INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Alternative Architecture Approaches for Distributed Control of Smart Buildings

Débora Eulália Manhique Cuco

Master in Telecommunications and Computer Engineering

Supervisor:
PhD Rui Miguel Neto Marinheiro, Associate Professor
Iscte- Instituto Universitário de Lisboa

October, 2023

TECNOLOGIAS
E ARQUITETURA

Department of Information Science and Technology

Alternative Architecture Approaches for Distributed Control of Smart Buildings

Débora Eulália Manhique Cuco

Master in Telecommunications and Computer Engineering

Supervisor:
PhD Rui Miguel Neto Marinheiro, Associate Professor
Iscte- Instituto Universitário de Lisboa

October, 2023

Acknowledgment

I would like to express my thanks to those who made this thesis possible in some way.

First, I sincerely thank my professor and supervisor Rui Marinheiro for his guidance and support during the thesis.

I would like to acknowledge Instituto de Telecomunicações, IT-IUL for the support and providing all necessary material for this project.

I thank my family, parents, sister, and brothers for their unconditional support and motivation during this arduous process.

Finally, I want to thank my friends and colleagues at the university for their constant encouragement and support.

To all, my sincere thanks.

Resumo

Com a crescente distribuição e densidade de dispositivos em comunidade ou edifício inteligente, fatores como fiabilidade e a integridade são postos à prova nos processos de gestão de recursos e de plataformas de automação, quando se pretende delegar o controlo de dispositivos. Por esta razão, existe a necessidade de uma arquitetura que possa responder a tais desafios.

Este trabalho tem como objetivo estudar diferentes abordagens para controlo distribuído em ambientes IoT (internet das coisas) e as respetivas arquiteturas, e pretende ainda analisar como estas abordagens podem ser instaladas e executadas no contexto do controlo distribuído em plataformas de gestão de edifícios inteligentes.

O principal objetivo desta dissertação é conceber e comparar arquiteturas alternativas para integração de plataformas de automação de edifícios que prevejam a gestão distribuída de recursos.

Para atingir o objetivo desta dissertação, foram propostas quatro alternativas de integração que combinam duas abordagens de comunicação, MQTT e Rest API, com duas colocações de serviços de comunicação, em servidores de borda e nuvem.

De acordo com os testes, todas as alternativas alcançam a integração desejada, sendo que o melhor resultado de tempo de resposta foi de 33,49ms na comunicação via Rest API dos serviços de borda. No entanto, a integração via Rest API deixa expostos itens que poderiam ser controlados de forma indesejada e, como tal, é menos segura. Este não é o caso quando a comunicação via MQTT apresenta um tempo de resposta aceitável de pouco mais de 100ms, quer utilizando servidores de borda ou de nuvem.

Palavras-chave: Internet das coisas, domótica, controle distribuído.

Abstract

With the increasing distribution and density of devices in smart communities or buildings, factors like reliability and integrity are put to the test in the resource and management processes of automation platforms, when delegation of device control is intended. For this reason, there is a need for an architecture that can address such challenges.

This work aims to study different approaches for distributed control in IoT (Internet of Things) environments and their architectures, and it also intends to analyze how these approaches can be installed and performed in the context of distributed control in smart building management platforms.

The main goal of this dissertation is to design and compare alternative architectures for the integration of building automation platforms envisaging the distributed management of resources.

To achieve the aim of this dissertation, four integration alternatives have been proposed combining two approaches for communication, MQTT and Rest API, with two placements of communications services, in edge and cloud servers.

According to tests, all alternatives achieve the desired integration, and the best response time result was 33.49ms when communicating via Rest API edge services. However, the integration via Rest API left exposed items that could be undesirably controlled and, as such, it is less secure. However, this is not the case when communication via MQTT, which presents an acceptable response time of slightly over 100ms, whether using edge or cloud servers.

Keywords: Internet of Things, home automation, distributed control.

Table of Contents

List of tables

List of Figures

Abbreviations

API – Application Programming Interface

HTTP – Hypertext Transfer Protocol

ICT – Information and Communication Technology

IoT - Internet of Things

ISO – International Organization for Standardization

IEC – International Electrotechnical Commission

JSON – JavaScript Object Notation

P2P - Peer to Peer

MQTT – Message Queuing Telemetry Transport

openHAB – Open Home Automation Bus

QoS – Quality of Service

REST – Representational State Transfer

VMs – Virtual machines

Glossary

Bindings – function as small devices or libraries manage the devices and functionality.

Channel – they are the software connection between Things and Items.

Entity – Participants of the system solution.

Item – Group of things that generate the functionalities.

Things – entities that manage devices (devices management) everything devices is configured in “things”.

Rules - that allow you to automate the system.

Sitemap – is the user interface generated by OpenHab.

ZigBee – is a wireless communication data transfer protocol that has been widely used in IoT devices.

Zwave – is a wireless communication and data transfer protocol, used primarily for residential building automation.

Wi-fi – is a wireless network technology that allows computers, or mobile devices to connect to the internet.

Bluetooth – is a wireless for exchanging data over short distance, from fixed and mobile devices.

Chapter 1. Introduction

1.1. Contextualization

The use of the internet of things (IoT) has proved to be revolutionary due to the impact generated in each area or industry that is used, presenting as one of the characteristics the remote control and automation of devices and processes that give rise to intelligent life, smart factories, smart vehicles, smart homes, smart production, smart security, and all other possible areas of application.

The Internet of Things technology promises to be useful for people with disabilities and the elderly, allowing for improved levels of independence quality of life at reasonable cost [1]. Internet of Things systems such as networked vehicles, smart traffic systems, and sensors embedded in roads and bridges bring us closer to the idea of “smart cities”, which help reduce congestion and energy consumption.

The energy management platform is an excellent example of using the Internet of Things to monitor and manage energy. This solution transforms the way people and businesses use and control electricity, electrical appliances, loads, and energy storage. It consists of intelligent hardware, software, and data tools. Once equipped with this system, a home, office building, or any other facility becomes a smart space with a rich set of features and capabilities to monitor and control energy consumption. Among them, the optimized installation and performance of energy storage and solar panels, smart homes, and smart buildings.

Families and smart building residents get multiple benefits from using the customer's energy monitoring and management system. In addition to responsive control over electricity distribution, consumption, load, and cost, the system provides tools for safer, more environmentally friendly, and less wasted energy use. Real-time data processing for electricity consumption for the entire house and separate circuits, energy input from solar panels, and energy stored in storage.

The present dissertation contributes to this direction and presents different protocols, i.e., different communication approaches as well as different locations for communication proxies that will manage the interconnections and will test the best architecture for a distributed control of buildings that can be located within a condominium or a resort.

1.2. Definition of Problem

Remote communities or smart buildings have a range of services growing over time. Services such as energy consumption control, food supply, water supply, newspaper supply, and service provision, cleaning, and communications mechanisms require a system capable of integrating all devices that are part of these communities to a single management window. On other hand, the system should allow the delegation of the management to entities that provide services in a particular area in order to optimize the use of services, make the services more flexible, and access the data inherent to activities in real time with a certain level of reliability, immutability, and availability.

Unfortunately, with the growing number of smart buildings that use the conventional structure of access and management of devices, it has not been easy to implement the delegation of management functions to third parties due to the centralized and autonomous control of their resources.

1.3. Motivation

Nowadays, homes are the most prevalent areas of the IoT. Smart buildings can control lights, temperature, humidity, and other devices. In this context, the opportunity arises for solutions based on the distributed control of smart buildings using integration platforms such as openHAB[1].

1.4. Background

The number of technologies and solutions for smart buildings has progressed significantly, and their adoption has grown exponentially in recent years. Traditionally, the management and control of the various functions have been centralized. However, there has been a need for these functions to be increasingly delegated to third parties. Some examples are the distributed management of electricity production and consumption in self-sustaining communities or alarm management in the event of fire or security breaches.

Nevertheless, delegating management functions has not been easy, as each building generally has autonomous and centralized control.

[1] https://www.openhab.org/

1.5. General Objective

The main objective of this dissertation is to design and compare alternative architectures for the integration of building automation platforms, aimed at the distributed management of resources.

1.6. Research Questions

The main question that supports the development of this dissertation are:

1. How decentralization can be implemented?
2. Which interconnection method is more effective and will have better performance results, regarding location and protocol to use?
3. What risks do distributed systems offer to the community of smart building by allowing the decentralization of control to provide services by third parties?

1.7. Research Method

The development of this dissertation will follow the research method of Design Science Research (figure 1) and this method consist in six distinct activities.

This model, specifically organized for technological areas, has been structured to guide and assist scientific research based on following a set of various essential steps for a successful investigation [2].

- **Problem Identification:** The problem definition will be used to develop an artefact that can provide the solution. The problem detected was to identify whether it is possible to have distributed building control using automation platforms installed in a building.
- **Define objectives of solution:** Once the problem has been identified, existing solutions, or presenting a new one solution [2]. The main goal of this dissertation is to develop main a design a conceptual model of Internet of Things solution architecture that uses a secret key for OpenHab, cloud in an integrated way in smart building management platforms.
- **Design and Development:** Includes determining the artifacts desired functionality and architecture [2]. Design a conceptual model of IoT solution architecture in smart building.

- **Demonstration:** Demonstrate the use of the artefact to solve one or more instances of the problem, this could involve its simulation or use case [2]. The implementation of the system will be tested in a building, where the light is switched on and switched off in a room of a smart building.
- **Evaluation:** Observe and measure how well the artefact supports a solution to the problem. Evaluation can take many forms: a comparison of the artefact's functionality with the solution objectives, quantitative performance measure such as the results of satisfaction, feedback, or simulations [2]. After evaluating the coherence between the achieved objectives of system, through the analysis of the results of the reanalyzed implementation, it can be concluded about the efficiency and security of the system.
- **Communication:** Communicate the problem and its importance. Communicate the artefact, its utility, and novelty, the rigor of the design, and its effectiveness to researchers and other relevant audiences, such as practicing professionals, when appropriate [2]. The last one will be the results obtained in the dissertation.

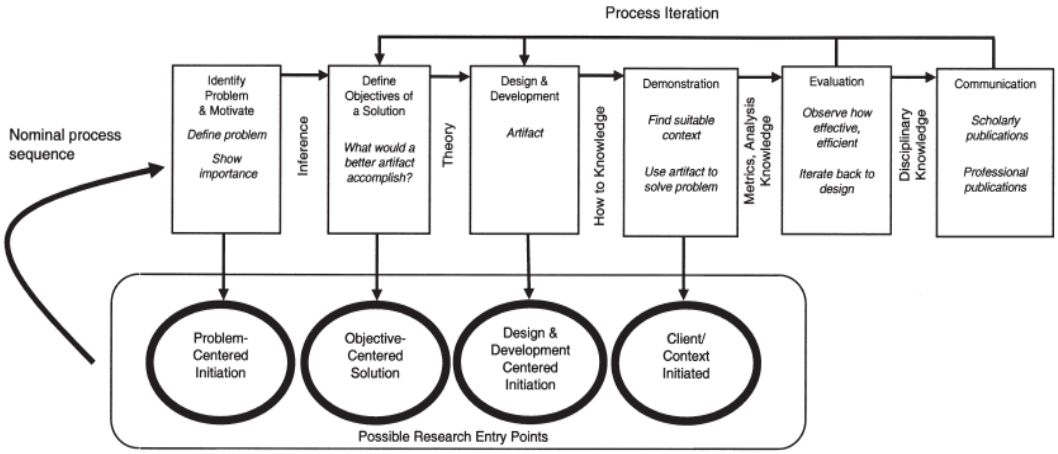

Figure 1. Design science research methodology [2]

1.8. Structure and organization of dissertation

The dissertation is organized in five chapters. Chapter 1 is about introduction, followed by a literature review, then the development of the architecture and the implementation and, finally the conclusion.

- Chapter 1 introduces the subject of investigation and the objectives, as well as a brief description of the structure of the project.
- Chapter 2 provides an overview of the essential literature that allow to have more knowledge, to better understand how the architecture will be developed. This chapter

is divided in 3 sections. In the first section, a brief introduction about the Internet of Things and smart building. In the second section, studies which IoT integration platform will be used to develop the project and, finally in third section studies which protocols used to interconnect home automation platforms.

- Chapter 3 describes the architecture that offers various alternatives; and described how the architecture functioning.
- Chapter 4 describes the implementation of the system, to demonstrate the validity of the architecture and answer the objective and research questions, a demonstrator or prototype was implemented that integrates multiple services and how they interact.
- Chapter 5 describe results and test and presents the result it's obtained through integration with local servers.
- Chapter 6 presents the conclusion and contribution of the study.

Chapter 2. Literature Review

Considering the complexity of the Internet of Things and to know the best architecture for the distributed control of smart buildings, this works tried to point out references that address applications focused on the building's domain. The diagnosis of new applications made possible by advances in Information and communication technologies research, and especially in the IoT area, allows us to understand that the concept of a smart building has also evolved over time. Currently, there are numerous definitions in the literature. This chapter covers the state of art of IoT, IoT integration platform and protocols used to interconnect platforms.

2.1. Internet of Things - IoT

The basic premise and goal of IoT is to connect and unconnected, this means that objects that are not currently joined to a computer network, namely the Internet, will be connected so that they communicate and interact with people and other objects.

The term Internet of Things was first mentioned by Kevin Ashton, one of the founders of MIT's Auto-ID Center, in 1999. At the time, the author was thinking about the possibility of computers being able to know everything they could about the environment around them without needing the help and intervention of users. In this way, according to the author, it would be possible to reduce losses, costs, and waste. They need computers so that they can see, hear, smell, and understand the world on their own. and understand the world on their own [3].

IoT is about connecting the unconnected enabling smart objects to communicate with other objects, systems, and people [4].

The devices contain sensors and, or actuators, collect, analyze, and share data with other devices, programs, and platforms, and this data can be used to monitor and interact with various equipment, as well as to provide better planning, control and coordination of system, these connected things can improve autonomy, communication and facilitate knowledge sharing [5] which helps to create smart solutions.

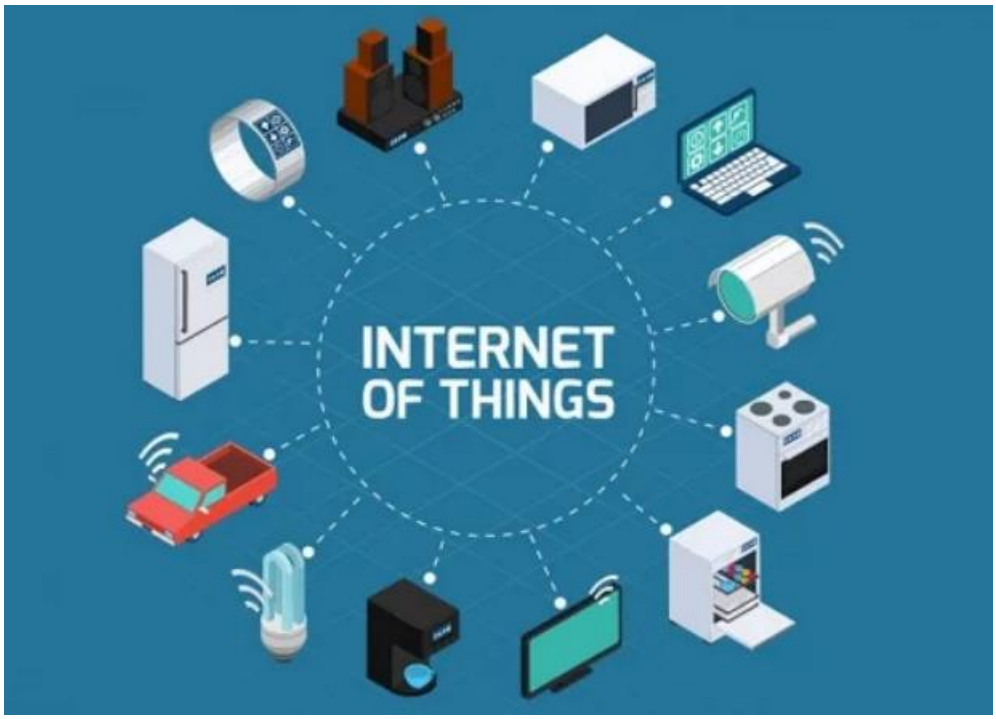

Figure 2. Internet of Things [6]

The figure 2 illustrate the connection between physical objects with the user and the internet, using intelligent sensors, such as Bluetooth and GPS, as well as software, employed in the collection and transmission of data to the network, allowing to control various devices by mobile.

2.1.1 Advantages and Disadvantages of IoT

The IoT is a network of physical objects connected to the internet that collect and share data. This technology has potential to transform the way we live, work, and interact with the world around us. Here are some of the main advantages [6]:

- It can help with smarter control of homes and cities via cell phones. It increases security and offers personal protection.
- By automating activities, we save time.
- Information is easily updated in real time, even if we are far from our actual location.

IoT also presents some challenges including:

- Hackers can gain access to the system and steal personal information. Since we have added so many devices to the Internet, there is a risk that our information will be misused.

- They rely heavily on the Internet and cannot function effectively without it.
- With the complexity of systems, there are many ways for them to fail.

2.2. Smart Buildings

The term "smart buildings" [7] has been used for more than two decades. During the second half of the 1970s, the term referred to buildings constructed based on energy efficiency concepts; in 1980, was used to refer buildings that could be controlled via a computer. Technological advances have made it possible to construct buildings capable of managing their infrastructure and services, such as lighting or air conditioning systems, in a more efficient and autonomous way, using as a basis and autonomous way, using some pre-defined configurations, via software as a basis for decision-making.

Thanks to the power of the Internet of Things, entire cities are becoming digitally interconnected and thus smarter. By collecting and analyzing huge amounts of data from IoT devices across different city systems, cities improve the lives of citizens. Smart cities can make better decisions through the data they collect on infrastructure needs, transportation requirements, crime, and safety. A study shows that using existing smart city applications, cities improve quality of life indicators, such as crime, traffic, and pollution, by between 10% and 30%. Internet of Things technologies in everyday life as part of your home, transportation, or city, relate to a more efficient and enjoyable life experience. Internet of Things promises a better quality of life through routine chores and increased health and wellness [8]

The concept of Smart Building could be defined as a set of communication technologies enabling different objects, sensors, and functions within a building to communicate and interact with each other and to be managed, controlled, and automated in a remote way. Indeed, technologies help to connect a variety of subsystems that originally operated independently. Automated processes allow the control of the building’s operations including HVAC (Heating, Ventilation, Air Conditioning), lighting, security, and other systems [9].

The following is a set of characteristics for smart buildings [10]:

- IoT sensors and devices for real-time monitoring and control of environmental conditions.
- Energy management systems to reduce consumption and CO2 emissions.
- Automation of security functions, such as surveillance cameras and access systems.

2.2.1 Advantages and Disadvantages of Smart Buildings

A smart building is a building that uses technology to improve efficiency, safety, and occupant comfort. These technologies include automation, sensors, connectivity, and data analysis. Here are some of the main advantages of smart buildings [11]:

- More security at home - The Internet of Things (IoT) has made it possible to connect various devices and sensors to a control center, or hub. This, in turn, can be controlled by voice or via cell phone so that you always have the information in your pocket. In this way, you can install a doorbell and video intercom whose alerts are displayed on your smartphone. The same goes for a smart lock, motion sensor or light that will immediately notify the user via cell phone.
- Greater personalization of the space. The smart home can give the user control of the climate control, or entertainment systems. Even cleaning with the best robot vacuum cleaners, automatic watering systems, or lighting with solution from Philips Hue and others. The smart home can give the user control of the climate control, or entertainment systems. Even cleaning with the best robot vacuum cleaners, automatic watering systems, or lighting with solutions from Philips Hue and others. It's possible to interconnect several appliances in a smart home, from the fridge to the washing machine. Internet connection is one of the requirements, with several manufacturers preparing their products to operate in conjunction with other systems in the home.
- Greater savings and energy efficiency. The smart home helps save money by improving energy efficiency, reducing waste, and making it easier to control sockets. For example. using a smartphone or voice commands, you can put an end to stand-by consumption.
- More convenience and comfort. Planning a smart home involves choosing the sensors and automations that will make everyday life easier for the inhabitants, properties will be more attractive to potential tenants by simplifying various tasks.
- A practical example of this potential is smart locks on external gates or internal doors, which can also facilitate any transition between tenants.

Here are some of the main disadvantages of smart buildings:

- Initial cost. Although technology is becoming increasingly advanced and accessible, the initial cost of purchasing, installing, and configuring a smart home system can be considerable. However, there are solutions to suit all budgets.
- Networking is mandatory. This is how the various appliances, sensors, and automation communicate with each other. The same is true of control hubs, or command centers, so having a good network service is a must. Security concerns. An encryption system is crucial for the Wi-Fi network, as well as for the connected IoT devices, and strong passwords are needed. If possible, you should only add trusted devices to your home network.

2.2.2 Architecture for developing a Smart Building

The development architecture comprises the following points:

- Heterogeneity
- Context sensitivity
- Practicality
- Information security and privacy.

M2M (Machine-to-Machine) communication takes place between two or more devices without the need for direct human intervention. This communication is a premise for IoT applications and, consequently, for the new possibilities of solutions to be used in the field of buildings applications. The proliferation of wireless sensor-to-actuator networks (WSN - Wireless Sensor Network) has enabled M2M communications connectivity. WSN networks, as previously analyzed, are characterized by their low cost, low consumption, and self-configuration. WSN networks can be built involving different devices and protocols. For example, the ZigBee and Z-Wave protocols are the most common for home applications. For a long time, networks with Bluetooth communication allowed the connection of up to seven devices limited to up to 100 meters between them. However, with Bluetooth Low Energy (BLE), mesh-type networks can be built and compete with other protocols available on the market [12].

Context sensitivity refers to the ability to react based on physical changes (changes in the position of the object or changes in the state) or changes in the environment around it, such as

temperature, devices, sounds, and lighting. Being context-sensitive allows for a real-time reaction.

In the case of buildings, it is possible to know people's location using a cell phone, which allows for the instant readjustment of the building according to the passage or permanence of people.

The recognition protocol performs the authentication of this user and automatically conducts a control check of the user's access, alerting the local security, or blocking its entry. By having access to people who are using a certain environment, the building management system can trace the best lighting and air conditioning conditions based on the physical aspects of these users (sex, height, weight) and based on the activities to be performed in the room (meeting, workshops, fairs, contests, computational research). Instant modelling of utilities is an effective way to guarantee the correct use of building resources, avoid waste, and increase the building's energy efficiency.

It can therefore be seen that context sensitivity is the basic for building intelligence. To make this sensitivity possible, some research [13] demonstrated the ability to capture activities using flexible electronics installed in watches, tags, or cards. Another alternative [14] shows that it is possible to recognize emotional characteristics through cameras by recognizing facial patters, posture, and gestures. In addition, [15] points out that the analysis of audio clips and interpretation of sound amplitudes and energy also allows certain emotions to be recognized. In this way, the building could, for example, be able to trigger the building’s medical department or call an ambulance when it detects a health problem in one of its users, or call security in cases of conflict, aggression, harassment, or theft. Being able to interpret user’s emotions is also a way to assess people’s level of comfort and evaluate which configurations result in their well-being.

Technological progress should aim to improve people's quality of life. of people's lives, and the usability of technology plays an important role in making this happen. For this to happen, improving the interface between man and machine is one of the main objectives when designing a new solution. Failure to synchronize technological advances in the interface can jeopardize the success of new technologies or applications. In buildings, operations are usually carried out via consoles such as computers, monitors, projectors, or televisions associated with keyboards, mice, touch screens, or remote controls, screens, or remote controls. However, some researches are being carried out to make the computer's presence less important during the

control of less visible activities, through compact and embedded systems with cameras, sensors, microphones, and other sensors, or even grouping all control functions in smartphones [16].

The new technological resources made available by the advancement of information and computing technologies enable new forms of interaction between man and machine. Context sensitivity allows voice commands to become a natural means of commanding actions in a [17] building. to command actions in a building. The technological challenge lies in the fact that the building's system must be sensitive enough to pick up voice in any location, or at least in most areas.

Virtual reality has emerged as a more intuitive form of interaction between human being environment by translating digital information into actions in a virtual environment captured by cameras. Users can simulate walks in virtual environments and interact with objects that reproduce the real environment. The detection of gestures and movements by artificial intelligence algorithms makes it possible for the elderly or people with special needs, for example, to carry out predefined actions. As such, the technology added to smart buildings should be seen as a tool to guarantee accessibility. It must be able to enable the target public to carry out simple domestic tasks, guaranteeing their quality of life and independence.

In addition to these possibilities, the popularization of social networks and their growing importance in people's daily lives makes it possible to combine the services offered by smart buildings and social platforms. and social platforms. By integrating social media with the building infrastructure, it is possible to send alerts via social media to the residents of a particular home or to people who work in the same building, indicating the conditions of use of utilities or informing them about the traffic situation or parking spaces. The new forms of interaction between man and machine are the subject of research aimed at bringing together and improve the relationship between users and the utilities in intelligent buildings.

Information security and privacy are related to issues at the top levels of protection against external attacks or the non-permissibility of access to privileged information. With the increase in applications involving networks connected to the internet in which the exchange of information between sensors, computers, and servers occurs, this issue theme has been the main concern of companies and developers.

Illegal acquisition of confidential materials, registration data, videos obtained through IP cameras, personal photos from cloud servers, and password acquisition are some of the main activities of hackers. By increasing the number of devices connected to a network, smart

buildings will consequently increase the risks related to this information's protection and security. In order to avoid invasions, smart buildings must be able to provide the necessary means to meet the confidentiality, integrity, availability, and authenticity of your information. Studies [17] argue that wireless sensor and actuator networks must have an information protection system in each nodes of their network.

The major security concerns regarding IoT applications revolve around two aspects. The first aspect deals with the relationship of trust that must be established between the network of objects and a new device to be added to that network. The second concern involves applications in which information is transmitted for processing on cloud servers. The sending of personal data or media requires the consent of the users, who must accept the legal terms of service providers.

2.3. IoT integration platforms

IoT integration platforms are solutions that allow devices, data, and application to be connected and integrated. They are essential for developing IoT solutions, as they allow developers to focus on creating applications and functionalities without worrying about complexity of integrating devices and data. In this dissertation three integration platforms will be compared which are: openHAB, Home Assistant, and Domoticz.

2.3.1. openHAB

The open Home Automation Bus (openHAB)[2] is an open source, technology agnostic home automation platform which runs as the center of your smart home.

OpenHAB is an open-source home automation project that began in 2010. It is a Java based project that supports a wide variety of home automation technologies and devices. It requires the use of a Java virtual machine and can be installed on servers running various operating systems. It can be run on any operating system, including Linux, macOS, Windows, Raspberry Pi, PINE64, Docker, Synology, and others.[3]

[2] https://www.openhab.org/docs/

[3] https://www.dusuniot.com/pt/blog/openhab-vs-home-assistant-which-one-is-right-for-you/

The main features of openHAB:

- The ability to integrate multiple devices and systems. openHAB may include other home automation systems, (smart) devices and other technologies into a single solution.
- Provides a uniform user interface and a common approach to automation rules across the entire system, regardless of the number of manufacturers and sub-systems involved.
- Give the most flexible tool available to make almost any home automation wish come true.[4]

The main advantages of openHAB includes:

- Compatibility – openHAB supports many different protocols and devices, so you can integrate a wide range of smart home devices.
- Ease of use – openHAB provides a user-friendly web interface that allows you to easily manage and configure a smart home.
- Automation – openHAB allows to create rules and scenarios to automate your smart home.
- Voice control – openHAB integrated voice control via services such as Amazon Alexa and Google Assistant.

2.3.2. Open Remote

Open remote it is an open source IoT platform that can integrate, design, and manage solutions focus on smart cities, buildings, home automation and health care. Using Open Remote Designer, it is possible to adapt a specific solution to meet the needs of each user. needs of each user, also giving the possibility of retrofitting devices that were not designed to be smart and designing specific rules to control lighting, entertainment, climate and other entertainment, climate control and other [18].

Open Remote allows you to integrate all the devices in a smart home and home and create a universal remote control to control them from your smartphone or tablet. Examples of automation that can be created using Open Remote are:

- When you arrive home, Open Remote can turn on the living room lights, turn on TV.
- If the thermostat detects that it is cold outside, Open Remote can turn on the heating.

[4] https://www.openhab.org/docs/

- If the motion sensor detects movement at your front door, Open Remote can turn on the video intercom.

2.3.3. Home Assistant

Home Assistant is free and, open-source home automation alternative used to develop decentralized home automation systems that control actuators, interpret data collected by sensors implement automation rules and manage communication between devices [19].

Home Assistant contains many extensions for various purposes, such as in the openHAB, other programs such as Visual Studio Code, Spotify, and others [20].

The problem that exists in the Home Assistant is the documentation [21]. New users will notice some difficulty configuring and interacting with the system. However, Home Assistant has an active community that continues to grow, so if users experience difficulties the community can respond.

The remote connection part has limited management and requires a subscription to use to voice assistants Google Assistant and, Amazon Alexa. If want to remotely access the system ends up being exposed on the Internet, making it more vulnerable [21].

One of the most significant benefits of an innovative smart home automation platform is the ability to connect all your devices.

The advantages of Home Assistant include:

- Compatibility – with many different smart home devices Home Assistant supports a wide range of smart devices, including Amazon Echo, Google Home, Philips Hue, Nest and many more. This allows you to centralize and manage your devices in one place.
- Ease of use – Home Assistant has a user-friendly interface that is easy to navigate and allows you to see your devices and settings in one place.
- Automation – Home Assistant lets you create automation rules to make your smart home even smarter. For example, you can create rules that make sure the lights are turned off when you leave the house, or that the heat is turned down when everyone in the house is asleep.
- Integration – with other services: Home Assistant offers integrations with many other services such as IFTTT, Google Assistant, and Alexa making it more powerful.

2.3.4. Domoticz

Domoticz is a Home Automation System that lets you monitor and configure various devices like lights, switches, various sensors or meters like temperature, rain, wind, UV, electra, gas, water and much more. Notifications or Alerts can be sent to any mobile device.[5]

This system is designed to operate in various operating systems. The user-interface is a scalable HTML5 web frontend, is automatically adapted for desktop, mobile devices and compatible with all recent browsers.

Domoticz Advantages includes:

- Easy to set up and use.
- Large community and support.
- Integrates with many devices and protocols.

The major Disadvantages are:

- Limited customization options.
- Resource intensive usage for larger systems.

2.4. Protocols used to interconnect home automation platforms

Protocols used to interconnect home automation platforms are sets of rules and standards that allow devices from different manufactures to communicate with each other. These protocols are essential for creating integrated and efficient home automation systems.

There are many different protocols available, the most popular protocols are: Zigbee, Zwave, Wi-Fi, Bluetooth, Thread [22], and others.

MQTT and Rest API where the protocols chosen because they are more common and particularly available in the platform that will be used. The best protocol for interconnecting home automation platforms via IP an WAN in this case its MQTT because its efficient and reliable, and HTTP because it's a protocol that is widely supported. For example, MQTT can be used to interconnect Home Assistant and openHAB that could allow the two platforms to share data and control devices.

[5] https://github.com/domoticz/domoticz

2.4.1. MQTT

MQTT a foundational Internet of Things standard developed by the OASIS consortium, has now been approved for release by the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) [23].

The MQTT [24] is a lightweight machine to Machine-to-Machine communication protocol designed to be easily implemented on low bandwidth networks.

MQTT offers several benefits [25]:

- Efficient and lightweight – MQTT minimizes the resources required by clients and network bandwidth.
- Scalable – MQTT can scale to support millions of devices or "things" in an IoT or IoT ecosystem.
- Bidirectional communication – MQTT facilitates communication between devices and servers and supporting publishing and subscribing.
- QoS levels – MQTT specifies QoS levels to ensure reliable message delivery.
- Persistent sessions – MQTT supports persistent sessions between devices and severs, reducing reconnection time over unreliable networks.
- Security – MQTT supports TLS[6] encryption for message confidentiality and authentication for client verification.

The MQTT Protocol uses a Publish/Subscribe model which allows the client to make posts and/or capture information while the server manages the sending and receiving of the respective data. In other words, in an MQTT there will be a publisher who will be responsible for publishing messages in a particular topic where a subscriber will subscribe to this topic to access the message.

As there is no direct connection between the subscriber and the publisher, for these messages to take place, the MQTT protocol will need a message manager called a Broker [26].

[6] https://www.cloudflare.com/pt-br/learning/ssl/transport-layer-security-tls/

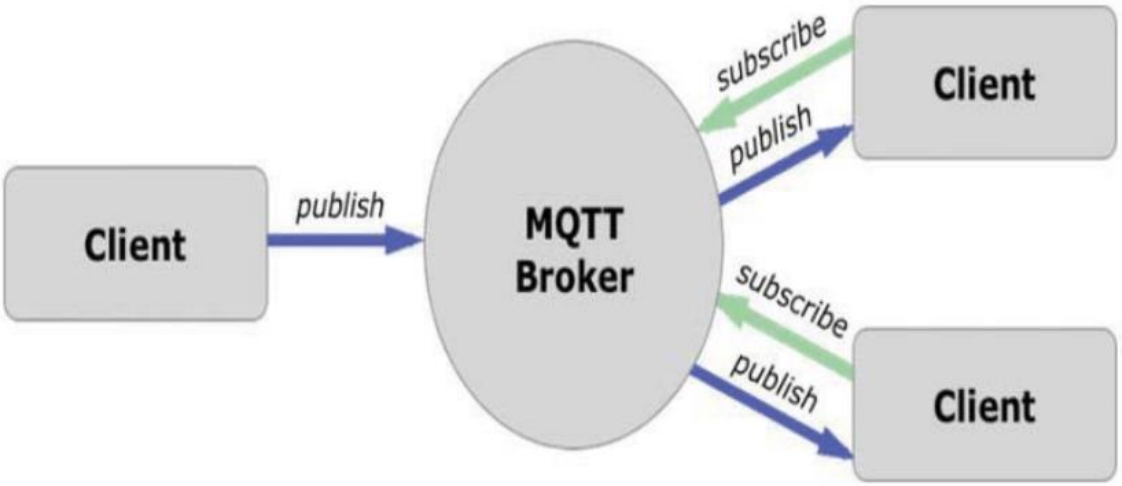

Figure 3. Publish – Subscribe MQTT Protocol [27]

The Figure shows that in an MQTT protocol, data is published and received via a type of server called a Broker. In other words, in this communication there will be a client which will play the role of Publisher and will transmit the message, with a destination topic and its Payload (the content of the message). This message will then be transmitted to the Broker, who in turn will be responsible for managing it and forwarding it to the Subscriber who was previously subscribed to the topic. Along the same lines, if a customer wants to become a Subscriber to a topic, they simply must send a request to the Broker, who will be able to link the customer to the topic in question.

The MQTT integration in openHAB is done in two ways:

- Subscribing to topics associated with items – this is the most common way of integrating MQTT into openHAB, it's necessary to create an item for each topic you want to subscribe to. The item type must be the same as the type of message you expect to receive in the topic.
- Using a binding: OpenHAB provides bindings for various MQTT devices and services. These bindings provide an easier way to integrate MQTT into OpenHAB. To use a binding, you need to install the binding in OpenHAB. After installing the binding, you can configure the binding to connect to your MQTT broker. Once the binding is configured, you can create items to represent the MQTT devices or services you want to integrate. The binding will automatically subscribe to the topics required for the items.

2.4.2. REST API

Representational State Transfer (REST) is a software architecture style for Application Programming Interfaces (APIs) that consists of guidelines and best practices for creating scalable web services. REST uses simple HTTP to make calls between machines. This happens via a request/response mechanism between the server and the client. For example, a client, let's say an Android application, make a request for most recent post from the website. The server knows how to interpret this request, through REST, and satisfies the response by providing the most recent posts in a format understood by the client.

Roy Fielding, in the year 2000, introduced Representational State Transfer (REST) [27]. He explains that World Wide Web, although seen as the world's largest distributed application it is necessary to understand the core architecture principles behind the Web. Furthermore, has asserts that if known will greatly translate to improving other distributed applications to avoid undesirable changes with respect to standards based on the web architecture. He introduces REST, a significant model or framework which is the underlying principle as a guideline in designing and evaluating a real software system. The web represents a loosely couple application framework, where resources are uniquely identified and accessed via Uniform Resources Identifies (URI's) by clients including smart phones, tablets, laptops etc. The resources are accessed in a request/response model using methods like GET, PUT, POST, DELETE.

Moreover, as asserted [27], system performance is usually reliant on the communication network when it comes to network-based applications. With respect to distributed hypermedia system, he asserts than the focus being on computation-intensive tasks, large data is rather sent 26 among components during their communication.

Thus, brings in the idea of REST, to resolve these problems identified to get the needed functional, performance, and social properties required of an architecture [27]

On the openHAB platform, there is a binding that allows communicating with remote openHAB servers. The Remote openHAB connection allows you to communicate with remote openHAB servers. Communication is two-way. The connection on the local server listens for any item status updates on the remote server and updates the channel connected on the local server accordingly. Transfers any item command from the local server to the remote server. You can map anything remote to a local thing. Through this mapping, in your rules (local

server), you can perform actions based on state updates or state changes generated by remote things, and you can perform actions based on trigger events generated by the trigger channels defined on the remote thing.

The Rest API can be used by openHAB to control openHAB devices and items remotely, for example turning a light on or off.

2.5. Comparative Table

Table 1. Comparison between IoT platforms

	OpenHab	**Open Remote**	**Domoticz**	**Home Assistant**
Open source	Yes	Yes	Yes	Yes
The language that has been developed	Java	Java	C/C++	Java, Phyton
The difficult of installation of new devices and the platform	Easy	Medium	Easy	Easy
The number of supported devices	Large	Large	Limited	Large
The devices in with it can be run	Linux MacOS Windows RaspeberryPi PINE64	MacOS Windows Debian RaspeberryPi	Linux MacOS Windows Docker RaspeberryPi	MacOS Android Windows RaspeberryPi
The number of worldwide users	Large numbers of users	Medium numbers of users	Large numbers of users	Large numbers of users
If it allows automation rules	Yes	Yes	Yes	Yes
Number of Binding supported and example of protocols	42748 users[7] Zigbee Z-wave MQTT HTTP InfluxDB Grafana KNX Modbus Dali IFTT HomeKit	675 users[8] Zigbee Z-wave MQTT HTTP InfluxDB Grafana KNX Modbus Dali	33512 users Zigbee Z-wave MQTT HTTP InfluxDB Grafana KNX Modbus Dali HomeKit, IFTT	193448 users[9] Zigbee Z-wave Thread Matter MQTT HTTP InfluxDB Grafana KNX Modbus, Dali

[7] https://community.openhab.org/u?order=likes_received
[8] https://forum.openremote.io/u?order=likes_received
[9] https://community.home-assistant.io/u?order=likes_received

Chapter 3. Proposed Architecture

The main goal of this dissertation is to design and compare alternative architectures for the integration of building automation platforms.

This section intends to explain the proposed design. The alternative architectures must be designed to be easy to integrate and implement, comprising of several approaches for the communication protocols and message structures to be used.

For this endeavor two dimensions, with alternative solutions, are considered:

- Message Transfer versus State Transfer
- Edge Integration versus Cloud Integration

The table 2 represents the combination of these different forms of integration, where, as it will be seen later, the Message Transfer protocol to be used is MQTT, and the State Transfer will be achieved by using Rest. These protocols will achieve integration by communicating through services implemented either at a remote cloud or at a near edge server.

Table 2. Different forms of integration

	Edge Integration	**Cloud Integration**
Message Transfer	Resort Message Service	Cloud Message Service
State Transfer	Resort Connector Service	Cloud Connector Service

For the edge integration, a server is installed near the premises of the buildings, where the automation platforms to be integrated run. This server deployment could be done in common areas of a condominium or a resort, where the buildings belong. For a cloud integration, remote services, possibly running in distinct servers, may be used, and they are usually located distant from the building automation platforms to be integrated.

The proposed alternatives for the integration architecture are detailed in the component diagram in the figure 4. In this diagram, on the left-hand side, there is a home automation platform, here referred to as Alice, from where remote services or items are usually controlled. On the right-hand side, there is a home automation platform, here referred to as Bob, that hosts services or items to be controlled. Both these entities are in smart buildings and communicate through remote and/or local services, that run on one or multiple servers located at the edge or on a remote cloud. This is depicted in the central components of the diagram.

For the sake of simplicity, in the diagram Alice is shown as a client who wants to control a remote service in Bob´s home, but in the integration, approaches proposed here, any home automation platform can assume both roles.

To achieve remote control, Alice sends commands via the ‘Item Control’ component, which is connected to both ‘Message Binding’ and the ‘Connector Binding’ components. Afterwards, the ‘Message Binding’ can alternatively use the communication services of the ‘Cloud Message Service’ or the ‘Resort Message Service’. The same happens for the ‘Connector Binding’ that can use the ‘Cloud Connector Service’ or the ‘Resort Connector Service’ for the communication.

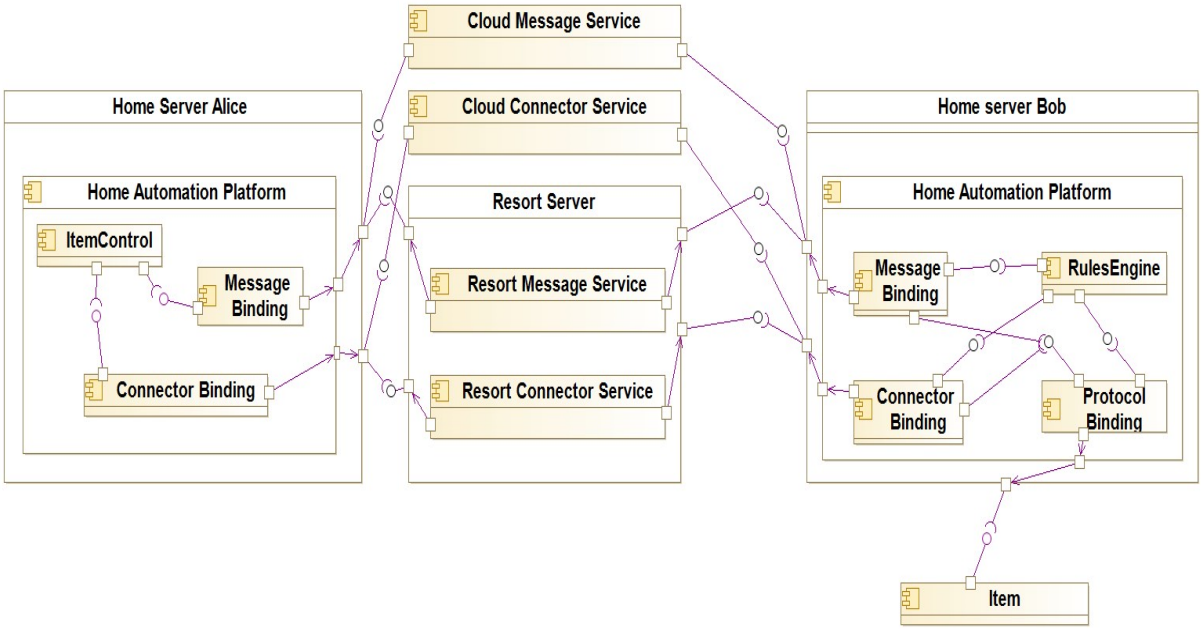

Figure 4. Component Diagram

At Bob’s, his ‘Message Binding’ or ‘Connector Binding’ components triggers the control of a remote service or items that is accessed with the ‘Protocol Binding’ component. This service or item is represented in the diagram by the ‘Item’ component. When the state of the service or items changes, a rule is then triggered and thereafter implemented by Bob’s ‘Rules Engine’ component. This rule uses the services of the ‘Message Binding’ or the ‘Connector Binding’ components of Bob to communicate back to Alice the item state change that she commanded. This communication is once again going through respectively the ‘Cloud Message Service’ or ‘Resort Message Service’ components, and the ‘Cloud Connector Service’ or the ‘Resort Connector Service’.

At Alice's, the state change is received either by the 'Message Binding' or the 'Connector Binding' components of Alice, that pass the state change back to the representation of the remote item at Alice's, i.e., the 'Item Control' component.

To further clarify the integration, a step-by-step description, that uses message communications services, where Alice wants to turn on the lights in Bob´s, follows:

Step 1: Alice´s command

Alice emits a command to turn on the lights at Bob´s. That happens through the 'Item Control' in her own house.

Step 2: Sending the command

The command is sent by the 'Item Control' that passes it to the 'Message Binding' which is connected to a remote 'Cloud Message Service' or to a 'Resort Message Service' close to Alice´s, in a 'Resort Server'.

Step 3: Delivering to Bob

The 'Message Binding' at Bob´s receives the control message with the command to turn on the lights.

Step 4: Activating the command in Bob´s house

The 'Message Binding' at Bob´s house delivers the turn on request of the lights to the 'Protocol Binding'.

Step 5: Turning on the lighting device

The 'Protocol Binding' implements the command, and the lights are turned on. Due to the state change at the 'Protocol Binding', a pre-programed rule is activated at the 'Rules Engine'.

Step 6: Giving Alice feedback

The activated rule, at the 'Rules Engine', sends back to Alice the feedback of the state change through the same communications channels.

Step 7: Display the feedback at Alice's

Alice receives feedback through the 'Message Binding', that, in its turn, passes it on to the 'Item Control' component. The 'Item Control' state will change to display that the lights have been turned on.

Through the above-described steps, we can see that Alice can turn on the lights at Bob´s and that she can receive the feedback about the state change, which assures that the lights have been successfully turned on.

Chapter 4. Implementation

The aim of this chapter is to explain the technologies used and the steps taken for implementing the proposed solution.

The figure 5 represents the Deployment Diagram. This diagram shows how each component of the architecture has been implemented.

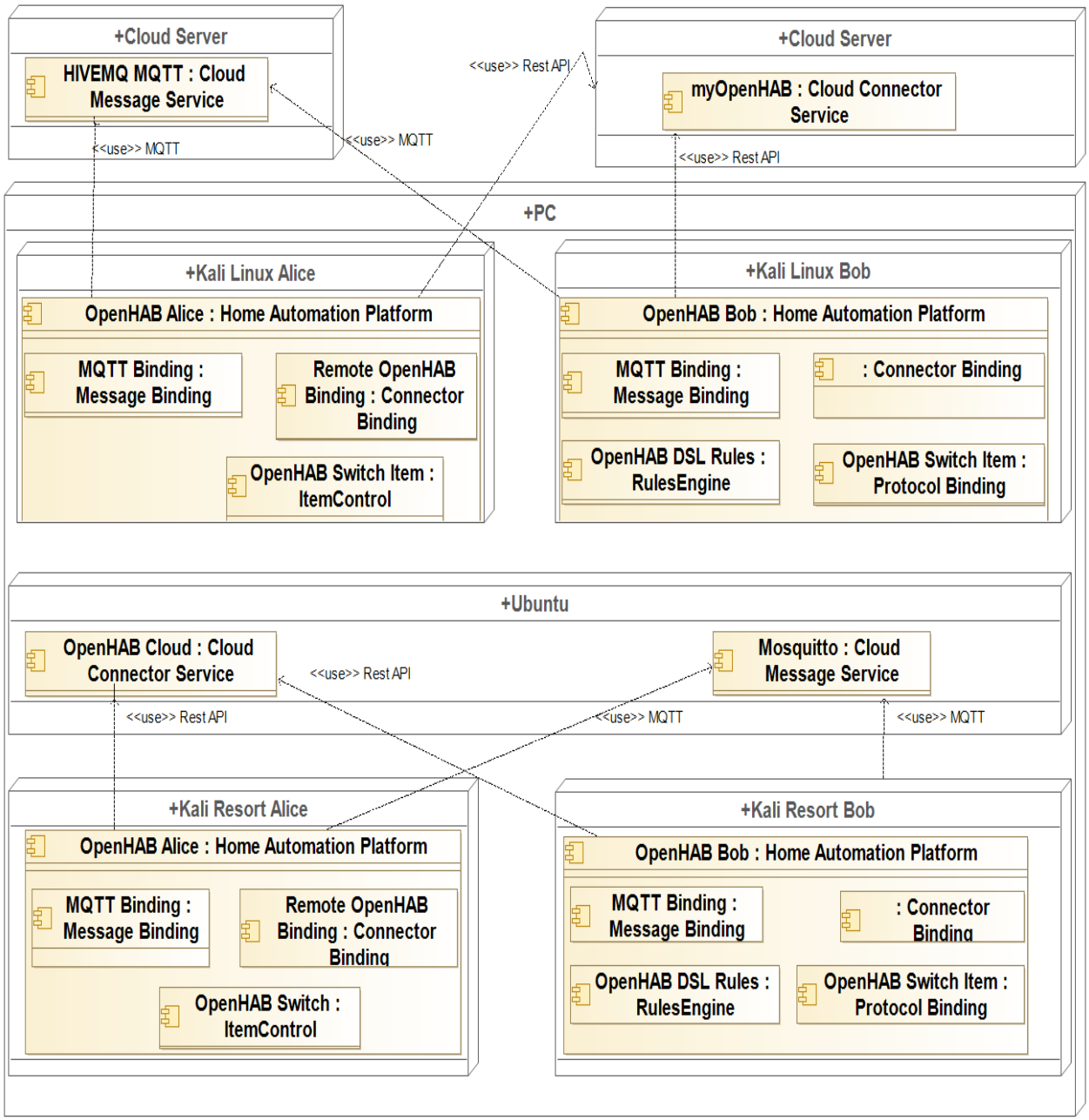

Figure 5. Deployment Diagram

All the implementation work performed for this dissertation has been carried out on a HP Spectre laptop that runs the Windows11 operating system. However, virtual machines that run on a VMware Workstation Pro installation have been extensively used during the implementation. Details of the host and the virtual machines specifications is shown in Table 3.

Table 3.Specification of Host and Virtual Machine

	Virtualization Platform	CPU	Logical Processors	RAM	Storage	OS
Host/laptop	-	Intel® Core™ i7-1195G7 2.92 GHz	8	16GB	457GB	Windows
Kali Linux Alice	VMware	Intel® Core™ i7-1195G7 2.92 GHz	4	1.9GB	80.1GB	Kali Linux-3.41
Kali Linux Bob	VMware	Intel® Core™ i7-1195G7 2.92 GHz	4	2.1GB	80.1GB	Kali Linux-3.41
Ubuntu	VMware	Intel® Core™ i7-1195G7 2.92 GHz	8	4GB	32GB	Linux 2022.4
Kali Resort Alice	VMware	Intel® Core™ i7-1195G7 2.92 GHz	4	1.9GB	80.1GB	Kali Linux-3.41
Kali Resort Bob	VMware	Intel® Core™ i7-1195G7 2.92 GHz	4	2.1GB	80.1GB	Kali Linux-3.41

For the virtual machines, different flavors of the Linux operating system have been chosen, because of its stability and small footprint regarding the required resources computational resources on the host machine. A total of five virtual machines have been created, four of them have a Kali Linux distribution and one has an Ubuntu distribution.

Kali Linux Alice and Kali Linux Bob emulate a middlebox, home server, or set-top box where the home automation platform is running respectively for Alice's and Bob's. The home automation platform used for the implementation was the open source openHAB[10], for the reasons already explained in the literature review. These platforms communicate with each other via remote cloud services. When Alice and Bob communicate via the 'Cloud Message Service' they use the open access HIVEMQ [11]service, with the MQTT protocol. When Alice and Bob communicate via the 'Cloud Connector Service' they use the Rest API service provided by the open access myOpenHAB[12]. HIVEMQ and myOpenHAB are services hosted in independent servers in a remote cloud.

Kali Resort Alice and Kali Resort Bob were initially created as clones of respectively Kali Linux Alice and Kali Linux Bob and provide the same set of functionalities. However, Kali Resort Alice and Kali Resort Bob have been reconfigured to use local implemented communication services, instead the equivalent services hosted in a remote cloud. For this, a local edge server, that implements the 'Resort Server' component, has been also created, in an Ubuntu virtual machine. The Ubuntu server has been installed with two services that support the communication. A Mosquitto[13] installation for the 'Resort Message Service', that uses the MQTT protocol, and an 'openHAB Cloud'[14] installation for the 'Cloud Connector Service', that uses Rest API.

The versions of software and tools installed and configured on the Kali and Ubuntu virtual machines are shown in table 4.

Table 4. Software's and version

	Kali	Ubuntu
Automation Platform	openHAB 3.41	openHAB cloud
Java JDK	Zulu 11.62.17-ca-jdk11.0.18-linux	
MQTT Broker		Mosquitto Version 2.0.11 V5.0/v3.1.1/v3.1

[10] https://www.openhab.org/docs/

[11] https://www.hivemq.com/

[12] https://www.myopenhab.org/

[13] https://mosquitto.org/

[14] https://www.openhab.org/addons/integrations/openhabcloud/

Aiming at the MQTT integration, the Ubuntu virtual machine has been configured with a Mosquitto installation, Version 2.0.11, that implements the MQTT protocols versions V5.0/v3.1.1/v3.1. In addition, OpenHAB instances can communicated via a Rest API openHAB Cloud connector, an add-on for openHAB that allows to connect local openHAB instances to an openHAB Cloud server. A remote server like myOpenHAB may be used, but for the openHAB instances to use the openHAB Cloud connector, connected to local resort server, it is necessary to install a custom openHAB Cloud service. For configuring this, on Ubuntu, the steps in the GitHub repository for openHAB Cloud[15] have been followed. During the installation and configuration process some difficulties made this task very time-consuming because some steps in the GitHub were incorrect. Finally, when the openHAB Cloud was successfully installed, two accounts for the Kali Resort Alice and Kali Resort Bob have been created.

Similar accounts have also been previously created in the myOpenHAB remote cloud service.

All Kali virtual machines have been installed with the openHAB open-source home automation platform, version 3.41, that run on a Java JDK Zulu 11.62.17-ca-jdk11.0.18-linux. The openHAB instances for Alice's and Bob's have also been specifically configured with items that control or are controlled via the four alternatives for the integration architecture.

The figure 6 shows the implemented Alice's control panel. When a user clicks on Alice's Home, it can see the control of local items at Alice's. When a user clicks on Bob's Home icon, it can see the control of remote items at Bob's. The figure 7 how's this last control panel, that has several visual representations of items that can be remotely controlled by Alice, either by message transfer, the 'MQTT Bob Kitchen Light' item, or state transfer, the 'Remote BobKitchenLight' item.

[15] https://github.com/openhab/openhab-cloud/blob/master/README.md

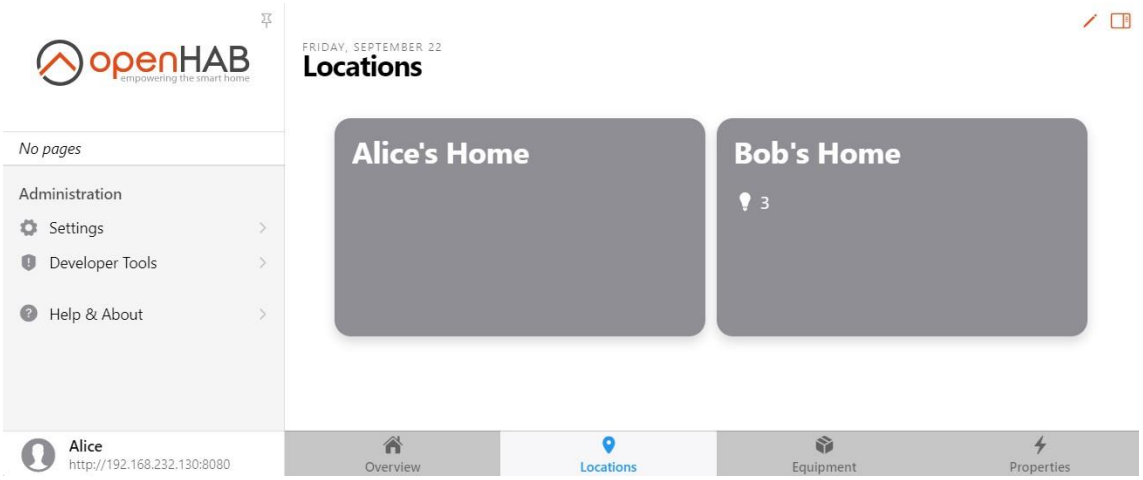

Figure 6. Alice's control panel for all items

Figure 6. shows the control panel, on the left are the items in Alice's house that Alice controls. On the right are the items that Alice controls remotely in Bob's house. The user controls the lights by clicking on the light's items. These items have been configured to send commands via a configured 'MQTT Binding' on openHAB, that implements the 'Message Binding' component of the architecture, or via a configured 'Remote OpenHAB Binding' on openHAB, that implements the 'Connector Binding' component of the architecture. These happens respectively for the 'MQTT BobKitchenLight' and the 'Remote BobKitchenLight'.

After sending the control of the items, Bob will process the request and it communicates back the state change of the remote items.

Figure 7. Alice's control panel for items located at Bob´s

The figure 7. Shows the state feedback of the 'MQTT Bob Kitchen Light' and the 'Remote BobKitchenLight' items are respectively received by Alice in the 'MQTT BobKitchenLightStatus' and the 'Remote BobKitchenLightState' items, also

respectively using the configured ‘MQTT Binding’ and the configured ‘Remote openHAB Binding’.

Rules have been created in openHAB instances to automate tasks and actions in the home automation platform. The rules are written in scripts, which are a subset of Java.

To carry out the implementation, it was necessary to automate the process in which Bob sends the feedback of the change of state of lights, controlled according to Alice’s request.

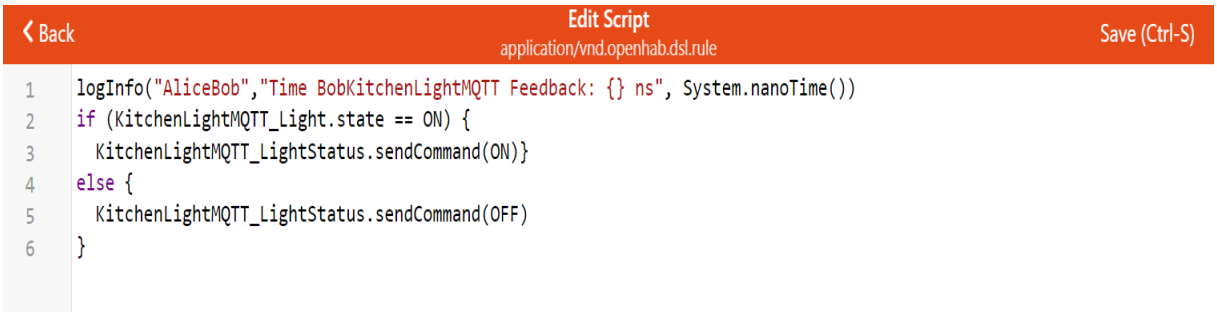

Figure 8. Bob MQTT feedback to Alice

The figure 8. show how Bob’s rule has been implemented to send the feedback of a ON or OFF state of the ‘MQTT Bob Kitchen Light’ item, via MQTT, to Alice’s ‘MQTT BobKitchenLightStatus’ item.

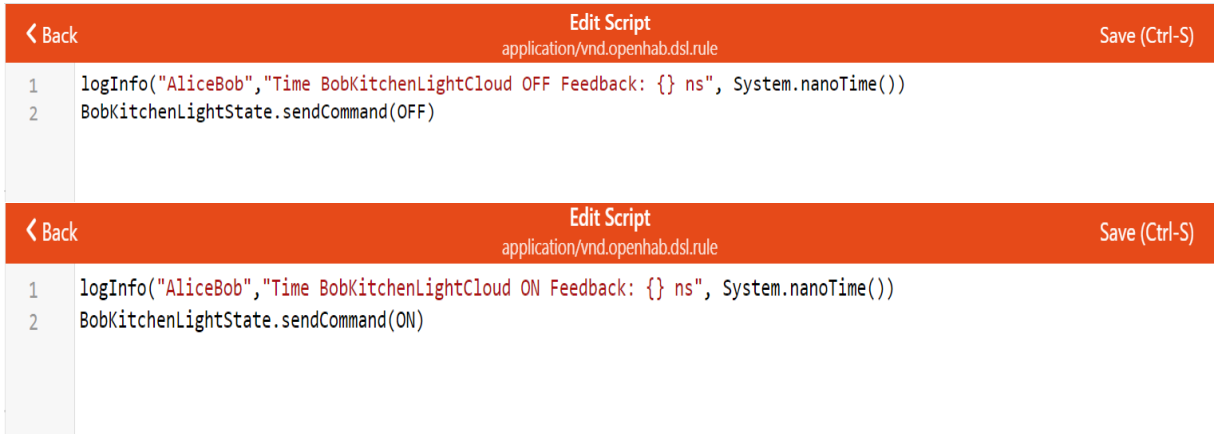

Figure 9. Bob Rest openHAB Cloud feedback to Alice

The figure 9. shows the two rules implemented at Bob rule to send the feedback state of the ‘Remote BobKitchenLight’ item, via Rest API, to Alice’s ‘Remote BobKitchenLightState’

item. In one rule, Bob sends the feedback for state of light off, when Bob Kitchen Light has been updated to OFF. On the other rule, Bob sends the feedback for the state of light on, when Bob Kitchen Light has been updated to ON.

```
 - Item 'BobKitchenLight' received command OFF
11:11:31.209 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLight' changed from ON to OFF
11:11:31.211 [INFO ] [rg.openhab.core.model.script.AliceBob] - Time Bob Item Kitchen Light OFF Feedback: 5367195193527 ns
11:12:05.172 [INFO ] [openhab.event.ItemCommandEvent       ] - Item 'BobKitchenLight' received command ON
11:12:05.179 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLight' changed from OFF to ON
11:12:05.236 [INFO ] [rg.openhab.core.model.script.AliceBob] - Time Bob Item Kitchen Light ON Feedback: 5401219624652 ns
11:12:05.239 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLightState' changed from OPEN to CLOSED
11:12:06.538 [INFO ] [openhab.event.ItemCommandEvent       ] - Item 'BobKitchenLight' received command OFF
11:12:06.544 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLight' changed from ON to OFF
11:12:06.545 [INFO ] [rg.openhab.core.model.script.AliceBob] - Time Bob Item Kitchen Light OFF Feedback: 5402529246439 ns
11:12:06.549 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLightState' changed from CLOSED to OPEN
11:12:07.764 [INFO ] [openhab.event.ItemCommandEvent       ] - Item 'BobKitchenLight' received command ON
11:12:07.769 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLight' changed from OFF to ON
11:12:07.768 [INFO ] [rg.openhab.core.model.script.AliceBob] - Time Bob Item Kitchen Light ON Feedback: 5403752413182 ns
11:12:07.774 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLightState' changed from OPEN to CLOSED
11:12:08.867 [INFO ] [openhab.event.ItemCommandEvent       ] - Item 'BobKitchenLight' received command OFF
11:12:08.873 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLight' changed from ON to OFF
11:12:08.873 [INFO ] [rg.openhab.core.model.script.AliceBob] - Time Bob Item Kitchen Light OFF Feedback: 5404857369872 ns
11:12:08.881 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLightState' changed from CLOSED to OPEN
11:15:22.169 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLightState' changed from NULL to ON
11:15:22.170 [INFO ] [openhab.event.ItemUpdatedEvent       ] - Item 'BobKitchenLightState' has been updated.
11:15:56.144 [INFO ] [openhab.event.RuleUpdatedEvent       ] - Rule 'BobFeedbackAliceOFFOpen' has been updated.
11:16:38.830 [INFO ] [openhab.event.RuleUpdatedEvent       ] - Rule 'BobFeedbackAliceOFFOpen' has been updated.
11:16:54.148 [INFO ] [openhab.event.RuleUpdatedEvent       ] - Rule 'BobFeedbackAlice' has been updated.
11:17:16.404 [INFO ] [openhab.event.RuleUpdatedEvent       ] - Rule 'BobFeedbackAlice' has been updated.
11:17:30.103 [INFO ] [openhab.event.RuleUpdatedEvent       ] - Rule 'BobFeedbackAlice' has been updated.
11:17:45.932 [INFO ] [openhab.event.ItemCommandEvent       ] - Item 'BobKitchenLight' received command ON
11:17:45.939 [INFO ] [openhab.event.ItemStateChangedEvent  ] - Item 'BobKitchenLight' changed from OFF to ON
11:17:46.022 [INFO ] [rg.openhab.core.model.script.AliceBob] - Time BobKitchenLight ON Feedback: 5742006065269 ns
11:17:46.025 [INFO ] [openhab.event.ItemCommandEvent       ] - Item 'BobKitchenLightState' received command ON
11:17:48.500 [INFO ] [openhab.event.ItemCommandEvent       ] - Item 'BobKitchenLight' received command OFF
```

Figure 10. Alice´s logs

In addition, rules have also been created that will allow Bob to keep a record for future analysis whenever the status of lights are changed, or a user performs certain actions. The same logging rules have also been implemented at Alice's openHAB instance of her's home automation platform. Figure 10 shows some results obtained with these logging rules.

The logging rules print the timestamp when relevant states changes. These timestamps have been used in the next chapter to analyze the performance obtained with the alternative approaches to for the integration of the building automation platforms.

Chapter 5. Tests and Results

This chapter presents the results obtained from the tests carried out, showing tests scenarios and which architecture performed better.

Four tests were carried out using the following configurations:

1. For testing local services: Resort Connector service and, Mosquitto.
2. For testing remote services: Cloud connector service and, HIVEMQ

For each configurations tested, the switch was flicked every five seconds, and it was done twenty times, and the log was obtained. The average value was calculated from the values extracted from the log.

The results in the table 5. were obtained through calculations made in Excel to obtain average and standard deviation.

Table 5. Average and standard deviation

	Average Time [ms]	Standard deviation [ms]
Cloud Connector Service myOpenHAB	1358,47	369,31
Resort Connector Service local OpenHAB Cloud	33,49	7,52
Cloud Message Service HIVEMQ	147,13	54,23
Resort Message Service Mosquitto	124,59	43,05

Based on the results in table 5, it can be concluded that Resort services are faster than cloud services and are good for getting fast feedback. However, using only the average values from table 5. it is not good enough to make a definitive evaluation. For this end, it necessary to know if there are extreme situation of very significant delay times. This can be done through the Box Plot, also known as a box and whisker plot, which is a graphical method for displaying the distribution of data.

Histograms require a sample of at least thirty elements to be useful, while the box plots require a sample of only five elements and provide more detail in the tails of the distribution and more readily compared across three or more samples.

To calculate the range, it is necessary to find the largest observed value of variable, the maximum value and subtract the smallest observed value, the minimum value. The range only considers these two values and ignore the data points between the two extremities of the distribution, it is used as a supplement to other measures, but is rarely used as the sole measure of dispersion because it's sensitive to extreme values.[16]

Box plots characterize a sample using percentiles, also known as the lower quartile (Q1), median (m or Q2), and upper quartile (Q3). Quartiles are insensitive to outliers and preserve information about the center and spread [28].

Analyzing the four alternatives tested, the following can be seen:

- The second hypothesis (Resort Connector Service) presents the best time, so it would be the best solution. However, it has security problems (the item are exposed), where someone can control other items in an unwanted way. Therefore, this solution is not the best.
- The Messaging Service (MQTT): HIVEMQ & Mosquito, which present a time within acceptable limits for an interactive application. According to the literature, times below 150 ms are within acceptable limits for an interactive application, i.e. from the moment you press the button to receiving a response.
- Of all the alternatives, the Cloud Connector Service is the only one that would not be acceptable for integration in the case under study (items that require interactive responses), as it has a time outside the acceptable limits (1358.47 ms). It should be noted that integration can also be carried out using remote services in the cloud, at the level of state propagation with Rest API, but for items that do not require interactive responses.

[16] https://www150.statcan.gc.ca/n1/edu/power-pouvoir/ch12/5214890-eng.htm

The figure 11. shows that the graph of resort connector on the right has a lower scale than the cloud connector service graph on the left, in magnitude of ten times lower. The Figure shows that response time is higher when using the cloud server.

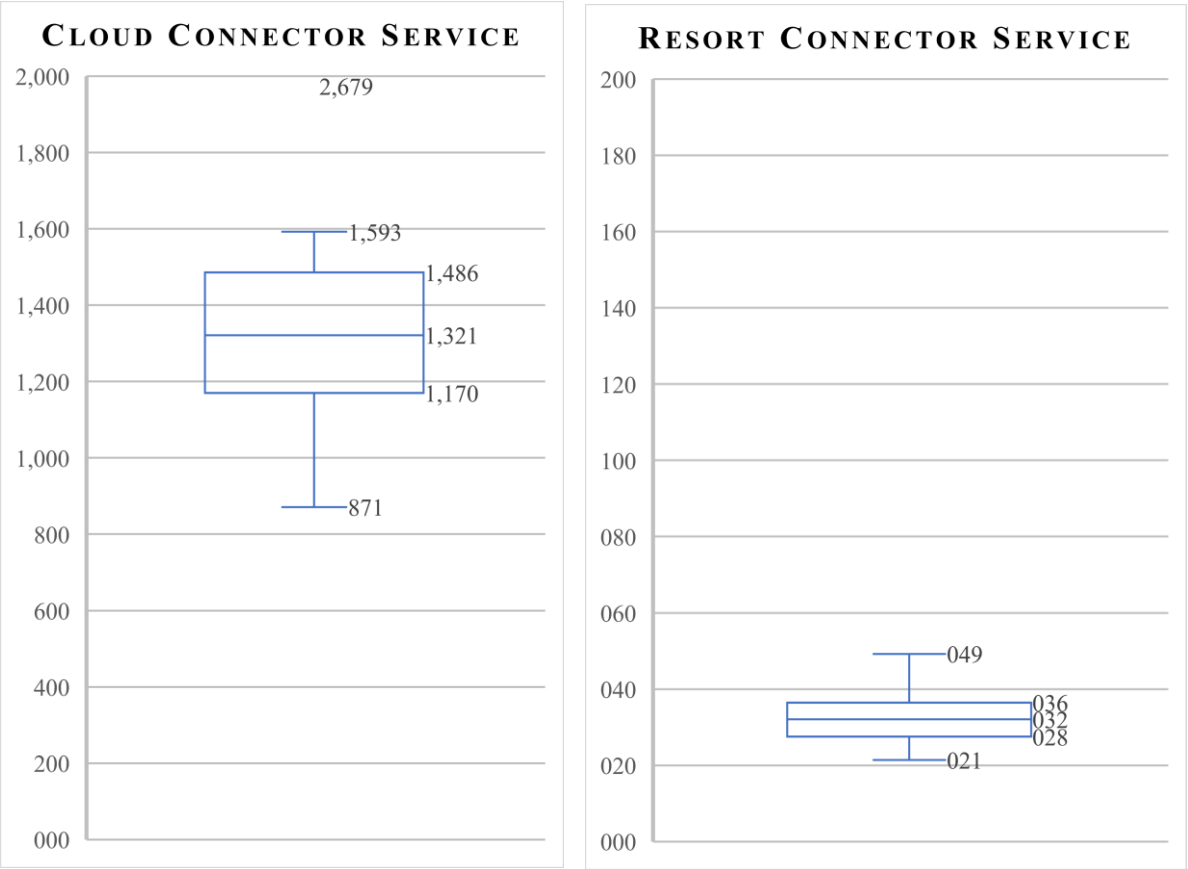

Figure 11 Comparison between cloud connector and resort connector

According to the calculations, there was one extreme situation, which was a reading error, but in general there were no other extreme phenomena.

Most of the samples in the cloud connector service fall between 1170 ms, that it is the lower quartile, and 1486 ms, that it is the upper quartile, and has average 1358,47ms.

For the resort connector service there is no deviation or sample outside from normal. It presented an average of 33,49 ms, and the values of the quartiles are between 36,45 ms for the upper quartile and 27,51 ms for the lower quartile.

In myopenhab, it can take almost 3 seconds to get feedback in a borderline situation and need to be careful with the services we use in the Cloud, especially when they have network access.

Comparing the two graphics from Figure 12, the Resort connector service has better performance.

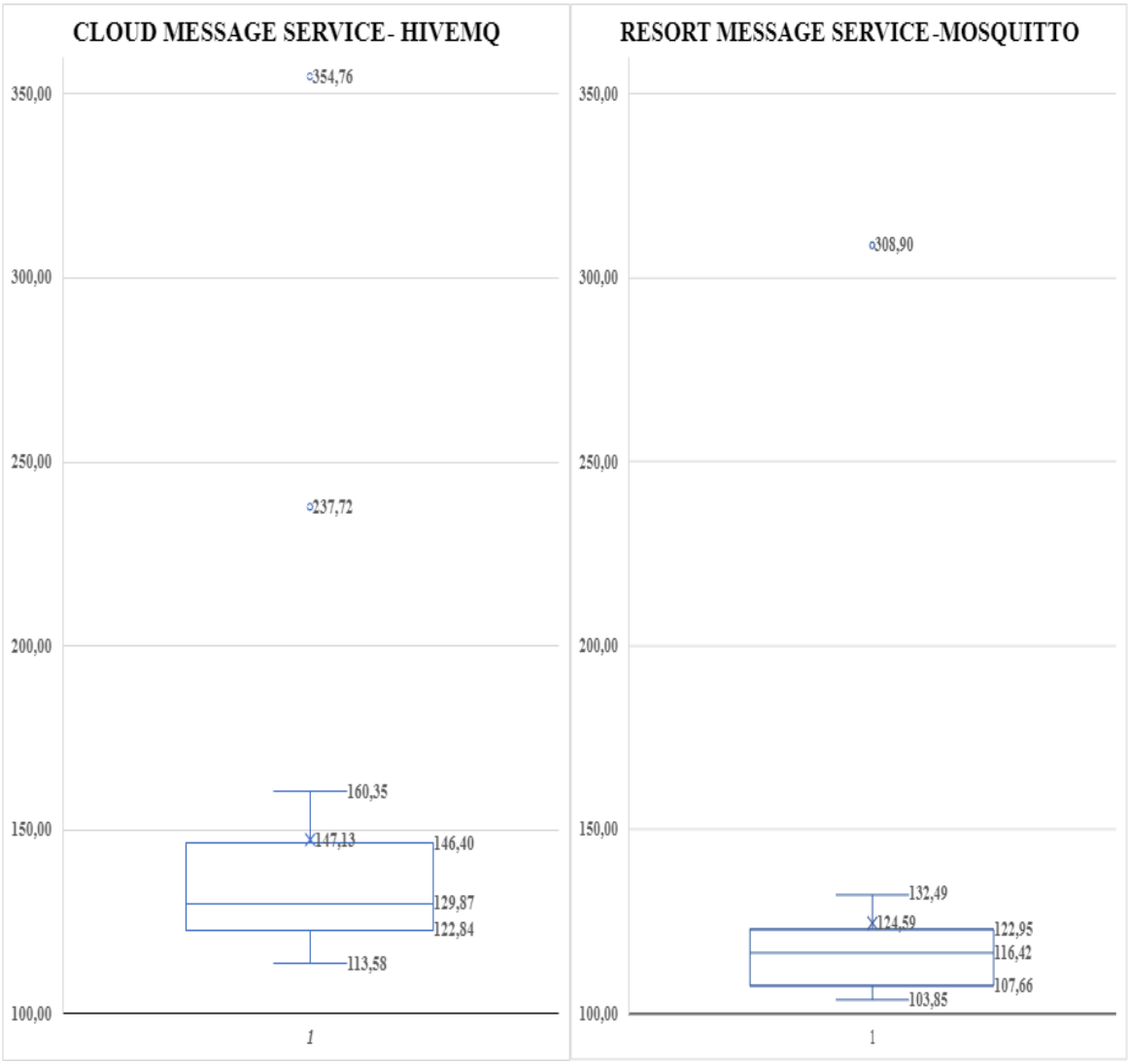

Figure 12.Comparison between HIVEMQ and Mosquitto

The figure above, 124.5 shows que Resort Message Service presents the best time compared to Cloud Message service. However, the integration via Rest API left exposed items that could be undesirably controlled and, as such, it is less secure.

It was noted that in the implementation in myOpenHAB remote are exposed, which is not the case when communication via Message services like MQTT. The MQTT allows the configuration of the item to be controlled individually, indicating a topic to use for each one, which represents an important advantage.

Furthermore, the response times obtained with Message services like MQTT, as shown in Figure 12 represents an acceptable response time below 150ms, whether using edge or cloud servers.

These times are within the limits for interactive communications, below 150ms which is the limit of perception for human interactive communications and well below 400ms which is the acceptable limit that does not harm the user's interactive experience [29].

Chapter 6. Conclusions and Future work

This chapter presents the main conclusions of the work carried out, the obstacles and limitations encountered, as well as future work that could be developed.

6.1. Main conclusion

Throughout the dissertation, several obstacles were encountered due to the nature of the proposed integration. However, the initial objectives were achieved, and a stable work base was created an architecture for future developments has been created.

From the tests carried out, it can be concluded that the best results were obtained from the Resort connector service, because has better performance, and the reaction time is faster. On the other hand, it was noticed that local servers are more efficient than remote servers.

Analyzing the research questions:

1. How decentralization can be implemented?
2. Which interconnection method is more effective and will have better performance results, location, or various servers MQTT, remote openHAB connector?
3. What risks do distributed systems offer to the community of smart building by allowing the decentralization of control to provide services by third parties?

It is possible to conclude that:

1. Decentralization can be implemented based on communication services at the edge or cloud.

2. Resort Connector Service presents the best delay time, so it could be considered the best solution. However, this integration via Rest API left items exposed that could be controlled in an undesirable way. Therefore, this option is problematic due to security concerns.

3. An integration achieved through message service, with topic subscription, does not present the above security issue. Moreover, actuation delays are within acceptable limits for interactive control.

6.2. Future work

Future work could be the development of security and communication using REST API.

- Integration with Blockchain Security was not possible this time, but it could be considered in the future.
- By adding IFTTT to the local connector service integration, it should be possible to limit the items to be exposed for remote control.

References

[1] Domingo, M.C. (2012) Na Overview of the Internet of Things for People with Disabilities. Journal of Network and Computer Applications, 35, 584-596.
https://doi.org/10.1016/j.jnca.2011.10.015

[2] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research", Journal of Management Information Systems, vol. 24, pp.45-77, 3 Dec. 2007, ISSN: 0742-1222. DOI: 10.2753/MIS0742-1222240302.

[3] ASHTON, K. That "Internet of Things" Thing. Em RFiD Journal. 2009.

[4] DAVID, Hanes., JEROME, Henry, "IoT Fundamentals-Networking Technologies, Protocols, and Use Cases for the Internet of Things-CISCO (2017), pp 72, ISBN-13: 978-1-58714-456-1.

[5] A. Koohang, C. S. Sargent, J. H. Nord, and J. Paliszkiewicz, "Internet of things (iot): From awareness to continued use," International Journal of Information Management, vol. 62, Feb. 2022, ISSN: 02684012. DOI: 10.1016/j.ijinfomgt.2021.102442.

[6] https://acervolima.com/vantagens-e-desvantagens-da-iot/

[7] ABDENNADHER I., KHABOU N., RODRIGUEZ I. B., JMAIEL M., Designing energy efficient Smart Buildings in ubiquitous environments, 15th International Conference on Intelligent Systems Design and Applications (ISDA), Marrakech, 2015, pp. 122-127.

[8] Durán-Sánchez, A, et al. (2017) Sustainability and Quality of Life in Smart Cities: Analysis of Scientific Production. In: Sustainable Smart Cities. Creating Spaces for Technological, Social and Business Development, Springer, Berlin, 159-181. https://www.researchgate.net/publication/308963358_Sustainability_and_Quality_of_Life_in _Smart_Cities_Analysis_of_Scientific_Production

[9] MEDINA, Bruno Eduardo. Internet das coisas em edifícios inteligentes: Desenvolvimento de uma rede de sensores e atuadores sem fio para o controle de sistemas de climatização. Dissertação (Mestrado) - Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. Campinas – SP, 2017.

[10] https://biblus.accasoftware.com/ptb/smart-building/amp/

[11] https://4gnews.pt/vantagens-casa-inteligente-smart-home/

[12] T.K.L. HUI, et al. Major requirements for building smart homes in smart cities based on internet of things technologies. Em Future Generation Computer Systems. 2016.

[13] Y.-G. HA, Y.-C. BYUN. A ubiquitous homecare service system using a wearable user interface device. Em IEEE/ACIS 11th International Conference on Computer and Information Science, ICIS. 2012, pp. 649-6

[14] C.S.S. TAN, J. SCHÖNING, K. LUYTEN, K. CONINX. Informing intelligent user interfaces by interfering affective states from body postures in ubiquitous computing environments. Em Proceedings of the International Conference on Intelligent User Interfaces. IUI'13, ACM, New York, NY. USA. 2013, pp. 235-246

[15] I. BISIO, A. DELFINO, F. LAVAGETTO, M. MARCHESE, A. SCIARRONE. Gender-driven emotion recognition through speech signals for ambient intelligence applications. IEEE Trans. Emerg. Top. Comput. 1 (2). 2013, pp. 244-257

[16] J. CORN. User Unfriendly: Consumer Struggles with Personal Technologies, from Clocks and Sewing Machines to Cars and Computers. The John Hopkins University Press. 2001.

[17] K. ISLAM, W. SHEN, X. WANG. Security and privacy considerations for wireless sensor networks in smart home environmnets. Em IEEE International Conference on Big Data, Big Data. 2014. pp.1-8.

[18] Rui, J. Passinhas (2019). Integration of Mobile Devices in Home Automation with Use of Machine Learning for Object

[19] Leonardo, José Silva (2019). Utilizando o Home e o NodeMCU para um modelo genérico de automação moderna: https://ri.ufs.br/bitstream/riufs/12305/2/Leonardo_Jose_Nascimento_Silva.pdf

[20] Josué, Duarte Ferreira (2022). Home Assistant versus Hubitat https://digituma.uma.pt/bitstream/10400.13/5101/1/Tese%20-%202066513.pdf

[21] Home Assistant for newcomers: What is hassio, hassos, hassbian, 101 and cookies – Home Assistant Community. (n.d.). Retrieved August 29, 2021, from https://community.home-assistant.io/t/homeassistant-for-newcomers-what-it-is-what-ishassio-hassos-hassbian-101-and-cookies/123004

[22] SANTOS, Pedro. Internet das coisas: O desfio da privacidade. Dissertação- Instituto Politécnico de Setúbal, 2016.

[23] https://www.oasis-open.org/news/pr/oasis-mqtt-internet-of-things-standard-now-approved-by-iso-iec-jtc1/

[24] OASIS, "MQTT,", 2018, [Online] Available: http://mqtt.org/faq

[25] https://www.hivemq.com/article/how-to-get-started-with-mqtt/

[26] https://www.automacaoindustrial.info/mqtt/

[27] R. T. Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” Building, vol. 54, p. 162, 2000.

[28] Krzywinski, M., Altman, N. Visualizing samples with box plots. *Nat Methods* 11, 119–120 (2014). https://doi.org/10.1038/nmeth.2813

[29] KUROSE, James (2016) Computer Networking: A Top–Down (7th ed) ISBN-10 : 9780133594140ISBN-13: 978-0133594140