages of IoT in healthcare

The advantages and beneficial effects of IoT in healthcare are diverse and varied.

- Lower expenses: Patients who are not in immediate danger can stay at home and utilize cloud-connected medical IoT devices to collect, track, and transfer health data to a medical facility. Patients can also plan e-visits with nurses and physicians using telehealth technology instead of going to the hospital [38];

- Better treatment of patients: With an IoT-based health monitoring system, doctors can access real-time data about patients via technologies, such as cloud computing and medical device connection [39]. As a result, symptoms are assessed more quickly and intervention is provided earlier, resulting in improved care and prevention of severe diseases;

- Better disease control: Healthcare practitioners may keep track of patients using real-time data [39]. This means that they may be able to detect an illness before it becomes a severe problem;

- Fewer mistakes: IoT enables precise data gathering, automated operations, and reduced waste, but most significantly, it lowers the chance of human mistake;

- Remote patient care: Patients can be monitored in the privacy of their own homes thanks to Machine To Machine (M2M) technology [40]. Sensors can be attached to various medical devices (such as heart rate monitors) at the patient's bedside. The information obtained is forwarded to the hospital, where it is analyzed;

- Maintenance of medical devices: Health professionals may inspect the status of mobile equipment and patient updates in real time from anywhere through the Internet [33], saving maintenance time and increasing productivity;

- More trust towards doctors: According to [41], digital healthcare has the potential to improve the Doctor-Patient relationship. With the right technology, healthcare can become more transparent and accessible as patients can decide what is convenient for them, such as remote consultations.

2.1.2 Challenges of IoT in healthcare

When it comes to integrating IoT for healthcare facilities, healthcare organizations may encounter several challenges.

Data collection is one of the most significant challenge in healthcare, but it is also an important opportunity. Before deploying an IoT ecosystem in a healthcare environment, it is critical to define the system's design, data collection and utilization objectives to produce meaningful data reports. This sort of data is paramount to improve the overall healthcare experience [42].

Due to the features of the IoT environment, where diverse entities (e.g., actuators, sensors, platforms, frameworks, concepts, and users) are linked and exchange information, **interoperability** and **security** must be developed concurrently. Secure identification (e.g., authentication and authorization) and interoperable identification are necessary so that an IoT system can be designed and built without allowing losses of information. In addition, when data is transmitted between various IoT entities, it must also be translated without loss of information. An interoperable data exchange protocol as well as a secure transport technique are necessary to accomplish this. Finally, security and interoperability should be taken into account in different parts of the healthcare IoT system, such as architecture and framework design, platform development, and scenario creation.

2.2 Technologies, Protocols and standards used in IoT for healthcare

The most common architectures for IoT are based on the reference model created by the IoT World Forum (IWF) architecture committee in October 2014 [43]. This reference model, illustrated in Figure 2.1, provides a common framework to allow deploying IoT systems easily and quickly in the industry.

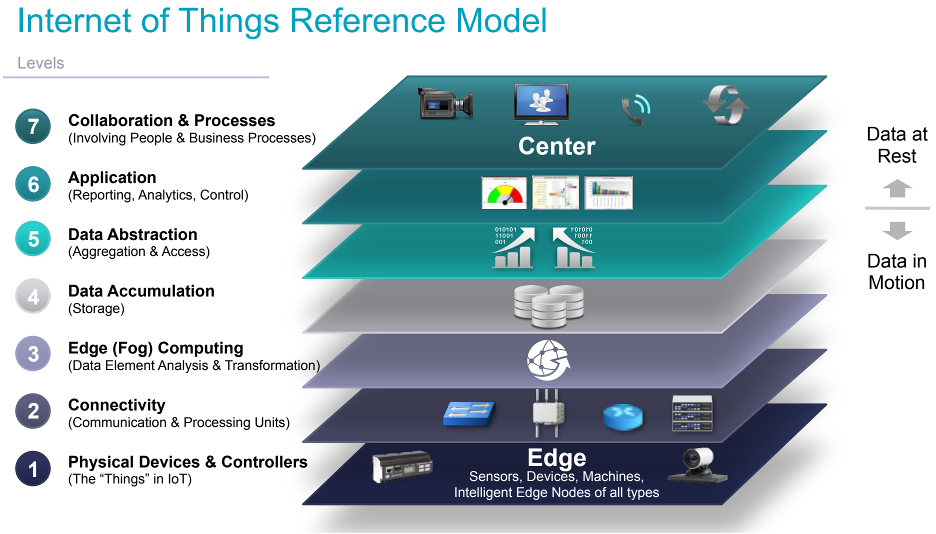

Internet of Things Reference Model
Levels
7 Collaboration & Processes
(Involving People & Business Processes)
6 Application
(Reporting, Analytics, Control)
5 Data Abstraction
(Aggregation & Access)
4 Data Accumulation
(Storage)
3 Edge (Fog) Computing
(Data Element Analysis & Transformation)
2 Connectivity
(Communication & Processing Units)
1 Physical Devices & Controllers
(The "Things" in IoT)
Center
Edge
Sensors, Devices, Machines,
Intelligent Edge Nodes of all types
Data at Rest
Data in Motion

Figure 2.1: IoT World Forum Reference Model. Source: Juxtology [43].

In the reminder of this section, we focus on technologies, protocols and standards to foster security and interoperability in IoT for healthcare, by reviewing important aspects related to connectivity, data accumulation, data abstraction and additional security and interoperability guidelines.

2.2.1 Networking Technologies

Different networking technologies are used in communications between devices and cloud services or gateways as shown in Table 2.1:

- Ethernet is the most commonly used Local Area Network (LAN) technology. It provides a wired communication to connect the devices to the Internet [44];
- Wireless Fidelity (Wi-Fi) is ideally suited for devices that need to transfer huge volumes of data yet do not care about power consumption [45]. As a result, it is adequate for tight spaces in buildings but an inadequate option for battery-powered gadgets;
- The Near Field Communication (NFC) protocol is used to exchange information between two electronic devices at very short distances [46];
- BLE is a light-weight subset of classic Bluetooth and was introduced as part of the Bluetooth 4.0 core specification. BLE popularity has risen in recent years, owing to its extremely low power consumption, which makes it ideal for many Personal Area

Network	Connectivity	Frequency	Data Rates (Min-Max)	Security Features
Ethernet [44]	Wired, Short-Range (Max 100 m)	Various	10 Mbps to 400 Gbps	IEEE 802.1AE
Wi-Fi [45]	Wireless, Short-Range (Max 45 m)	2.4 GHz or 5 GHz or 6 GHz	1 Mbps – 9.6 Gbps	RC4 stream cipher(WEP) + AES blockcipher + WPA2
NFC [46]	Wireless, Ultra-Short-Range (Max 20 cm)	13.56 MHz	up to 424 kbps	Non-Standard[1]
BLE [47]	Wireless, Short-Range (Max 400 m)	2.400 – 2.4835 GHz	125 kbps - 2 Mbps	AES-128 block cipher + AES-CCM
LPWAN [48]	Wireless, Long-Range (Max 40 km)	Various	0.3 kbps to 50 kbps per channel	Depends on Architecture[2]
ZigBee [49]	Wireless, Short-Range (Max 100 m)	2.4 GHz	250 kbps	Stream cipher + AES block cipher + CBC-MAC/ Extension of CCM
Cellular Networks [50, 51]	Wireless, Long-Range (Max 70 km)	900, 1800, 1900, 2100 MHz	2G (100-400 kbps) 3G (0.5-5 Mbps) 4G (1-50 Mbps) 5G (1-400 Mbps)	Layer two Tunnelling Protocol (L2TP) + Datagram Transport Layer Security (DTLS)
RFID [52, 53]	Wireless, Short-Range (Max 460 m)	125–134 kHz 13.56 MHz 860–960 MHz 2.45 GHz	4 kbps–804 kbps	Tame Transformation Signatures (TTS) Algorithm

[1] Despite not having a defined standard for NFC encryption, non-standard approaches have been discussed, as in [54], [55] and [56].

[2] For example, LoRa makes use of Over-The-Air-Activation (OTAA) or Activation by Personalisation (ABP) + AES-128 and Message Signing [57].

Table 2.1: Major networking technologies used in IoT.

Network (PAN) applications. BLE also supports point-to-point, star, and mesh topologies. It is appropriate for IoT applications since it operates with frequent data transfers, which greatly decreases battery use [47];

- ZigBee is based on the IEEE 802.15.4 radio standard. It became popular for monitoring applications in the smart home area due to its low power, low rata rate protocol, which enables mesh topology. It is commonly used to send little quantities of data across short distances. The mesh architecture allows range expansion by sending a packet across several nodes using a multi-hop mechanism [49];

- Radio-Frequency Identification (RFID) makes use of radio waves to send small quantities of data from an RFID tag to a reader over a short distance. Until recently, technology has enabled a significant shift in retail and logistics. RFID continues to be established in the retail industry, allowing new IoT applications such as smart shelves, self-checkout, and smart mirrors [52, 53].

In the field of healthcare, wearable devices are frequently organized in networks called Wireless Body Area Networks (WBANs). Since this is a short-range wireless network, BLE, ZigBee, RFID and Wi-Fi are the most commonly used protocols [58–60].

2.2.2 Communication Protocols

In order to properly share data over the numerous networking technologies specified in the previous subsection, a few communications protocols are available for IoT, allowing smooth data sharing. These communication protocols are employed at the application level, and as previously stated, they are used to transfer data across devices. Below, a list of the major protocols currently available in the IoT architectural layers is presented (see Table 2.2):

- Data Distribution Service (DDS) represents a M2M real-time messaging framework in IoT systems;

- Advanced Message Queuing Protocol (AMQP) provides server protocols via peer-to-peer data exchange;

- Constrained Application Protocol (CoAP) defines protocols for constrained devices that use low power and low memory, such as wireless sensors;

- MQTT represents the messaging protocol standard for low-powered devices using TCP/IP for seamless data communication.

2.2.3 Data Accumulation

Typical IoT systems create a huge amount of data (big data) [63], data generated by connected IoT devices is expected to reach 73.1 ZB by 2025, and companies are likely to have problems controlling the flow and storage of this data [64]. One solution to this problem is cloud computing, the ability to provide computing services over the internet, enabling companies to manage and analyze the data, therefore improving the overall efficiency and effectiveness of IoT systems [65].

According to [66], Microsoft Azure IoT Suite, Google Cloud's IoT Platform, IBM Watson IoT Platform and AWS IoT Platform are the most popular for IoT development.

	DDS	**AMQP**	**CoAP**	**MQTT**
Type	M2M and Device to Device	Middleware messaging	Web Transfer Protocol	Lightweight M2M
Communication Protocol	Publish/Subscribe and Request/Response	Point to Point, Publish/Subscribe	Request/Response	Publish/ Subscribe
Transport Layer Protocol	TCP/IP and UDP/IP	TCP/IP	UDP/IP	TCP/IP
Security	TLS & DTLS	SSL/TLS, SASL	DTLS	SSL/TTS
Data exchange	Bus-based	Broker based	Broker Based	Broker based

Table 2.2: Common messaging protocols used in IoT. Adapted from [61, 62].

One major challenge when it comes to IoT data storage is maintaining security in the face of potential threats. While organizations need to store and organize their IoT data effectively in order to derive insights from it, those same activities can make data a prime target for cyber criminals. In 2018, there were over 32.7 million malware attacks, and that number more than doubled in 2019 [67].

The following are some of the measures that companies should take to create secure IoT-cloud environments:

- Data encryption: Data kept on any type of storage medium, including backup devices and solid-state drives should be protected by an encryption method. To secure data at rest, many levels of encryption might be utilized [68], such as encryption of sensitive data before storage and encryption of the storage device itself. Data in transit is thought to be more vulnerable to security breaches. Consequently, it is critical to have an end-to-end security policy. Prior to transferring the data, encryption is activated to safeguard it in transit. Hypertext Transfer Protocol Secure (HTTPS), File Transfer Protocol over SSL (FTPS), Secure Sockets Layer (SSL) and Transport Layer Security (TLS) can be utilized for this;

- Protection of the data flow: The data flow from IoT endpoints to the cloud must be protected from the start, if the data arrives corrupted at the cloud, it can compromise the whole system;

- Regular check for vulnerabilities: Vulnerability testing can be used to discover errors. Enterprises can choose to undertake vulnerability testing throughout the entire ecosystem or just a single component, as long as they do it on a regular basis;

- Device identity: Each device in an IoT deployment should have its own device identification. This identity is used to verify and authorize secure connection with other components of the IoT ecosystem when a device goes online.

2.2.4 Data Abstraction

Once the data collection step is complete, it is necessary to simplify how the applications access the data, reconcile the different data stores, and ensure the information is complete and consistent [69].

Data abstraction is essential since it is the process where interoperability is introduced. Thus, it is critical to understand the information provided and how it is shared. In healthcare, someone might easily have health records distributed across many doctor's offices with the traditional approach of physical file folders. If there is a relocation to a new office, patient's files must be sent to the new office so that the doctors have a complete picture of their health. This can result in a variety of difficulties, including a patient forgetting their previous office's contact information, missing or incomplete papers, and difficult-to-read handwriting, among others.

With this in mind, Eletronic Health Record (EHR) were created. EHRs are patient-centered, real-time records that make information available to authorized users quickly and securely. While an EHR system does contain a patient's medical and treatment history, it is designed to go beyond typical clinical data collected in a provider's office and can encompass a broader perspective of a patient's care [70].

FHIR is a standard describing data formats and elements (known as "resources") and an API for exchanging EHR. The standard was created by the Health Level Seven (HL7) healthcare standards organization. The philosophy behind FHIR is to build a base set of resources that, either by themselves or when combined, satisfy the majority of common use cases. FHIR resources aim to define the information contents and structure for the core information set that is shared by most implementations [71]. FHIR is built on internet standards that are extensively used outside of the healthcare industry. The Representational state transfer (REST) method, for example, specifies how discrete packets of information may be readily exchanged. FHIR greatly lowers the barriers of adoption for new software developers to meet healthcare requirements by using current standards, technologies and common data formats, such as JavaScript Object Notation (JSON) and Extensible Markup Language (XML) [71].

2.2.5 Additional Security and Interoperability Guidelines

According to [72, 73], there are fundamental steps that can help to achieve a **secure** IoT system:

1. Ensure Data Encryption: Encryption renders data unreadable to anybody who does not have authorized access. Once data has been encrypted, a key is required to decrypt it, protecting it from unwanted access or usage. According to I. Hübschmann [74], the most commonly IoT encryption algorithms are Advanced Encryption Standard (AES), Data Encryption Standard (DES), Triple DES, Rivest–Shamir–Adleman (RSA) Algorithm, Digital Signature Algorithm (DSA) and Blowfish;

2. Minimize physical security threats to field devices, such as ensuring all devices are identified, using network segmentation to improve defense, adopting secure password practices and patching and updating firmware when available;

3. Secure end-to-end communications: Networks should act as a level playing field for users to install security measures that protect all connected devices;

4. Ensure devices remain accessible after deployment: Devices may require battery changes, repairs and maintenance over time. Security risks can be reduced by appropriately managing this before deploying the devices.

5. Security by Design: The only way to ensure effective IoT security is to build it into deployment from the start.

6. Consider different protocols: IoT devices interact using not only internet protocols, but also a diverse range of networking protocols, such as BLE, Wi-Fi, etc. To decrease risks and eliminate threats, developers must be aware of the whole variety of protocols utilized in their IoT systems;

7. Secure IoT-cloud convergence and deployment of cloud-based solutions to provide IoT edge devices with additional security and processing capabilities;

8. Assign an administrator of things to reduce security oversights and vulnerabilities.

In terms of **interoperability**, IoT interoperability can be seen from different perspectives [75]:

- **Device interoperability** refers to the capacity of devices to integrate and interoperate with multiple communication protocols and standards supported by heterogeneous IoT devices.

- **Network interoperability** refers to techniques that enable smooth message exchange across systems over multiple networks (networks of networks) for end-to-end communication.

- **Syntactical interoperability** refers to the interoperation of the format as well as the data structure utilized in any transferred information or service between heterogeneous IoT system components.

- **Semantic interoperability** is defined as "enabling different agents, services, and applications to exchange information, data and knowledge in a meaningful way, on and off the Web" [76].

- **Platform interoperability** difficulties in IoT occur as a result of the availability of many operating systems (OSs), such as, Contiki, RIOT, TinyOS, etc., programming languages, data formats (e.g. JSON, XML, etc.), architectures, and access mechanisms for things and data.

Researchers have used a variety of techniques and technologies to improve IoT interoperability [75]:

- Adapters/gateways address interoperability by developing an intermediary tool, sometimes known as mediators to provide a bridge between various specifications, data, standards, and middleware, among other things. The gateway can be enhanced with plug-ins to conduct protocol conversions between the protocol of the transmitting device and the protocol of the receiving device.

- Virtual networks/overlay-based solutions have been proposed under the "Managed Ecosystems of Networked Devices" (MENO) [77] with the goal of seamlessly connecting sensor and actuators, as well as other IP-smart objects to the Internet for end-to-end communication. MENO's core idea is to build a virtual network on top of real networks, allowing connection with various sorts of devices, such as sensor nodes. End-to-end communication is available within each virtual network using multiple protocols.

- Syntactic interoperability specifies the format and structure of the data. International standards development organizations (SDOs) promote the organized sharing of data. In the case of SDOs in healthcare revelant organizations include Health Level Seven International (HL7) and Integrating the Healthcare Enterprise (IHE).

- Semantic interoperability addresses terminologies, nomenclatures, and ontologies. For health data, standards such as FHIR and OpenEHR ensure that the meaning of concepts may be communicated across systems, resulting in a standard language for health that is, ideally, comprehensible by people and machines all around the world.

2.3 Related Work on IoT Healthcare Applications

This section presents an introduction on IoT healthcare applications, however since this dissertation focus on the development of a secure and interoperable IoT system, relevant related works regarding security and interoperability on healthcare IoT applications are primarily described and reviewed in this section.

In [58], Chaudhury et al. propose a healthcare monitoring system based on IoT. The proposed system monitors vital health parameters, such as body temperature, ECG and heart rate. This data is then forwarded to a base station or Gateway server via a Wi-Fi module and stored in the form of files, in order to be accessed anytime. There is also a GUI that displays this data in both graphical and text format, and a GSM modem to send a message to the caretaker in case of an extreme situation based on threshold values. Although the authors describe a real-time monitoring system, they also mention that the GUI has a refresh rate of 0.066 Hz (every fifteen seconds), which may not be the most appropriate strategy for monitoring systems, considering the significance of being able to check the patient's vital signals at any time. The authors also mention that this system can be used in hospitals but there is no mention to any integration with an HIS. Finally, the authors make no indication of what happens if the Wi-Fi connection fails.

In [59], Islam et al. propose a smart healthcare IoT system that can monitor a patient's basic health signs as well as the room condition where the patients are in real-time. In this system, five sensors are used to capture the data, such as an heartbeat sensor, a body temperature sensor, a room temperature sensor, a gas sensor and an air quality sensor. The data is then processed using an ESP32 module before being sent to a gateway server using Wi-Fi. The data is displayed using a mobile application called ThingSpeak. Although the authors mention that the proposed system monitors a patient's health signals, it is missing several of the most important ones, such as ECG and respiration. The authors mention use of authentication in order to access the application but do not mention any encryption while sending it through Wi-Fi. This raises serious privacy concerns, since a cyberattack could expose any private information that can be detrimental to the patients if revealed. Lastly, the authors also mention that the system is currently too bulky and that it may be necessary to combine these sensors into a smaller device to make the patients more comfortable.

Motivated by the COVID-19 pandemic, Raposo et al. [60] have developed an IoT system called "e-CoVig" used for monitoring COVID-19 patients during the quarantine. The physiological measurements are made using a specialized wearable device that sends the collected data to a mobile device through BLE or it can be manually inserted. The data acquired at the mobile application is sent automatically to the web/cloud application and made available in real-time to the medical staff. The BrainAnswer data visualization and monitoring platform serves as a cloud-based storage platform, analytics engine, and patient and healthcare professional dashboard interface. Unfortunately, the authors make no indication of what happens if the Wi-Fi connection fails. The ability to manually input measurements might potentially mislead medical professionals into believing there is a health problem with the patient. Finally, the authors mention that this system can be used in hospitals but they do not mention any possible integration with external healthcare systems.

Verisense by Shimmer is a commercial wearable sensing platform designed specifically for clinical trials. The Verisense IMU sensor allows for continuous remote monitoring of participants' activity and sleep. In addition, the Verisense Pulse+ sensor captures photoplethysmogram (PPG) and galvanic skin response (GSR), allowing the measurement of participants heart rate, oxygen saturation and GSR. This platform includes three main components: the Verisense sensor responsible by acquiring the vital signs; the Verisense Base Station, an Android device with an in-built app that receives data from the sensors through Bluetooth; and the Verisense Cloud Platform based on an AWS, which server collects data from base stations through cellular or Wi-Fi.

The Verisense platform typically only outputs activity and sleep data. However, it can be adapted to generate additional metrics for numerous clinical research applications such as gait and falls, Parkinson's tremor classification, etc., yet this requires a participant to use more sensors. Shimmer[1] also mentions that the base station notifies the participant and the web server of any connection problems, but does not mention any data synchronization with the cloud in case the Wi-Fi or cellular communications fail. Although this system does not typically integrate with external HIS, Shimmer mentions that the cloud platform is scalable and customizable and can be fully integrated with any existing EHR system. This IoT healthcare solution has twice been awarded "Best Clinical Trials Wearable Sensor Technology Company" and "Best Researcher Wearable Wireless Technology Provider 2019".

Although other commercial IoT systems could have been analyzed, this is outside the scope of this dissertation and therefore will not be discussed further.

[1] Shimmer, `https://shimmersensing.com`

2.3.1 Security

El Zouka and Hosni [78] propose a trusted environment to collect authenticated physiological data from a patient's body. Raw data is sent through a Global System for Mobile Communications (GSM) module to Azure IoT Hub, where it is converted into linguistic representation. With the help of a logic-based algorithm, which is trained in a fuzzy-based inference system (FBIS) to obtain the status of the patient. The FBIS is integrated with a secure healthcare monitoring system to retrieve the states of the patient and send it to a medical advisory for preliminary precautions.

In the aforementioned work, besides using GSM for data encryption, a secure, lightweight authentication and key agreement protocol is proposed to meet the requirements of healthcare professionals and their patients. It includes three phases: a registration phase, a login phase, and an authentication phase.

- The registration phase provides strong authentication between the patient and the provider within a strict policy framework.
- The login phase computes a one-time password and a session key by using a private encryption algorithm every time the server receives a registration request with a dynamic password and the patient ID, through a secure channel.
- The authentication Phase which after the key agreement and login processes are completed enables the patient and the server to encrypt/authenticate communications.

Wu et al. [79] present a hybrid wearable sensor network system with edge computing ability to promote safe working environments and reduce health risks in the construction industry. The proposed IoT infrastructure incorporates two networks: a WBAN for data collection using BLE and a Low Power Wide Area Network (LPWAN) for internet connection using Long Range (LoRa). The environmental conditions (temperature, humidity, UV and CO2) and vital signs (heart rate and body temperature) of the subject are measured by wearable sensors deployed in the WBAN. The data from individual sensors are transmitted using BLE within the WBAN, being collected and transmitted to a gateway using LoRa within the LPWAN. The gateway can act as a local server for edge computing, namely pre-processing sensor signals, displaying data and triggering alerts when an emergency occurs. Finally, an IoT cloud server is designed and implemented for data storage and further functionalities, such as web monitoring and support for mobile applications.

For data encryption, both the sender (Safe Node) and the receiver (Gateway) use the same cipher (Speck) and the same encrypt-key. The RHEncryptedDriver library[2] is used to

[2] RadioHead, `https://github.com/adafruit/RadioHead`

add encryption and decryption to the LoRa (RFM95) driver by using the Speck cipher. The aforementioned library encrypts the LoRa data using an encrypt-key and then transmits the data to the remote LoRa gateway. After the gateway receives the encrypted data, it decrypts the data using the same key.

Gope and Hwang [80] propose an architecture called BSN-Care, as illustrated in Figure 2.2. The BSN architecture is composed of wearable and implantable sensors. Each sensor node is integrated with bio-sensors, including Electrocardiogram, Electromyography, Electroencephalography, Blood Pressure, thermometer and motion. These sensors collect the physiological parameters and forward them to a coordinator called Local Processing Unit (LPU), which can be a portable device such as Personal Digital Assistant (PDA), smartphone etc. The LPU works as a router between the BSN nodes and the central server called BSN-Care server, making use of wireless communication mediums such as mobile networks 3G/CDMA/GPRS. Besides, when the LPU detects any abnormalities, it provides immediate alert to the person that is wearing the bio-sensors.

Security in this work is divided in two parts:

- Network security comprising authentication, anonymity, and secure localization.
- Data security including data privacy, data integrity, and data freshness.

To achieve the network security requirements proposed, a lightweight anonymous authentication protocol is developed. When a LPU sends the periodical updates to the BSN-Care server, the server needs to confirm the identity of the LPU using a lightweight anonymous authentication protocol. This proposed authentication protocol consists of two phases:

- Registration Phase: the BSN-Care server issues security credentials to a LPU through a secure channel.
- Anonymous Authentication Phase: before data is transmitted from the LPU to the BSN-Care server, both the LPU and the server will authenticate each other.

To meet the data security requirements with a reasonable computational overhead, the authors chose an authenticated encryption scheme in offset codebook (OCB) mode. OCB is well suited for fast and secure data communication, where only encryption can guarantee both security and integrity of data in a single pass, without the need for additional cryptographic primitives such as hash functions, MAC, or CRC. Therefore, OCB is also well suited for the power-constrained sensors of LPU devices.

K. Binu et al. [81] propose a smart gateway that safeguards the entire system using modified Host Identity Protocol Diet EXchange (HIP-DEX) key exchange protocol and a new key exchange scheme based on Low Energy Adaptive Clustering Hierarchy (LEACH) routing protocol. The proposed architecture can be seen at Figure 2.3.

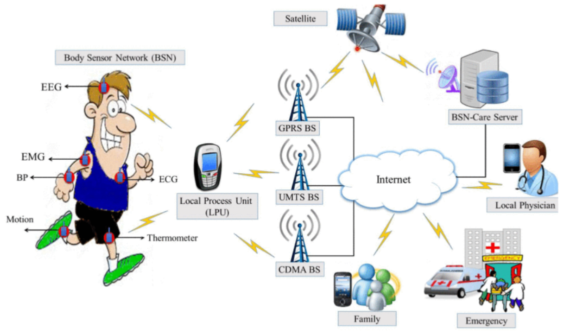


Figure 2.2: BSN-Care architecture. Source: Gope and Hwang [80].

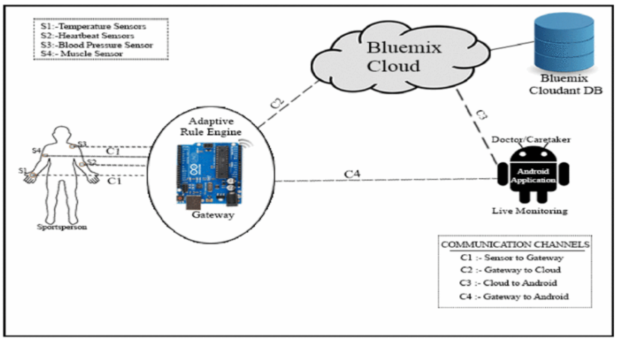


Figure 2.3: Bluemix System Architecture. Source: K. Binu et al. [81].

In the aforementioned work, the secure gateway is implemented using the Arduino MKRzero microcontroller board. To measure the biosignals, the authors have chosen to use temperature, heart rate, muscle and blood pressure sensors. The collected data is transmitted to the gateway, which is based on the cloud environment IBM Bluemix for data storage. The gateway will then analyze this data. When anomalies are detected, a notification is sent to the monitoring device through a fast and secure channel. End users, such as healthcare professionals, can monitor the collected data using an Android application developed by the authors.

To enable data security in the communication process the system was divided into four channels:

- Sensors to Gateway channel which makes use of pre-shared keys in order to secure this communication;
- Gateway to Cloud channel which makes use of a lightweight modified HIP-DEX key exchange protocol;
- Cloud to Android device channel which is also secured using the modified HIP-DEX protocol;
- Gateway to Android device channel which makes use of a key exchange protocol scheme based on LEACH routing protocol.

There are also three additional stages: the initial authentication process, smart device registration, and smart device key update. During these phases, the network is kept secure and data is constantly transmitted through a secure channel.

Yasin et al. [82] propose an ultra-low power and secure IoT sensing/pre-processing platform for prediction of ventricular arrhythmia (VA) using ECG signals. The ECG data is collected and processed on-chip, and only the relevant features are sent to the medical facility through the intermediate "programmer" device.

In order to secure the platform, the authors have developed a multi-layered defense scheme, comprising the following four layers:

- Prediction rate manipulation integrating logic locking to break the functionality of the VA processor when an incorrect key is applied;
- SAT attack resilience on logic locking, using SARLock;
- Denial of service which offers protection against hacking attacks, which can take control of the processor, by locking the processor FSM in an intermediate stage;
- ECG key generation by extracting the chip-specific key from the ECG signal. This key is further used to encrypt the communication on the telemetry interface.

A comparison of the key aspects of the works examined focusing on security is provided in Table 2.3. Aspects such as goals, sensors used, networking topologies, messaging protocols, data storage, data security and network security are presented.

Works	**Goals**	**Sensors Used**	**Networking Tecnologies**	**Messaging Protocols**	**Data Storage**	**Data Security**	**Network Security**
El Zouka and Hosni [78]	Measure and monitor patient physiological data and assess health status and fitness.	Temperature Pulse Oximeter Blood Pressure	WLAN	Not Mentioned	Azure IoT	GMS encryption	three-part lightweight authentication and key agreement protocol
Wu et al. [79]	Hybrid wearable sensor network system to improve safety and reduce health risks in the construction industry.	Temperature & Humidity UV and CO2 Heart Rate Temperature	WBAN: BLE LPWAN: LoRa	MQTT	DigitalOcean	RHEncryptedDriver data encryption by using the Speck cipher	Not mentioned
Gope and Hwang [80]	Body sensor network to detect abnormalities and provide alerts.	EEG and EMG and BP ECG and Motion Thermometer	Cellular Networks	Not Mentioned	Server Database	OCB encryption	Lightweight anonymous authentication protocol and safe localization
K. Binu et al. [81]	Smart gateway responsible of securing the entire system using HIP-DEX and key exchange scheme.	Temperature Heart Rate Muscle and Blood Pressure	Not Mentioned	Not Mentioned	IBM Bluemix	Not mentioned	HIP-DEX and LEACH key exchance protcols + authentication process + smart device registration
Yasin et al. [82]	Ultra-low power and secure IoT pre-processing platform for prediction of VA using ECG signals.	ECG	Not Mentioned	Not Mentioned	Not Mentioned	Encryption using chip-specific keys	SAT attack resilience + Denial of service + logic locking

Table 2.3: Comparison between the different pervasive healthcare applications reviewed with emphasis on security.

2.3.2 Interoperability

Alamri [83] proposes a semantic middleware that exploits ontology to support integration and functional collaborations between IoT healthcare Information Systems and EHR systems. The goal of the proposed model is to provide a interoperation middleware for IoT and EHR data that will facilitate data interoperability, integration, information search and retrieval, and automatic inference. To achieve this goal, a semantic middleware architecture is proposed which has three main components:

- Semantic EHR triplestore[3] that manages and store EHR data;
- Semantic IoT triplestore that stores and manages the data that is gathered by the IoT healthcare devices and sensors;
- Semantic integration process to integrate the EHR healthcare information with the IoT healthcare system, transforming the data subject (patient data) to semantic IoT triplestore.

According to the author, adopting a semantic EHR system simplifies information transmission, achieves interoperability, integrates IoT with EHR, and improves information search and retrieval. The work also addresses automated inference, which is possible thanks to the combination of the Web Ontology Language (OWL) and Semantic Sensor Network (SSN) ontologies. However, the author makes no indication of how the model manages complex data, since health data can be originated from several sources, which is a key component of semantic methods. Also, no approaches for data security are presented and there is no reference to tests applied to the middleware

Hong et al. [84] propose an interconnected Personal health record (PHR) system built in adherence with healthcare data communication standards and an IoT Cloud platform. This system includes an interoperable hospital information system to store electronic medical records (EMRs) and transfer them to a PHR system via email, a public cloud that supports encrypted data sharing for big data analysis services, a PHR gateway repository based on an IoT module which communicates with hospitals and public clouds, and a mobile application to manage and view PHRs. This system can store and share raw EMR and life log data like consumed calories, step count and calories burned from exercise based on the healthcare communication standard HL7 FHIR.

The authors do not go into details on how they would implement features such as, machine learning and security. However, an Hospital Information System (HIS) that contains a FHIR

[3] A triplestore is a purpose-built database for the storage and retrieval of triples through semantic queries. A triple is a data entity composed of subject-predicate-object.

server to store wellness data from mobile devices is mentioned, which allows this system to be interoperable. Additionally, an hospital-backed clinical trial with the aim of creating a gene-based obesity management model is reported.

Rubi et al. [85] propose an Internet of Medical Things (IoMT) platform for pervasive healthcare that ensures interoperability, scalability in an M2M-based architecture, and processing of high volumes of data with knowledge extraction, and common healthcare services.

The platform uses the semantics described in OpenEHR for both data quality evaluation and standardization of healthcare data stored by the association of IoMT devices and observations defined in OpenEHR. Moreover, it enables the application of big data techniques and online analytic processing (OLAP) through Hadoop Map/Reduce and content-sharing through a FHIR API. The authors mention the consideration of standardized data formats for the storage and transmission of data, which promotes interoperability regarding data representation formats. Also, the platform automatically prepares the data for the application of OLAP through the anonymization of records, thus granting patient's privacy.

Boutros-Saikali et al. [86] have designed a platform that allows companies to integrate various IoT technologies. Data is collected by connectors, which are specific programs for a systematic extraction of data from each provider. This data is then saved in a non-standardized EHR, and then transformed into FHIR and Open mHealth resources (author-defined format). Third parties can query the FHIR resources using a set of REST APIs defined by the FHIR standard. The authors, on the other hand, make no mention of how to incorporate standalone devices or data processing.

These studies help in identifying important aspects about interoperability, such as the importance of characterizing the devices used in order to achieve device interoperability, and standardizing IoT following international initiatives such as HIMSS (Healthcare Information and Management Systems Society). These promote the use of e-health standards like FHIR that facilitate the exchange of information between systems, in order to achieve true interoperable systems.

A comparison of the key aspects of the works examined focusing on interoperability is provided in Table 2.4. Aspects such as proposed solution, data types, data protocols and any mention to tests are presented.

2.3.3 Weaknesses Commonly Found in the Literature

In the literature, many authors propose healthcare monitoring systems based on IoT including multiple components. However, we could not find any mention to techniques for synchronizing information among the various components if communication suddenly fails. Another major consideration that several works overlook is the ability to integrate the pro-

Works	Proposed Solution	Data Types	Data Protocols	Tests
Alamri [83]	Semantic EHR system to support integration between IoT HISs and EHR systems	Data gathered by IoT healthcare devices and sensors	Standard EHR	Not mentioned
Hong et al. [84]	Interconnected PHR system built in adherence with healthcare data communication standards and an IoT Cloud platform	EMR and life log data	HL7 FHIR	Hospital-backed clinical trial
Rubi et al. [85]	IoMT platform for pervasive healthcare that ensures interoperability, scalability in an M2M-based architecture	Data gathered by IoMT devices	OpenEHR and HL7 FHIR	Not mentioned
Boutros-Saikali et al. [86]	Platform that allows companies to integrate various IoT technologies	Data collected by connectors	non-standardized EHR and HL7 FHIR and Open mHealth resources	Not mentioned

Table 2.4: Comparison between the different pervasive healthcare applications reviewed with emphasis on interoperability.

posed systems with external healthcare systems.

Regarding security, some works address both data and network security, considering encryption for data security and authentication for network security. However, a subset of these works deal explicitly one of these strategies, which is not the most adequate option for achieving a really **secure** system. Other works mentions secure transmission which can be achieved by using TLS. This protocol ensures integrity, confidentiality, and authentication by combining public key cryptography to validate the communicating parties' identities, symmetric-key algorithms to encrypt the transmissions, and message integrity checks to ensure that transmissions are not tampered with during transmission. Furthermore, access control must be considered. Systems have many entries, which need to be secured. For instance, securing the devices physically and limiting the access to topics in messaging protocols or network topologies allows developers to decide which devices access which information.

Following the survey conducted, we can observe that there is no specific way to achieve **interoperability**. Many manufacturers develop their own proprietary data formats and communication protocols due to a lack of clear and concise industry standards and regulations. However, as previously stated, there are designed standards like HL7 FHIR that are recognized by the ITU (International Telecommunication Union) and the European Commission.

2.3.4 Statement of Contributions

Following the review of the various techniques adopted by previous works and their weaknesses, in the context of this dissertation, we propose to:

- Materialize the IoT architecture planned (see chapter 3), by developing and consolidating the data acquisition and data transmission components, as well as the GUI in use on the WoW project.

- Implement a redundancy mechanism for retransmitting data between Smart box and the Gateway if the Wi-Fi connection suddenly falls.

- Implement secure communication mechanisms (encryption, authentication, etc.) for transmission and management of patient biomonitoring parameters.

- Design and develop a FHIR translation layer based on the HAPI FHIR library, which is an implementation of the HL7 FHIR specification, to promote interoperability with the hospital's patient management system.

- Conduct a detailed performance study through unit tests and architecture integration tests to validate all the mechanisms developed.

- Perform experimental validation in a real world scenario.

2.4 Summary

The relevance of IoT systems, their benefits and challenges, and how they might improve digital healthcare have been covered in this chapter. In addition, an IoT reference system has been presented, along with a quick review of the most significant aspects in the context of this dissertation, with an emphasis on security and interoperability.

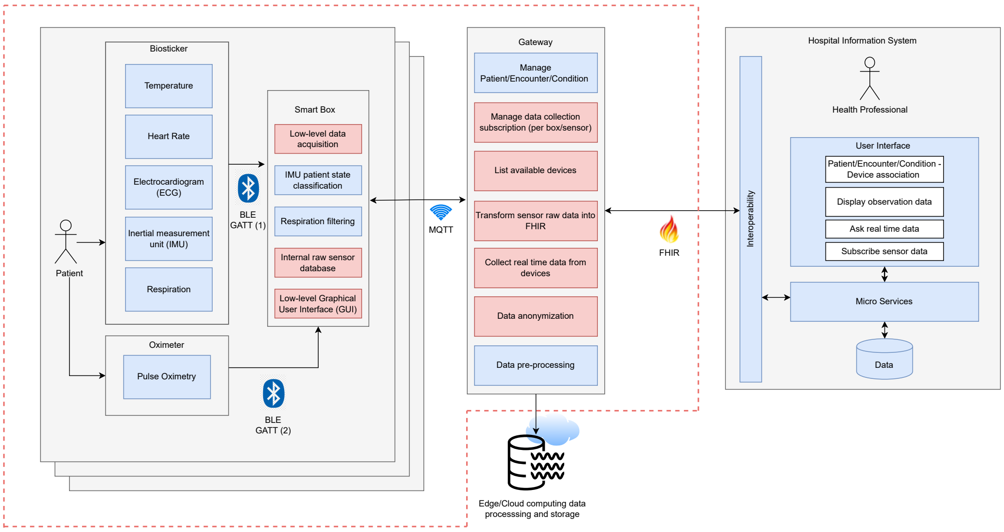
In the next chapter, we present and discuss the IoT architecture in use, focusing on the work developed with regards to data acquisition, data transmission and GUI in the scope of this dissertation.

3 IoT System Development

This work is developed in the scope of the WoW R&D project[4], which focuses on wireless biomonitoring of patients in hospital beds, equipped with Biostickers (i.e., electronic tags with embedded sensors that acquire vital signs) placed non-intrusively on the skin [87]. In this chapter we present and discuss the IoT architecture of the WoW project, and the work developed with regards to the BLE sensor data acquisition, the MQTT wireless data transmission to the central Gateway, and the GUI that presents real time data to the user.

3.1 System Architecture

Figure 3.1 shows a logical view of the proposed architecture, which is followed by a brief technical description of its components. This system is divided into four major components: the Biosticker and Oximeter, the Smart box, the Gateway and the Globalcare HIS. The red square and colors in the figure highlight the components and submodules that are the focus of the work developed in this dissertation.



Biosticker
Temperature
Heart Rate
Electrocardiogram (ECG)
Inertial measurement unit (IMU)
Respiration
Patient
Oximeter
Pulse Oximetry
BLE GATT (1)
BLE GATT (2)
Smart Box
Low-level data acquisition
IMU patient state classification
Respiration filtering
Internal raw sensor database
Low-level Graphical User Interface (GUI)
MQTT
Gateway
Manage Patient/Encounter/Condition
Manage data collection subscription (per box/sensor)
List available devices
Transform sensor raw data into FHIR
Collect real time data from devices
Data anonymization
Data pre-processing
Edge/Cloud computing data processing and storage
FHIR
Hospital Information System
Health Professional
Interoperability
User Interface
Patient/Encounter/Condition - Device association
Display observation data
Ask real time data
Subscribe sensor data
Micro Services
Data

Figure 3.1: System architecture of the WoW project. Source: WoW Deliverable D1.2 [11]

[4] WoW, https://inovglintt.com/financiamento/wow/

- **Biostickers** consist of electronic patches that are attached to the patient that contain a variety of sensors, including body temperature, heart rate, ECG, respiration rate and Inertial Measurement Unit (IMU), which were chosen following a user requirements analysis from the interviews with health professional staff from CHUC (see Figure 3.2). These stickers communicate vital signs data to the respective Smart box embedded in patient's bed through a wireless connection. As such, the Biostickers are in charge of transmitting electrophysiological data to be collected by the Smart boxes.

- **Oximeters** are used to measure the levels of oxygen in the patient's blood by using a pulse oximetry sensor. Oximeters also transmit data wirelessly to the Smart boxes.

- The **Smart box** consists of a single board computer (Raspberry Pi 4 Model B running Ubuntu 20.04) that retrieves the low-level data supplied by the sensors embedded in the Biostickers and Oximeters through BLE (see Figure 3.3). This component includes a respiration filter to compute the respiratory rate from the respiratory values, as well as a GUI to show sensor data in real time, among others. In the context of the WoW project, it was decided to use the Raspberry Pi 4B due to its low scale form factor and because it already includes Wi-Fi and BLE networking.

- The **Gateway** is based on an Intel NUC NUC8i7BEH also running Ubuntu 20.04, this component connects the Smart boxes to the GlobalCare HIS used in the project, and it is in charge of managing users and data. The Gateway maintains a list of available Smart boxes and sensors, as well as their associated devices. Furthermore, it gathers data from devices through Wi-Fi, pre-processes raw data, converts it into the FHIR interoperability standard for transmission to the interoperability layer, and more.

- The **Globalcare** HIS software developed by Glintt-Healthcare Solutions, S.A. covers administrative duties linked to patient identification and administration. It enables the registration and monitoring of all patient procedures and flows, from the initial interaction with the health entity through the patient's release.

To integrate these components, three distinct interactions are considered:

- **Biosticker-Smart box** uses BLE technology from the latest Bluetooth Core specification (version 5.2)[5]. This technology provides a secure and low power communication protocol for constrained applications. We decided to use this technology since it allows the system to communicate to the Biosticker and oximeter simultaneously. We have been developing data acquisition using the official Linux implementation of the BLE

[5] Bluetooth Core, `https://www.bluetooth.com/specifications/specs/core-specification/`

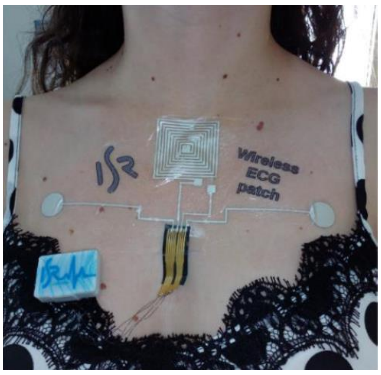


Figure 3.2: Biosticker.

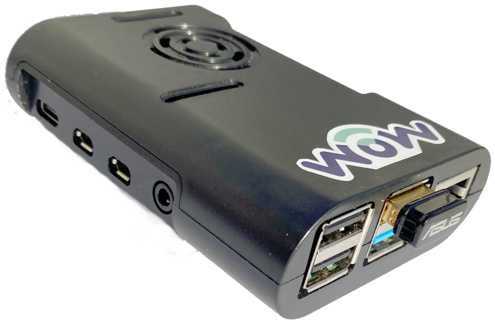


Figure 3.3: Smart box.

protocol stack, BlueZ[6], which promotes interoperability by allowing us to make use of multiple BLE adapters with the same codebase.

- **Smart box-Gateway** uses Wi-Fi, a technology commonly used in healthcare applications to link networks at the hospital or at home (via a private network). The Wi-Fi implementation combined with the MQTT protocol provides reliable communications. We have decided to use this technology since it allows communication between devices with limited resources, such as the Smart box, using encryption, authentication, and authorization to ensure system security.
- **Gateway-HIS** uses Wi-Fi or Ethernet (when the Gateway and the Globalcare server are in the same network). In this interaction the FHIR standard is used. The interoperability component (from the HIS) may make health information systems collaborate inside and beyond organizational boundaries. This enables systems of various types to communicate with one another.

3.2 BLE Data Acquisition

As previously stated, Bluetooth Low-Energy (BLE) version 5.2 is used for communication between Biostickers and Smart boxes, since it allows for interoperability, consumes low power, employs data formats that are approved by medical authorities, and secures data transmission [88]. The Smart box must be able to collect data simultaneously from the Biosticker and a pulse oximeter. For this, an external adapter, the ASUS USB-BT500, is used to connect to the Biosticker (see [89]), since it collects several vital signals, and the oximeter connects to the internal BLE adapter of the Raspberry Pi 4B.

The acquisition is only possible by combining BLE with Generic ATTribute Profile (GATT) since BLE is used for communication and GATT controls the connections and

[6] BlueZ, `http://www.bluez.org`

advertisements between devices, dividing it in multiple layers [90]. In Table 3.1, the data collected by the Biosticker and the oximeter is presented.

Signals	**Data size**	**Data range**	**Unit of measure**	**Frequency**	**Data type**
Heart rate	2 bytes	0 to 65535	beats per minute	0.2 Hz	unsigned byte
Temperature	5 bytes	-3.4×10^{38} to $+3.4 \times 10^{38}$	°C	1/60 Hz	float IEEE 11073
Respiration	4 bytes	0 to 4.294.967.295	none	10 Hz	unsigned int
Electrocardiogram	20 bytes	-32768 to 32767	none	10 Hz	20 byte array
IMU	24 bytes	-3.4×10^{38} to $+3.4 \times 10^{38}$ for each value	acc: g (1 g $= 9.8m/s^2$) (Bytes 0:11) gyr: degrees/sec ((Bytes 12:23))	20 Hz	float
Battery	1 byte	0 to 255	percentage (%)	0.1 Hz	int
Pulse Oximetry	4 bytes	0 to 100	percentage (%)	1 Hz	int

Table 3.1: Information about data collected by the Biosticker and the Oximeter. Adapted from: [91]

The data flow is depicted in Figure 3.4, beginning with the Smart box connection to the Bluetooth external or internal adapter for data collection from the Biosticker or oximeter sensors. The Smart box also has a Keep-alive and reconnection system, which checks if the connection is working and prevents it from breaking permanently.

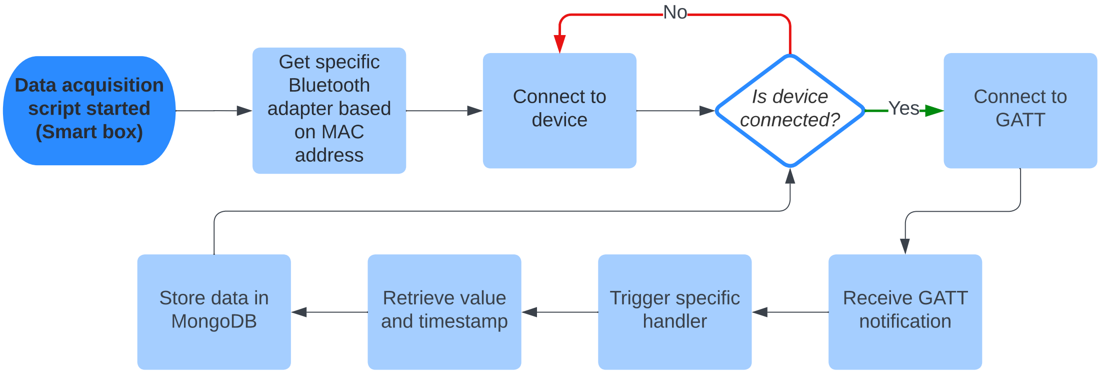

Data acquisition script started (Smart box)
Get specific Bluetooth adapter based on MAC address
Connect to device
No
Is device connected?
Yes
Connect to GATT
Receive GATT notification
Trigger specific handler
Retrieve value and timestamp
Store data in MongoDB

Figure 3.4: The communication flowchart between the Biostickers or Oximeters and Smart box.

Initially, in order to acquire the data from the Biosticker and the oximeter, two scripts had been previously created [91]. One communicates with the Biosticker and the other one with the oximeter.

A drawback of the implementation with two scripts, is the lack of ability to communicate with both the Biostickers and the Gateway concurrently. This implementation required the creation of two MQTT clients which would attempt to connect to the Gateway using the same Smart box identifier, so when the second MQTT client connected, it would cause the disconnection of the first one. In order to solve this issue, two threads have been created using Python's Threading[7] library, each executing one script in parallel, while utilizing the same MQTT client, thus allowing for simultaneous BLE data acquisition and MQTT data transfer through WiFi for the Biosticker and oximeter.

Meanwhile, the raw sensor data received at the Smart box is also kept in a local database in order to feed a graphical user interface (GUI), to be pre-processed, and to be preserved in the event that the connection with the Gateway is interrupted. The software chosen for the database is MongoDB[8]. MongoDB is a document-oriented NoSQL database, which is appealing for IoT applications, since it can handle unstructured or semi-structured data and generally scales better than traditional SQL databases as the amount of data stored increases [92].

Smart Box Database Collections

MongoDB stores data records as documents, specifically Binary JSON (BSON) documents, which are gathered together in collections. A collection is the equivalent of a Relational Database Management System (RDBMS) table, however it does not enforce a schema.

The system's MongoDB database contains two collections:

```
{
    "name": $sensor_name,
    "value": $value,
    "timestamp": $timestamp
}
```

where the *name* field can be heart rate, temperature, battery level, respiratory values, respiratory rate, ECG, oxygen saturation and pulse rate which corresponds to the sensor name, *timestamp* is the ISO date[9] of the measurement and the *value* contains the value measured by the respective sensor; or for the IMU's sensors where the *value_ acc* field is the IMU accelerometer measurement for each coordinate x,y,z, *value_ gyr* field is the IMU gyroscope measurement for each coordinate x,y,z and *timestamp* is the ISO date of the measurement.

```
{
    "name": IMU,
    "value_acc_x": 1.873779058456421,
    "value_acc_y": 0.6645510196685791,
    "value_acc_z": -0.3310549855232239,
    "value_gyr_x": 1.774657964706421,
    "value_gyr_y": 0.8632810115814209,
    "value_gyr_z": -0.42773398756980896,
    "timestamp": ISODate("2022-13-02T02:13:50.570Z")
}
```

[7] Threading, https://docs.python.org/3/library/threading.html

[8] MongoDB, https://www.mongodb.com

[9] ISO date, https://www.w3.org/QA/Tips/iso-date

3.3 Graphical User Interface (GUI)

Each Smart box is assigned to a patient (1 to 1 relationship), and a GUI has been developed to display data collected by the patient's Biosticker and Oximeter in real time. The GUI's primary users are healthcare professionals, but it is also useful for the development team, specifically for debugging purposes.

Since the healthcare professionals are expected to use the GUI to monitor the patient's vital signals, it must be fluid and intuitive to use. The version 1.0 of the GUI, which was available before the work developed in this dissertation can be seen in Figure 3.5.

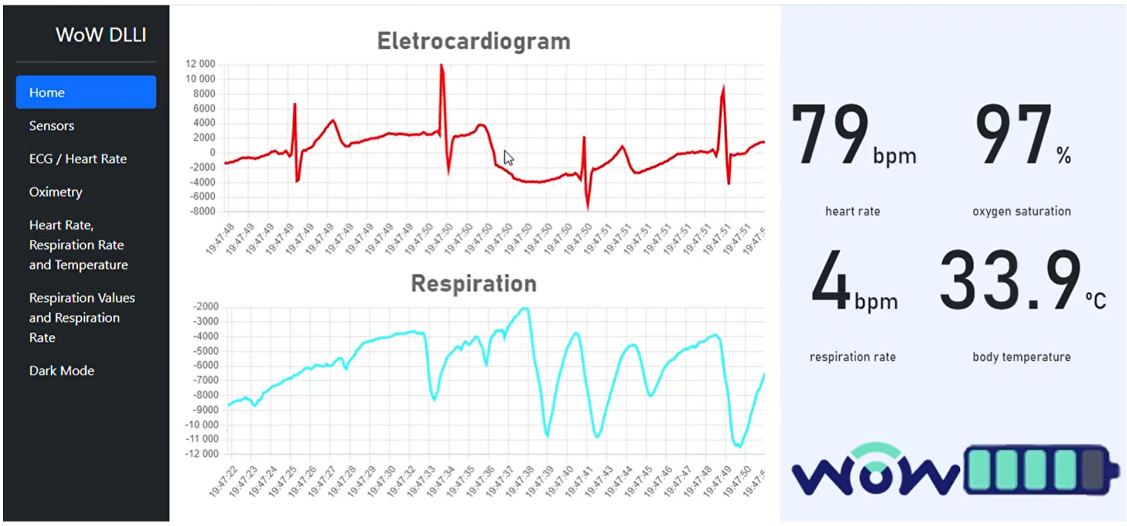


Figure 3.5: Home page of version 1.0. Source: [91].

This GUI can be accessed locally through any web browser using each Smart box fixed Internet Protocol (IP). JavaScript[10], CSS[11], and HTML[12] were used to develop the GUI.

The main library used was Node.js[13], as it employs an event-driven, non-blocking I/O model, allowing the initiation of multiple requests to the mongoDB database in parallel and the presentation of data from different sensors in the GUI simultaneously. The library used to plot the data charts was Charts.js[14] which is a free open-source JavaScript library for real time data visualization.

After the first pilot of the WoW project, a list of potential improvements was established, together with end-users, i.e., health professionals from CHUC (Table 3.2) which resulted on the development of a new version of the GUI, version 2.0, as can be seen in the Figure 3.6.

Since the responsiveness of version 1.0 was extremely low (average 5 FPS), it was properly

[10] JavaScript, `https://www.javascript.com`

[11] CSS, `https://www.w3.org/Style/CSS`

[12] HTML, `https://html.spec.whatwg.org/multipage`

[13] Node.js, `https://nodejs.org`

[14] Chart.js, `https://www.chartjs.org`

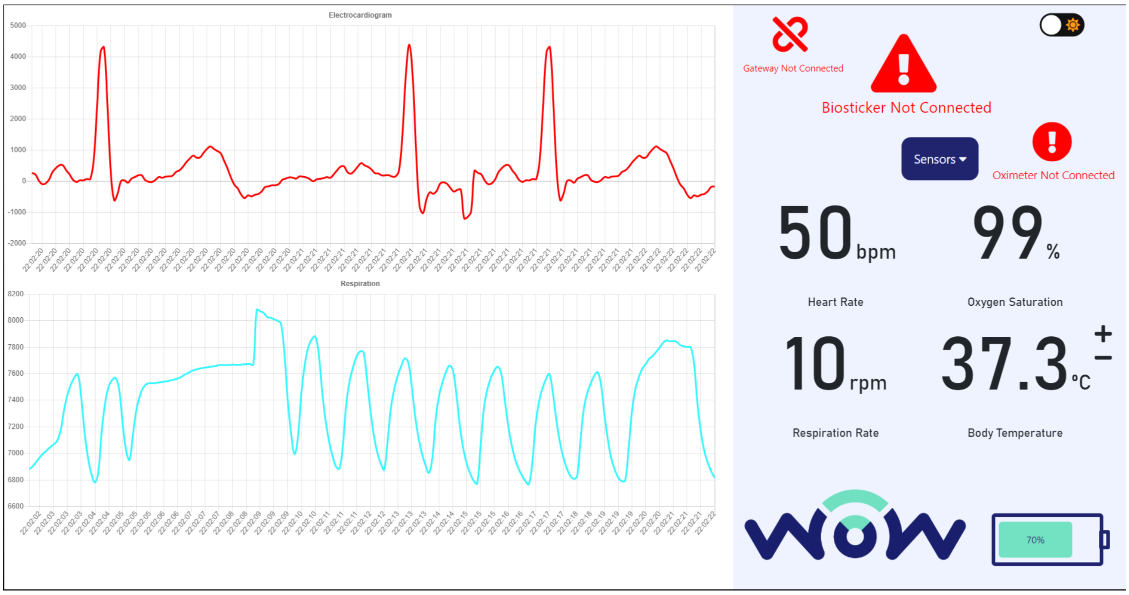


Figure 3.6: Home page of version 2.0.

Requirement	Before	After
Responsiveness	≈ 4 FPS	≈ 20 FPS
Design	Sidebar on the home page	New widgets with same features on home page
Intuitiveness	New page to show sensors list	Dropdown on the home page with sensors list
Usability	Color mode resets after the page reloads	Color mode is saved
Design	The battery icon does not automatically update	New battery icon with better quality that updates the battery level based on the real battery value
Intuitiveness	No warnings	Warnings presented on the home page based on disconnections
Usability	Only Tablet/PC modes	Everything on GUI adapts to the device size

Table 3.2: List of GUI improvements.

concluded that improving the GUI's performance was critical. Several design changes were made to make the GUI more user-friendly and intuitive, and additional pages for comparing sensor data, such as a page containing IMU gyroscope and IMU accelerometer data for comparing their values (Appendix A), were added. A summary of changes, as well as a full overview of the version 2.0 can be viewed at: `https://youtu.be/XIZ2NS1PpvA`.

3.4 MQTT Data Transmission

The communication between the Gateway and the Smart boxes has been developed using the MQTT protocol 5.0. The MQTT protocol provides a lightweight method for performing communication using a publish/subscribe method and is considered a suitable protocol for low-power and efficient data transfer with a fast messaging model, making it a suitable solution for the WoW project. The MQTT broker is a service implemented in the Gateway and is responsible for the communication between the Smart boxes and the Gateway. The open source programme Eclipse Mosquitto[15] has been used to create this broker. Eclipse Mosquitto is an open source message broker that supports the MQTT protocol. We chose Mosquitto since it provides web socket support, as well as supports Quality of Service (QoS) levels 0, 1, or 2. The QoS level is an agreement between the sender and the receiver of a message that defines delivery guarantees.

To establish a connection using the MQTT protocol, three entities are required: the publisher, which is responsible for sending the collected data; the subscriber, which receives the data; and the broker, which manages the data exchange. As mentioned earlier, in this system the Gateway runs the MQTT broker, but the subscribers and publishers in the clients are determined by the type of data delivery.

In order to overcome existing issues, such as connection failures of MQTT clients, the fact that the gateway was not able to manage the pairing of Biosticker and Smart box, etc., several **MQTT features** have been implemented:

- Seamless communication of the measurements made by the Biosticker's sensors. As soon as the Smart box receives a new sensor measurement, it immediately transmits it to the Gateway.

- Pairing feature that is used to manage the Smart box and Biostickers pairs at the Gateway. When a Smart box is paired with a new Biosticker or oximeter, it sends this information to the gateway so that it can manage the pairings, i.e., if a Smart box was already paired with a Biosticker, it will unpair it and pair it with the new Biosticker;

- Keep alive feature that keeps the MQTT connection alive even when the Smart box is not transmitting measurements due to preceding issues in the BLE connection. When the BLE connection is interrupted, there are no data transmissions to the Gateway, so the MQTT connection stops. To fix this, a status endpoint has been set up. The gateway sends a status request to each Smart box that is online every 30 seconds, and the Smart box must respond to avoid interrupting the MQTT connection.

[15] Eclipse Mosquitto, `https://mosquitto.org`

Disconnection Period	30min	1h	2h	6h	12h	24h
Amount Of Data To Be Recovered	26.08 MB	52.50 MB	105.32 MB	316.47 MB	633.22 MB	1.23 GB

Table 3.3: Amount of data to be recovered per disconnection time.

- Synchronization feature that allows the Smart box to relay missing data to the Gateway when they reconnect after an unexpected MQTT disconnection.

During the implementation of the **synchronization functionality**, several tests were designed. Since this procedure involves transferring a considerable amount of data, e.g., 30 minutes of disconnection equals about 26.08 MB of data, the main purpose of these tests is to determine whether we should split the message with missing data into multiple messages and how many. With the goal to minimize synchronization time without loss of information, we evaluate the impact of message size on synchronization. To summarize, we want to define the maximum payload size of each sent message to restore the missing data while minimizing the synchronization time without information loss.

In these experiments, the following configurations have been considered:

- Different period of interruptions: 30min, 1h, 2h, 6h, 12h and 24h. See Table 3.3 for a conversion from time to the corresponding MQTT payload size. The disconnection time refers to the total amount of data to be synchronised.

- Maximum payload size: 14.84 KB, 159.29 KB, 445.08 KB, 890.16 KB, 6.52 MB, 13.04 MB, 26.08 MB,39.12 MB and 52.16 MB. These have been chosen, as they correspond to 1s, 10s, 30s, 1min. 7.5min, 15min, 30min, 45min and 1h (See Table 3.4). The maximum payload size is limited to 52 MB, as values above this number would lead to memory shortage errors. The maximum payload size affects how many messages are needed for transmitting all missing data, i.e. if we have a disconnection time of 1h and a maximum payload size of 26.08 MB (30min), this requires the transmission of 2 distinct messages.

- The synchronization approach is tested firstly without considering the transmission of data being acquired in real time, and subsequently we test it with the simultaneously transmission of real-time data in order to understand its impact on the synchronization feature.

The following performance metrics have been defined:

- Time required to synchronize all missing data (s) - time elapsed between the request

MQTT Payload Size	14.84 KB	159.29 KB	445.08 KB	890.16 KB	6.52 MB	13.04 MB	26.08 MB	39.12 MB	52.16 MB
Corresponding Message Time	1s	10s	30s	1min	7.5min	15min	30min	45min	1h

Table 3.4: Corresponding message time for each MQTT payload size considered in the tests.

of the synchronization mechanism by the Gateway and its receipt of the final message from the Smart box with type *"sync_ rep_ end"*.

- Throughput (Mbps) - Amount of data successfully transmitted from the Smart box to the Gateway per second.

These tests were performed by simulating each disconnection and then having the Gateway query the information from the beginning of the disconnection to its end. For each configuration, the procedure has been automated and repeated 100 times for statistical significance, thus for our metrics, we evaluate average values of each configuration. The experiments were performed using a Biosticker to collect 24 hours of data, a Smart box, and a Gateway. The same 2.4 GHz home Wi-Fi network has been used for all tests which lasted for 3 weeks.

Experimental Assessment of the MQTT Synchronization Functionality

Following the design criteria described in the previous subsection, a total of 4300 consecutive tests have been performed with different combinations of disconnection period *vs.* maximum payload size. It is important to note that we have not employed all configurations possible, as some values of payload sizes have been added during the course of experiments to obtain cleaner results.

In Figure 3.7, we provide an overview of the average time required to synchronize all missing data in all tested configurations and illustrate the impact of the maximum payload size with different disconnection periods. As expected, the longer the connection is interrupted, the longer it takes, until all information is sent again.

We can also see that the time taken to restore the missing data tends to increase with the maximum payload size. Howeverm the highest average time required is usually achieved with the smallest payload size. For instance, if we use 14.84 KB as the maximum payload size with 30 minutes of disconnection period, we send about 300 messages per second, which overloads the gateway as it is not able to receive all messages in due course and builds up a large queue, which delays the synchronization process.

Moreover, we can observe that in each scenario least time needed to resend all the information, is obtained when using 13.04 MB as the maximum payload size per message sent,

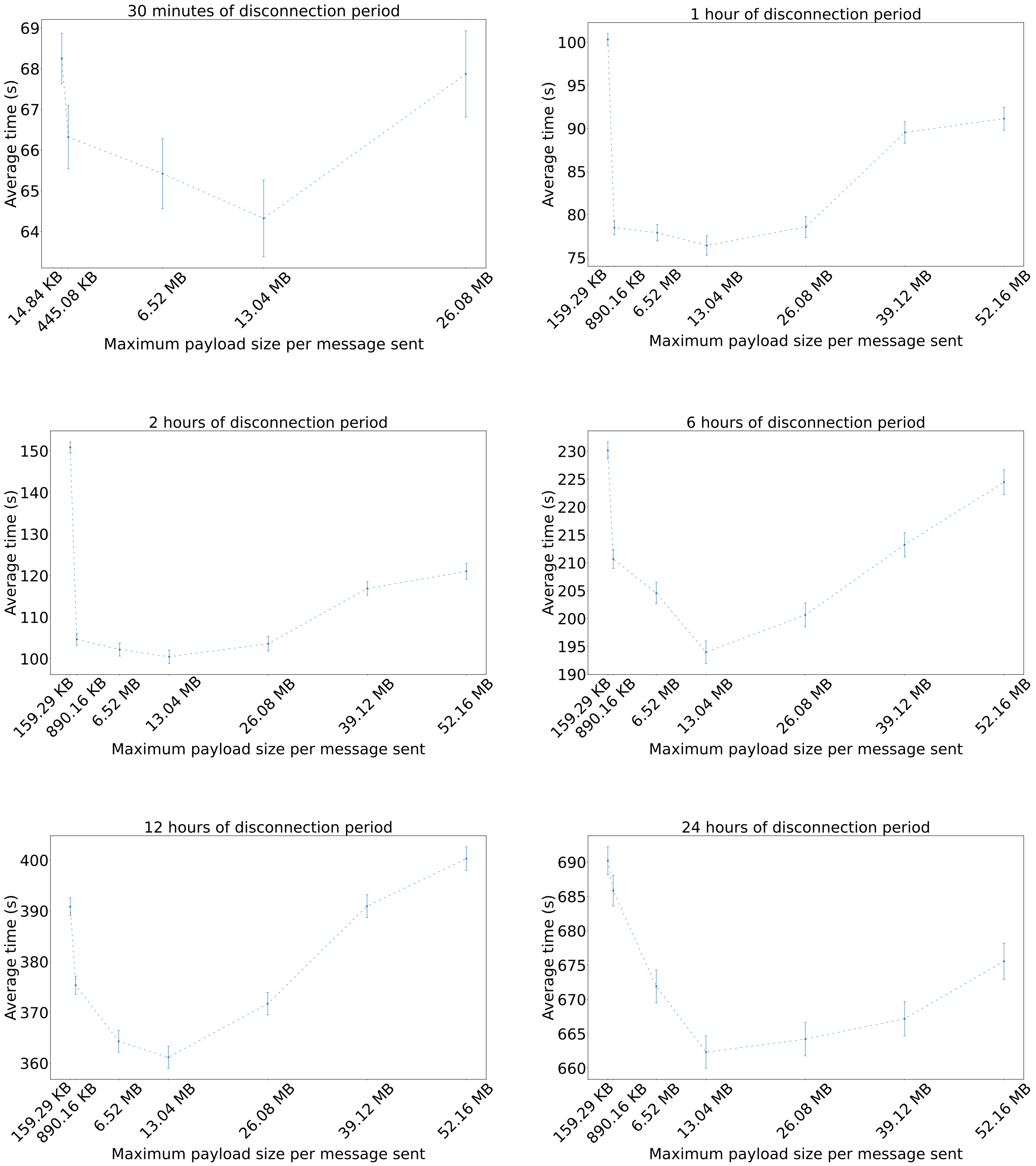


Figure 3.7: Average time required to sync all missing data per configuration (s).

independently of the disconnection period tested. This is also confirmed by Figure 3.8, which shows that the highest data throughput is always achieved with a maximum size per sent message of 13.04 MB.

From Figure 3.8 we can also see that the throughput increases with the disconnection period, this might be explained by the fact that the time required to resend the missing information does not increase as the amount of data increases, e.g., as seen in Table 3.3, 24 hours of disconnection period corresponds to twice the amount of data lost when considering 12 hours of disconnection, but if we compare the time required to resend the missing information with a payload size of 52.16 MB, it is 675.56 seconds and 400.30 seconds respectively, which means that the rate of increase to restore the data is only 1.68 times.

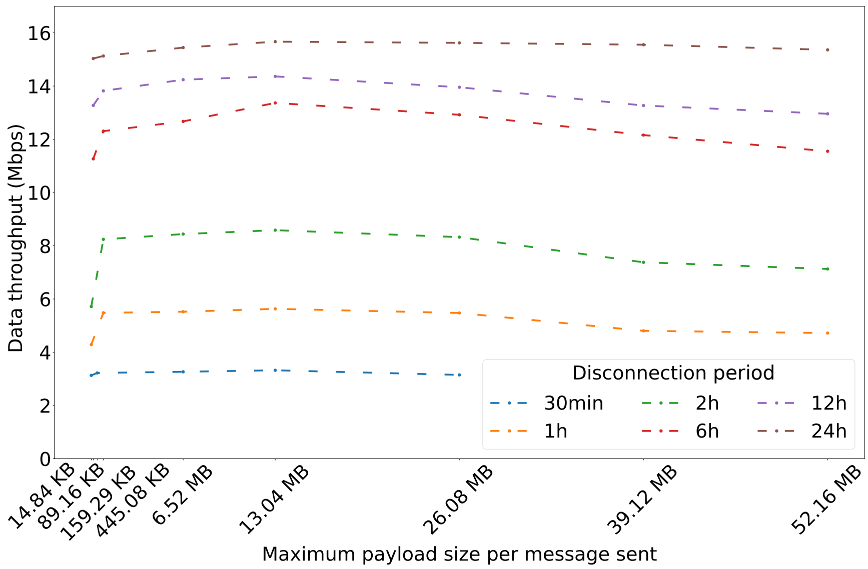


Figure 3.8: Data Throughput in all synchronization tests performed (Mbps).

Additionally, we can also see that the simultaneous transmission of the actual data acquired in real time had no affect in the synchronization feature. This can be confirmed by the example illustrated in Figure 3.9. This happens because the messages that send the current data in real time can be disregarded when comparing with the messages with the lost data.

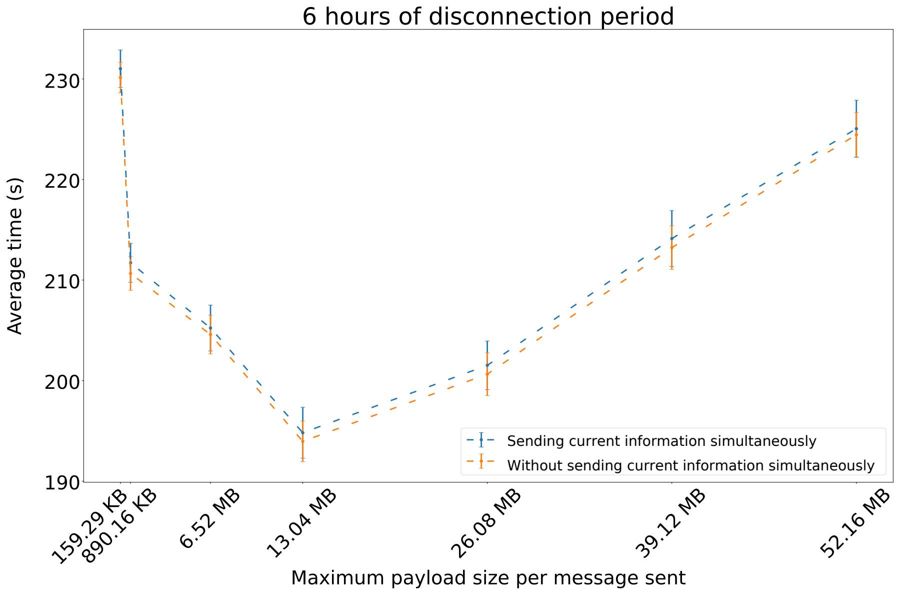


Figure 3.9: Comparison of synchronization method with and without transmitting the actual data acquired.

In conclusion, this experimental assessment allowed us to dimension the payload size of the synchronization messages with a mean optimal value of 13.04 MB.

3.4.1 Proposed MQTT Specification

To promote interoperability, we have defined that all MQTT messages must adhere to the JSON data standard with the following structure:

```
{
    "client_id": $client_uuid,
    "timestamp": $timestamp,
    "message_type": $message_type,
    "payload": {
        // ...
    },
}
```

where *client_id* is the unique UUID of the MQTT client, *timestamp* is the UNIX[16] timestamp that is synchronized between all Smart boxes and the Gateway using Chrony[17], an implementation of the Network Time Protocol (NTP), and the *payload* contains the specific content of the message that is associated to the *message_type*. The field **message_type** defines the type of message sent, and can be consulted in Appendix B

The **payload** field must contain the message's content and its format varies depending on the *message_type* and can be consulted in Appendix C.

To communicate the sensor data, the Smart box must publish to different **endpoints** (see Appendix D), depending on the message type.

When data arrives at the gateway, it is preprocessed to validate messages and filter information, to avoid issues that might occur with unvalidated data, and then stored in a database. This database is a traditional Relational Database Management System (RDBMS), which is used to store data in the system, enforcing a consistent and logical representation of the information, which would not be possible by using a NoSQL database. In this system, PostgreSQL[18] is used due to its overall performance and scalability [93].

3.4.2 Gateway Database Schema

Figure 3.10 represents the database model used in the PostgreSQL database. It represents all the information contained in the Gateway, as well as the relationships within the data, arranged according to the use of the information, i.e. the service/functionality it is associated with. The data stored in the system can be divided into five categories:

[16] UNIX, https://www.unixtimestamp.com

[17] Chrony, https://chrony.tuxfamily.org

[18] PostgreSQL, www.postgresql.org

- **Sensor observation data**: Biosignals measured by the Biostickers and transmitted by the Smart boxes via MQTT. Each measured signal is associated with the sensor that measured it and the Smart box connected to the Biosticker containing that particular sensor;

- **System data**: Information about the Smart boxes, Biostickers and sensors in each Biosticker, including Smart boxes and Biosticker pairs managed by the Gateway. The data model is flexible so that each Smart box can be associated with multiple Biostickers and each Biosticker can be associated with multiple sensors.

- **MQTT related data**: Information about MQTT clients and their permissions described in the Security section.

- **FHIR related data**: Data related to FHIR communications, such as subscription requests from the HIS to transmit sensor measurements received.

- **Stored procedures**: User-defined subroutines that specify operations that other services (e.g., the MQTT broker) use to deal with stored data (insert, delete, search, etc.), such as precompiled SQL statements that are simply a set of statements that perform a specific task.

3.5 Summary

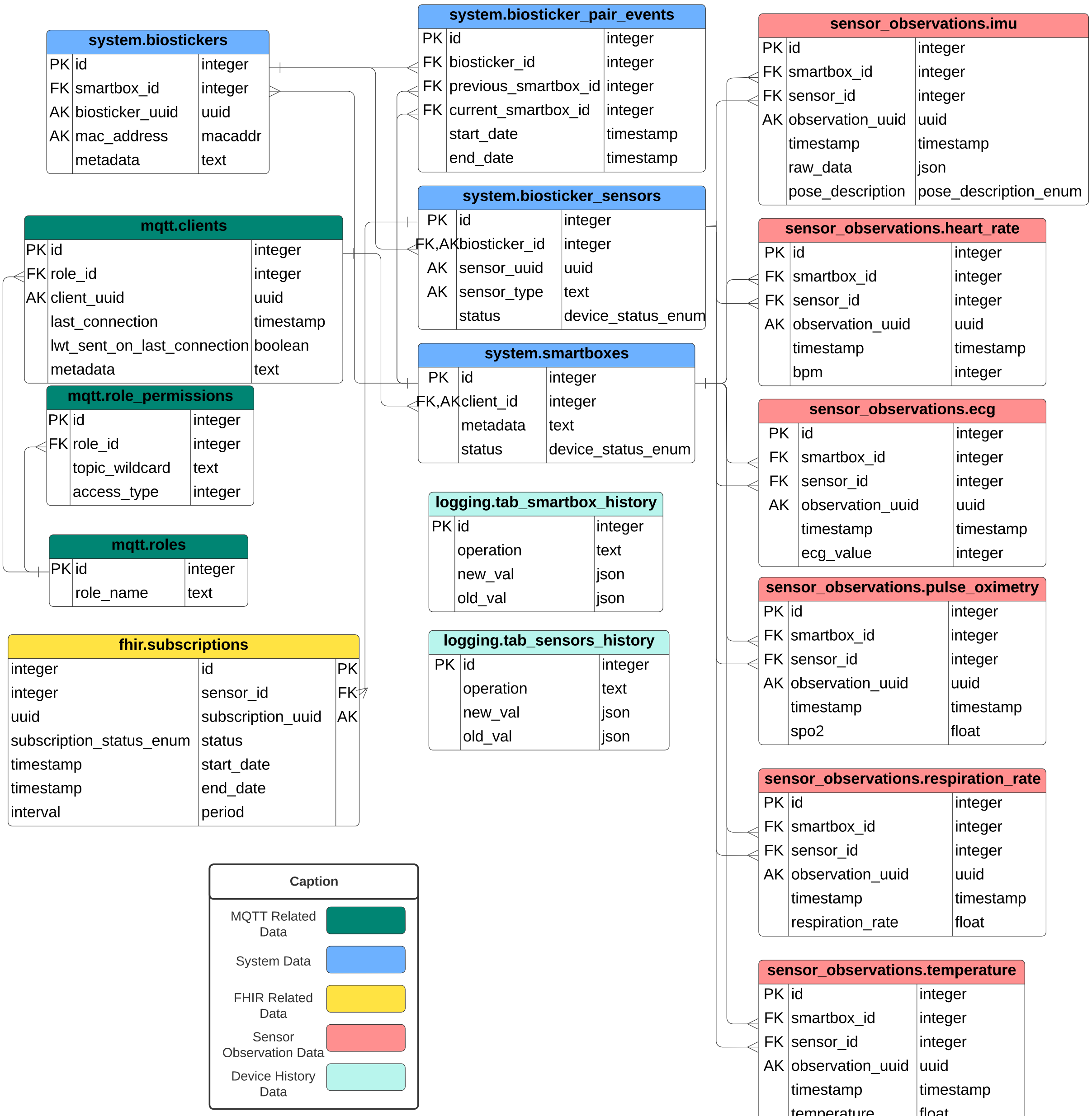
In this chapter, we have described the WoW architecture and the work developed on BLE sensor data acquisition, MQTT wireless data transmission to the central Gateway, and GUI.

As for the BLE data acquisition submodule, the implementation has been redesigned to include threads to address a critical issue that had been hindering the system.

In addition, the GUI has been completely redesigned and improved based on a list of potential improvements established together with CHUC health professionals.

For the MQTT data transmission submodle, a MQTT specification has been proposed to promote interoperability. Furthermore, a pairing functionality that allows the Gateway to manage each pairing of the Smart box and Biosticker has been developed, as well as a Keep Alive feature that prevents the MQTT connection from being interrupted, and a synchronization functionality that allows the Smart box to resend missing data to the Gateway when it reconnects after an unexpected MQTT disconnection. Regarding the synchronization functionality an in-depth experimental evaluation has been designed to define the maximum payload size of each sent message to restore missing data.

The next chapter presents the design and development of a FHIR translation layer based on the library HAPI FHIR towards interoperability and a vulnerability analysis towards securing the overall system.



system.biostickers		
PK	id	integer
FK	smartbox_id	integer
AK	biosticker_uuid	uuid
AK	mac_address	macaddr
	metadata	text

system.biosticker_pair_events		
PK	id	integer
FK	biosticker_id	integer
FK	previous_smartbox_id	integer
FK	current_smartbox_id	integer
	start_date	timestamp
	end_date	timestamp

system.biosticker_sensors		
PK	id	integer
FK,AK	biosticker_id	integer
AK	sensor_uuid	uuid
AK	sensor_type	text
	status	device_status_enum

system.smartboxes		
PK	id	integer
FK,AK	client_id	integer
	metadata	text
	status	device_status_enum

mqtt.clients		
PK	id	integer
FK	role_id	integer
AK	client_uuid	uuid
	last_connection	timestamp
	lwt_sent_on_last_connection	boolean
	metadata	text

mqtt.role_permissions		
PK	id	integer
FK	role_id	integer
	topic_wildcard	text
	access_type	integer

mqtt.roles		
PK	id	integer
	role_name	text

fhir.subscriptions		
integer	id	PK
integer	sensor_id	FK
uuid	subscription_uuid	AK
subscription_status_enum	status	
timestamp	start_date	
timestamp	end_date	
interval	period	

logging.tab_smartbox_history		
PK	id	integer
	operation	text
	new_val	json
	old_val	json

logging.tab_sensors_history		
PK	id	integer
	operation	text
	new_val	json
	old_val	json

sensor_observations.imu		
PK	id	integer
FK	smartbox_id	integer
FK	sensor_id	integer
AK	observation_uuid	uuid
	timestamp	timestamp
	raw_data	json
	pose_description	pose_description_enum

sensor_observations.heart_rate		
PK	id	integer
FK	smartbox_id	integer
FK	sensor_id	integer
AK	observation_uuid	uuid
	timestamp	timestamp
	bpm	integer

sensor_observations.ecg		
PK	id	integer
FK	smartbox_id	integer
FK	sensor_id	integer
AK	observation_uuid	uuid
	timestamp	timestamp
	ecg_value	integer

sensor_observations.pulse_oximetry		
PK	id	integer
FK	smartbox_id	integer
FK	sensor_id	integer
AK	observation_uuid	uuid
	timestamp	timestamp
	spo2	float

sensor_observations.respiration_rate		
PK	id	integer
FK	smartbox_id	integer
FK	sensor_id	integer
AK	observation_uuid	uuid
	timestamp	timestamp
	respiration_rate	float

sensor_observations.temperature		
PK	id	integer
FK	smartbox_id	integer
FK	sensor_id	integer
AK	observation_uuid	uuid
	timestamp	timestamp
	temperature	float
	is_celsius	boolean

Figure 3.10: Database model implemented in the Gateway.

4 Interoperability & Security

4.1 HIS FHIR Integration

To achieve interoperability with the Globalcare HIS, we use Fast Healthcare Interoperability Resources (FHIR). FHIR is a standard for data formats and elements, as well as an Application Programming Interface (API) for the exchange of electronic health information. In our architecture, the Gateway must be able to handle requests from the HIS, such as requests to receive sensor readings or information when a new device is added/removed. It is also worth noting that any communication to/from the HIS must be in FHIR format. Therefore, the Gateway must be able to translate low-level messages into the FHIR format. A flowchart describing the communication of sensor data to the Globalcare HIS can be seen in Figure 4.1.

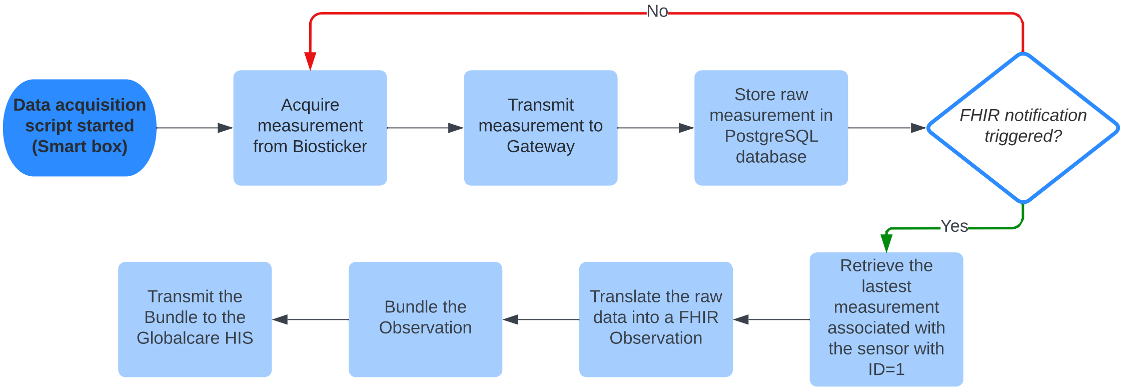

No
Data acquisition script started (Smart box)
Acquire measurement from Biosticker
Transmit measurement to Gateway
Store raw measurement in PostgreSQL database
FHIR notification triggered?
Yes
Retrieve the lastest measurement associated with the sensor with ID=1
Translate the raw data into a FHIR Observation
Bundle the Observation
Transmit the Bundle to the Globalcare HIS

Figure 4.1: Flowchart describing the communication of sensor data to the HIS.

To implement these features, we have developed a new service in the Gateway; the HIS FHIR Integration service, which is based in a FHIR RESTful HTTP server. The FHIR server has been implemented with the open-source HAPI FHIR library[19]. The HAPI FHIR Java library was created in 2001 and it is mantained by Smile CDR[20], a health information

[19] HAPI FHIR, `https://hapifhir.io`

[20] Smile CDR, `https://www.smilecdr.com`

technology company. The HAPI FHIR library offers a simple and straightforward API to interact with FHIR resources, that can be used to exchange and/or store data.

Since we need to to transform the sensor data that we currently have into the FHIR format, we have opted to utilize a plain server. When using a plain server, the HAPI FHIR library takes care of HTTP processing, parsing/serialization and FHIR REST semantics, which showcases its flexibility. The HAPI FHIR Plain Server implementation is based on Java Servlet 3.1 API[21]. A servlet is a Java programming language class designed to improve the capabilities of servers that host applications accessible by means of a request-response programming architecture. So, in order to develop the FHIR server, we require a web server capable of hosting the HAPI FHIR plain server servlet.

According to the HAPI FHIR documentation, it is suggested to adopt the Eclipse Jetty[22] as the web server, thus it has been chosen for the development of the HIS FHIR Integration service.

4.1.1 FHIR Specification

In the development of the HIS FHIR Integration service we have collaborated with Global Intelligent Technologies (Glintt) to establish a technical specification for the functionalities required. A brief summary of the specification is included in Appendix E. A more formal, complete and confidential specification has been prepared by Glintt within the WoW project.

The specification is composed by three major components:

- **Vital Signs Specification**, describes how sensor measurements are converted to FHIR format. For instance, if a temperature measurement of the sensor with ID=1 is requested by the HIS, the Gateway would need to transmit a transaction-type bundle[23] containing the observation resource containing the last measurement of the temperature sensor with ID=1, a device resource containing the information of the sensor and the information of the Smart box connected to the Biosticker containing that sensor.

- **Devices Specification**, the device creation information is sent in transaction-type bundles, sending the device resource of the sensor and also the device resource of the associated Smart box, so that the Globalcare can associate the sensor with the Smart box. For example, when a sensor is paired with a Smart box, the Gateway transmits a transaction-type bundle containing a device resource with that sensor's information

[21] Java Servlet, `https://docs.oracle.com/javaee/7/tutorial/servlets.htm`

[22] Eclipse Jetty, `https://www.eclipse.org/jetty/`

[23] In FHIR, a bundle is a container for a collection of resources. A resource represents a healthcare category, i.e., patients, measurement, devices, etc. The transaction-type bundle allows the user to submit a bundle containing a number of resources to be created/updated/deleted as a single atomic transaction.

and a device resource with the Smart box's information so that the Globalcare can associate the sensor with the Smart box.

- **Frequency Specification**, describes the subscription requests of the measurement frequency and measurement update. For example, if a healthcare professional intends to receive information about a patient every 30 minutes, they should select the device associated with that patient, and the Globalcare sends a ServiceRequest to the Gateway, processes that service request, and sets up notifications to send the last sensor reading associated with that device to the Globalcare every 30 minutes.

4.1.2 FHIR Implementation

In order to fully implement the FHIR specification, the servlet is composed by three essential components:

- **Transaction Bundle Provider**, responsible for carrying out the required tasks, such as setting up the notifications for a new device subscription, after receiving a Bundle from the Globalcare HIS through the Subscription Handler.
- **Subscription Handler**, manages the scheduling and sensor data transfer to the HIS.
- **Device Handler**, manages the notifications about new/removed device.
- **Database Handler**, responsible for reading and translating the data stored in the Gateway database into valid FHIR resources, such as devices or sensor measurements.

The Transaction Bundle Provider reads the incoming bundles of ServiceRequests from GlobalCare and calls the appropriate Subscription Handler function responsible for scheduling, unscheduling or updating a Smart box subscription.

A Smart box subscription is essentially a job that periodically reads data from all sensors connected to the respective Smart box and transmits the data to the GlobalCare. When a new subscription is created, it remains active as long as there is no request to deactivate it. If the server is shut down for any reason, the Subscription Handler will make a request to the Database Handler upon restart to retrieve the list of all existing subscriptions. The subscriptions are scheduled using the Quartz Scheduler library[24] to trigger notification events on a regular basis.

When a new trigger occurs, the Database Handler retrieves the latest measurement from the database and translates it into its FHIR representation: the FHIR Observation resource, and the FHIR Device resources of the associated sensor and Smart box (see Figure 4.2). This

[24] Quartz Scheduler, `http://www.quartz-scheduler.org`

is necessary because the GlobalCare HIS does not contain any information about the link between the sensors embedded in the Biostickers and the Smart boxes. Finally, everything is bundled into a Bundle resource and sent to GlobalCare, which processes the information accordingly.

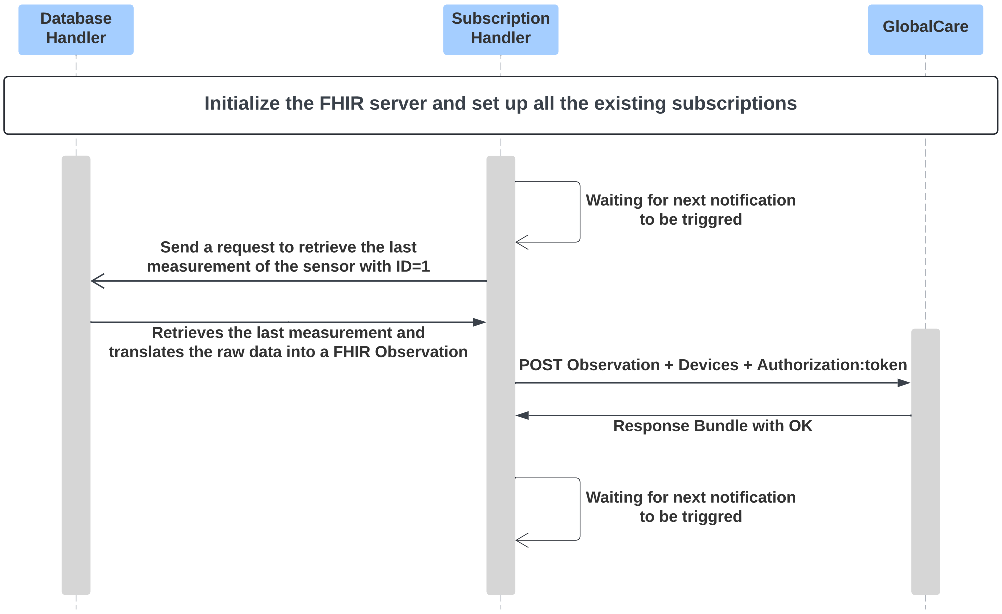

Database Handler
Subscription Handler
GlobalCare
Initialize the FHIR server and set up all the existing subscriptions
Waiting for next notification to be triggred
Send a request to retrieve the last measurement of the sensor with ID=1
Retrieves the last measurement and translates the raw data into a FHIR Observation
POST Observation + Devices + Authorization:token
Response Bundle with OK
Waiting for next notification to be triggred

Figure 4.2: Sequence diagram describing the handling of subscription notification triggers.

There is also another job that is scheduled when the FHIR server boots up that routinely checks the database for new devices and for devices that may have been removed (see Figure 4.3). For example, when a new sensor is associated to a Smart box, a bundle is created with the FHIR device resources of the corresponding sensor and Smart box. A sequence diagram describing the transmission of sensor measurements to the HIS is shown in Figure 4.4.

It is important to mention that our server is fully prepared for use in a real-time system, it is only necessary to set up the infrastructure and start the server. A disadvantage of this implementation is that it was developed for this specific specification and therefore it works only with Globalcare. In addition, we can only transmit FHIR messages to a single HIS. If this system is used in multiple hospitals in the future, the simpler solution is to use multiple Gateways, one at each hospital. A positive aspect of our implementation is that it is fully available for remote monitoring, since we use Wi-Fi for data transmission, the Gateway and the HIS do not need to be on the same network.

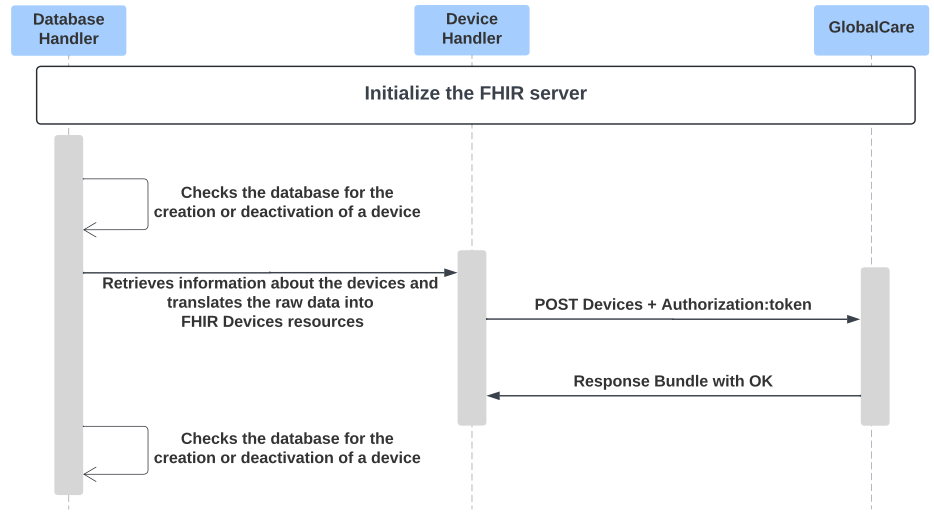

Database Handler
Device Handler
GlobalCare
Initialize the FHIR server
Checks the database for the creation or deactivation of a device
Retrieves information about the devices and translates the raw data into FHIR Devices resources
POST Devices + Authorization:token
Response Bundle with OK
Checks the database for the creation or deactivation of a device

Figure 4.3: Sequence diagram describing the handling of device notifications.

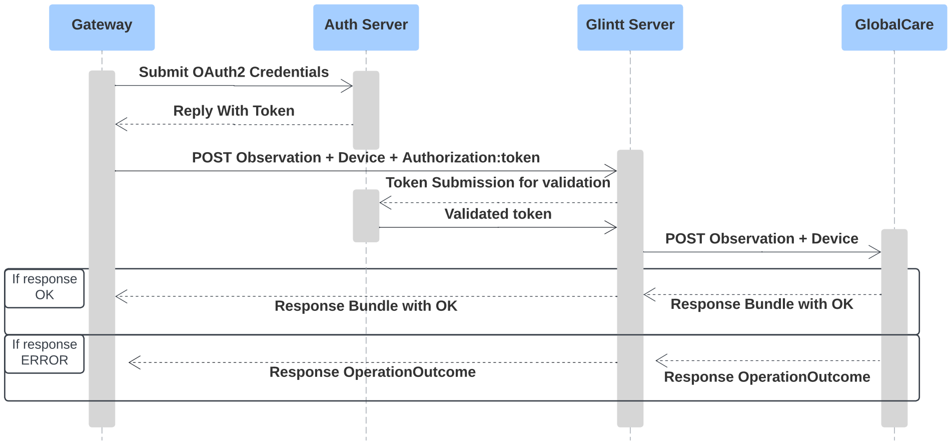

Gateway
Auth Server
Glintt Server
GlobalCare
Submit OAuth2 Credentials
Reply With Token
POST Observation + Device + Authorization:token
Token Submission for validation
Validated token
POST Observation + Device
If response OK
Response Bundle with OK
Response Bundle with OK
If response ERROR
Response OperationOutcome
Response OperationOutcome

Figure 4.4: Sequence diagram describing the transmission of sensor measurements to the HIS.

4.2 Vulnerability Analysis

One of the main aspects of this dissertation is security. In order to assess which mechanisms are needed to ensure the security of the overall system, a vulnerability analysis of the system has been performed.

4.2.1 Data Privacy

A crucial aspect about privacy is that all data transmitted through the system is not linked to a patient. That is, even if there were a way to penetrate our system and retrieve data from it, it would only disclose sensor readings that could not be linked to a specific person.

In our specific implementation, we assign Biostickers to Smart boxes, while Smart boxes are assigned to patients at the level of the Hospital Information System (HIS). Since, the WoW architecture consists of several components which often interact, they must be kept secure. In the following subsections a detailed analysis of each system's component and interaction is presented.

4.2.2 BLE Communication

First of all, the Biosticker communicates measurements to the Smart box via BLE. Since the Biosticker implementation falls outside the scope of this dissertation, we have decided to conduct a vulnerability analysis of this communication and inform the responsible parties.

Table 4.1 presents the several BLE security modes and levels.

	BLE Security Modes	
	Mode 1[1]	Mode 2[2]
Levels	1. No security (No authentication, no encryption); 2. Unauthenticated pairing with AES-CMC encryption; 3. Authenticated pairing with encryption; 4. Authenticated LE Secure Connections pairing with ECDHE encryption.	1. Unauthenticated pairing with data signing; 2. Authenticated pairing with data signing.

Table 4.1: BLE security modes. Source: Microchip Technology[25]

[1] Each level of security meets the standards of the ones above it.

[2] This mode is only used for connection-based data signing.

In BLE security levels 3 and 4 there are several pairing methods that can also prevent man-in-the-middle (MITM) attacks, such as [94]:

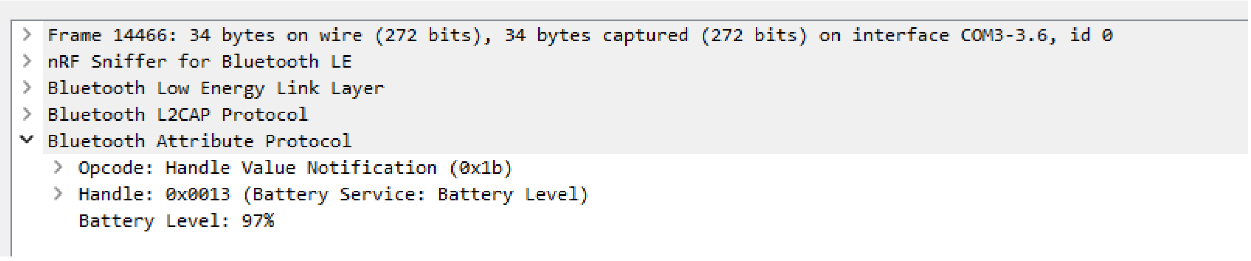
- **Just Works** is the default pairing method for most BLE networks. The value of the temporary key exchanged between devices in the second phase of pairing is set to 0 for BLE. Legacy connections are also set to 0, and devices create the short-term key value based on this. It is clear that such a pairing technique is very insecure and does not provide protection, but only a way to establish a connection.

- **Out of Band Pairing** (OOB) allows some data packets to be transmitted over an alternate wireless protocol. OOB pairing can be used during the second phase of pairing so that keys exchanged between devices are not sent over the less secure BLE protocol, and also when a device is sending sensitive data. Near Field Communication (NFC) is commonly applied for OOB pairing [95], e.g. when two devices physically connect to each other. Since the devices are so close, it is assumed that the intention is to pair them.

- **Passkey Pairing** makes users part of the security process. Basically, the initiating device displays a six-digit number between 000000 and 999999. Then the user must enter that same number into the responding device, if it has an input function. The main drawback of this pairing is that the devices require input/output functions (such as keyboards or touchscreens) to interafce with users, which may be difficult to integrate on tiny portable devices.

According to Kacherovska [94], the only method that does not protect against MITM attacks is Just Works. While passkey pairing protects against MITM, this method does not protect against passive eavesdropping [96], an attack that allows a foreign device to eavesdrop on data transmitted between devices on a BLE network.

For testing the security of BLE communications from the Biostickers, a BLE sniffer has been employed. We have adopted the nRF sniffer because it is deployed by the same company (Nordic Semiconductor[26]) as the Biosticker hardware. The nRF Sniffer for BLE allows near real-time display of BLE packets, enabling us to check whether the data is encrypted or not.



```
> Frame 14466: 34 bytes on wire (272 bits), 34 bytes captured (272 bits) on interface COM3-3.6, id 0
> nRF Sniffer for Bluetooth LE
> Bluetooth Low Energy Link Layer
> Bluetooth L2CAP Protocol
v Bluetooth Attribute Protocol
    > Opcode: Handle Value Notification (0x1b)
    > Handle: 0x0013 (Battery Service: Battery Level)
      Battery Level: 97%
```

Figure 4.5: Packet transmitting battery level captured by the nRF sniffer using the Wireshark software.

As can be seen in Figure 4.5, we can easily see the battery level (97%) without having to decrypt the information, so there is **no data encryption**. In addition, there is also **no pairing method**, as we can simply connect to the Biosticker without requiring authentication, and there is also **no data signing**, as it would be visible in Figure 4.5 if it were signed, so we can confirm that the BLE communication adopts mode 1 and level 1, which makes this communication a major vulnerability in the system.

Our recommendation to the team responsible for the development of Biostickers in the project is to use Mode 2 (signing the data) and Level 4 (encrypting the data using the ECDHE protocol) and to further use NFC for OOB pairing, given that the Biosticker is a very small scale device that has neither input nor output functionalities, making it unfeasible to use the passkey pairing method.

[26] Nordic Semiconductor, https://www.nordicsemi.com

4.2.3 MQTT Communication

As previously mentioned, the connection to the Smart boxes is achieved through MQTT.

According to IBM [97], there are four concepts that are fundamental to ensure MQTT security:

- Identity: In order for a client to connect with a broker, it must first send a connect request. When seeking a connection, the MQTT protocol requires a client to report a client id. Every client should in theory have a unique client identifier; most devices have a universal unique identifier (UUID) or the MAC address of the network device used to connect the client. When a broker receives a connect command from a client, it should check whether the message has a valid client id, username, and password to make sure that the client is qualified to connect;

- Authentication: In addition to username and password authentication, the MQTT protocol allows a device to authenticate with a X.509 certificate. X.509 is a digital certificate that uses a public key infrastructure to verify that a public key belongs to a client. To be able to use X.509 authentication, the client must use TLS as its encryption mechanism;

- Authorization: Authorization is not part of the MQTT protocol, so it needs to be provided by MQTT servers. When a client connects to the broker, it may execute two actions: publish and subscribe to topics. Topics are the primary resource available to clients and should require protection against unauthorized access. Role Based Access Controls (RBAC) and Access Control Lists (ACL) are two kinds of authorization. A role in RBAC offers a level of abstraction between a client and the main resource. Permissions are always associated with a certain role, which allows the broker to authorize a client's ability to post or subscribe to a specific topic. ACL assigns a set of permissions to certain clients. These permissions provide policies on which topics a client can subscribe/publish to.

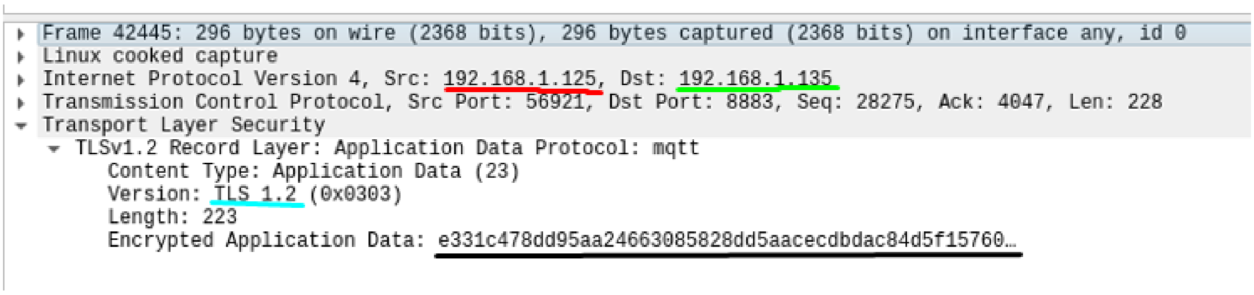
- Encryption: Encryption is also not part of the MQTT protocol, so it needs to be implemented by developers. There are two ways of securing data: TLS security or payload encryption. This security is part of the TCP/IP protocol and provides an encrypted pipe where the MQTT messages can flow, which will protect all parts of the MQTT message, and not just the message payload. Payload encryption is done at the application level and not by the broker. This means that data is encrypted end to end and not just between the broker and the client.

With these concepts in mind, we decided to use TLS v1.2[27], the latest version of TLS compatible with Mosquitto, to authenticate and encrypt transmissions between devices. For authentication and identification, each MQTT client has its own X.509 v3[28] certificate, which is the latest version of the X.509 certificates, and UUID to identify it. As for authorization, the system uses a RBAC policy to authorize access to the MQTT topics. We have decided to use RBAC instead of ACL because in this type of access control the system allows or revokes access to resources according to the role of the device, which means that all devices with a given role have the same list of permissions.

As mentioned in the IBM guide, authorization needs to be provided by MQTT servers. According to [98], there are three choices for authorization: password files, authentication plugins, and unauthorized/anonymous access. Unauthorized/anonymous access, as the name implies, provides no security. Password files are the standard authorization mechanism that stores usernames and passwords in a single file that is processed at program startup. Authentication plugins provide developers additional control over authentication than the password file mechanism.

A custom plugin to meet the security requirements is used, as described in [89]. This plugin intercepts authorization and authentication requests from the MQTT broker and validates the information contained in them. It queries the list of permissions associated with the client role and then checks whether any permission in that list explicitly grants PUBLISH access to the topic.

Figure 4.6 shows a packet sent from the Smart box (IP underlined in red) to the Gateway (IP underlined in green) showing that TLS v1.2 (blue underline) is being used with MQTT (same line) as expected, and that the data is being encrypted (black underline).



```
▸ Frame 42445: 296 bytes on wire (2368 bits), 296 bytes captured (2368 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 192.168.1.125, Dst: 192.168.1.135
▸ Transmission Control Protocol, Src Port: 56921, Dst Port: 8883, Seq: 28275, Ack: 4047, Len: 228
▾ Transport Layer Security
  ▾ TLSv1.2 Record Layer: Application Data Protocol: mqtt
      Content Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 223
      Encrypted Application Data: e331c478dd95aa24663085828dd5aacecdbdac84d5f15760…
```

Figure 4.6: Secured MQTT packet captured by Wireshark.

When attempting connect to the Gateway without the validated X.509 certificate, it will not grant permission to the device trying to connect.

Considering that the data is encrypted, that we cannot connect to the devices unless we

[27] TLS v1.2,https://datatracker.ietf.org/doc/html/rfc5246

[28] X.509 V3,https://tools.ietf.org/html/rfc5280

have validated X.509 certificates, and that we cannot publish messages unless we have the permissions to do so, we believe that MQTT data transmission is extremely secure.

4.2.4 FHIR Communication

According to Gupta [99], there are several steps to achieve secure REST APIs.

First, the author points out that it is important to always use Hypertext Transfer Protocol Secure (HTTPS) instead of HTTP because HTTPS uses TLS to encrypt normal HTTP requests and responses, as authentication credentials can be simplified to a randomly generated access token. Therefore, we have decided to use this in our server.

Additionally, other measures mentioned by the author were considered and applied during the development of the server, such as non-disclosure of information on URLs and validation of input parameters.

Finally, the *OAuth2*[29] protocol is used for authorization. OAuth2 relies on authentication scenarios, called *flows*, that allow the resource owner (user) to share the protected content of the resource server without revealing its credentials. For this purpose, an OAuth2 server issues access tokens that client applications can use to access protected resources on behalf of the resource owner. This protocol has been designed to allow websites and applications to access resources hosted by other web apps on behalf of users. When the FHIR server launches, a new OAuth2 token is requested and it is then used for every interaction with the GlobalCare HIS. When the token expires, a new one is requested. An example of an OAuth2 interaction can be seen in Figure 4.7.

Considering that we use the de facto industry standard for online authorization, our FHIR communication is limited to approved users. Also, we use HTTPS instead of HTTP, which makes our FHIR communication private because even if it is intercepted during transmission, it is encrypted. For these reasons, we believe that the FHIR communication is extremely secure.

4.2.5 Device Security

As previously mentioned, there are two physical devices developed in the context of this dissertation: the Smart box and the Gateway. Since these devices store the vital signs acquired by the Biostickers, they must also be secure.

There are several practices that can be employed to achieve a secure system, according to AWS [100]. With this in mind, we have implemented all the feasible techniques from this guide, including:

[29] OAuth2, `https://oauth.net/2/`

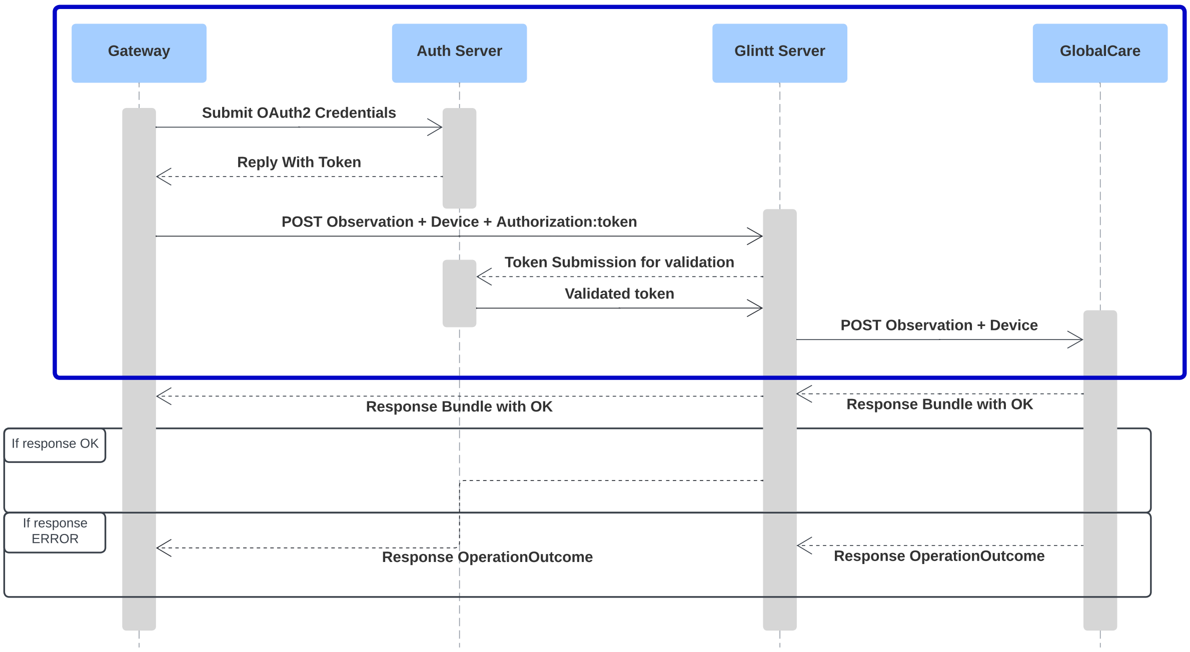

Gateway
Auth Server
Glintt Server
GlobalCare
Submit OAuth2 Credentials
Reply With Token
POST Observation + Device + Authorization:token
Token Submission for validation
Validated token
POST Observation + Device
Response Bundle with OK
Response Bundle with OK
If response OK
If response ERROR
Response OperationOutcome
Response OperationOutcome

Figure 4.7: Sequence diagram describing the request and use of an OAuth2 token.

- Assign unique identities to all devices: as mentioned in the previous sections, each device has a unique UUID to identify it.
- Assign unique and cryptographic credentials such as X.509 certificates to each identity.
- Monitor the numerous services implemented in the devices: all of the services implemented in the devices have a logging system to track every event that occurs when the software is running.
- Minimize the attack surface of our IoT ecosystem: we have identified and eliminated unused entry points on the devices using the firewall ufw[30].
- Avoid unnecessary data access, storage, and transmission: we use the RBAC policy for MQTT authorization, which allows to restrict a device's access to its own individual topics.

With these methods, we firmly believe that our devices have achieved an appropriate level of security. However, some improvements can be made, such as creating an automatic device provisioning mechanism and a continuous update mechanism for deploying security updates and patches.

[30] UFW, `https://code.launchpad.net/ufw`

4.3 Summary

In this chapter, we explain how we achieve interoperability with the Globalcare HIS and the steps we took to achieve a secure system.

As for interoperability, a FHIR integration service has been developed in the Gateway. This service consists of a HAPI FHIR plain server based on a Java servlet API with Eclipse Jetty as the web server. In developing this service, we worked with Glintt to establish a technical specification for the required functionalities, i.e., processing incoming requests to schedule smart box subscriptions, sending messages to the HIS each time a device is created or removed.

Regarding security, a vulnerability analysis of the overall system has been performed and the necessary improvements were implemented. For BLE communication, several recommendations were made to the responsible team for the development of Biostickers. For MQTT data transmission, TLS is used for data security, X.509 certificates for authentication and identification, and a RBAC policy for authorization. Finally, a custom plugin is used to validate authorization and authentication requests. For FHIR communication, HTTPS is used instead of HTTP, and the OAuth2 protocol is used for authorization. Finally, several techniques have been implemented for device security, such as assigning unique identities to all devices, using a firewall to minimize the attack surface, and avoiding unnecessary data access.

Next, the performance of the proposed system is evaluated through system testing.

5 Experimental Validation

Experimental tests were conducted to assess and validate the performance of the system. Due to logistical issues, these tests were not conducted in a real-world scenario, since the responsible parties were not able to timely set up an infrastructure at Centro Hospitalar e Universitário de Coimbra (CHUC), therefore these tests were conducted in a controlled environment.

5.1 Experimental Setup

The objective is to validate the system, its stability and reliability, taking into account the new security measures and the new service introduced to achieve interoperability with the HIS. For these tests, we prepared a Smart box, a Gateway, a nRF board to replace the Biosticker and the HIS was set up at the Glintt's facility (see Figure 5.1). This board was conveniently used to eliminate the battery-related limitations of the Biosticker, and it mimicks a Biosticker, sending simulated data instead of real patient data collected by the sensors. The Wi-Fi network used to connect the Smart box to the Gateway is a domestic 2.4 GHz network, whose bandwidth is shared with other home residents.

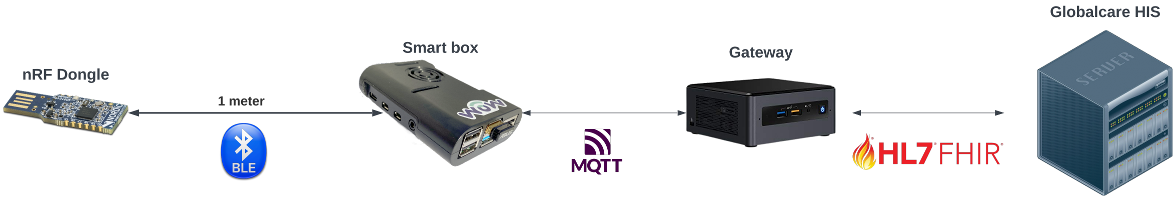


Figure 5.1: Experimental setup diagram.

The main objective is to evaluate the system with a single patient during 24 hours. Sensor acquisition rates and data information are described in Table 3.1. Regarding FHIR subscriptions, the subscription rate was changed during testing and varied between 20 and 30 minutes for all sensors, i.e. the latest measurement of each vital sign, specified in Chapter 4, is communicated to the HIS every 20/30 minutes. The subscription rate was changed only twice, from 20 minutes to 30 minutes after 6 hours and back to 20 minutes after 8 hours. The main goal was to see if this would have any effect on the results.

To evaluate the reliability of the system, the bandwidth used by the BLE between Biosticker and Smart box, bandwidth used by MQTT between Smart box and Gateway and the latency of the FHIR communication protocol between Gateway and HIS are measured. Any interruptions in each communication link are also taken into account, and the synchronization functionality was tested once by interrupting the Wi-Fi connection after 12 hours for 25 minutes. The resource consumption of each service and the GUI frame rate are monitored to evaluate the stability of the system. To evaluate the performance of the proposed system, the following performance metrics have been defined and measured during testing:

- BLE bandwidth (kbps) - Rate of data exchanged between Biosticker and Smart box;
- BLE and MQTT packet loss - Number of BLE and MQTT packets not received;
- GUI frame rate (FPS) - Measurement of how quickly a number of frames appears within a second;
- MQTT bandwidth (kbps) - Rate of data exchanged between Smart box and Gateway;
- FHIR round-trip time (ms) - Time period required by the Gateway to send a specific message to the HIS and to receive the acknowledgement;
- Resource usage of each service (%) - CPU and RAM usage of each service.

5.2 Results and Discussion

5.2.1 BLE Data Acquisition

Considering the sensor acquisition rates and data information described in Table 3.1, the BLE bandwidth is expected to be 5.76 kbps. Yet, the BLE connection is very inconsistent. In the course of the 24 hours test, there were 152 disconnections with an average duration of 14.56 ± 6.01 seconds and a total disconnection duration of 2213.69 seconds. Considering the duration of the tests, a connection interruption occurred every 569 seconds, i.e. about every 9 minutes and 30 seconds, which is a percentage of 2.56 % of the test duration. Considering that the nRF board was constantly at a distance of about 1 meter from the Smart box, the interruptions could not be due to the range. The reason for these disconnections is that the firmware of the Biosticker, which falls outside the scope of this dissertation, contains interruptions in the script that make the BLE transmission restart randomly. Furthermore, this relatively high average duration of disconnections can be explained by the fact that the

Biosticker restarts if it does not receive a keep-alive flag every 3 seconds, which increases the time to reconnect to the Biosticker. If we exclude these disconnections, we obtain an average measured bandwidth of 5.76 ± 0.2 kbps, which means that the Smart box receives every measurement sent by the Biosticker, thus we have no packet loss when the devices are connected to each other.

Comparing these results with those of the previous trial of the project discussed in [89, 91], which took place before the start of this dissertation, we can observe that the BLE connection is more reliable and stable. In those tests, there was a disconnection every 80 seconds, and now we have no data loss, while in previous the trial some sensors reached more than 15% of data lost. It is important to mention that in the trial, a real Biosticker was used, which could explain why there were so many data losses.

5.2.2 GUI

As for the GUI's responsiveness, the average frame rate value is 21.90 ± 7.56 FPS, the maximum value is 32.2 FPS and the minimum is 4.4. This high standard deviation is due to the moments when the BLE acquisition fails, causing the frame rate to drop. When comparing the frame rate of the current version of the GUI with the version used in the previous trial, the average FPS increased 4.5 times (5 vs 22 Hz), proving that the changes made to the GUI were successful (see Chapter 3).

5.2.3 Resource Usage of the Smart box Services

Figures 5.2 and 5.3 depict the CPU and RAM use of the two Smart box services, i.e. BLE acquisition and MQTT transmission, and the GUI. It can be observed that the BLE acquisition and MQTT transmission CPU usage is relatively high, considering that the average CPU usage is 23.41%. This could be due to the fact that this service, as the name suggests, uses two threads, one to acquire data via BLE and another to send the data to the gateway via MQTT. It is also important to mention that the Raspberry Pi 4 Model B is a low computational power device, yet it has the necessary processing power to be used in the Smart box and fulfill its purposes. In the last trial, the BLE acquisition and MQTT transmission service used almost 40% of CPU, which means that this service is currently more efficient and that the Smart box is ready to receive more services, e.g., the respiration filtering and the IMU patient state classification sub-modules present on Figure 3.1.

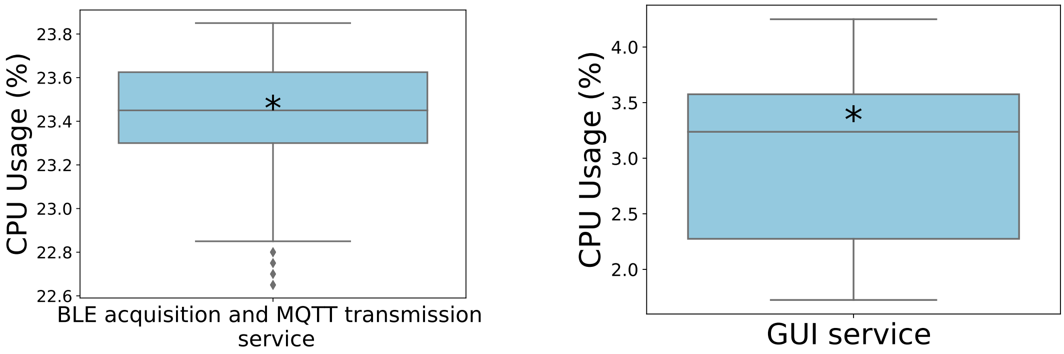


Figure 5.2: CPU usage services of the Smart box measured over time during the tests. The mean is indicated with the symbol "*".

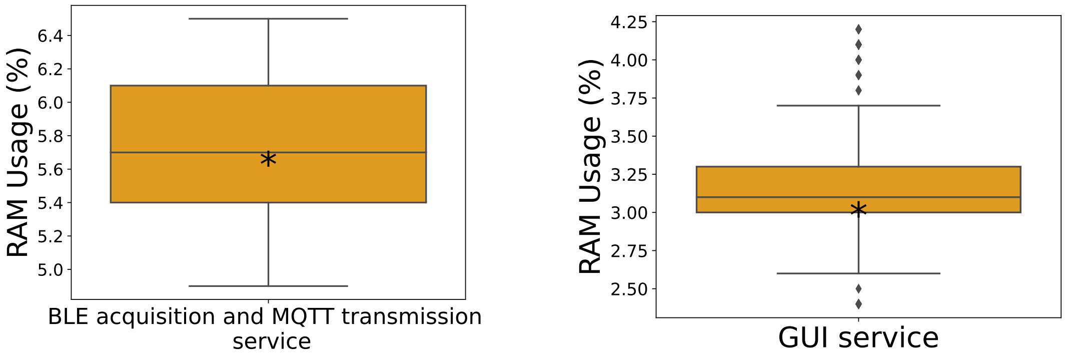


Figure 5.3: RAM usage of the services of the Smart box measured over time during the tests. The mean is indicated with the symbol "*".

5.2.4 MQTT Data Transmission

As for the bandwidth of MQTT transmissions, the expected bandwidth is 175.07 kbps. Unlike BLE, MQTT communication via Wi-Fi is robust, and no connection failures were reported. If we disregard the periods when no data was transmitted due to BLE acquisition breaks, we obtain an average measured bandwidth of 175.07 kbps, which means that the Smart box sends every measurement that should be sent and the Gateway receives it, and, in turn, there is no packet loss when the Smart box is acquiring data from the Biosticker. It is also important to mention that the synchronization functionality was tested with a disconnection period of 25 minutes. The time needed to resend the missing information was 63 seconds and did not affect the simultaneous transmission of the current information. When comparing the MQTT data transmission with the previous trial of the project, we can observe that the results have improved due to the implementation of the keep-alive feature. For instance, previously the MQTT connection would be interrupted when the BLE acquisition was stopped for more than 30 seconds and it was necessary to restart the MQTT connection, while this is not necessary in the current tests.

In terms of the overhead of the TLS security layer, the packet shown in Figure 4.6 weighs 296 bytes and carries a temperature measurement of 180 bytes. From the figure, we can see that the TLS weighs only 223 bytes, so the TLS overhead is about 40 bytes (223-180 = 43 bytes). If we analyze the remaining packets, we also obtain an average overhead of 40 bytes, which is the expected value [101]. Considering that there was no packet loss during the test, this overhead can be neglected given the benefit it brings.

5.2.5 FHIR Communication

Regarding FHIR communication, the Gateway successfully sent 210 measurements to the HIS during the 24 hours, which is the expected value, considering 9 messages per hour during 22 hours and 6 messages per hour during 2 hours. Figure 5.4 shows the round-trip time of each FHIR message sent during testing.

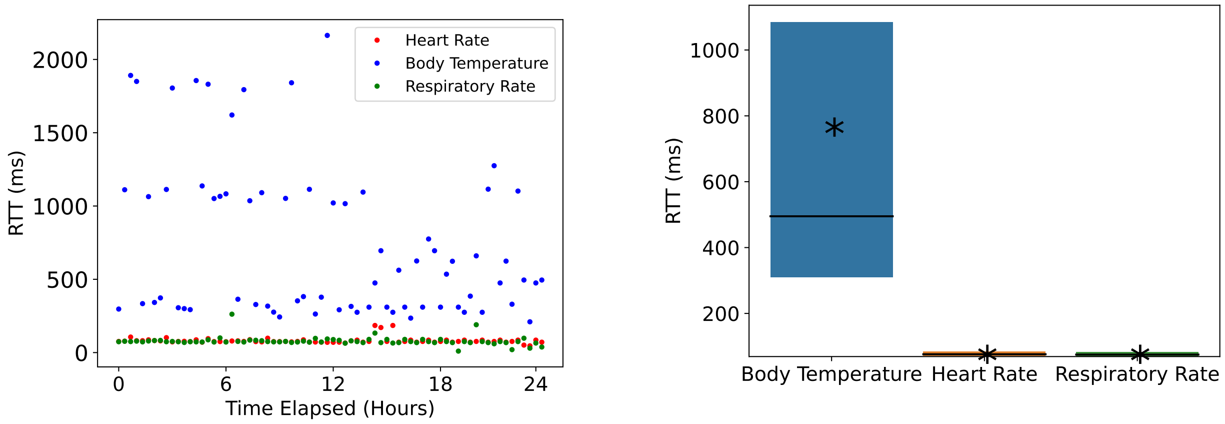


Figure 5.4: FHIR round-trip time measured. The mean is indicated with the symbol “*”.

As seen in the graph, the body temperature messages have the highest average RTT duration. If we measure the average RTT time for each vital sign, we obtain 740.0±529.99 ms for body temperature, 81.65 ± 22.64 ms for heart rate and 78.63 ± 30.56 ms for respiratory rate. This can be explained by the order in which the messages are sent. In this case, every 20/30 minutes, the last three values corresponding to these sensor readings are retrieved from the database and sent to HIS. The important aspect of this is that the temperature is always the first value, which results in the body temperature having a greater RTT than the others. When we changed the order of the transmissions, respiratory rate was first and its RTT increased to 728.35 ± 487.45 ms , while body temperature decreased to 90.42 ± 27.34 ms. This is because the first message transmitted always takes longer to be acknowledged by HIS, as the FHIR server deployed verifies whether the incoming bundles are compliant or not. It is worth noting that the same tests were performed on a public FHIR server[31] and

[31] `https://hapi.fhir.org/baseR4`

the results were identical. Regarding the obtained values, it is important to mention that in these tests, there is no processing of messages on the HIS side, which means that these RTT values would increase as soon as Glintt adds processing routines on their server.

5.2.6 Resource Usage of the Gateway Services

Figure 5.5 show the CPU and RAM use of the several Gateway services. The overall resource consumption is very low for all Gateway services, with less than 5% CPU and 7% RAM, indicating that the services were implemented efficiently. From the last experiment, it appears that the FHIR service now requires more RAM (0.8% vs. 5%) than before, which is to be expected since the FHIR implementation was still in its early stages.

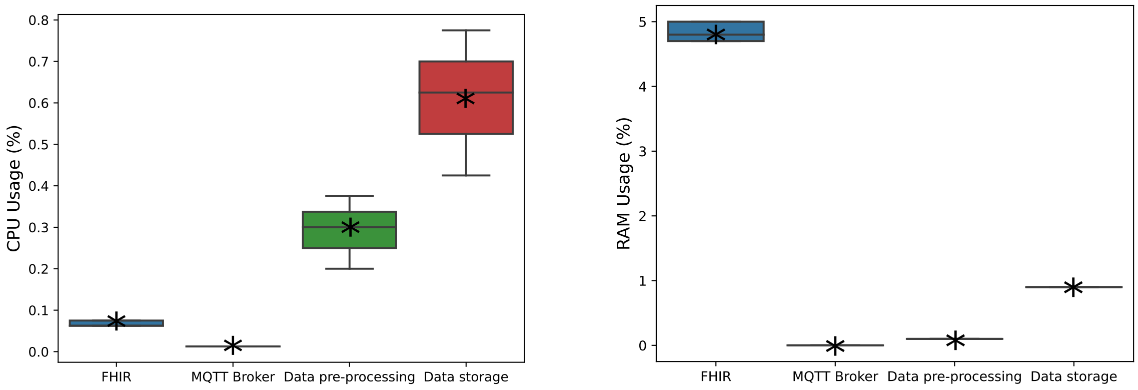


Figure 5.5: Average CPU and RAM usage of the services of the Gateway measured over time during the tests. The mean is indicated with the symbol “*”.

5.3 Summary

Overall, the reported results are extremely positive, since we can conclude that the system is significantly more stable and reliable with the new mechanisms and improvements introduced in this dissertation. In particular, BLE acquisition and the GUI have been significantly improved by reducing BLE connection drops from every 80 seconds to every 10 minutes, improving the responsiveness of the GUI by about 400%, and improving usability by adding several features. MQTT data transmission has also been consolidated by implementing multiple features that prevents disconnections and allows retransmission of lost data. FHIR communication is also extremely robust as there were no connection drops throughout the 24 hours, all measurements were received by the HIS even when the subscription rate was changed.

6 Conclusion

The area of digital healthcare still has vast research possibilities and the work presented in this dissertation addresses one of the core subjects in this context: an IoT architecture for wireless monitoring of patients.

In this last chapter, a global and self-critical overview of the proposed approach is given to summarise the work. Final conclusions and possible related future directions of work are also discussed.

6.1 Main Outcomes

In this dissertation, an interoperable and secure IoT architecture for wireless monitoring of untethered patients in smart beds was developed.

Initially, a thorough survey of the literature on IoT with a focus on IoT for digital healthcare was presented. Reviewing the current state of the art allowed for building general knowledge and experience, as well as identifying problems and important issues in the field, thus capturing important research opportunities and gaining new insights. Several gaps have been found and pointed out in related work, such as no mention of redundancy mechanisms, lack of interoperability with external healthcare systems, some authors focus only on data or network security, and there is no specific way to achieve interoperability, so standards recognized by the ITU (International Telecommunication Union) and the European Commission should be used. With this in mind, several objectives have been presented in section 2.3.4.

Regarding the implementation, the BLE data acquisition from the Biostickers was redesigned to include threads to address a critical issue that was hindering the system, resulting in a significant reduction in BLE connection drops. Based on a list of potential improvements established in collaboration with CHUC health professionals, the Smart Box graphical user interface was completely redesigned, resulting in an approximately 400% increase in frame rate and usability by adding features such as warnings when a BLE or MQTT connection interruption occurs or new pages. MQTT data transmission has also been consolidated by implementing several features, such as a redundancy mechanism to retransmit data between a Smart box and the Gateway if the Wi-Fi connection suddenly drops, a pairing functionality that allows the Gateway to manage each pairing of the Smart box and Biosticker, and a keep-alive feature that prevents the MQTT connection from being interrupted. As for the synchronization functionality, several tests were performed to define the maximum payload size of each sent message in order to restore the missing data while minimizing the synchronization time without information loss. A total of 4300 consecutive tests were performed, which allowed us to dimension the messages with a mean value of 13.04 MB for the minimum time required to retransmit the missing information.

As for interoperability, a FHIR integration service was developed in the Gateway. This service consists of a HAPI FHIR plain server based on a Java servlet API with Eclipse Jetty as the web server. In developing this service, the author worked with Glintt to establish a technical specification for the required functionalities, i.e., processing incoming requests to schedule smart box subscriptions, sending messages to the HIS each time a device is created or removed.

Regarding security, a vulnerability analysis of the overall system has been performed and the necessary improvements were implemented. For BLE communication, several recommendations were made to the responsible team for the development of Biostickers. For MQTT data transmission, TLS is used for data security, X.509 certificates for authentication and identification, and a RBAC policy for authorization. Finally, a custom plugin is used to validate authorization and authentication requests. For FHIR communication, HTTPS is used instead of HTTP, and the OAuth2 protocol is used for authorization. Finally, several techniques have been implemented for device security, such as assigning unique identities to all devices, using a firewall to minimize the attack surface, avoiding unnecessary data access and every service is continuously being monitored to track every event that occurs while the software is running.

Due to external circumstances, testing in real-world scenarios could not be conducted. However, a full system test was performed in a controlled environment. The results showed that the system is more reliable and robust than prior to the developments reported in this dissertation. As for the security, it can be concluded that the TLS overhead can be neglected given the benefit it brings and that the major security gap is BLE communication.

Throughout the implementation of the project, there has always been a sense of confidence and motivation for its development, mainly due to two reasons: the challenge it represents and the enhancement that the project can bring to the lives of many patients. Generally, the author considers that all main objectives have been accomplished in the elaborated work, considering that the contributions of this dissertation have been directly applied in the WoW R&D project.

As part of the WoW project, the author currently holds a fellowship and has co-authored an article on the usage of different QoS levels with the MQTT protocol, pending decision at the time of writing. In addition, in connection with the experiments on the synchronization functionality (see Section 3.4) the author is preparing an article, pending submission at the time of writing.

6.2 Future Work

Some issues are still left open and correspond to future guidelines that can be used to improve the current work. BLE acquisition needs to be secured and stabilized to prevent disconnections. This should be one of the main focuses of future related work in the WoW project, as this dissertation found that the major security gap is in BLE communication. Additionally, more features can be added to the GUI, such as a translation button to promote the usage of this system in multiple countries. The Smart box setup can also be automated, including credentials for efficient, scalable, and secure communication mechanisms to facilitate deployment for system users, as it is still under the responsibility of the development team. Additional security improvements can be made, such as creating an automatic device provisioning mechanism and a continuous update mechanism for deploying security updates and patches. Furthermore, a redundancy mechanism for FHIR communication, like the one implemented for MQTT data transmission, should be implemented in the FHIR integration service of the Gateway to avoid data loss. Once the infrastructure is in place at the Centro Hospitalar e Universitário de Coimbra (CHUC), experimental validation in a real-world scenario should be performed. It is also important to evaluate the performance of the system with multiple patients to assess its scalability.

Finally, a different avenue to explore would be to develop AI techniques for multimodal sensor fusion based on patient monitoring data to detect sudden changes in patient condition, such as low respiratory cycles, fall events, and high temperature, for faster intervention.

7 Bibliography

[1] Ageing and health. https://www.who.int/news-room/fact-sheets/detail/ageing-and-health.

[2] The top 10 most common chronic conditions in older adults. https://www.ncoa.org/article/the-top-10-most-common-chronic-conditions-in-older-adults, Apr 2021.

[3] S. Karthikeyan, K. Vimala Devi, and K. Valarmathi. Internet of things: Hospice appliances monitoring and control system. In *2015 Online International Conference on Green Engineering and Technologies (IC-GET)*, pages 1–6, 2015. doi: 10.1109/GET.2015.7453776.

[4] Euijong Lee, Young-Duk Seo, Se-Ra Oh, and Young-Gab Kim. A survey on standards for interoperability and security in the internet of things. *IEEE Communications Surveys Tutorials*, 23(2):1020–1047, 2021. doi: 10.1109/COMST.2021.3067354.

[5] Cambridge english dictionary: Meanings& definitions. https://dictionary.cambridge.org/dictionary/english.

[6] C Pham, Y Lim, and Y Tan. Management architecture for heterogeneous iot data sources in home network. In *Proceedings of the 2016 IEEE 5th Global Conference on Consumer Electronics, Kyoto, Japan*, pages 11–14, 2016.

[7] Emna Mezghani, Ernesto Exposito, Khalil Drira, Marcos Da Silveira, and Cédric Pruski. A semantic big data platform for integrating heterogeneous wearable data in healthcare. *Journal of medical systems*, 39(12):1–8, 2015.

[8] James Manyika, Michael Chui, Peter Bisson, Jonathan Woetzel, Richard Dobbs, Jacques Bughin, and Dan Aharon. Unlocking the potential of the internet of things. =https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/the-internet-of-things-the-value-of-digitizing-the-physical-world, Feb 2020.

[9] Christy Pettey. The iot effect: Opportunities and challenges. https://www.gartner.com/smarterwithgartner/the-iot-effect-opportunities-and-challenges-2/, Mar 2017.

[10] Steve Alder. February 2020 healthcare data breach report. https://www.hipaajournal.com/february-2020-healthcare-data-breach-report/, Apr 2020.

[11] Wow Deliverable D1.2, "Functional Specification and System Integration Architecture, Dec 2020.

[12] What is the internet of things (iot)? https://www.oracle.com/internet-of-things/what-is-iot/, .

[13] Klaus Schwab. *The fourth industrial revolution.* Currency, 2017.

[14] Smart parking. Smartpark system. https://www.smartparking.com/smartpark-system, Apr 2021.

[15] Smart farming technology - advanced agriculture solution. https://www.cropin.com/smart-farming/, Aug 2019.

[16] Smart water systems: Smart utility network: Smart water metering. https://sensus.com/smart-utility-network/smart-water/, Apr 2021.

[17] Hesham El-Sayed and Gokulnath Thandavarayan. Congestion detection and propagation in urban areas using histogram models. *IEEE Internet of Things Journal*, 5(5): 3672–3682, 2018. doi: 10.1109/JIOT.2017.2665662.

[18] C. Arcadius Tokognon, Bin Gao, Gui Yun Tian, and Yan Yan. Structural health monitoring framework based on internet of things: A survey. *IEEE Internet of Things Journal*, 4(3):619–635, 2017. doi: 10.1109/JIOT.2017.2664072.

[19] Rosario Morello, Claudio De Capua, Gaetano Fulco, and Subhas Chandra Mukhopadhyay. A smart power meter to monitor energy flow in smart grids: The role of advanced sensing and iot in the electric grid of the future. *IEEE Sensors Journal*, 17(23):7828–7837, 2017.

[20] David Niewolny. How the internet of things is revolutionizing healthcare.

[21] Marti Leitch. Study shows what patients need to feel comfortable during hospital stay, May 2017. URL https://wexnermedical.osu.edu/mediaroom/pressreleaselisting/patient-room-comfort.

[22] Carly Weeks. Better by design: How a hospital room can help patients heal, Feb 2014. URL `https://www.theglobeandmail.com/life/health-and-fitness/health/the-hospital/better-by-design-how-a-hospital-room-can-help-patients-heal/article16748288/`.

[23] Carrie Meadows. Please enable cookies, Nov 2020. URL `https://www.ledsmagazine.com/blogs/article/14187345/surveyed-healthcare-workers-send-a-clear-message-about-lighting-control`.

[24] Igor. Somfy and igor integrate for seamless smart building control, Oct 2020.

[25] HBC Editors. Smart beds & rooms in hospitals for better patient care, May 2021. URL `https://healthcarebusinessclub.com/articles/healthcare-provider/technology/smart-beds-rooms-hospitals/`.

[26] Jessilyn Dunn, Ryan Runge, and Michael Snyder. Wearables and the medical revolution. *Personalized medicine*, 15(5):429–448, 2018.

[27] André Henriksen, Martin Haugen Mikalsen, Ashenafi Zebene Woldaregay, Miroslav Muzny, Gunnar Hartvigsen, Laila Arnesdatter Hopstock, and Sameline Grimsgaard. Using fitness trackers and smartwatches to measure physical activity in research: analysis of consumer wrist-worn wearables. *Journal of medical Internet research*, 20(3): e9157, 2018.

[28] Chulsung Park, Pai H Chou, Ying Bai, Robert Matthews, and Andrew Hibbs. An ultra-wearable, wireless, low power ecg monitoring system. In *2006 IEEE biomedical circuits and systems conference*, pages 241–244. IEEE, 2006.

[29] Toshiya Arakawa. Recent research and developing trends of wearable sensors for detecting blood pressure. *Sensors*, 18(9):2772, 2018.

[30] Priyan Malarvizhi Kumar, S Lokesh, R Varatharajan, Gokulnath Chandra Babu, and P Parthasarathy. Cloud and iot based disease prediction and diagnosis system for healthcare using fuzzy neural classifier. *Future Generation Computer Systems*, 86: 527–534, 2018.

[31] Parag Chatterjee, Leandro J Cymberknop, and Ricardo L Armentano. Iot-based decision support system for intelligent healthcare—applied to cardiovascular diseases. In *2017 7th International Conference on Communication Systems and Network Technologies (CSNT)*, pages 362–366. IEEE, 2017.

[32] Fardin Abdali-Mohammadi, Maytham N Meqdad, and Seifedine Kadry. Development of an iot-based and cloud-based disease prediction and diagnosis system for healthcare using machine learning algorithms. *IAES International Journal of Artificial Intelligence*, 9(4):766, 2020.

[33] T. Dylan McAllister, Samy El-Tawab, and M. Hossain Heydari. Localization of health center assets through an iot environment (locate). In *2017 Systems and Information Engineering Design Symposium (SIEDS)*, pages 132–137, 2017. doi: 10.1109/SIEDS.2017.7937703.

[34] Mert Bal and Reza Abrishambaf. A system for monitoring hand hygiene compliance based-on internet-of-things. In *2017 IEEE International Conference on Industrial Technology (ICIT)*, pages 1348–1353. IEEE, 2017.

[35] Carman Ka Man Lee, Mei Na Cheng, and Chun Kit Ng. Iot-based asset management system for healthcare-related industries. *International Journal of Engineering Business Management*, 7(Godište 2015):7–19, 2015.

[36] Y Chen and H Chang. Improve the management of pharmaceutical inventory by using an iot based information system. *Int J Soc Sci Humanit*, 7(8):569–573, 2017.

[37] Jose Cabra, D Castro, Julián Colorado, Diego Mendez, and L Trujillo. An iot approach for wireless sensor networks applied to e-health environmental monitoring. In *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 578–583. IEEE, 2017.

[38] Vanessa A Diaz and Marty S Player. Direct-to-patient telehealth: Opportunities and challenges. *Rhode Island Medical Journal*, 103(1), 2020.

[39] Abdelhak Kharbouch, Youssef Naitmalek, Hamza Elkhoukhi, Mohamed Bakhouya, Vincenzo De Florio, Moulay Driss El Ouadghiri, Steven Latre, and Chris Blondia. Iot and big data technologies for monitoring and processing real-time healthcare data. *International Journal of Distributed Systems and Technologies (IJDST)*, 10(4):17–30, 2019.

[40] Elisa Spanò, Stefano Di Pascoli, and Giuseppe Iannaccone. Low-power wearable ecg monitoring system for multiple-patient remote monitoring. *IEEE Sensors Journal*, 16 (13):5452–5462, 2016.

[41] Paula Muto. How digital healthcare technology can reinvent the doctor-patient relationship, Jul 2021. URL https://builtin.com/healthcare-technology/how-digital-healthcare-can-reinvent-doctor-patient-relationship.

[42] Arijit Ukil, Soma Bandyoapdhyay, Chetanya Puri, and Arpan Pal. Iot healthcare analytics: The importance of anomaly detection. In *2016 IEEE 30th international conference on advanced information networking and applications (AINA)*, pages 994–997. IEEE, 2016.

[43] Iot: Architecture. https://www.m2mology.com/iot-transformation/iot-world-forum/, Jun 2016.

[44] What is ethernet? https://www.linksys.com/us/r/resource-center/basics/whats-ethernet/.

[45] Introduction to iot: What is wifi. https://www.leverege.com/iot-ebook/iot-wifi, .

[46] Roland Minihold. Near field communication (nfc) technology and measurements, Jun 2011.

[47] Bluetooth technology overview. https://www.bluetooth.com/learn-about-bluetooth/tech-overview/.

[48] AV System. What is lpwan?, Oct 2020. URL https://www.avsystem.com/blog/LPWAN/.

[49] Zigbee faq. https://zigbeealliance.org/zigbee-faq/, Nov 2019.

[50] Ilya Grigorik. Performance of wireless networks: Mobile networks, Apr 2016. URL https://hpbn.co/mobile-networks/.

[51] GSMA. Security features of lte-m and nb-iot networks, Sep 2019.

[52] Josh Miller. Rfid frequencies: Low, high, and ultra high; what they are and why it matters. URL https://www.computype.com/blog/rfid-frequencies-low-high-and-ultra-high.

[53] Alexis Leibbrandt. Iot connectivity landscape, Jan 2021. URL https://akenza.io/blog/iot-connectivity-landscape.

[54] Roy Pramono Adhie, Yonatan Hutama, A Saleh Ahmar, MI Setiawan, et al. Implementation cryptography data encryption standard (des) and triple data encryption

standard (3des) method in communication system based near field communication (nfc). In *Journal of Physics: Conference Series*, volume 954, page 012009. IOP Publishing, 2018.

[55] Chalee Thammarat and Werasak Kurutach. A lightweight and secure nfc-base mobile payment protocol ensuring fair exchange based on a hybrid encryption algorithm with formal verification. *International Journal of Communication Systems*, 32(12):e3991, 2019.

[56] NXP Semiconductors. Advanced security and privacy for trusted iot applications, 2021.

[57] Robert Miller. Building a secure lora solution, 2016.

[58] Shreyaasha Chaudhury, Debasmita Paul, Ruptirtha Mukherjee, and Siddhartha Haldar. Internet of thing based healthcare monitoring system. In *2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON)*, pages 346–349, 2017. doi: 10.1109/IEMECON.2017.8079620.

[59] Md Islam, Ashikur Rahaman, et al. Development of smart healthcare monitoring system in iot environment. *SN computer science*, 1(3):1–11, 2020.

[60] Afonso Raposo, Luis Marques, Rafael Correia, Francisco Melo, João Valente, Telmo Pereira, Luis Brás Rosário, Filipe Froes, João Sanches, and Hugo Plácido da Silva. e-covig: A novel mhealth system for remote monitoring of symptoms in covid-19. *Sensors*, 21(10), 2021. ISSN 1424-8220. doi: 10.3390/s21103397. URL https://www.mdpi.com/1424-8220/21/10/3397.

[61] Eyhab Al-Masri, Karan Kalyanam, John Batts, Jonathan Kim, Sharanjit Singh, Tammy Vo, and Charlotte Yan. Investigating messaging protocols for the internet of things (iot). *IEEE Access*, PP:1–1, 05 2020. doi: 10.1109/ACCESS.2020.2993363.

[62] Chris Pietschmann. Top 5 iot messaging protocols, Jan 2020. URL https://build5nines.com/top-iot-messaging-protocols/.

[63] Iot growth demands rethink of long-term storage strategies, says idc. https://www.idc.com/getdoc.jsp?containerId=prAP46737220.

[64] Sam Saltis. Top 3 iot challenges: Data, data and data, May 2020. URL https://www.coredna.com/blogs/iot-challenges.

[65] Abdur Rahim Biswas and Raffaele Giaffreda. Iot and cloud convergence: Opportunities and challenges. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*, pages 375–376. IEEE, 2014.

[66] Diksha Rana. Top 11 cloud platforms for internet of things (iot) - dzone iot. https://dzone.com/articles/10-cloud-platforms-for-internet-of-things-iot, Aug 2020.

[67] Ionut Arghire. Encrypted threats, iot malware surge past 2018 levels: Report, 2019. URL https://www.securityweek.com/encrypted-threats-iot-malware-surge-past-2018-levels-report.

[68] Jayant D Bokefode, Avdhut S Bhise, Prajakta A Satarkar, and Dattatray G Modani. Developing a secure cloud storage system for storing iot data by applying role based encryption. *Procedia Computer Science*, 89:43–50, 2016.

[69] Cisco. The internet of things reference model, 2014.

[70] What is an electronic health record (ehr)? https://www.healthit.gov/faq/what-electronic-health-record-ehr, Sep 2019.

[71] Fhir overview. https://www.hl7.org/fhir/overview.html.

[72] Mihai Voicu. 5 essential steps to secure enterprise iot deployments, Jul 2021. URL https://www.telit.com/blog/five-steps-secure-iot-deployments/.

[73] Trend Micro. Iot security issues, threats, and defenses, Jul 2021. URL https://www.trendmicro.com/vinfo/us/security/news/internet-of-things/iot-security-101-threats-issues-and-defenses.

[74] Ida Hübschmann. A developers guide to iot encryption algorithms, Jan 2021. URL https://www.nabto.com/iot-data-encryption-algorithm-guide/.

[75] Mahda Noura, Mohammed Atiquzzaman, and Martin Gaedke. Interoperability in internet of things: Taxonomies and open challenges. *Mobile Networks and Applications*, 24(3):796–809, 2019.

[76] Mike Ushold and Christopher Menzel. Semantic integration & interoperability using rdf and owl, Nov 2005. URL https://www.iso.org/obp/ui/.

[77] Jeroen Hoebeke, Eli De Poorter, Stefan Bouckaert, Ingrid Moerman, and Piet Demeester. Managed ecosystems of networked objects. *Wireless Personal Communications*, 58(1):125–143, 2011.

[78] Hesham A. El Zouka and Mustafa M. Hosni. Secure iot communications for smart healthcare monitoring system. *Internet of Things*, 13:100036, 2021. ISSN 2542-6605.

doi: https://doi.org/10.1016/j.iot.2019.01.003. URL https://www.sciencedirect.com/science/article/pii/S254266051830088X.

[79] Fan Wu, Taiyang Wu, and Mehmet Rasit Yuce. An internet-of-things (iot) network system for connected safety and health monitoring applications. *Sensors*, 19(1), 2019. ISSN 1424-8220. doi: 10.3390/s19010021. URL https://www.mdpi.com/1424-8220/19/1/21.

[80] Prosanta Gope and Tzonelih Hwang. Bsn-care: A secure iot-based modern healthcare system using body sensor network. *IEEE Sensors Journal*, 16(5):1368–1376, 2016. doi: 10.1109/JSEN.2015.2502401.

[81] P. K. Binu, Karun Thomas, and Nithin P. Varghese. Highly secure and efficient architectural model for iot based health care systems. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 487–493, 2017. doi: 10.1109/ICACCI.2017.8125887.

[82] Muhammad Yasin, Temesghen Tekeste, Hani Saleh, Baker Mohammad, Ozgur Sinanoglu, and Mohammed Ismail. Ultra-low power, secure iot platform for predicting cardiovascular diseases. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 64(9):2624–2637, 2017. doi: 10.1109/TCSI.2017.2694968.

[83] Abdullah Alamri. Ontology middleware for integration of iot healthcare information systems in ehr systems. *Computers*, 7(4), 2018. ISSN 2073-431X. doi: 10.3390/computers7040051. URL https://www.mdpi.com/2073-431X/7/4/51.

[84] Jaeki Hong, Peter Morris, and Jonghwa Seo. Interconnected personal health record ecosystem using iot cloud platform and hl7 fhir. In *2017 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 362–367, 2017. doi: 10.1109/ICHI.2017.82.

[85] Jesús N S Rubí and Paulo R L Gondim. Iomt platform for pervasive healthcare data aggregation, processing, and sharing based on onem2m and openehr. *Sensors*, 19(19):4283, 2019.

[86] Nicole Boutros-Saikali, Karim Saikali, and Rodrigue Abou Naoum. An iomt platform to simplify the development of healthcare monitoring applications. In *2018 Third International Conference on Electrical and Biomedical Engineering, Clean Energy and Green Computing (EBECEGC)*, pages 6–11, 2018. doi: 10.1109/EBECEGC.2018.8357124.

[87] Fernanda Famá, José N. Faria, and David Portugal. An iot-based interoperable architecture for wireless biomonitoring of patients with sensor patches. *Internet of Things*, 19:100547, 2022. ISSN 2542-6605. doi: https://doi.org/10.1016/j.iot.2022.100547. URL `https://www.sciencedirect.com/science/article/pii/S2542660522000488`.

[88] Alf Helge Omre and Steven Keeping. Bluetooth low energy: wireless connectivity for medical monitoring. *Journal of diabetes science and technology*, 4(2):457–463, 2010.

[89] Jose Nuno da Cruz Faria. Wireless iot smart bed system. Master's dissertation, Universidade de Coimbra, 2022.

[90] Tina Wu and Andrew Martin. Bluetooth low energy used for memory acquisition from smart health care devices. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 1256–1261, 2018. doi: 10.1109/TrustCom/BigDataSE.2018.00173.

[91] Bernardo Henriques Plácido Fernandes. Smart bed iot-based wireless data acquisition for untethered patients. Master's dissertation, Universidade de Coimbra, 2022.

[92] MongoDB. Internet of things applications. URL `https://www.mongodb.com/scale/internet-of-things-applications`.

[93] Christodoulos Asiminidis, George Kokkonis, and S. Kontogiannis. Database systems performance evaluation for iot applications. *SSRN Electronic Journal*, 10, 01 2018. doi: 10.2139/ssrn.3360886.

[94] Daryna Kacherovska. How secure is the ble communication standard? - dzone security, Aug 2019. URL `https://dzone.com/articles/how-secure-is-the-ble-communication-standard`.

[95] Kai Ren. Bluetooth pairing part 5:legacy pairing - out of band, Dec 2020. URL `https://www.bluetooth.com/blog/bluetooth-pairing-part-5-legacy-pairing-out-of-band/`.

[96] Tom Spring. Bluetooth hack leaves many smart locks, iot devices vulnerable, Aug 2016. URL `https://threatpost.com/bluetooth-hack-leaves-many-smart-locks-iot-devices-vulnerable/119825/`.

[97] IBM Corporation, Aug 2021. URL `https://www.ibm.com/docs/en/ibm-mq/7.5?topic=m2m-mqtt-security`.

[98] Documentation, Jul 2020. URL https://mosquitto.org/documentation/.

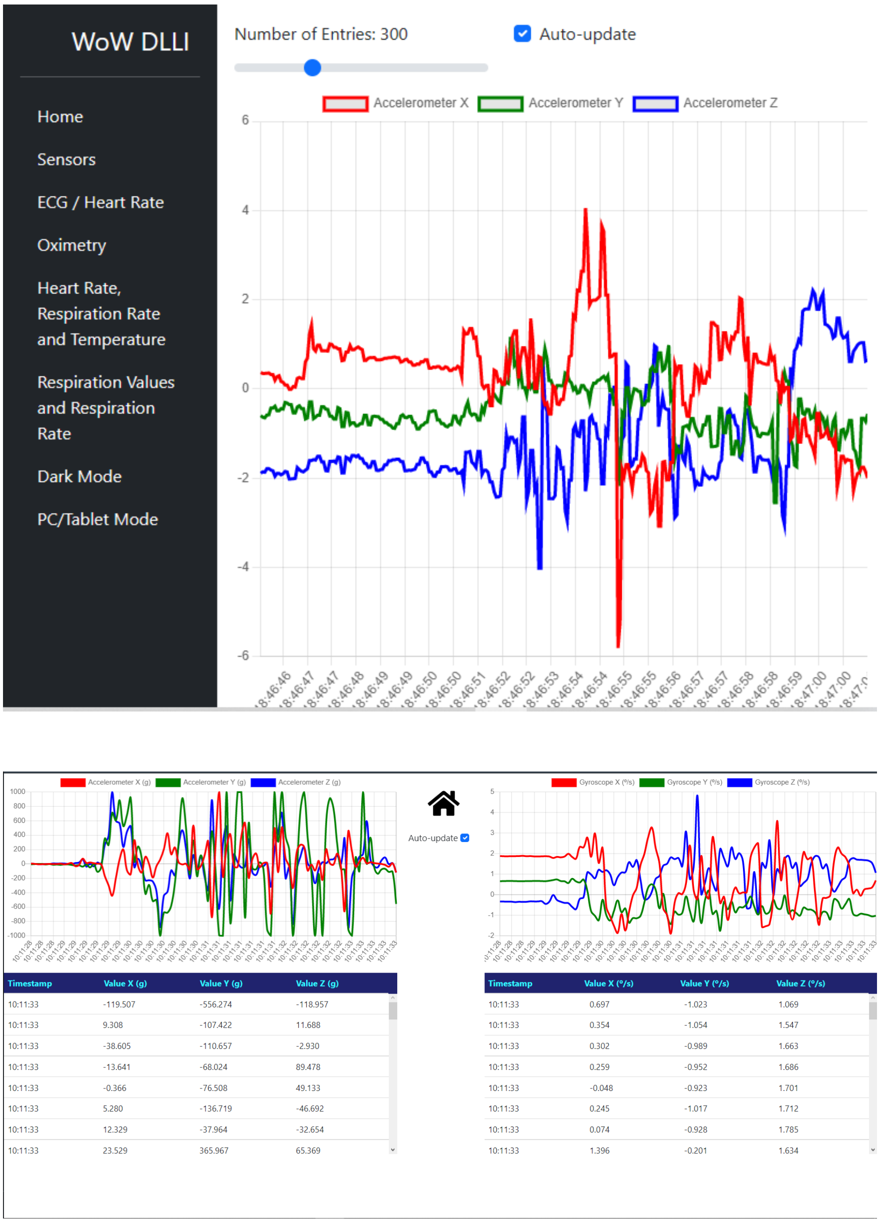
[99] Lokesh Gupta. Rest api security essentials, Sep 2021. URL https://restfulapi.net/security-essentials/.

[100] AWS. The ultimate iot security best practices guide - amazon web services, inc.. URL https://pages.awscloud.com/rs/112-TZM-766/images/IoT_Security_Best_Practices_Guide_design_v3.1.pdf.

[101] Nasko Oskov. Netsekure rng, Mar 2010. URL http://netsekure.org/2010/03/tls-overhead/.

Appendix A

GUI version 1.0 vs version 2.0



Timestamp	Value X (g)	Value Y (g)	Value Z (g)
10:11:33	-119.507	-556.274	-118.957
10:11:33	9.308	-107.422	11.688
10:11:33	-38.605	-110.657	-2.930
10:11:33	-13.641	-68.024	89.478
10:11:33	-0.366	-76.508	49.133
10:11:33	5.280	-136.719	-46.692
10:11:33	12.329	-37.964	-32.654
10:11:33	23.529	365.967	65.369

Timestamp	Value X (°/s)	Value Y (°/s)	Value Z (°/s)
10:11:33	0.697	-1.023	1.069
10:11:33	0.354	-1.054	1.547
10:11:33	0.302	-0.989	1.663
10:11:33	0.259	-0.952	1.686
10:11:33	-0.048	-0.923	1.701
10:11:33	0.245	-1.017	1.712
10:11:33	0.074	-0.928	1.785
10:11:33	1.396	-0.201	1.634

Figure A.1: IMU's page of version 1.0 (top) vs IMU's page of version 2.0 (bottom).

Appendix B

MQTT Message Types

"MEASUREMENT_TEMPERATURE", "MEASUREMENT_ECG", "MEASUREMENT_PULSEOXIMETRY", "MEASUREMENT_RESPIRATION", "MEASUREMENT_IMU", and "MEASUREMENT_HEARTRATE" for each of the measurements or:

- "SYNC_REQ", "SYNC_REP", "SYNC_REP_END": Indicating that the message corresponds to the data syncing feature;
- "STATUS_REQ", "STATUS_REP": Indicating that the message corresponds to the keep alive feature;
- "WILL_BRD" : Indicating that the message corresponds to the last will and testament feature;
- "PAIR_REQ", "PAIR_REP" : Indication that the message corresponds to the pairing feature;

Appendix C

MQTT Payload Formats

- “MEASUREMENT_TEMPERATURE”: The payload format is:

```
    "payload": {
        "temperature": 10.0,
        "is_celsius": true
    }
```

  where the “temperature” field is the temperature measurement, and “is_celsius” field indicates whether the measurement value is in Celsius or Fahrenheit.

- “MEASUREMENT_IMU”: The payload format is:

```
    "payload": {
        "imu": {
            "linear_acceleration": {"x": 0.00, "y": 0.00,  "z": 0.00},
            "angular_velocity": {"x": 0.00, "y": 0.00, "z": 0.00}
    }
        "pose_description": "SITTING"
    }
```

  where the “linear_acceleration” field is the IMU accelerometer measurement, “angular_velocity” field is the IMU gyroscope measurement, and “pose_description” is the text description of the current body pose of the patient.

- “MEASUREMENT_ECG”: The payload format is:

```
    "payload": {
        "ecg": 10
    }
```

  where the “ecg” field is the ECG measurement.

- “MEASUREMENT_PULSEOXIMETRY”: The payload format is:

```
    "payload": {
        "spo2": 10.0
    }
```

where the “spo2” field is the pulse oximetry measurement.

- “MEASUREMENT_HR”: The payload format is:

```
"payload": {
    "bpm": 10.0
}
```

where the “bpm” field is the heart rate measurement.

- “MEASUREMENT_RESPIRATION”: The payload format is:

```
"payload": {
    "respiration": 10.0
}
```

where the “respiration” field is the respiration rate computed.

- “SYNC_REQ”: The payload format is:

```
"payload": {
    "last_message_timestamp": 16148840000
}
```

where the “last_message_timestamp” field is the timestamp of the last message sent by this Smart box to the Gateway.

- “SYNC_REP”: The payload format is:

```
"payload": {
    "message_list":[ // List of messages in backlog
        {
            "timestamp": 16148840000,
            "message_type": "MEASUREMENT_TEMPERATURE",
            "payload": {
                "temperature": 10.3
            }
        },
        {
            "timestamp": 16148840001,
            "message_type": "MEASUREMENT_ECG",
            "payload": {
                "ecg": 10.4
            }
        },
    ]
}
```

where the “message_list” field is a list of the messages in backlog.

- “SYNC_REP_END”: The payload format is:

```
"payload": ''
```

- “STATUS_REQ”: The payload format is:

```
"payload": ''
```

- “STATUS_REP”: The payload format is:

```
"payload": {
    "connection_start_timestamp": 1644693581
}
```

  where the “connection_start_timestamp” field is the timestamp of the start of the connection.

- “WILL_BRD”: The payload format is:

```
"payload": ''
```

- “PAIR_REQ”: The payload format is:

```
"payload": {
    "mac_addr": "00:00:5e:00:53:af"
}
```

  where the “mac_addr” field is the MAC address of the Biosticker to be paired with this Smart box.

- “PAIR_REQ”: The payload format is:

```
"payload": ''
```

Appendix D

MQTT Endpoints

The several MQTT endpoints depending on the message type can be seen at the table :

Message Type	**MQTT Endpoint**
MEASUREMENT_TEMPERATURE	smartbox/client_UUID/temperature
MEASUREMENT_IMU	smartbox/client_UUID/imu
MEASUREMENT_ECG	smartbox/client_UUID/ecg
MEASUREMENT_PULSEOXIMETRY	smartbox/client_UUID/pulseoximetry
MEASUREMENT_HEARTRATE	smartbox/client_UUID/heartrate
MEASUREMENT_RESPIRATION	smartbox/client_UUID/respiration
WILL_BRD	smartbox/client_UUID/lwt
STATUS_REQ	status
STATUS_REP	status/client_UUID
SYNC_REQ	smartbox/client_UUID/sync
SYNC_REP	smartbox/client_UUID/sync/response
SYNC_REQ_END	smartbox/client_UUID/sync/response
PAIR_REQ	smartbox/client_UUID/pair
PAIR_REP	smartbox/client_UUID/pair/response

Table D.1: MQTT endpoints.

Appendix E

FHIR Specification

FHIR SPECIFICATION

Version 1.5

July 16, 2022

Author: Marco Domingues

Contents

1 Vital Signs Specification

This chapter of the document serves as a specification for the format of the vital signs measurement, which are:

- Temperature;
- Heart Rate;
- Respiratory Rate;
- Oxygen Saturation;

The sensor measurements are transmitted by the Gateway to the Hospital Information System (HIS) in transaction-type Bundles, where each Bundle must contain the Observation resource with the measurement in question, the Device resource of the sensor that took the measurement, and the Device resource of the associated Smart box.

In order to respect the official FHIR documentation[1], the measurement Bundle must follow a specific structure, detailed in the Table 1.1.

A flowchart of the vital signs measurements can be seen in the Figure 1.1.

Furthermore, the observation resources also follow a specific structure that only varies for each vital sign and can be seen at Table 1.2.

[1] https://www.hl7.org/fhir/

ResourceType	Bundle
Bundle.id	Transaction ID
Bundle.type	”Transaction”
Bundle.timestamp	Message Date Time
Bundle.entry[0].fullUrl	**Sensor ID**
Bundle.entry[0].resource	**Sensor Device Resource**
Bundle.entry[0].request.method	”POST”
Bundle.entry[0].request.url	”Device”
Bundle.entry[1].fullUrl	**Smart box ID**
Bundle.entry[1].resource	**Smart box Device Resource**
Bundle.entry[1].request.method	”POST”
Bundle.entry[1].request.url	”Device”
Bundle.entry[2].fullUrl	**Measurement ID**
Bundle.entry[2].resource	**Measurement Observation Resource**
Bundle.entry[2].request.method	”POST”
Bundle.entry[2].request.url	”Observation”

Table 1.1: Measurement Structure. The values indicated in bold vary for each vital sign.

ResourceType	Observation
Observation.meta.profile	FHIR profile associated with the measurement
Observation.status	”Final”
Observation.category.coding.system	”observation-category”
Observation.category.coding.code	”vital-signs”
Observation.category.coding.display	”Vital Signs”
Observation.code.coding.system	**Measurement Code**
Observation.code.coding.code	
Observation.code.coding.display	
Observation.code.text	
Observation.bodySite.coding.system	**Measurement Body Location**
Observation.bodySite.coding.code	
Observation.bodySite.coding.display	
Observation.bodySite.coding.text	
observation.effectiveInstant	**Measurement Instant**
Observation.valueQuantity.value	**Measurement Value**
Observation.valueQuantity.unit	**Measurement Unit**
Observation.valueQuantity.system	
Observation.valueQuantity.code	

Table 1.2: Observation Structure. The values indicated in bold vary for each vital sign.

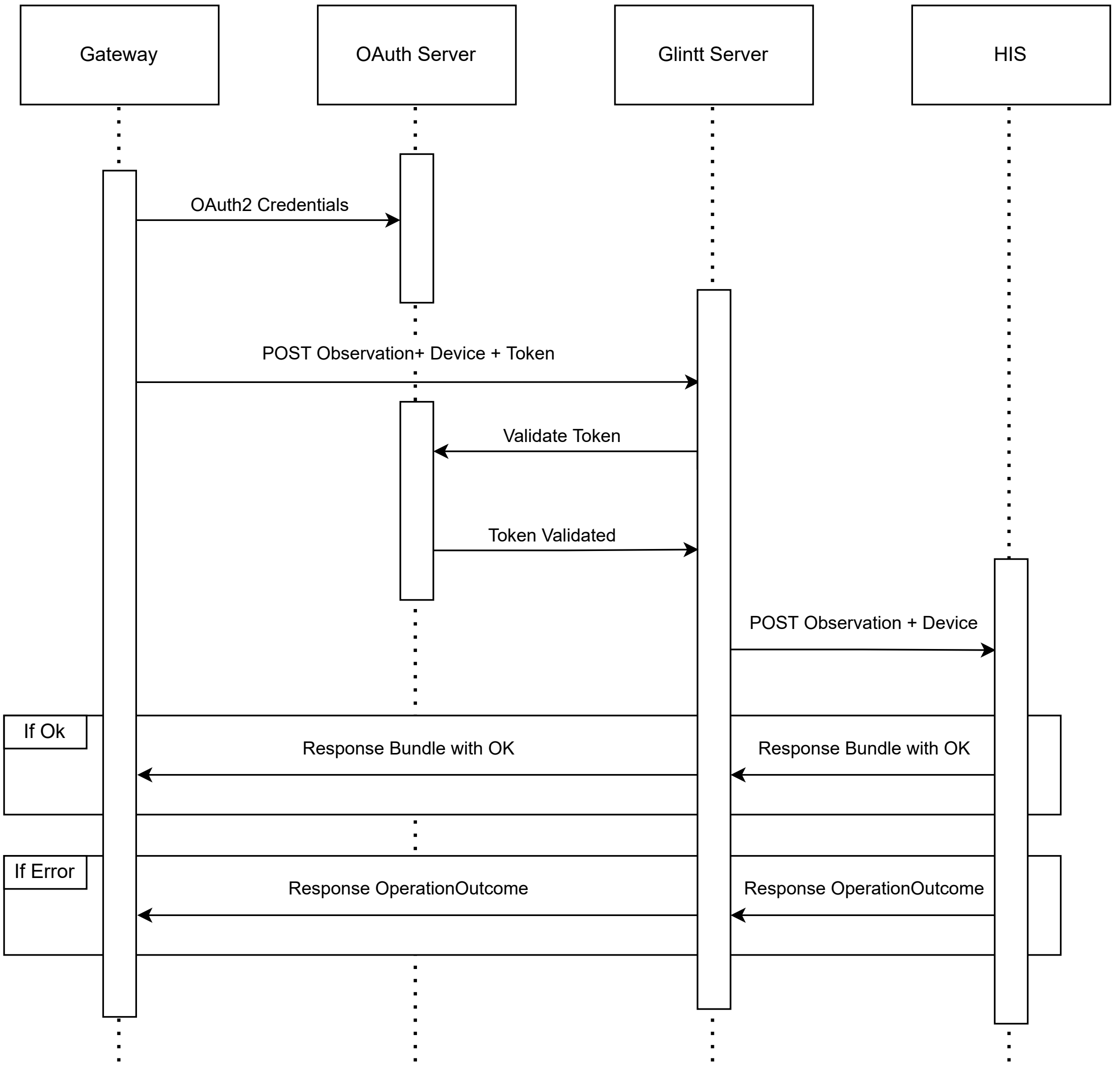


Figure 1.1: Flowchart of the vital signs measurements sending.

Example of a FHIR resource of a temperature measurement:

```
–
  ”resourceType”: ”Bundle”,
  ”type”: ”transaction”,
  ”entry”: [
    –
      ”resource”: –
        ”resourceType”: ”Observation”,
        ”status”: ”final”,
        ”category”: [
          –
            ”coding”: [
              –
                "system": "http://terminology.hl7.org/CodeSystem/observation-category",
                "code": "vital-signs",
                "display": "Vital Signs"
              ˝
            ]
          ˝
        ],
        "code": –
          "coding": [
            –
              "system": "http://loinc.org",
              "code": "8310-5",
              "display": "Body temperature"
            ˝
          ]
        ˝,
        "effectiveInstant": "2022-03-02T00:59:25.905+00:00",
        "valueQuantity": –
          "value": 40.630001068115234,
          "unit": "C",
          "system": "http://unitsofmeasure.org",
          "code": "Cel"
        ˝,
        "bodySite": –
          "coding": [
            –
              "system": "http://snomed.info/sct",
              "code": "728678006",
              "display": "Entire surface region of upper chest"
            ˝
          ],
          "text": "Entire surface region of upper chest"
        ˝,
        "device": –
          "reference": "urn:uuid:2bed2e0b-a3d7-4278-a7d2-383cdd3b16ff"
        ˝
      ˝,
      "request": –
        "method": "POST",
        "url": "Observation"
      ˝
    ˝,
    –
      "fullUrl": "urn:uuid:2bed2e0b-a3d7-4278-a7d2-383cdd3b16ff",
      "resource": –
        "resourceType": "Device",
        "identifier": [
          –
            "system": "urn:ietf:rfc:3986",
            "value": "urn:uuid:2bed2e0b-a3d7-4278-a7d2-383cdd3b16ff"
          ˝
        ],
        "status": "active",
        "type": –
          "coding": [
            –
              "system": "http://snomed.info/sct",
              "code": "27991004",
              "display": "Thermometer"
            ˝
          ],
          "text": "Term metro"
        ˝,
        "parent": –
          "reference": "urn:uuid:1ef8a769-7221-48df-a8db-8bc9079996ef"
        ˝
      ˝,
      "request": –
        "method": "POST",
        "url": "Device"
      ˝
    ˝,
    –
      "fullUrl": "urn:uuid:1ef8a769-7221-48df-a8db-8bc9079996ef",
      "resource": –
        "resourceType": "Device",
        "identifier": [
          –
            "system": "urn:ietf:rfc:3986",
            "value": "urn:uuid:1ef8a769-7221-48df-a8db-8bc9079996ef"
          ˝
        ],
        "status": "active",
        "type": –
          "coding": [
            –
              "system": "http://snomed.info/sct",
              "code": "5159002",
              "display": "Physiologic monitoring system"
            ˝
          ],
          "text": "Smartbox"
        ˝
      ˝,
      "request": –
        "method": "POST",
        "url": "Device"
      ˝
    ˝
  ]
```

2 Devices Specification

This chapter of the document serves as a specification for device creation and deactivation notifications.

The information about the devices is sent by the Gateway in Bundles resources containing the Device resource of the sensor and also the Device resource of the associated Smart box to the HIS.

When a new device is created, the resource status must be “active“ and when a device is deactivated, the resource status must be “inactive“.

The Device resource must follow a structure detailed in the Table 2.1.

A flowchart of the creation of a new device can be seen in the Figure 2.1.

ResourceType	Device
Device.identifier.system	**Sensor ID**
Device.identifier.value	
Device.type.coding.system	**Sensor Type**
Device.type.coding.code	
Device.type.coding.display	
Device.type.text	
Device.parent.reference (only if sensor)	**Associated Smart box ID**

Table 2.1: Device structure. The values indicated in bold vary for each device (Sensor or Smart box).

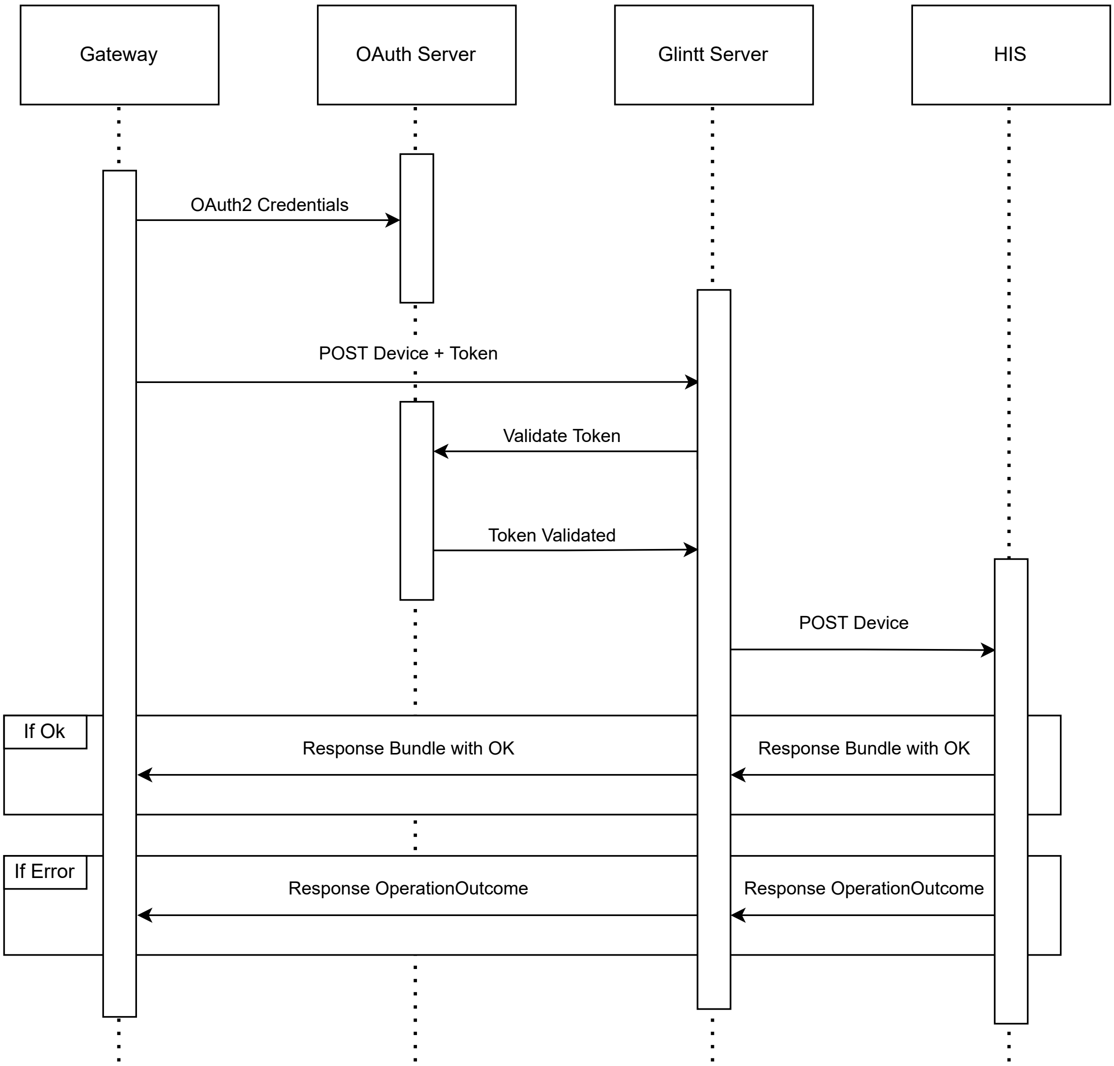


Figure 2.1: Flowchart of the creation of a new device.

When deactivating a device the ”PUT” method must be used instead of the ”POST” method.

Example of a FHIR resource of a notification of a device deactivation:

```
–
    ”resourceType”: ”Bundle”,
    ”type”: ”transaction”,
    ”entry”: [ –
      ”fullUrl”: ”Device/189bc38a-8a0f-460e-9129-36537d8cfd23”,
      ”resource”: –
        ”resourceType”: ”Device”,
        ”id”: ”189bc38a-8a0f-460e-9129-36537d8cfd23”,
        ”identifier”: [ –
        "system": "urn:ietf:rfc:3986",
        "value": "urn:uuid:189bc38a-8a0f-460e-9129-36537d8cfd23"
      ˝ ],
      "status": "inactive",
      "type": –
        "coding": [ –
          "system": "http://snomed.info/sct",
          "code": "701914001",
          "display": "Emergency heart rate monitor"
        ˝ ],
        "text": "Monitor de Emerg ncia da Frequ ncia Card aca"
      ˝,
      "parent": –
        "reference": "urn:uuid:1ef8a769-7221-48df-a8db-8bc9079996ef"
      ˝
    ˝,
    "request": –
      "method": "PUT",
      "url": "Device/189bc38a-8a0f-460e-9129-36537d8cfd23"
    ˝
  ˝, –
    "fullUrl": "urn:uuid:1ef8a769-7221-48df-a8db-8bc9079996ef",
    "resource": –
      "resourceType": "Device",
      "identifier": [ –
        "system": "urn:ietf:rfc:3986",
        "value": "urn:uuid:1ef8a769-7221-48df-a8db-8bc9079996ef"
      ˝ ],
      "status": "active",
      "type": –
        "coding": [ –
          "system": "http://snomed.info/sct",
          "code": "5159002",
          "display": "Physiologic monitoring system"
        ˝ ],
        "text": "Smartbox"
      ˝
    ˝,
    "request": –
      "method": "POST",
      "url": "Device"
    ˝
  ˝ ]
˝
```

3 Frequency Specification

This chapter of the document serves as a specification for notification of subscription request creation, update, and deactivation.

This information is transmitted by the HIS via Bundles resources, sending the ServiceRequest resource with the request to be made, specifying the frequency with which this request should be fulfilled.

The healthcare professional should select a device and determine how often the gateway should transmit these measurements to the HIS (e.g., every 15 minutes).

If the subscription was successfully created, the Gateway should return a Bundle of type transaction-response with the information that the request was created (response.status = '201 Created') and the ServiceRequest with status = 'active'.

A flowchart of the creation of a new subscription can be seen in the Figure 3.1.

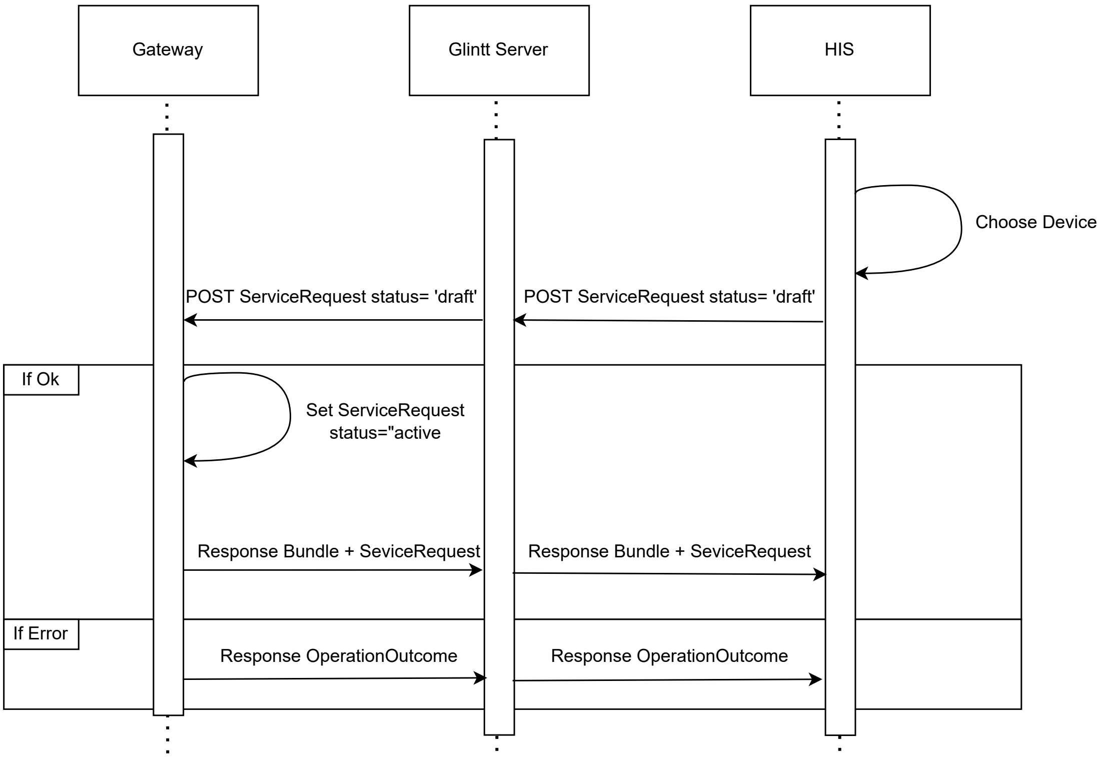

Gateway
Glintt Server
HIS
Choose Device
POST ServiceRequest status= 'draft'
POST ServiceRequest status= 'draft'
If Ok
Set ServiceRequest status="active
Response Bundle + SeviceRequest
Response Bundle + SeviceRequest
If Error
Response OperationOutcome
Response OperationOutcome

Figure 3.1: Flowchart of the creation of a new subscription.

To update or deactivate a subscription, the HIS must send the resource to the Gateway

ResourceType	ServiceRequest
ServiceRequest.identifier.system	Request ID
ServiceRequest.identifier.value	
ServiceRequest.status	Request Status
ServiceRequest.intent	Request Intent
ServiceRequest.code.coding.system	What is being requested
ServiceRequest.code.coding.code	
ServiceRequest.code.coding.display	
ServiceRequest.code.text	
ServiceRequest.subject	Requested Device
ServiceRequest.ocurrenceTiming	When the service should occur
ServiceRequest.authoredOn	Request Date
ServiceRequest.note	Notes

Table 3.1: Subscription Request structure.

using the PATCH method, which contains a ServiceRequest with the fields to be changed. When updating the subscription, the status must be ”active” and when deactivating, the status must be ”completed”.

If the request is successfully updated or deactivated, the Gateway should return a Bundle of type transaction-response with the information that the transaction was successful (response.status = ’204 No Content’) and the corresponding ServiceRequest. If an error occurs, an OperationOutcome should be sent by the Gateway.

The ServiceRequest resource must follow the structure detailed in the Table 3.1.

Example of a FHIR resource of a notification of a subscription creation:

```
–
    "resourceType": "Bundle",
    "type": "transaction",
    "entry": [
        –
            "fullUrl": "5cce01ff-1134-402d-8d3f-cb360885478",
            "resource": –
                "resourceType": "ServiceRequest",
                "identifier": [
                    –
                        "system": "urn:ietf:rfc:3986",
                        "value": "5cce01ff-1134-402d-8d3f-
                           cb360885478"
                    ˝
                ],
                "status": "draft",
                "intent": "order",
                "type": –
                    "coding": [
                        –
```

```
                            "system": "http://snomed.info/sct",
                            "code": "410188000",
                            "display": "Taking patient vital signs
                                assessment"
                        ˝
                    ],
                    "text": "Avaliar sinais vitais"
                ˝,
                "subject": –
                    "reference": "Device/ urn:uuid:61ebe359-bfdc
                       -4613-8bf2-c5e300945f0a"
                ˝,
                "occurrenceTiming": –
                    "repeat": –
                        "frequency": 1,
                        "period": 30,
                        "periodUnit": "min"
                    ˝,
                    "code": –
                        "system": "https://www.rfc-editor.org/rfc/
                           rfc5545.html",
                        "code": "RRULE:FREQ=MINUTELY;INTERVAL=30"
                    ˝
                ˝,
                "authoredOn": "2022-03-02T10:00:07+00:00",
                "note": [
                    –
                        "text": "example"
                    ˝
                ]
            ˝,
            "request": –
                "method": "POST",
                "url": "ServiceRequest"
            ˝
        ˝
    ]
˝
```

Example of a FHIR resource of a transaction response:

```
–
    "resourceType": "Bundle",
    "type": "transaction-response",
    "entry": [
        –
            "fullUrl": "17d2d24e-3e53-4b74-a60d-edc99061c081",
            "resource": –
                "resourceType": "ServiceRequest",
                "id": "17d2d24e-3e53-4b74-a60d-edc99061c081",
                ”identifier”: [
                    –
                        ”system”: ”urn:ietf:rfc:3986”,
                        ”value”: ”17d2d24e-3e53-4b74-a60d-edc99061c081”
                    ˝
                ],
                ”status”: ”active”,
                ”intent”: ”order”,
                ”code”: –
                    ”coding”: [
                        –
                            ”system”: ”http://snomed.info/sct”,
                            ”code”: ”410188000”,
                            ”display”: ”Taking patient vital signs assessment”
                        ˝
                    ],
                    ”text”: ”Avaliar sinais vitais”
                ˝,
                ”subject”: –
                    ”reference”: ”Device/ urn:uuid:1ef8a769-7221-48df-a8db-8bc9079996ef”
                ˝,
                ”occurrenceTiming”: –
                    ”repeat”: –
                        ”frequency”: 1,
                        ”period”: 50,
                        ”periodUnit”: ”s”
                    ˝
                ˝,
                ”authoredOn”: ”2022-03-02T10:00:07+00:00”,
                ”note”: [
                    –
                        ”text”: ”example”
                    ˝
                ]
            ˝,
            ”response”: –
                ”status”: ”201 Created”
            ˝
        ˝
    ]
˝
```

Example of a FHIR resource of a OperationOutcome:

```
–
    ”resourceType”: ”OperationOutcome”,
    ”id”: ”5cce01ff-1134-402d-8d3f-cb360883595c”,
    ”issue”: [
        –
            ”severity”: ”error”,
            ”code”: ”invalid”,
            ”details”: –
                ”text”: ”Invalid Syntax”
            ˝,
            ”diagnostics”: ”[ServiceRequest/5cce01ff-1134-402d-8d3f-cb360885478] An error occurred.
        ˝
    ]
˝
```