COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

Communication, optimization, and control aspects of smart homes

Autor: Alicia Sanz De La Escalera

Director: Tomislav Capuder

Co-Director: Ivan Pavic

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Communication, optimization, and control aspects of Smart homes

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2019/20 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

[illegible]

Fdo.: Alicia Sanz de La Escalera Fecha: 04/09/ 2020

Autorizada la entrega del proyecto

EL COORDINADOR DE TRABAJOS FIN DE GRADO

[illegible]

Fdo.: Aurelio García Cerrada Fecha: 09/09/ 2020

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

Communication, optimization, and control aspects of smart homes

Autor: Alicia Sanz De La Escalera

Director: Tomislav Capuder

Co-Director: Ivan Pavic

Madrid

UNIVERSITY OF ZAGREB

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

BSc THESIS ASSIGNMENT No.

COMMUNICATION OPTIMIZATION AND CONTROL ASPECTS OF SMART HOMES

Alicia Sanz De La Escalera

Zagreb, June 2020.

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

BSc THESIS ASSIGNMENT No.

COMMUNICATION, OPTIMIZATION AND CONTROL ASPECTS OF SMART HOMES

Alicia Sanz De La Escalera

Zagreb, June 2020.

ABSTRACT

Smart homes systems have been traditionally present only in the concept of providing additional comfort to their users with limited or no interaction with the out-of-home environments. This paradigm is changing and new approaches to intelligent communication and control creates benefits not only for smart home users but also to the upstream systems, energy systems. To monitor and control smart home devices one must be able to read information from sensors and, after processing those signals, send information back to the device actuators or other equipment.

The thesis will provide an in-depth review of existing open source smart home platforms as cloud solutions where one can connect his devices with a goal of providing solutions to the following tasks:

i) Kind of data that can be sent/received between devices and platforms.
ii) Can the signal be aggregated as a household and then send to platforms?
iii) Can those platforms be customized in terms of signals provided or they are hardcoded?
iv) Can one connect custom made device to those platforms?

Furthermore the thesis will describe and details the process of monitoring and acquiring all possible data from the smart devices (smart ac conditioner, refrigerator, boiler, washing machine etc.) as well processing them in order to create an optimized plan of future behaviour of those devices with a final goal of lowering smart house electricity bill.

RESUMEN

El concepto de casas inteligentes ha estado tradicionalmente presente únicamente en el ámbito del confort y la comodidad de sus usuarios, con una interacción con el exterior limitada. Este concepto está cambiando, están surgiendo avances que aportan beneficios no sólo para los usuarios de casas inteligentes sino también para los sistemas energéticos.

Para controlar los equipos de una casa inteligente, es necesario leer información procedente de sensores, procesar las señales que generan y enviar de vuelta dicha información a los equipos inteligentes para ejecutar una acción.

La tesis proporcionará un estudio en profundidad sobre las plataformas de código abierto de hogares inteligentes, donde los usuarios pueden conectar sus dispositivos con el objetivo de dar respuesta a las siguientes cuestiones:

i) Tipo de datos que pueden ser enviados/recibidos entre los dispositivos y la plataforma de código abierto.
ii) ¿Se pueden integrar las señales en la casa inteligente y enviarlas a la plataforma?
iii) ¿Se pueden personalizar las señales que envían las plataformas de código abierto?
iv) ¿Puede el usuario conectar dispositivos personalizados y controlarlos a través de las plataformas de código abierto?

Además, la tesis describirá en detalle el proceso de monitorizar y adquirir los datos de los dispositivos inteligentes (aires acondicionados, termostatos, calderas, etc.) para procesarlos con el objetivo de crear un plan óptimo de futuro con el objetivo final de reducir el consumo energético y las facturas eléctricas.

TABLE OF CONTENTS:

INDEX OF FIGURES:

INDEX OF TABLES:

Acronyms

API:	Application Programming Interface
IoT:	Internet of Things
UI:	User Interface
MEMS:	Micro Electro-Mechanical Systems
WSNs:	Wireless Sensors Networks
HA:	Home Assistant
A/D Converter:	Analog-to-digital Converter
D/A Converter:	Digital-to-analog Converter
OSS:	Open-Source Software
HVAC:	Heating, Ventilating, and Air Conditioning
AC:	Air conditioner
MQTT:	Message Queuing Telemetry Transport
HTPC:	Home Theater Personal Computer
DIY:	Do-it-yourself

1. Purpose of the thesis

According to energy-saving, smart technology can help in this process, allowing homeowners to take control of their energy usage through a variety of smart-home energy-saving strategies. Smart-home hubs can turn appliances off completely when not in use, eliminating the energy drain caused by idling appliances and improving energy conservation. The HVAC system, (heating, ventilation, and air-conditioner) is responsible for about 46 percent of the home´s energy usage. The water heater represents the 14 percent, and appliances as washer, dryer, electric oven, electric stove, dishwasher account for 13 percent of the electricity cost. [1] To reduce power consumption, it is a good point to start controlling HVAC systems, as they take most of the energy waste. A way to control them is using smart thermostats, because they can be programmed to de adaptable to the homeowner's lifestyle. As smart thermostats are connected to the Internet, they can be turned on/off depending on the outside weather, or even on the user´s location, what will allow to save energy because HVAC systems will be working only when necessary. Controlling these functions by an open source smart home platform, helps the user having all smart devices included and controlled together.

In first place, the thesis will explain how data is sent and received between smart home platforms and devices in order to communicate. For that, there will be described parameters of a control installation as sensors and actuators, and how the smart home platform´s architecture is organized.

Secondly, as each platform allows performing different actions and functions, some have more limitations than others, the thesis will provide an in-depth review of existing open source smart home platforms and will make a comparison between them. Then, the purpose is to compare the process of controlling and connecting a custom-made device with a non-custom one, in one of the platforms. Specifically, the thesis will explain the process of controlling an AC device. Hardware and software implementation will be described and some programming functions to reduce energy-waste. Once both processes have been determined, there will be done a comparison to figure out which would be more profitable.

1.1. Thesis structure

In first place, an introduction of what is IoT and its importance on smart houses is explained, along with a brief explanation of what is needed to make communication between devices and Internet compatible.

Then in chapter four, Sensors and actuators- a conceptual overview, describes the components of a control installation involved in a smart house as sensors, actuators, and the sensor nodes. It also describes wireless sensors networks, and how data is managed by sensors, how they read information, and store it. There is explained types of sensors within their output data and the functions they implement.

In chapter five, Open source smart home platforms, is described the purpose of an open-source software and which benefits offer to a smart home platform. In this chapter a study of the different platforms has been done, to compare between them which devices can be connected, and if there is the possibility to connect custom-made devices. Then, in chapter six and seven, two smart home platforms have been studied in detail as OpenHAB and Home Assistant respectively. Its architecture and how they manage data flow are explained. Also, in both cases is used a light example, to understand the functionality of the components of each architecture.

In chapter eight, a comparison between these two platforms has been done to work later one of them to study how to control HVAC systems.

In chapter nine, a study about house temperature control has been done using Home Assistant, by two different process. Firstly, connecting a custom-made AC equipment, and implementing an Interface to make it compatible with the platform. Implementations as MQTT and Z-Wave are explained in detail. Programming functions to control it based on reducing energy waste are also detailed. Secondly, connecting a Tado device, which already has an interface with the platform, and there is any implementation needed.

Lastly, an economic comparison has been realised to figure out which previous process is more economically profitable.

2. Motivation

Due to the fact that the perspective of smart houses is changing, and more users tend to implement it in their homes, to simplify their day-to-day, it is important to take advantage of it for reducing energy waste, instead of just implementing it for comfort. Usually, there are three aspects that smart homes try to cover, which are resource usage, as water conservation, power consumption, security, and comfort. Reducing power consumption, help users to reduce their electricity bills, which is another relevant aspect to consider.

When implementing a smart house with an open source smart home platform, it allows the users to include all type of smart devices together to have them programmed to work with the same objective. Also, as implementing smart devices in a house can have a high cost, it is of great interest to connect and control a custom-made device, taking advantage of the non-smart devices that anyone can have at their homes, instead of investing on smart manufacturers that are usually more expensive.

3. Introduction

IoT, Internet of Things, is known as the aggrupation and interconnexion between devices and objects through a network, where all can interact between them. These devices can be from sensors to everyday objects as microwave, clothes, light bulbs, etc. The main objective is to connect everything together to improve people lifestyle. IoT has many appliances, it can be helpful for the business world, industrial appliances, livestock but also for the smart homes. For example, IoT allows having devices that can be controlled by voice, due to the fact they are connected to the Internet or helps connecting alarm systems to the internet for improving security.

Technologies used by IoT collect data and send it to networks to be analysed. From this point it is of great importance the way they communicate, the “language” spoken between devices and networks, there are sensors or devices which communication and connexion to the Internet is direct and easy, but on the other hand there are others which its protocol of communication is not that ambiguous. Each device manufacturer has its own communication protocols which makes that not all smart devices are compatible. One of the open communication protocols is MQTT (Message Queuing

Telemetry Transport), it allows manufacturers to participate enabling communication between different devices. [2]

The interface between devices can be established by connecting cables, but there is also another communication protocol more commonly used in smart homes, which is wireless communication, this can be both Z-Wave and ZigBee technology. This technology establishes communication between nodes network and the embedded gateway, which transmits the data to the monitoring PC through Wi-Fi network. [3]

IoT is the core part of the architecture of a smart house because it oversees managing web server data, and making communication compatible, even with non-living things in a comfortable way. The Interface of IoT based on smart houses, is shown in the figure below.

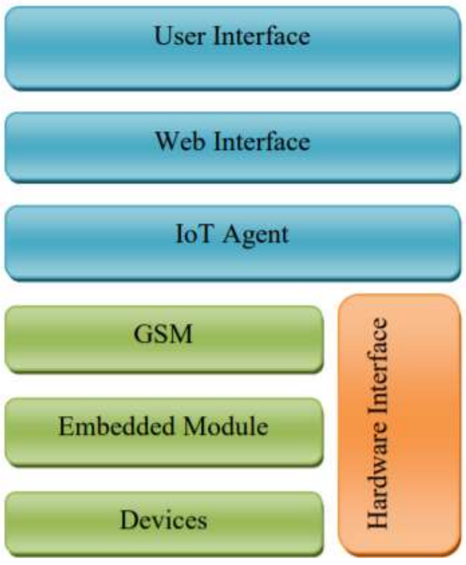

Figure 1: 7 Functional Interface of IoT based Smart Home [3]

Where, GSM stands for Global System for Mobile, which is another protocol for communication within the smart homes, that enables users seeing the status of their home devices, through the web client. As it has been mentioned the relevant point of IoT is to make a compatible communication, it is of great importance to establish a user and web interface with Internet and IoT agent, which is in charge of maintaining the current status of the devices. For Hardware Interface there are the communication protocols mentioned, as Z-Wave and ZigBee, because both hardware and software of any device must be compatible with the monitor. [3]

To conclude, IoT has many appliances and can be used on many different aspects, the thing that devices are connected and that they are intelligent is not that special, but to make communication between them compatible, and to collect and analyse data in order to send it to other devices.

Open source smart home platforms provide several communication protocols to integrate and control different devices. They have established interfaces, both in hardware and software, with the device manufacturers making them compatible so data flow between platforms and devices can be easily managed. Each smart home platform allows controlling lights, thermostats, alarm systems, boilers, etc. On the other hand, when connecting custom-made devices, (that not all smart home platforms allow it), the user needs to implement the interface by him/herself, including for example wireless communication protocols and a broker as MQTT to create the interface.

4. Sensors and actuators– a conceptual overview

A control installation has three fundamental parts which are, sensors, control systems and actuators or equipment. Sensors gather data from the outer condition and send it to the control system such that it can peruse and process. At that point, this information is sent to the actuators. As the data gathered requirements to have a similar organization to be investigated by the control framework, the sensor makes an interpretation of this data to electric signs. This implies, the sensor will change over some physical phenomenon into an electrical impulse that can be deciphered to decide a perusing.

A sensor is what is known as the input of the system or installation. It is a machine that detects events or changes in its environment and send the information to other electronics, frequently a computer processor. [4]

A good sensor has a high sensitivity, it is sensitive to the measured property. Where, the sensitivity is known as the ratio between the output signal and measured property.
There are alternative options that ought to be thought about once selecting a sensing element, as: accuracy, environmental conditions (usually they need limits for temperature or humidity), standardisation, resolution or price.

On the other hand, an actuator is the part of the installation that is accountable of moving and dominant a mechanism or system. It needs an impact signal and a supply of energy. The control signal is comparatively low energy and should be electrical voltage or current. Once it receives an impact signal, associate mechanism responds by changing the source´s energy into mechanical motion. An actuator operates in the reverse direction of a sensor. It takes an electrical input and turns it into a physical action. There are several types of actuators, some can be: electric motors, comb drive, hydraulic cylinder, or pneumatic cylinder.

On the other hand, one type of sensors that have become essential due to the fact of the advances in micro electro-mechanical systems (MEMS) are the wireless sensors networks (WSNs). These have many applications but are mainly focused on power conservation. WSNs are composed of individually embedded systems that are capable of:

(1) Interacting with their environment through various sensors
(2) Processing information locally
(3) Communicating this information wirelessly with their neighbours. [4]

In addition, the main element is the sensor node, which performs some processing, gathering sensory information and communicates with other connected nodes in the network. It generally consists of three components:

- The key components of the sensor network are wireless modules. They possess the communication capabilities and the programmable memory. These modules usually consist on a microcontroller, transceiver, power source, memory unit and include some sensors.
- The sensor board, which is mounted on the mote and integrates multiple types of sensors. The sensor board can also include prototyping area which is useful to connect custom-made sensors.
- The programming board or gateway board also provides multiple interfaces as Wi-Fi, USB or serial ports to connect different motes. The boards are used for programming the motes or gathering data from them.

4.1. Data Flow

Most of the sensors are discrete components that take the voltage in and produce either digital or analog data. It is also required another component to read the data from sensors and to send to the system control, this can be Arduino. Arduino in this case would be the sensor node, and the system control would be a computer. Related then to how sensors communicate their data to the other nodes in the network, there are two basic forms, wired and wireless networks.

- Wired networks: they require a wire or cable to send electric signals from one device to another. The hardware uses wired communication and must be aggregated to the hardware implemented on the nodes in the network.
- Wireless networks: in these cases, a wireless device such as Wi-Fi is used for each Arduino, it also requires a wireless router. The drawback is that Wi-Fi does not reach a very long distance, so wireless networks cannot be suitable for some networks. [4]

According to the output data there are two types of sensors:

- Analog sensors: these are devices that generate a voltage range, typically between 0 and 5 volts. An analog-to-digital circuit is needed to convert the voltage to a number. Most microcontrollers have this feature built in, as Arduino. Analog sensors work as resistors and, when connected to microcontrollers. Often require another resistor to “pull up” or “pull down” the voltage to avoid spurious changes in voltage known as floating. This is because voltage flowing through resistors is continuous in both time and amplitude. Thus, even when the sensor is not generating a value or measurement, there is still a flow of voltage through the sensor that can cause spurious readings. It is necessary to have a clear distinction between OFF (zero voltage) or ON (positive voltage). Pull-up and pull-down resistors ensure that these states have been determined correctly. In addition, for an analog sensor signal to be processed, it needs to be converted to a digital signal, using analog-to-digital converter, if a digital equipment is going to be used.

- Digital/Binary sensors: they are designed to produce a string of bits using serial transmission (one bit at a time). However, some digital sensors produce data via parallel transmission (one or more bytes at a time). The bits are represented as voltage, where high voltage, (it can be 5 volts, or ON state) is defined as '1'; low voltage (0 or even -5 volts, or OFF state) is '0'. Digital sensors can be sampled more frequently than analog signals because they generate the data more quickly and because no additional circuitry is needed to read the values (such as A/D converters and logic software to convert the values to a scale).

As a result, digital sensors are generally more accurate and reliable than analog sensors. (the accuracy of a digital sensor is directly proportional to the number of bits it used for sampling data). If the signal is digital and the equipment used is analog, it is needed a digital-to-analog converter.

4.2. Storing sensor data

There are two features to consider when storing data, firstly how data is interpreted and secondly how it will be used. Sometimes to store data it is necessary to use an A/D converter to store a sequence of voltages from an analog signal, but on the contrary it may not have any sense. To store digital conversion of the voltage it is necessary to consider the scale and range to derive the values intended to be represented. When using digital sensors, it is necessary to store them in an embedded database or in a file-storage device, and it is very important to consider what unit of measure is being used, (Celsius, Fahrenheit, feet, meters, etc.). Related to systems to storage, these can be hard-copy printer, secure digital card, USB hard drive or database server. [4]

4.3. Types of sensors

There exist different classifications of the sensors depending on, the output data, internal structure or type of parameters that can be detected.

There are two types depending on power or energy supply requirement, which are active sensors, for example, light detection. The ones that do not require power supply as radiometers, are known as passive sensors.

On the other hand, some groups of sensors are:

- Accelerometers: these are based on the Micro Electromechanical sensor technology. They are used for patient monitoring which includes pacemakers and vehicle dynamic systems.
- Biosensors: these are based on the electrochemical technology. They are used for food testing, medical care device, water testing, and biological warfare agent detection.
- Image sensors: they are used in consumer electronics, biometrics, traffic, etc.
- Motion detectors: these are based on the Infra-Red, Ultrasonic and radar technology. They are used in videogames and simulations.

In addition, there are sensors which are relevant for a smart house, these can be:

- Proximity sensors: these sensors use infrared or sound waves to detect distance to/from an object. For example, they are used for security as the main part of a smart alarm.
- Water sensor: it will alert if a leak is detected, or if the device has been inadvertently moved or tampered with.
- Weather sensors: sensors for temperature, barometric pressure, rain fall, humidity, wind speed, etc. Most generate digital data and can be combined to create comprehensive environmental sensor networks.
- Light sensor: sensors that measure the intensity or lack of light. Sometimes they are called photoresistors, these sensors use analog data. [4]

5. Open source smart home platforms

A Smart House is an automation house in which audio/video, alarm, boiler, lights, etc. can be automatized. Network service can be accessed everywhere in house, such that home appliances at any place may be interconnected with other devices. [5]

The purpose of a smart house is to provide safety, comfort and to save energy. Automating using an open source smart home platform has a wide range of possibilities, as every element or device that is automated is under the same control unit, so different devices can be connected between them and controlled at the same time.

Open-source software (OSS) dictates that the source code of an open-source project is publicly accessible and may be redistributed and modified by a community of developers. Open source projects have a commitment with its community and developers, so both platform and users are benefited. This commitment to community pushes developers to constantly contribute new features and to ensure old ones perform properly. The benefits of OSS are that they are more cost-effective, flexible, and secure. OSS can be altered and extended by any developer. [6]

An example of an open source smart home can be shown on the figure below. Where there are some important elements as 1) processor unit, 2) power unit, 3) by pass switch connector and relay unit.

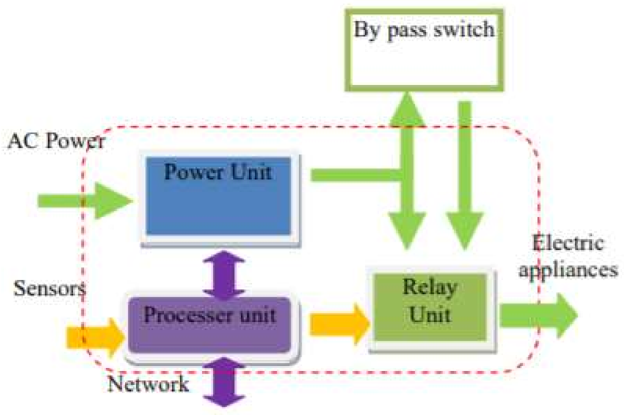

Figure 2: Smart Home Platform scheme

Open source home automation platforms have open-source software, and people working behind them achieve a strong codebase that users can use to monitor their homes. There are several platforms in the landscape of IoT (Internet of Things), each of them has different compatibility with each device manufacturer, have a bigger community of users, or is more flexible. Some of them just allow the user to buy the smarts devices on their own market store rather than customizing their equipment using Arduino or a Raspberry Pi. Some smart home platforms are EventGhost, Jeedom, MajorDoMo, Pimatic, MyController, Domoticz, and two which are the biggest because of the number of users and developers contributing are OpenHAB and Home Assistant. [7]

The table shown below collects some characteristics of 8 different smart home platforms, which are important to consider. When choosing a smart home platform, it is of great interest that the community language is understandable, because they are the ones who upload how to program each device and how to control them. In second place, not all platforms can control power consumption and energy systems. Thirdly, it is needed to consider the option of customizing. All platforms allow to control the functions between the possibilities of control they have. For example, if it is possible to control light bulbs, the user will be able to set the time he/she wants to turn them on, or if the user is controlling a thermostat he/she will be able to set the room temperature as he/she wants. But on the other hand, not all platforms allow incorporating custom-made devices, due to the interface, because they do not offer the possibility to the user to create an own interface so any device can communicate with the platform. Some platforms just include their own market, so the user is forced to buy their items, as in the case of Pimatic.

	Community language	Power consumption control	Functions Customizable	Custom-made devices	Limitations, only runs on:
EventGhost	English	NO	YES	YES	Python
Jeedom	French	YES	YES	YES	Linux
MajorDoMo	Russian	YES	YES	YES	Raspberry Pi
Pimatic	English	YES	YES	NO	Raspberry Pi
MyController	English	YES	YES	YES	Raspberry Pi
Domoticz	English	YES	YES	NO	C/C+
OpenHAB	English	YES	YES	YES	Java
Home Assistant	English	YES	YES	YES	Python 3

Table 1: Characteristics Open Source Smart Home platforms

EVENTGHOST

[8] EventGhost is a home automation program that enables competing technologies to work together by providing them with a platform through which they can communicate with one another. The program accomplishes this through an interface.

An automation example would be, pausing a movie on any HTPC (Home Theater Personal Computer) when someone rings the doorbell. The event would be the doorbell ringing and the action would be pressing the pause button. To implement this action, it is necessary to bundle both, doorbell, and television, together inside of a macro.

The platform has over 300 plugins. It gives the user a list to configure macros that do all kind of things like: launching applications, emulating keystrokes, emulating mouse movements and clicks, control the sound card, control external hardware devices like projectors and other media equipment and extensive control of programs that have special communication interfaces, such as some media players.

The characteristics of this platform are that it is written in Python, it runs on Microsoft Windows and it is event based.

Related to the actions it can implement, there are:

- Media control
- Audio control
- Lighting control
- Keyboard/Mouse emulation

JEEDOM

[9]This is a free open-source software that can be installed on any Linux system. It is compatible with various protocols such as Z-Wave, EnOcean, KNX, Legrand Bus, etc. The plugin system, though Jeedom market, guarantees compatibility with numerous current and future protocols.

It is an autonomous platform; it does not need access to external servers to work. Each user can create their own Jeedom home automation, using widgets, views, and designs. This home automation platform allows users to connect their custom-made devices by creating and interface. Jeedom also includes its own market, which allows users adding home automation features and to assure compatibility with home automation modules.

Jeedom allows to control some functions as:

- Manage the safety of goods and people
- Automate heating for better comfort and energy savings
- View and manage energy use to anticipate expenses and reduce use

- Communicate by voice, SMS, e-mail or mobile applications
- Manage all of the home´s automatic devices: shutters, gate, lights, etc.
- Manage multimedia audio and video devices, and connected objects.

MYCONTROLLER

[10] This is an IoT automation controller for home, office, or any place. It is a very lightweight server, designed to run with limited resources. For instance, it can only run on the first-generation Raspberry Pi.

Its resource requirements are modest. Disk (100 MB, may require more space to store metrics data for many days), memory (256 MB), Java (1.8 version or later). MyController is a Java-based application server. It can run on any platform where Java support is available.

Some characteristics of this platform are:

- The program can be scheduled with time-based operations. It supports a user-defined scripts.
- Metrics data can be stored in SQL database or in influx DB.
- Scripts support (JavaScript, Python, Ruby, Groovy)

On the other hand, MyController was created starting from MySensors. This is an open source hardware and software community focusing on DIY (do-it-yourself) home automation and IoT. This platform is designed for helping people creating sensors and actuators based on these components:

- Arduino
- ESP8266
- Raspberry Pi
- NRF24L01 +
- RFM69

MyController allows the user customizing their own devices by using MySensors.

MySensors provides an open platform for collecting sensor data and help the users building their own wireless sensors and actuators. The user will be able to receive a message when temperature changes, to collect humidity levels, to control LED lights, to control the door, etc.

PIMATIC

[11] This platform is a smart home automation for Raspberry Pi. It defines several home devices and sensors, so that all devices can be controlled uniformly and are presented in a common interface.

Automation tasks can be defined by rules in the form “if this, then that” where the “this” and the “that” can be fully customized by plugins.

Related to devices connection, this home automation platform only supports specific devices that have been already tested by its community. The devices they offer are such as switches, dimmers, shutters, smoke detectors, thermostats and sensors (that control temperature and humidity, doorbell, windows, movement, lights and magnetic switches). This means, the user is not able to integrate custom-made devices, as he/she cannot make his/her own interface.

6. OpenHAB

OpenHab is an opensource home automation hub, developed in Java that allows users to run the system in many devices and depend on the Eclipse Smart Home framework. It has an HTTP server called Jetty, and can work on Linux, Windows, and Mac OSx.

In OpenHab there are more than 2000 integrations. Related to what can be controlled there is a wide range of features as lighting, thermostats, climate control, door locks, garage doors, cameras, security systems, water management, shades/blinds, voice/telephony, audio/video/media, energy management, weather, automobile, etc. Everything that can be searched in the internet can be configured. OpenHab offers the opportunity to install a platform that informs of the weather, the air quality, etc. [12]

6.1. Architecture

To get data into OpenHab it is needed to define an Item which uses the appropriate binding to read/receive/retrieve and transform the data.

First, some terms are going to be explained to understand better the structure.

Things, entities that can be physically added to a system. Things may provide more than one function, for example, they may provide a motion detector and measure room temperature. They do not have to be physical devices; they can also represent a web service or any other manageable source of information and functionality. Things expose their capabilities through **Channels**. When the user configures the system, does not need to use every capability offered by a Thing. It can be found out what Channels are available for a Thing.

Bindings, can be thought as software adapters, making Things available to the home automation system. They are add-ons that provide a way to link items to physical devices.

Items, represent capabilities that can be used by applications.

The glue between Things and Items are Links. A link is an association between exactly one Channel and one Item.

Related to the communication, OpenHab has two different internal communication channels:

- The Event Bus: is the base service of OpenHab. All bundles that do not require stateful behaviour should use it to inform other bundles about events and to be updated by other bundles on external events. There are two types of events. The first one, commands which trigger an action or a state change of some device; the second one, status updated which inform about status change of some device (often as a response to a command). All protocol bindings (which provide the link to the real hardware devices) should communicate via Event Bus.
- Item Repository: this is connected to the Event Bus and keeps track of the current status of all items. This can be used whenever need the current state of items/devices.

The following diagram shows how these communication channels are used:

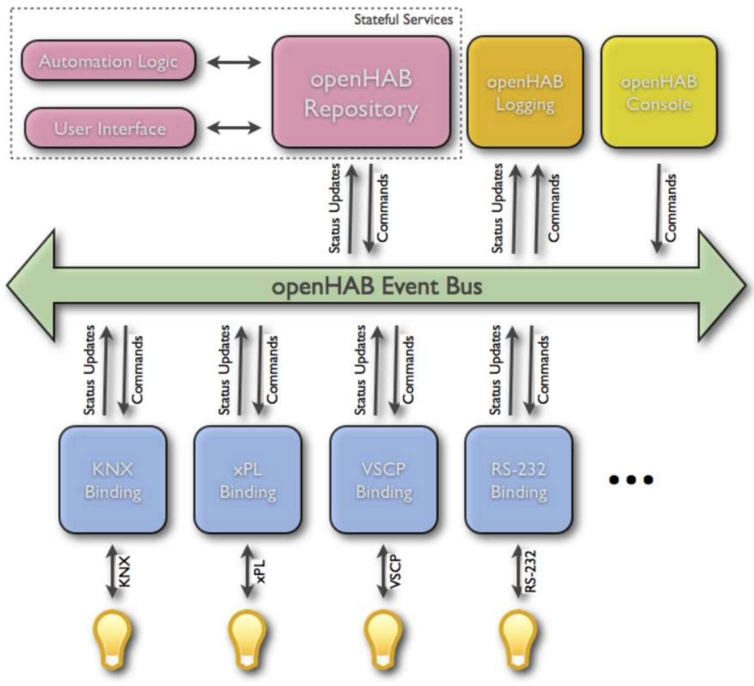

Figure 3: OpenHAB architecture

6.2. Light example

The example can be implemented with a two-cannel actuator that controls two lights. The actuator is a Thing that might be installed in an electrical cabinet. It has a physical address and it must be configured to be used. In order for the user to control the two lights, he/she access the capability of the actuator Thing (turning on and off two separate lights) through two Channels that are Linked to two switch Items presented to the user through a user interface.

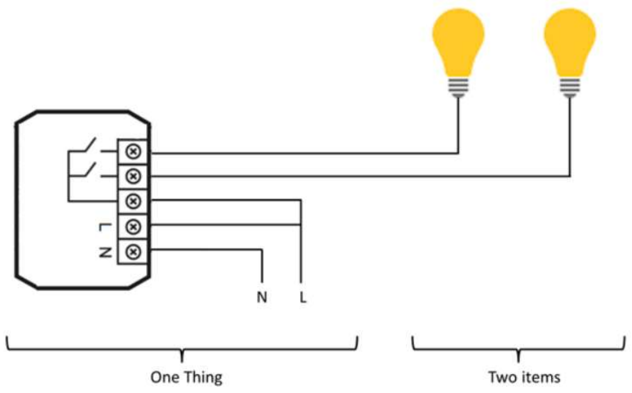

Figure 4: Light Control

7. Home Assistant

[13] Home Assistant offers a large list of integrations, there are 1611. Between all these possibilities there are several software interfaces, and a variety of devices that can be connected. Related to the devices or functions that can be controlled, they are: alarm, light bulbs, HVAC system, energy consumption, fan devices, temperature, water heater, lock, covers, car, doorbell and everything that can be accessed via Internet, as time and date, weather, etc.

7.1. Data flow

The Home Assistant core is event driven. This means that everything that happens is represented as an event: a light being turned on; a motion sensor being tripped, or an automation triggered. Each event has an attached context. The context can be used to identify which events have been triggered as a response to other events.

Events

The Home Assistant core is using a couple of built-in events to wire all the pieces together. Each event has a type and data.

- state_changed: the event will contain the entity identifier and both the new and old state of the entity. If old state is not present, it means the state was set for the first time. If new state is not present, it means the state has been removed.
- automation_triggered: An automation will trigger an action that can result in more changes being made to the system. The resulting changes can be tracked because all related events will share the same context as this event.
- script_started: this event is fired when a script is run. A script can be invoked by a user or triggered by an automation. The resulting changes can be tracked because all related events will share the same context as this event.
- service_registered: this event is fired when a new service is registered
- home_assistant_start, home_assistant_stop: these events are fired when Home Assistant starts and stops. There is no other data attached to it.

All events are stored in the database. That information can be used to figure out what happened, when, and how it is related to other events.

States

An IoT device or data from a service is represented as one or more entities in Home Assistant. An entity in the core is represented as a state. Each state has an identifier for the entity in the format of <domain>. <object_id>, a state and attributes that further describe the state. An example of this would be "light. Kitchen" with the state on and attributes describing the colour and the brightness.

The <domain> part of an entity identifier is equal to the Home Assistant component that is maintaining the state. This domain can be used to figure out what kind of state attributes to expect.

The difference between "last_changed" and last_updated is that last_changed only updates when the state value was changed while last_updated is updated on every state change. For example, if a light turns on, the state changes from off to on, so both last_updated and last_changed will update. But if a light changes colour from red to blue, only the state attributes change.

Services

A service can be used to control one of the entities of that component or it can be used to call an external script or service. A service is identified by a domain, which is equal to the component offering the service, and a name. Each service can take optional service data object to indicate what to control. An example of a service is light. turn_on with service data {"entity_id": "light.kitchen"}.

Context

Context is used to tie events and states together in Home Assistant. Whenever an automation or user interaction triggers a new change, a new context is assigned. This context will be attached to all events and states that happen as result of the change.

In the following example, all events and states will share the same context:

- Paulus arrives home, which updates device_tracker.paulus_pixel from not_home to home
- The automation "Paulus is home" is triggered and fires automation_triggered event.
- The automation calls service light. turn_on, which fires the service_call event.
- The light.turn_on service turns on the light which causes an update to the state of light.living_room.

Data flow within Home Assistant is shown in figure below.

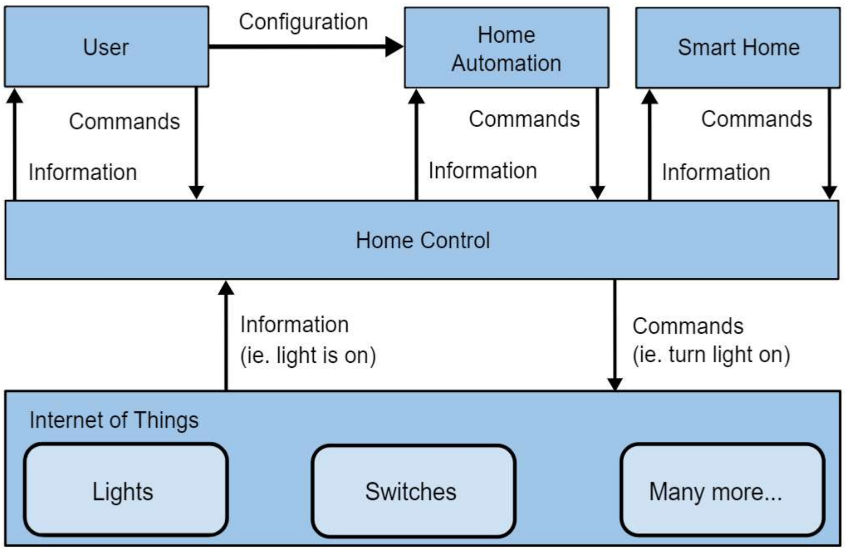

Figure 5: Home Assistant data flow

Home Assistant architecture is shown in the figure below, where the components are:

- Home Control: is responsible for collecting information and controlling devices.
- Home Automation triggers: commands based on user configurations.
- Smart Home triggers: commands based on previous behaviour.

7.2. Architecture

[14] The Home Assistant core is event driven, this means that everything that happens is represented as an event, for example, a light being turned on. Each event has an attached context. All events are stored in the database. That information can be used to figure out what happened, when, and how it is related to other events.

The Home Assistant core is responsible for Home Control. It is based on four principal components:

- Event Bus: facilitates the firing and listening of events -- the beating heart of Home Assistant.
- State Machine: keeps track of the states of things and fires a state_changed event when a state has been changed.
- Service Registry: listens on the event bus for call_service events and allows other code to register services.
- Timer: sends a time_changed event every 1 second on the event bus.

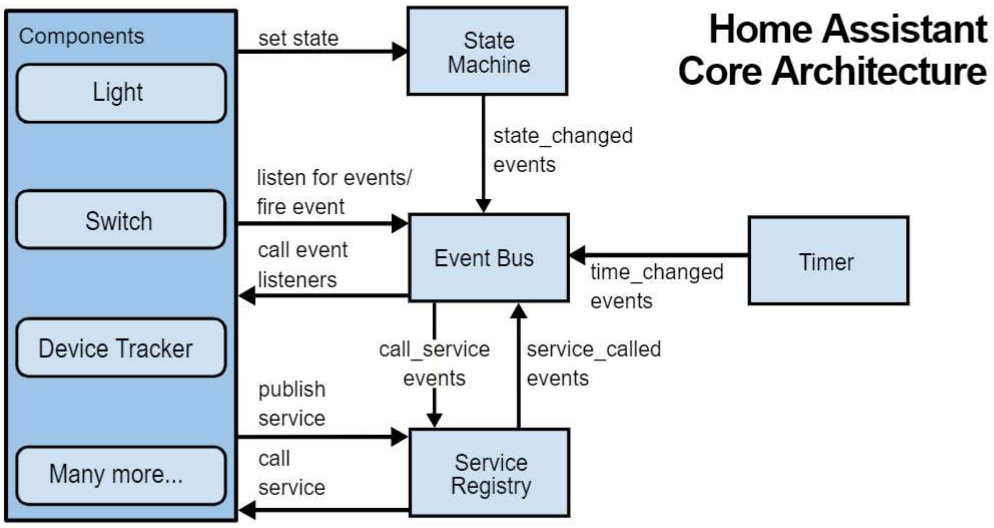

Figure 6: Home Assistant core

7.3. Components of architecture

Home Assistant can be extended with components. Each component is responsible for a specific domain within Home Assistant. Components can listen for or trigger events, offer services, and maintain states. Components are written in Python and can do all the possibilities that Python has to offer.

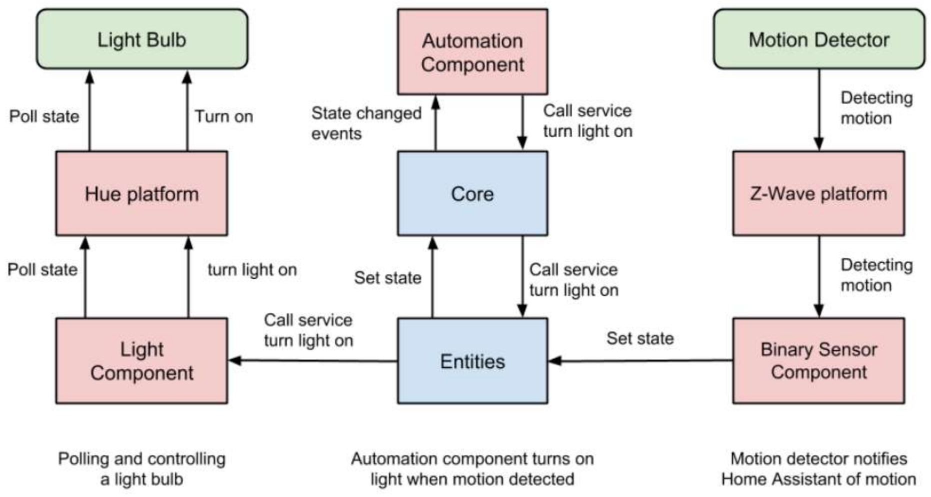

Figure 7: Home Assistant Components

There are two types of components within Home Assistant: components that interact with an Internet of Things domain, and components that respond to events that happen within Home Assistant.

Components that interact with an Internet-of-Things domain

These components track devices within a specific domain and consist of a core part and platform-specific logic. These components make their information available via the State Machine and the Event Bus. The components also register services in the Service Registry to expose control of the devices.

Components that respond to events that happen within Home Assistant

These components provide small pieces of home automation logic or involve services that do common tasks within your house.

7.4. Integrations: Energy and Climate control

Home Assistant includes 1604 different integrations to automate. Each component allows the user to control different equipment. For controlling HVAC (heating, ventilating, and air conditioning) systems and house temperature there two components that explain how this can be done and what the user would need. These are called “Generic Thermostat platform” and “climate”, the first one offers the commands to implement a thermostat, and the possibility to the user to customize all functions.

“Generic Thermostat platform”: it is a thermostat implemented in Home Assistant. It uses a sensor and a switch connected to a heater or air conditioning under the hood. When in heater mode, if the measured temperature is cooler than the target temperature, the heater will be turned on and turned off when the required temperature is reached. When in air conditioning mode, if the measured temperature is hotter than the target temperature, the air conditioning will be turned on and turned off when required temperature is reached. One Generic Thermostat entity can only control one switch. If the user needs to activate two switches, one for a heater and one for an air conditioner, he/she will need two Generic Thermostat entities.
The variables can be configured in the code.
In the case of thermostat, it can be a generic one. In case of a heater, for heater switch, it has to be a toggle device. To target any sensor, its entity has to be written correctly in the code, the temperature we want to reach has to be fixed, also the maximum/minimum temperature, and an initial target temperature.
Related to the AC, it can be treated as a cooling device instead of a heating device specifying it in the programming code. Also, we can set an amount of time that the switch specified in the heater option must be in its current state prior to being switched either off or on.
Also, the user can set a hot or cold tolerance, this means setting a minimum amount of difference between the temperature read by the sensor and the target temperature that must be changed.
The user also can set an initial HVAC mode, this is valid initial values as “off”, “heat” or “cool” for the devices.
Finally, any HVAC device can be programmed to be turned on in any minute/hour/day that the user wants.

“Climate”: integration that allows the user to control and monitor HVAC devices and thermostats.

Functions can be customized, this means, if the user wants to turn auxiliary heater on/off for climate device, we can set a list that define the entity of climate device to control. Also, the same action can target all climate devices.

For example, any device can be turned on/off at any time specifically, but it can also be controlled depending on other parameters.

Also, a "vacation mode" can be configured. This means, the temperature can be changed when the device is set to save energy.

For temperature, it also optional to set the highest/lowest temperature that the climate device will allow.

This integration also allows to set target humidity of climate device, set swing operation mode for climate device, set climate device´s HVAC mode, turn climate device on (only supported if the climate device supports being turned off).

8. Comparison of OpenHAB with Home Assistant

[15] OpenHAB has been created in 2010 by Kai Kreuzer. It is developed in Java and mainly based on the Eclipse Smart Home framework. OpenHAB is a modular software that can be extended through "Add-ons". Add-ons give OpenHAB a wide array of capabilities, from User Interfaces to the ability to interact with a large and growing number of physical Things.

OpenHAB runs on many popular platforms including Linux, Windows and Mac OSx. Many users find that the simplest way to experiment with OpenHAB is to get a Raspberry Pi and install OpenHABian. While OpenHABian offers a streamlined and simplified way to get up and running quickly, it is a complete OpenHAB home automation system capable of automating the entire home.

<u>OpenHAB</u>

Benefits	Drawbacks
Setup and installation procedure are easy	Raspberry Pi 3 or older version may cause problems due to limited RAM
Automation rules can be customizable	The uptadet topics published per week is less than in Home Assistant
Custom-made devices can be connected	
Extended user interface	
The online community gets around 200 topics in 7 categories in each week	
383 add-ons and 2004 things	
Supports both Alexa and Google Assistant	

Table 2: OpenHAB benefits and drawbacks

<u>Home Assistant</u>

Benefits	Drawbacks
It does not have to store data in cloud which ensures user privacy	Based on the use of YAML, which can be difficult to use.
Step by step instruction for setup in website	It can be difficult to use and comprehend for new users.

Automation rules allow users to customize as they like	
Custom-made devices can be connected	
All codes can be found in its forum, Github	
Supports both Alexa and Google Assistant	
It is more stable in terms of notifications	
Around 500 topics published in 10 categories per week	

Table 3: Home Assistant benefits and drawbacks

Home Assistant offers greater flexibility than OpenHAB, better user interface and has a more flexible platform to control the smart devices

Related to supported devices and pairing both systems can also work with main IoT protocols such as Bluetooth, Z-Wave or Zigbee. Home Assistant is compatible with around 1611 devices, and OpenHAB with 2004.

9. House temperature control within Home Assistant

Controlling a house by an open source platform has several benefits as it has been mentioned, lights, boiler, alarm HVAC systems, etc. can be controlled and optimized. HVAC systems is responsible for the 46 percent of the energy consumption of a house, in average, so controlling it can help to reduce that waste.

An open source home platform offers different ways to control devices, in particular Home Assistant, allows any user to control specific smart devices or even his/her custom-made devices. There are several functions that can be controlled as light, temperature, humidity, alarm, etc. To control light for example, Home Assistant offers many options of smart lights brands that can be easily connected. The same happens with the HVAC systems in order to control a room temperature.

Between all 1604 possible integrations, they are all compatible with Home Assistant and can communicate, send and receive data, just by connecting them. Home Assistant has its own API (Application Programming Interface) already created between the platform and each device manufacturer to allow communication. In these cases, if the user wants to control house temperature, for example, would have to choose any smart thermostat between all possibilities offered and connect it as the platform indicates. Then, the control will be done automatically.

On the other hand, one of the possibilities that Home Assistant offers, is to create an interface to control custom-made devices, or any device that is not on that large list of integrations. This process of customizing devices must ensure that both hardware and software of the device are compatible with platform. For that, it is needed a software implementation, so the equipment that is being customized can communicate with the platform. Related to the hardware implementation, it must include sensors that already have an interface with Home Assistant. Between those 1604 integrations, Home Assistant includes different types of brokers to create an interface, as MQTT; or devices for hardware implementation as Z-Wave items.

9.1. MQTT

[16] The MQTT protocol is one of the most important items of IoT due to its light weight and simplicity. These are important aspects as most devices from IoT have limitations related to power and consumption. MQTT stand for MQ Telemetry Transport. The design principles are to minimise network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. It has been used in sensors communicating to a broker via satellite link, over occasional dial-up connections with healthcare providers, and in a range of home automation and small device scenarios. It is also ideal for mobile applications because of its small size, low power usage and efficient distribution of information to one or many receivers.

The broker acts as a post office, MQTT does not use the address of the intended recipient but uses the subject line called “Topic”, and anyone who wants a copy of that message will subscribe to that topic. Multiple clients can receive the message from a single broker (one to many capability). Similarly, multiple publishers can publish topics to a single subscriber (many to one).

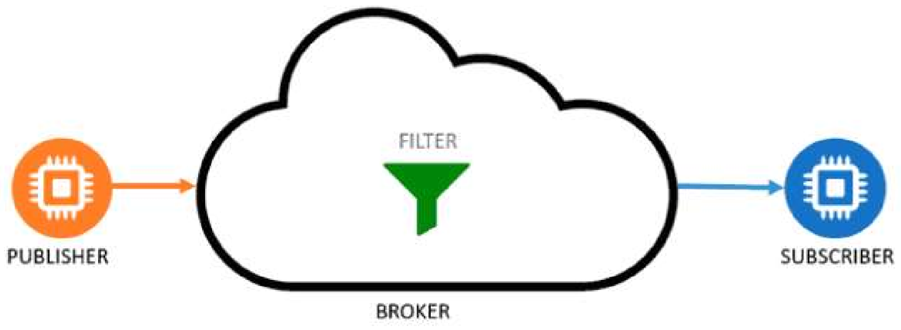

Figure 8: Broker scheme

Configuration within Home Assistant

The MQTT integration that is compatible with Home Assistant, allows any user creating his/her own interface between his/her custom-made device and the platform, because to create an API first is needed a broker.

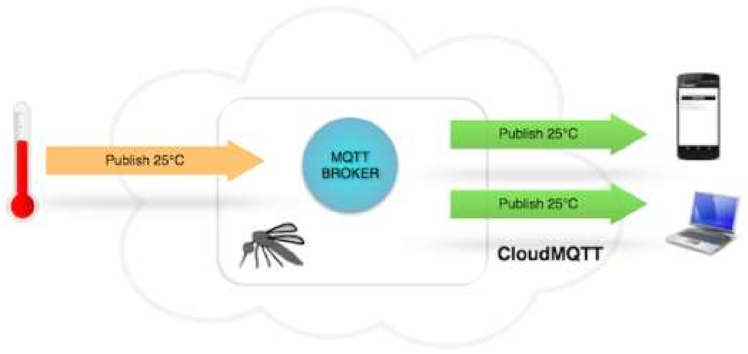

Figure 9: MQTT broker architecture [17]

The MQTT options compatible with Home Assistant are:

- Mosquitto broker: it is an open-source (EPL/EDL licensed) message broker that implements the MQTT protocol. Mosquitto is a lightweight and is suitable for use on all devices low power single board computers to full servers. There is the option "run your own" which is the most private one. The recommended setup method is to use the Mosquito MQTT broker add-on.

To install it in Home Assistant, some variables will be required, as the IP address or hostname of the MQTT broker, the network port to connect to, the client ID that Home Assistant will use (has to be unique on the server), the time in seconds between sending keep alive messages for the client and an username and a password.

The second Mosquitto broker option is the public broker. There is no privacy as all messages are public which is usually recommended to be used only for testing purposed and not for real tracking.

- CloudMQTT: it is a hosted private MQTT instance that is free for up to 10 connected devices. There is another option of charge per month, that allow the user connecting 1000 devices. The thing is, Home Assistant is not affiliated with CloudMQTT nor will receive any kickbacks.

9.2. Z-Wave

[18] Z-Wave is a wireless communication protocol designed for home automation. It is based on advances techniques of security to guarantee that data transmission is not interrupted by any cybercriminal. It allows the user to create a mesh network communicating with each other multitude of devices, there is a limit of 232 devices in a single Z-Wave network. It is similar to the Wi-Fi network, but the difference is that Z-Wave sends and receives an amount of data smaller, such as “turn on the light” “reduce the room temperature” or “close the door”. This allows operating on lower frequencies and with lower power, which avoids using common frequencies that are saturated.

Z-Wave integration for Home Assistant allows to observe and control connected Z-Wave devices. Z-Wave support requires a supported Z-Wave USB stick or module to be plugged into the host. There is support for climate, covers, lights, locks, sensors, switches, and thermostats.

Related to be connected, it is needed a controller, which configures the Z-wave integration, and some devices added using the control panel. The controller is a compatible Z-Wave stick or module, it must be a static controller. The USB sticks that are confirmed to work are:

- Aeotec Z-Stick Series 5
- Everspring USB stick - Gen 5
- GoControl HUSBZB-1 stick

- Sigma Designs UZB stick
- Vision USB stick - Gen5
- Zooz Z-Wave Plus S2 stick ZST10
- ZWave.me Razberry Board
- ZWave.me UZB1 stick

9.3. Customizing a smart thermostat

[19] Home Assistant includes the opportunity to build custom-made equipment by configuring both software and hardware to be compatible.

The need begins in a house where there are only two AC units that are not smart, and the time they take to cool an area is very long. Also, with non-smart devices is harder to save energy.

The units are like it is shown in *figure 10.* There are two constraints as these units are mounted in the walls, and they do not include their documentation. The apartment in this case does not counter with any sensor nor network appliances. These AC devices do not have a thermostat included to measure room temperature, so it is necessary to build one. Also, they only have on/off switches. On the other hand, as each AC unit is in different room of the apartment, since they have different sizes, both devices have to be controlled separately, for the main purpose of saving energy. This is going to be done in two different ways.

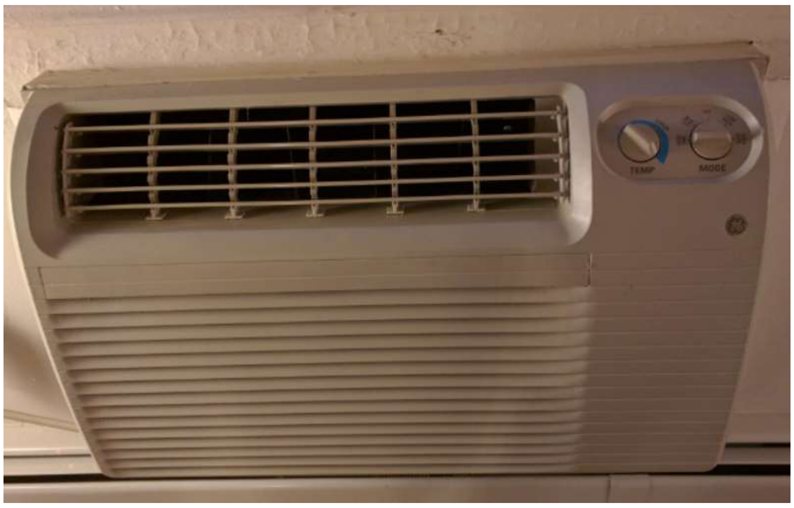

Figure 10: AC device

As it is important to minimize unnecessary power consumption, when nobody is at home the devices are turned off, but a way to control them remotely is going to be implemented in order to turn them on some time before somebody arrives at the house. To automate them, there are two processes, firstly the hardware and secondly the software. Also, cycle times are studied to have both AC units as efficient as possible.

Hardware

To control remotely the AC units, a first inclination was based on tapping into the existing controls with a microcontroller to access over the home network. The AC has only analog controls, a rotary selector switch for the mode and a potentiometer for fan speed, so it is easy to tap into that. The problem with this, is that the AC units are included in the apartment, so taking them apart is not a good idea compared to solder wires into them.

The second idea intended was to implement a form of robotic control to turn the dials on the AC unit. While this would have been a good option, it did not feel like workable. The knobs can break easily, so having something automatically turning them on/off can make it last little.

So, the last option for hardware implementation, is to control the power to the AC units instead of controlling the AC dials. The AC units can be in the on state the full time and just turning on the power when needed. For that, it is necessary a relay to control it. In addition, there are two constraints, as the AC units are mounted in the walls, a wireless control is needed. On the other hand, to configure any device within Home Assistant, the relay for the load of the AC unit has to be rated, as the devices do not include it´s model number, it will be measured with a clamp meter.

The next step is to select a device to control remotely power switches. There are many different options on the market for Internet of Things wireless communication protocols, but the most popular choices are Z-Wave and ZigBee. Both are included in the large list of integrations of Home Assistant, (it means both are compatible with the platform) so once any of them would be connected to the AC units, it would become easy to communicate and send data to the platform.

In this case Z-Wave devices have been chosen. They have a consistent ecosystem. For the hardware control it is necessary to have a USB Z-Wave controller and two Z-Wave enables power switches, one for each AC unit. Also, a temperature sensor is needed. As the control is done with Z-Wave devices the sensor is a Z-Wave Multisensor 6. It includes a variety of sensors, (including temperature) in a small device. This is a wireless and portable sensor that can be placed anywhere. From this point, begins another quandary about where to locate it. The final decision was to locate it far away from the windows so that the temperature does not depend too much of the outside temperature. The problem at this point, is that the single temperature sensor located in the living room will control both bedroom and living room AC units.

As it has been mentioned, the AC units are the same, but the bedroom and the living room where they are located, have different sizes. This means, that for the main purpose of saving energy, they should be controlled separately. So, all the hardware components mentioned before are used for the AC unit located at the living room.

For controlling the bedroom temperature, one option would be including another Z-Wave sensor or including through Raspberry Pi 2. In this case, instead of using wireless control, it is done by wiring a couple of temperature sensors into it and using it as the data source for the bedroom. Related to the temperature sensors, DS18B20, a Dallas 1-Wire sensor is used, which is extremely popular with the Raspberry Pi crowd.

The advantage of connecting 1-Wire sensor is that everything can be connected in parallel. All the devices are individually addressed, so the daughterboard just wires the two sensors in parallel, with a pull-up resistor. With things connected to the Raspberry Pi 2, there was already an interface to locally poll the temperature in the bedroom, but it was still needed a good interface for getting data into Home Assistant, which runs a different machine.

This point led to average the MQTT infrastructure, that can be installed within Home Assistant. This allows to periodically publish results from an arbitrary number of Dallas 1-Wire sensors, and also could work with other classes of sensors in the future. The new temperature sensor is added using MQTT sensor component. (It is a platform that uses the MQTT message payload as the sensor value. In this case, the sensor will receive an instant update with last known value, otherwise, the initial state will be defined).

As it has been explained in MQTT, point a broker is needed. In this case it is used the Mosquitto broker. As it is shown in the figure below, there are some sensors connected to the AC units and send data to the MQTT broker, that transforms that data to be understandable by Home Assistant, so the platform can manage that information.

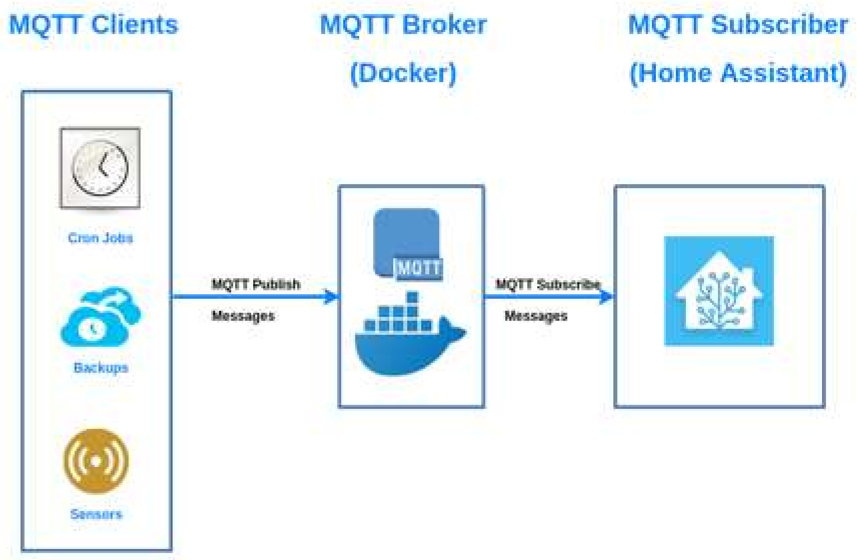

Figure 11: Relation MQTT with Home Assistant

Software

Once the devices are configured and the hardware is under control, it is needed a software to use the hardware acquisitions as a thermostat. In this point, configuring it with Home Assistant offers a variety of possibilities. Home Assistant supports a ton of different devices and services so other kind of items can be used at the same time. As for example, controlling a person´s location, or the outside temperature in order to monitor the AC devices based on them. The first point then, is to install Home Assistant and to configure its interface with the devices.

Related to the thermostat component, Home Assistant already has two components implemented for basically the exact use needed. These are called “climate” and “Generic Thermostat Platform”. Within these integrations the platform allows controlling HVAC system and any thermostat. Home Assistant provides an automation interface that enables to write rules that execute on a particular trigger from any device. The following programming code shows how the thermostat would be configured in Home Assistant, to control one of the AC devices. Both AC devices would be configured the same, but for the name of each sensor.

```
climate:
  - platform: generic_thermostat
    name: house_thermostat
```

```
        heater: false
        target_sensor: sensor.Multisensor6
        min_temp: 15
        max_temp: 21
        ac_mode: switch.house_AC
        target_temp: 17
        cold_tolerance: 0.3
        hot_tolerance: 0
        min_cycle_duration:
          seconds: 5
        keep_alive:
          minutes: 3
        initial_hvac_mode: "off"
        away_temp: 16
        precision: 0.1
```

When in heater is written “false” is because what is being configured is the cooling device. Values for min_cycle_duration and keep_alive are default values but these will be changed in order to make the cycle time of the AC device longer. The cold_tolerance allows not being changing each time the device reaches the desired temperature. If the target temperature is 17 and the cold_tolerance 0.3, the AC device will stop when the sensor equals or goes below 16.7. Also, initial_hvac_mode is set to “off” for security to help align any discrepancies between the thermostat and cooling state. [20]

As the main purpose is to save energy, it is of great interest to configure the cooling devices depending on the user´s position. Home Assistant can read location data, sent from the mobile phone, for example, using another component/integration called OwnTracks. This is an open-source and uses open protocols for communication, so the data stays secure and private. OwnTracks uses MQTT, and as it is compatible with Home Assistant. To install it, first the user must configure it in his/her mobile, by setting these parameters:

- Mode: HTTP
- URL: URL given when setting the integration
- Turn on authentication

- User ID: name

Full example for configuration is shown below:

```
owntracks:
  max_gps_accuracy: 200
  waypoints: true
  mqtt_topic: "owntracks/#"
  events_only: true
  waypoint_whitelist:
    - Alicia
    - Jon
```

Where, max_gps_accurancy is the distance in meter that the application can read away from home. This a security measure, so if the user is kilometres away from his/her home, location will not be read. Waypoints is to configure zone definitions, if the user do not want to configure, the default value is set to “true”. Waypoint_whitelist, are the names of the user´s location that the application is going to read. [21]

On the other hand, Home Assistant have access to the outside temperature, by using the integration called “Weather Underground”. In this case the AC will be turned off if the outside temperature is below 22ºC. At that temperature, to save energy and power consumption the window should be opened instead of running the AC. Also, the program will push notifications to remind to open a window.

To configure it, it is needed the following code:

```
sensor:
  - platform: wunderground
    api_key: 12345
    monitored_conditions:
      - alerts
      - dewpoint_c
      -temp_c
      -temp_high_avg_c
```

Monitored_conditions are conditions to display in the frontend. Where alerts are current severe weather advisories, and dewpoint_c is temperature in Celisius which water droplets beging to condense and dew can form. Temp_c is current temperature in Celisius, and temp_high_avg_c is the average high for a day in Celsius. The program will give information each 5 minutes. [22]

Cycle times

The AC units was being short cycling, because as soon as the room hit the desire temperature, the program would immediately power the switch. This means, the room temperature would increase, and some minutes later, the AC unit would be turned on again. This cycle would be repeating continuously. This turning on and turning off, determines a cycle of the air conditioner. The cycle time of an air conditioner is the amount of time the unit runs to maintain the temperature of the room equal to the temperature setting of the thermostat. If the thermostat is set a very low temperature as compared to the room temperature, the compressor of the AC unit must work for a longer time to decrease the room temperature, thus, increasing the cycle time. In other words, greater the temperature difference between the room temperature and thermostat temperature, greater will be the cycle time and vice versa. Additionally, the factors affected by cycle time are power consumption, humidity, efficiency, and comfort. An air conditioner consumes a lot of power every time it is turned on. This power is much greater than the power consumed by air conditioner for continuous operation, (long cycles). On the other hand, an air conditioner´s comfort level depends on its humidity removal capability. This capability improves as the cycle time increases. An air conditioner with frequent on/off cycles cannot remove humidity from the room. Those frequent short cycles affect the efficiency of the air conditioner. In addition, HVAC systems, are most efficient at steady state and much less efficient the first few minutes after they have been turned on. [23]

At this point, the configuration done above has to be reconfigured, to solve this it is needed to add hysteresis to the system, so it is not constantly cycling the units. One way a thermostat does this, is by adding a bit of hysteresis around the set temperature, so instead of switching at exactly the set temperature, it waits until it passes it by a couple of degrees. Another way is to set a defined maximum switching frequency, which sets an upper bound on how frequently the unit will cycle on/off in a room.

There is no specific ideal cycle time for air conditioner as it completely depends on its cooling capacity and the room temperature and humidity. Typically, 10-20 minutes on cycle time should be good enough.

```
climate:
- platform: generic_thermostat
name: house_thermostat
heater: false
target_sensor: sensor.Multisensor6
min_temp: 15
max_temp: 21
ac_mode: switch.house_AC
target_temp: 17
  cold_tolerance: 0.5
  hot_tolerance: 0
  min_cycle_duration:
    minutes: 10
  keep_alive:
    minutes: 5
  initial_hvac_mode: "off"
  away_temp: 16
  precision: 0.1
```

The minimum cycle duration takes 10 minutes, so that the AC units are not turning on/off every few minutes. Also, the keep_alive call is done with the current valid climate integration state.

9.4. Tado

Tado is a brand of smart thermostats for controlling temperature at any house. As it is part of the Home Assistant´s integrations, the Tado manufacturer has an interface with the platform, so the communication is done automatically after it has been connected.

It ensures a comfortable and healthy climate while saving up to 31% on the heating bill. It can heat or cool an area, and can detect presence, so if the user has left the house the device will turn off and in the same way, if the user is getting home it will be turned on. This device can also notice if a window is open and lets turning off the heating or air conditioner. Tado adapts to the behaviour of the user to ensure the optimal room climate and it can also have access to the boiler.

On the other hand, the most significant feature is the location-based-control function, which allows Tado to turn the heating down or even off depending on the user´s location. It is the most compatible solution for climate control in Europe and is able to efficiently regulate almost every heating system.

In order to reduce the energy bill, it is very efficient because every time it is not needed to have the heating turned on, it will switch down the devices. It also offers an energy saving report, which provides a breakdown of the ways that Tado helped the user saving over the previous month. Also, it can save energy even when someone is always home because the user can create blocks in the smart schedule and control the rooms individually. Tado uses the local weather forecast and the characteristics of the user´s building to determine to heat efficiently. This benefits residential houses or larger apartments.

<u>Heat control</u>

The first step before installing Tado devices to control heating and to save energy is to find the right Tado setup for the house. For that, it is necessary to evaluate if the heating existing system of the house is compatible with Tado devices. The only requirement for Tado to work with the boilers is that it communicates in a language that Tado can understand and process. The most common ways in which thermostats communicate with boilers are through simple relay on/off connections or through digital communication systems. Tado can control hot water only in boilers that are digitally controlled. Secondly, there are two ways for controlling heating, using a smart centralized thermostat, or using a smart radiator thermostat. The centralized thermostat works with own heating system in a single-family house, with gas heating, or underfloor heating in single-family homes or apartments. The radiator thermostat works with radiators in single-family homes and apartments, regardless of the heat source, for example, gas or oil burner, heat pump or district heating.

i. Two ways boilers can communicate with Tado
   - Relay (on/off) heating systems

A relay on/off connection works like a normal electric switch. When there is a call for heat, the electric circuit closes to switch on heating. And when the desired temperature is reached, the circuit opens to switch off the heating. Tado is compatible with almost all boilers that support relay connections.

- Digital heating systems

  There are digital systems that can modulate the amount of heat supplied by the boiler. If there is a request for a minor increase in temperature, the room thermostat can power the boiler accordingly to meet that specific request rather than firing at full capacity.

  Tado is compatible with OpenTherm boilers. A list of brands that Tado is compatible with is given below.

Brands	Compatible	with	Tado		
ACV	Baxi	Chaffoteaux	Frisquet	Ideal	Potterton
Alpha	Beretta	Chappee	Geminox	Immergas	Protherm
Ariston	Bosch	De Dietrich	Glow-worm	Intergas	Radson
ATAG	Brötje	Elco	Heatlipe	Junkers	Rapido
Atlantic	Buderus	Ferroli	Hermann	Nefiy	Remeha
AWB	Bulex	Firebird	Hoval	Oertlj	Rjello
Saunier Duval	Unical	Vaillant	Viessmann	Wolf	Worcester

Table 4: Brands compatible with Tado

ii. Tado devices options for controlling heating

- Smart centralized thermostat

  This device can control the heating of the house. This thermostat will measure the temperature of the room where it is located, (main room) and will set the temperature of the other rooms depending on that main room. As it is shown in the figure it is connected via Wi-Fi to the Internet, and it sets any desired temperature in the house. As temperature is controlled at once, there are some advantages and some drawbacks. If the user decides just to have this thermostat will save less energy, than having the temperature of the rooms controlled one by one. Sometimes in a house there are rooms that tend to be colder than others, so if the thermostat measures the temperature in one room and sets the same for the rest of the rooms there will be a waste of energy. Anyway, having a smart thermostat saves more energy than having a non-smart one, because it notifies when a window is open, or when anyone is at home to turn the heating off.

If the presence detection is required to be controlled, it is necessary to use the Tado smart thermostat and its support for person presence detection based on smartphone location by geofencing. This tracker uses the Tado API to determine if a mobile device is at home.

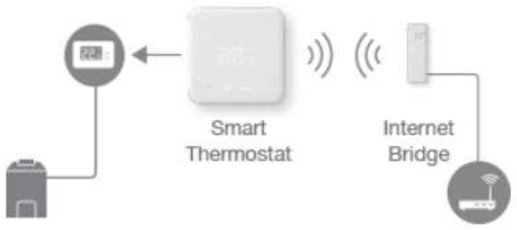

Figure 12: Tado Smart centralized Thermostat

- Smart radiator thermostat

  This device allows the user controlling each room separately as it goes connected to each radiator. By this way, the most save of energy is done. As it is shown in the figure it also goes connected to the internet via Wi-Fi. As it has been mentioned, this device works with radiators in single-family homes and apartments, regardless of the heat source, for example, gas or oil burner, heat pump or district heating. The only drawback having this system is the higher price rather than having just one smart device. The radiator thermostat does not detect person presence so the temperature cannot be controlled depending on the user location.

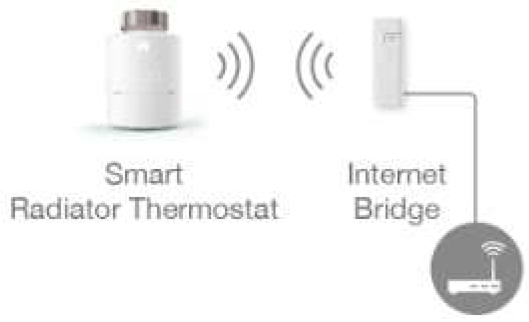

Figure 13: Tado Smart Radiator Thermostat

Air conditioner control

The Tado smart AC control device, works with air conditioners with an infrared remote control that displays the air conditioner´s current settings (for example mode, target temperature, and fan speed), be it a split, multi-split, in-window or portable unit from all manufacturers. It is connected to the internet via the home´s Wi-Fi and controls the air conditioner via IR.

Installation within Home Assistant

First, it is needed to configure Tado devices.

1. Register the internet bridge. The internet bridge connects the Tado devices to the internet and register the smart thermostat
2. Select the independent installation of Tado with individual step-by-step.
3. Select the heating setup, based on the heaters currently available. This way Tado can tailor the installation instructions for the user´s needs.

In addition, to configure it with HA, it is necessary to add to the configuration.yaml file in HA. [24] [25]

```
# Example configuration.yaml entry for Tado

device_tracker:
  - platform: tado
    username: Alicia_Sanz
    password: 1234
    home_id: 13579
```

10. Economic assessment

A comparison of the economic cost of each way for controlling house temperature is shown on the following tables.

Tado

There are two different ways for controlling house temperature, using a smart centralized thermostat, or using smart radiator thermostat.

Heating	Price (€)
Smart centralized thermostat	199,99
Smart radiator thermostat	129,99

Table 5: Tado heating prices

Cooling	Price (€)
Smart air conditioning	99,99

Table 6: Tado cooling price

Custom-made thermostat

To control by building a smart thermostat, as it has been explained there are two ways, the first one using Z-Wave controller, and the second one using Raspberry Pi.

Components	Price (€)
USB Z-Wave controller	37,46
Z-Wave power switch	45,90

Z-Wave Multisensor	62,51
Total	145,87

Table 7: Custom-made AC prices, Z-Wave controller

Components	Price (€)
Raspberry Pi 2 protoboard	54,25
Z-Wave power switch	45,90
Z-Wave Multisensor	62,51
Total	162,66

Table 8: Custom made AC prices, Raspberry Pi 2

In order to compare both ways previously explained of optimizing a house temperature, two temperature rooms of a house are going to be controlled separately, for a better optimization, as usually each room of a house has different size, or due to its orientation, gets colder/hotter easily than the other.

Using Tado devices would be necessary to use the smart centralized thermostat in the main room (Room1) to stablish a temperature for that area but to control also other spaces in the house. The smart thermostat is needed because is the only Tado device that allows configuring depending on the user´s position. On the other hand, for controlling the second room temperature separately it is needed a smart radiator thermostat, which will control only that room temperature.

This process can now be compared to the house temperature control based on the use of a custom-made AC, where each device is controlled separately by two different ways (one using Z-Wave protocol and the second one using Raspberry Pi 2). As the control made with Z-Wave is less expensive than the control with Raspberry Pi, the comparison is made with the first implementation.

	Tado	Custom-made AC
Room 1	Smart centralized thermostat: 129,99€	Z-Wave control: 145,87€
Room 2	Smart radiator thermostat: 99,99 €	Z-Wave control: 145,87€
Total	229,98€	291,74€

Table 9: Comparison of Tado with Z-Wave controller

Conclusion

The idea of smart homes is changing, and they are no longer being only for comfort, but for energy saving. The idea to control a smart house by an open source smart home platform is because they allow any user to connect any smart device under the same control unit, rather than having each device controlled separately.

Open source smart home platforms allow controlling and optimizing a house power consumption, due to the fact that HVAC systems can be programmed to work only when needed, and also they offer to the user comfort, as they can be adjusted to user´s position, weather, etc. In addition, platforms that allow connecting custom-made devices widens the range of possibilities for house control, as Home Assistant.

Home Assistant is one of the biggest open source smart home platform, including 1611 integrations, and a big community of users developing continuously codes to program those integrations. This platform offers MQTT as communication protocol to create an interface for connecting custom-made devices, and Z-Wave wireless communication protocol to implement the hardware of devices.

On the other hand, after studying how to control a house temperature within Home Assistant, by two different process, it has been demonstrated that it is more economically profitable to use Tado devices than to customize an Air Conditioner. In addition, Tado allows reducing up to 31% in heating bill, when customizing there is not any exact guarantee of saving, it depends on each custom-made device, in taking advantage of its cycle durations and in its power consumption. On the other hand, controlling any custom-made device offers more possibilities of control, due to the opportunity of connecting any other device the user wants.

References

[1] "Direct Energy," 2020. [Online]. Available: https://www.directenergy.com/learning-center/average-electric-bill.

[2] M. Gracia, "IoT-Internet of Things," *Deloitte.*

[3] T. Nadu, Internet of Things Based Architecture of Web and Smart Home Interface Using GSM, International Journey of Innovative Research in Science, Engineering and Technology, March 2014.

[4] M. C. V. Ian F. Akyildiz, Wireless Sensor Networks, Wiley, 2010.

[5] H. L. a. S. Y. C.H Chiu, A Content Delivery System for Storage Services in Cloud Environment, International Journal of Ad Hoc and Ubiquitous Computing, 2010.

[6] Z. staff, "The Benefits of Open Source Software," *Zivtech,* July, 2019.

[7] A. Lupori, "16 Open Source Home Automation Platforms to Use in 2020," *Ubidots,* October, 2019.

[8] "EventGhost," 2005-2017. [Online]. Available: http://www.eventghost.net/docs/index.html.

[9] "Jeedom," [Online]. Available: https://www.jeedom.com/fr/.

[10] "MyController.org," 2015-2018. [Online]. Available: https://www.mycontroller.org/#/home.

[11] O. Schneider, "Pimatic," 2020. [Online]. Available: https://pimatic.org/.

[12] o. community, "OpenHAB," 2020. [Online]. Available: https://www.openhab.org/docs/.

[13] O. t. Jekyll, "Home Assistant," [Online]. Available: https://www.home-assistant.io/integrations/.

[14] "Home Assistant- Architecture," 2020. [Online]. Available: https://developers.home-assistant.io/docs/architecture_index/.

[15] A. Brice, "Best of open source smart homes: Home Assistant vs OpenHAB," *SmartHome University,* January, 2020.

[16] L. Llamas, "¿Qué es MQTT? Su importancia como protocolo IoT," *Ingeniería, informática y diseño,* April, 2019.

[17] "CloudMQTT," [Online]. Available: https://www.cloudmqtt.com/docs/index.html.

[18] S. Laboratories, 2020. [Online]. Available: https://www.z-wave.com/ .

[19] M. Treinish, "Building a better thermostat with Home Assistant," *opensource.com ,* Aug 2018.

[20] O. t. Jekyll, "Generic Thermostat-Home Assistant," [Online]. Available: https://www.home-assistant.io/integrations/generic_thermostat/.

[21] GitHub, "OwnTracks-Home Assistant," [Online]. Available: https://www.home-assistant.io/integrations/owntracks/.

[22] “Weather Underground-Home Assistant,” [Online]. Available: https://www.home-assistant.io/integrations/wunderground/.

[23] F. E. Wicks, “Efficient heater and air conditioner”. United States Patent US4813242A, 1987.

[24] “tadoº,” [Online]. Available: https://www.tado.com/us/.

[25] GitHub, “Tadoº Integrations- Home Assistant,” [Online]. Available: https://www.home-assistant.io/integrations/tado/.

ODS ANNEX

The 2030 Agenda for Sustainable Development sets out 17 Sustainable Development Goals, with 169 integrated and indivisible targets covering economic, social, and environmental spheres.

The project bets on reducing power consumption by implementing two different process for controlling a house temperature, there is also a comparison between both to find out which is more economically profitable. Both processes are considered reliable, sustainable, and modern, therefore the project is related to objective number 7 from the ones proposed by the 2030 Agenda. It ensures access to affordable, reliable, sustainable, and modern energy for all.

In addition, the project is related to objective number 9, which is based on developing resilient infrastructures, promoting inclusive and sustainable industrialization, and innovation. Also, this project reflects objectives 12 and 13, which ensures sustainable consumption and production patterns, and taking urgent action to combat climate change and its effects. As it has been demonstrated, the main objective of the project is to optimize a smart home, with the main goal of reducing power consumption and reducing electricity bills.