ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

INGENIERO ELECTROMECÁNICO

TRABAJO DE FIN DE GRADO

DESARROLLO E INTEGRACIÓN DE UN CENTRO DE CONTROL DOMÓTICO

Autor: Carlos Cabrera Criado

Director: Juan Luis Zamora Macho

MADRID

agosto 2018

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. Carlos Cabrera Criado

DECLARA ser el titular de los derechos de propiedad intelectual de la obra: Desarrollo e integración de un centro de control domótico, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.
b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
e) Asignar por defecto a estos trabajos una licencia Creative Commons.
f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente)*.

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

a) Que la Universidad identifique claramente su nombre como autor de la misma
b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
c) Solicitar la retirada de la obra del repositorio por causa justificada.
d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.

d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

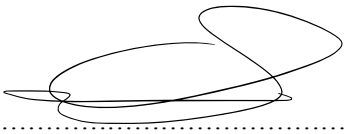
6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 3 de septiembre de 2018

ACEPTA



Fdo……

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título Desarrollo e integración de un centro de control domótico en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2017/2018 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.

Fdo.: Carlos Cabrera Fecha: 30/ 08/ 2018

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Juan Luis Zamora Fecha: 30/ 08/ 2018

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

INGENIERO ELECTROMECÁNICO

TRABAJO DE FIN DE GRADO

DESARROLLO E INTEGRACIÓN DE UN CENTRO DE CONTROL DOMÓTICO

Autor: Carlos Cabrera Criado

Director: Juan Luis Zamora Macho

MADRID

agosto 2018

This dissertation was typeset with LaTeX and compiled in TeXmaker. The font families used are Bitstream Charter, Utopia, Bookman, and Computer Modern. Unless otherwise noted, all figures were created by the author using Microsoft PowerPoint® and MATLAB®.

A mis padres

Resumen

Introducción y objetivos

En un mundo gobernado por los avances tecnológicos, la domótica está adquiriendo un papel protagonista. Tanto es así, que múltiples empresas ya han apostado por su desarrollo creando distintos periféricos, que, a pesar de ser eficientes, en la mayoría de los casos presentan problemas de compatibilidad entre dispositivos de diferentes marcas.

Este proyecto llamado El Nodo precisamente tiene como finalidad establecer enlaces entre dispositivos independientemente de las marcas que los han desarrollado, siendo un sistema agnóstico en lo que a tecnologías de comunicación se refiere. Este sistema pretende servir como centro de control universal para una extensa variedad de dispositivos del mercado, independientemente de las marcas y tecnologías empleadas.

Para cumplir con dicho objetivo, el centro de control debe por tanto disponer de una gran capacidad para establecer comunicaciones fiables a través de diferentes protocolos de comunicación. Además, se pretende dotar a dicho centro de control de la capacidad para implementar modelos de control avanzados dentro del mismo. Siendo por tanto capaz de cerrar un lazo de control gracias a su conectividad con periféricos de recepción de datos, procesamiento y posterior actuación.

En este proyecto se realizará un diseño completo del sistema tanto a nivel hardware como software, así como su posterior implementación y puesta en marcha.

Metodología

Para llevar a cabo los objetivos propuestos se seguirá el siguiente procedimiento:

- Diseño general de la arquitectura del sistema.
- Construcción del prototipo.
- Desarrollo de los componentes de software.
- Instalación de software de terceros. (Openhab, Matlab y Simulink)
- Programación de los enlaces entre módulos.
- Pruebas de funcionamiento.
- Ajustes finales. Redacción de la memoria.

Diseño del prototipo

El Nodo formará parte del mobiliario de la casa, y para garantizar una cómoda interacción con el usuario, el aspecto estético y su tamaño son fundamentales. El prototipo dispone de una pantalla táctil y altavoces a cada lado, micrófono y diferentes métodos de captación y visualización de información.

Todo ello se encuentra contenido en una estructura principalmente de madera elaborada gracias a una cortadora láser. Dicha estructura de madera en conjunción con diferentes piezas impresas en 3D y tornillería componen íntegramente la parte mecánica del prototipo.

Por otro lado, se han soldado los diferentes componentes electrónicos a diferentes placas de prototipado según el diseño electrónico realizado para garantizar el funcionamiento del dispositivo.

Arquitectura del sistema

Un aspecto importante del proyecto es el diseño de la arquitectura del sistema. Según lo propuesto, el sistema debe incorporar una gran conectividad junto con una versatilidad para implementar modelos de control.

Para cumplir con los criterios de conectividad, se hará uso del software Openhab, el cual nos permite añadir una capa de compatibilidad para la comunicación con periféricos demóticos independientemente del fabricante.

Por otro lado, las operaciones de control corren a cargo del software Matlab y Simulink. Este software proporciona al sistema la capacidad para realizar modelos matemáticos de sistemas del hogar y gestionar la actuación de los dispositivos en función de la información disponible.

Estos elementos se integran en el sistema gracias a la implementación modular del software, el cual podemos dividir en cinco componentes principales.

- Interfaz gráfica: Permite la interacción con el dispositivo.
- Servidor de El Nodo: Proceso principal encargado de la coordinación entre módulos.
- Gestor de hardware: Gestiona el hardware disponible en el sistema.
- Servidor Openhab: Ejecuta el software de Openhab en el sistema.
- Proceso Matlab y Simulink: Incorpora los modelos y scripts de

Matlab y Simulink.

Interfaz gráfica

La interfaz gráfica del sistema esta programada en lenguajes de programación web (HTML, CSS y Javascript). Se han escogido estos lenguajes por su fácil programación y su alta integración con el resto del sistema.

La interfaz gráfica permite al usuario acceder a las diferentes aplicaciones programadas, así como visualizar el estado de los periféricos conectados al dispositivo. Además, permite al usuario configurar el sistema de forma intuitiva sin tener que ejecutar código especifico.

Servidor de El Nodo

El servidor de El Nodo tiene por objetivo principal la regulación del funcionamiento del sistema en su conjunto, estableciendo las vías de comunicación necesarias entre módulos.

Este servidor esta conectado con el resto de los módulos principalmente mediante un servidor HTTP, el cual atiende a las peticiones del resto de módulos mediante POST o GET.

Para gestionar toda la información del sistema el servidor de El Nodo esta programado con Node.js. Este servidor permite un control a bajo nivel del hardware en el que se implementa, teniendo acceso a los elementos principales del hardware del dispositivo.

Gestor de hardware

El gestor de hardware se compone principalmente de un conjunto de componentes electrónicos instalados en la placa madre del dispositivo desde donde se gestiona el resto de placas electrónicas.

Este gestor ejecuta su proceso principal en un Arduino Mega, encargado de atender a los comandos enviados desde el

servidor de El Nodo. Dicho procesador tiene acceso por vía I2C a la placa de comunicaciones, el controlador del cubo led, así como de los altavoces y demás componentes.

Servidor Openhab

Openhab es un software de código libre pensado para la comunicación entre dispositivos domóticos con diferentes tecnologías de forma independiente a la marca usada. Gracias a la implementación de dicha tecnología el sistema es capaz de comunicarse con una gran variedad de dispositivos del mercado, haciendo efectiva la universalidad del dispositivo.

Matlab y Simulink

El proceso de Matlab y Simulink esta instalado en el sistema a modo de complemento para realizar tareas de control avanzado. Este software permite cargar modelos matemáticos en el sistema e interactuar con el hardware de El Nodo, teniendo a su disposición el resto de las herramientas de las que el sistema dispone.

Resultados y conclusiones

En este proyecto se ha desarrollado por completo un centro de control domótico capaz de comunicarse con multitud de dispositivos con tecnologías diferentes, así como realizar controles avanzados sobre dichos periféricos.

Para comprobar el funcionamiento del dispositivo se han realizado pruebas con diferentes periféricos verificando el funcionamiento de tanto las comunicaciones como del control. Dichas pruebas han mostrado ser exitosas, probando así el funcionamiento del sistema en su conjunto.

Abstract

Introduction and objectives

In a world governed by technological advances, home automation is taking on a leading role. So much so, that multiple companies have already opted for their development by creating different peripherals, which, despite being efficient, in most cases present compatibility problems between devices of different brands.

This project called El Nodo is precisely aimed at establishing links between devices independently of the brands that have developed them, being an agnostic system in terms of communication technologies. This system aims to serve as a universal control center for a wide variety of devices on the market, regardless of the brands and technologies used.

To fulfill this objective, the control center must therefore have a great capacity to establish reliable communications through different communication protocols. In addition, it is intended to provide said control center with the capacity to implement advanced control models within it. Being therefore able to close a loop of control thanks to its connectivity with peripherals of data reception, processing and subsequent action. In this project, a complete design of the system will be carried out both at hardware and software level, as well as its subsequent implementation and start-up.

Metodología

To carry out the proposed objectives, the following procedure will be followed:

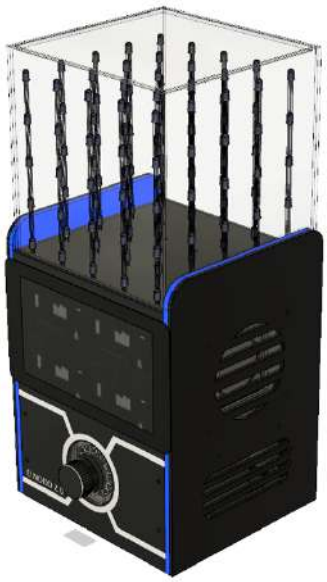
- General design of the architecture of the system.
- Construction of the prototype.
- Development of software components.
- Installation of third-party software. (Openhab, Matlab and Simulink)
- Programming of links between modules.
- Funcionality test.
- Final adjustments. Writing of memory.

Prototype design

"El Nodo" will be part of the furniture of the house, and to guarantee a comfortable interaction with the user, the aesthetic aspect and its size are fundamental. The prototype has a touch screen and speakers on each side, microphone and different methods of capturing and displaying information.
All this is contained in a structure mainly made of wood thanks to a laser cutter. This

wooden structure in conjunction with different 3D printed parts and hardware make up the mechanical part of the prototype.

On the other hand, the different electronic components have been welded to different prototyping boards according to the electronic design made to guarantee the operation of the device.

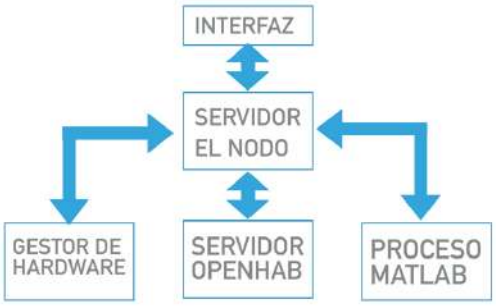


System architecture

An important aspect of the project is the design of the system architecture. As proposed, the system must incorporate a great connectivity together with a versatility to implement control models.
To comply with the connectivity criteria, Openhab software will be used, which allows us to add a compatibility layer for communication with demotic peripherals regardless of the manufacturer.
On the other hand, control operations are carried out by Matlab and Simulink software. This software provides the system with the ability to perform mathematical models of home systems and manage the performance of the devices based on the information available.

These elements are integrated into the system thanks to the modular implementation of the software, which can be divided into five main components.

- Graphic interface: Allows interaction with the device.
- El Nodo Server: Main process responsible for coordination between modules.
- Hardware manager: Manage the hardware available in the system.
- Openhab server: Run the Openhab software on the system.
- Matlab and Simulink Process: Incorporates Matlab and Simulink models and scripts.



Graphic interface

The graphical interface of the system is programmed in web programming languages (HTML, CSS and Javascript). These languages have been chosen for their easy programming and high integration with the rest of the system.

The graphic interface allows the user to access the different programmed applications, as well as to view the status of the peripherals connected to the device. In addition, it allows the user to configure the system intuitively without having to execute specific code.

El Nodo Server

The main objective of the El Nodo server is to regulate the operation of the system as a whole, establishing the necessary communication channels between modules.

This server is connected to the rest of the modules mainly through an HTTP server, which caters to the requests of the other modules through POST or GET.

To manage all the system information, the El Nodo server is programmed with Node.js. This server allows a low level control of the hardware in which it is implemented, having access to the main hardware elements of the device.

Hardware manager

The hardware manager consists mainly of a set of electronic components installed on the motherboard of the device from which the rest of the electronic boards are managed.

This manager executes its main process in an Arduino Mega, in charge of attending the commands sent from the El Nodo server. Said processor has I2C access to the communication board, the LED cube controller, as well as the loudspeakers and other components.

Openhab server

Openhab is an open source software designed for communication between domotic devices with different technologies independently of the brand used. Thanks to the implementation of this technology the system is able to communicate with a wide

variety of devices on the market, making effective the universality of the device.

Matlab and Simulink

The process of Matlab and Simulink is installed in the system as a complement to perform advanced control tasks. This software allows you to load mathematical models in the system and interact with the El Nodo hardware, having at your disposal the rest of the tools that the system has.

Results and conclusions

In this project, a domotic control center has been completely developed, capable of communicating with a multitude of devices with different technologies, as well as performing advanced controls on said peripherals.

In order to verify the operation of the device, tests have been carried out with different peripherals verifying the operation of both the communications and the control. These tests have proven to be successful, thus proving the functioning of the system as a whole.

Índice general

Índice de figuras

Índice de tablas

1

Introducción

1.1. Introducción

Imaginarnos un mundo sin conexión a Internet, sin teléfonos móviles o sin ordenadores es para muchos de nosotros una tarea complicada, por no decir, imposible. Sin duda, esto se debe a una clara consecuencia de la revolución tecnológica que hemos vivido en estas últimas cinco décadas. Una revolución que ha estado marcada por el exponencial desarrollo de la electrónica, tal y como ya postulaba Gordon Moore en 1965 con su famosa Ley de Moore[1].

Los avances tecnológicos han modificado nuestra relación con el mundo. A diferencia de lo que ocurría hace tan solo dos generaciones, nosotros hemos nacido en un entorno digital que se consolida y expande. Sin duda, la electrónica ha llegado para quedase, haciéndose rápidamente hueco en nuestros bolsillos con los teléfonos móviles, en nuestras relaciones humanas con las redes sociales, en nuestros trabajos con los ordenadores y en nuestras casas con la domótica.

Entendemos por domótica al conjunto de tecnologías y sistemas electrónicos instalados en el hogar con el fin de gestionar de forma inteligente los diferentes recursos disponibles, mejorando tanto la calidad de vida de sus ocupantes como su seguridad. Es por tanto que la domótica no puede catalogarse como una tecnología en concreto, sino como un sistema adaptable a cada hogar. La domótica tiene como objetivos principales, el confort de sus ocupantes mediante la instalación de sistemas de entretenimiento multimedia o la simplificación de algunas rutinas del hogar, la mejora de la seguridad con la implementación de sistemas de vigilancia como cámaras o alarmas y todo ello garantizando la máxima eficiencia energética gracias a un continuo consumo de los dispositivos del hogar [2].

Toda implementación domótica está constituida principalmente por tres elementos principales: sensores, actuadores y centro de procesado de datos. Estos elementos pueden estar integrados en un mismo dispositivo o lo que es más común, en dispositivos diferentes. Es por este motivo que uno de los aspectos más importantes de la domótica son las comunicaciones entre dispositivos. Para que un sistema domótico funcione, es necesario que todos sus componentes formen parte de una misma red de comunicaciones por la que compartir la información.

En este proyecto se estudiará la implementación de sistemas domóticos en el hogar haciendo especial hincapié en el aspecto de las comunicaciones y el procesado de datos centralizado en un centro de control domótico.

1.1.1. Organización del documento

Para facilitar la lectura y comprensión del proyecto en su totalidad, se describirá a continuación la organización del documento que lo desarrolla, con un breve resumen de los puntos a tratar en cada capítulo:

1. **Introducción** (Página: 1): Aproximación al proyecto.
   1.1. **Organización del documento** (Página: 3): Sección actual.
   1.2. **Motivación y problema** (Página: 5): ¿Por qué se realiza el proyecto? ¿Qué se trata de solventar con dicha implementación? ¿Qué repercusión se espera?
   1.3. **Objetivos del proyecto** (Página: 6): ¿Cuáles son los hitos importantes a conseguir en este proyecto? ¿Qué objetivos secundarios se plantean durante la ejecución?
   1.4. **Metodología de trabajo** (Página: 7): ¿Cómo se pretende afrontar el desarrollo del proyecto? ¿Qué recursos se van a emplear? ¿Cuál es la planificación?
   1.5. **Estado de la cuestión** (Página: 9): ¿Qué tecnologías hay actualmente en el mercado de la domótica? ¿Cuáles son las ventajas y desventajas que presenta cada una?
2. **Diseño general** (Página: 13):
   2.1. **Características y requerimientos** (Página: **??**): ¿Qué características indispensables debe reunir el proyecto para su mínimo funcionamiento? ¿Qué ventajas y desventajas presentan la tecnologías escogidas? ¿Qué prestaciones se prevén?.
   2.2. **Selección de componentes y tecnologías** (Página: 16): ¿Qué tecnologías se han decidido implementar en el proyecto? ¿Qué componentes se han escogido?
   2.3. **Diseño de funcionamiento general** (Página: **??**): ¿Qué funciones presentará el dispositivo fabricado? ¿Cómo se integra el software con el hardware? ¿Cómo se estructura el funcionamiento básico del sistema?
3. **Hardware** (Página: 75):
   3.1. **Diseño físico del prototipo** (Página: 25): ¿Qué características debe reunir el prototipo? ¿Cómo se ha realizado el modelado del sistema físico?.
   3.2. **Planos y esquemas de montaje** (Página: **??**): ¿Cuáles son los planos finales? ¿Cómo se realiza el montaje del dispositivo?
   3.3. **Diseño de sistemas electrónicos** (Página: **??**): ¿Cómo se distribuye la electrónica? ¿Qué componentes se usarán?
   3.4. **Esquemas eléctricos** (Página: **??**): ¿Cuáles son los esquemas electrónicos del sistema?
4. **Software** (Página: 43):
   4.1. **Estructura general del software** (Página: 43): ¿Cómo se organiza el software? ¿Cómo se administran los diferentes procesos y eventos?
   4.2. **Servidor de el Nodo** (Página: 46): ¿Cómo funciona el servidor del Nodo? ¿Qué tecnología se usa? ¿Qué funcionalidad aporta al sistema?

4.3. **Servidor Openhab** (Página: **??**): ¿Qué es OpenHab? ¿Qué funcionalidad aporta al sistema? ¿Cómo se integra OpenHab en el sistema?

4.4. **Proceso MATLAB y Simulink** (Página: **??**): ¿Que es MATLAB y Simulink? ¿Qué funcionalidad aporta? ¿Cómo se integra MATLAB y Simulink en el sistema?

4.5. **Gestor de hardware** (Página: 58): ¿Qué es el gestor de hardware? ¿Qué funcionalidad aporta? ¿Qué tecnología se usa? ¿Cómo se integra?

4.6. **Interfaz gráfica** (Página: **??**): ¿Cuál es el diseño de la interfaz gráfica? ¿Qué funcionalidad aporta al usuario? ¿Qué tecnología se usa? ¿Cómo se integra la interfaz en el sistema?

4.7. **Funcionalidades adicionales** (Página: **??**): ¿Cómo se gestiona la matriz led? ¿Cómo funciona el sistema de almacenado de datos en la nube? ¿Qué otras funcionalidades soporta el sistema?

5. **Pruebas** (Página: **??**):

5.1. **Tests de comunicaciones** (Página: **??**): ¿Con que productos se ha probado el sistema? ¿Cuáles son los resultados en el rendimiento y eficacia de los diferentes protocolos?

5.2. **Tests de sistemas de control** (Página: **??**): ¿Cómo se ha probado el funcionamiento de los sistemas de control? ¿Qué resultados se han obtenido?

6. **Conclusiones** (Página: **??**):

7. **Anexos** (Página: **??**):

7.1. **Sistema de seguridad** (Página: **??**): ¿Qué sistemas de seguridad se emplearán en el sistema?

1.1.2. Motivación y problema

El concepto de la domótica no es algo nuevo, desde hace ya muchos años se ha visualizado la automatización del hogar como algo futurista, quedando plasmada esta idea en gran cantidad de películas de ciencia ficción, donde la existencia de robots mayordomos nos hacen la vida en el hogar más fácil. En la actualidad, la tecnología ha alcanzado un punto de evolución considerable, y aunque no hasta el punto de robots mayordomos, sí que podemos decir que la domótica es una realidad.

Estamos por tanto hablando de una tecnología relativamente nueva, influenciada por la continua evolución de la electrónica. Una tecnología por la que ya muchas grandes compañías han apostado. Apple con Homekit, Amazon con Alexa o Google con Google Home son algunos de los productos que reflejan la nueva demanda de la sociedad por esta tecnología. El mundo de la domótica es un sector con un gran potencial, no obstante, como toda nueva tecnología, presenta aspectos que pueden ser objeto de mejora. Uno de los problemas más significativos de la domótica hoy en día radica en la integración entre dispositivos. Como ya hemos mencionado, gran cantidad de empresas han visto en la domótica una oportunidad, fabricando consecuentemente diferentes dispositivos para el hogar. Estos dispositivos en ocasiones están diseñados para comunicarse exclusivamente a través de una tecnología determinada, generando problemas de compatibilidad con dispositivos de otras compañías. Este tipo de incompatibilidad se traduce en una mayor rigidez a la hora de elegir dispositivos para el hogar, y en ocasiones, obligando al consumidor a utilizar únicamente los dispositivos de una misma marca.

Con motivo de proponer una solución a este problema nace este proyecto de integración de un centro de control domótico. Dicho centro de control tiene como propósito hacer de puente entre dispositivos con diferentes tecnologías, haciendo posible el control de todos ellos bajo una misma plataforma. Es por tanto que se pretende realizar un dispositivo capaz de gestionar las comunicaciones con una gran variedad de protocolos, así como ser capaz de realizar funciones de control en función de los datos de entrada.

Con este objetivo en mente se propone un esquema de funcionamiento del centro de control basado en tres etapas: Recepción, Procesado y Transmisión. (Figura 3)

En primer lugar, en la etapa de recepción, el centro de control dispondrá de un hardware capaz de recibir y transmitir información usando los protocolos descritos en el apartado 2.5 de este documento. Todas estas comunicaciones se realizarán a través de diversos dispositivos integrados dentro del encapsulado del centro de control. Dichos dispositivos transmitirán la información captada desde los periféricos al procesador principal, el cual ejecutará un proceso dedicado para OpenHab. Este proceso será el encargado de traducir los mensajes entrantes dependiendo de los diferentes protocolos usados por cada dispositivo.

Una vez obtenido el mensaje entrante, se procede a realizar el procesado de la información. El comando entrante se transfiere a otro proceso en el procesador encargado de la automatización del sistema. En él se tomarán las decisiones de actuación según se hayan programado previamente por el usuario. Por otro lado, en paralelo a este proceso tenemos el de MATLAB y Simulink, el cual será ejecutado si se considera que se debe realizar un control más exhaustivo. Finalmente, y una vez procesada la acción a realizar, se transmitirá la orden a los periféricos encargados de ejecutarla. Este proceso se realizará de forma inversa a la recepción de información. Se enviará el comando a OpenHab para su interpretación y consecuentemente se transmitirá el mensaje a través del hardware dedicado.

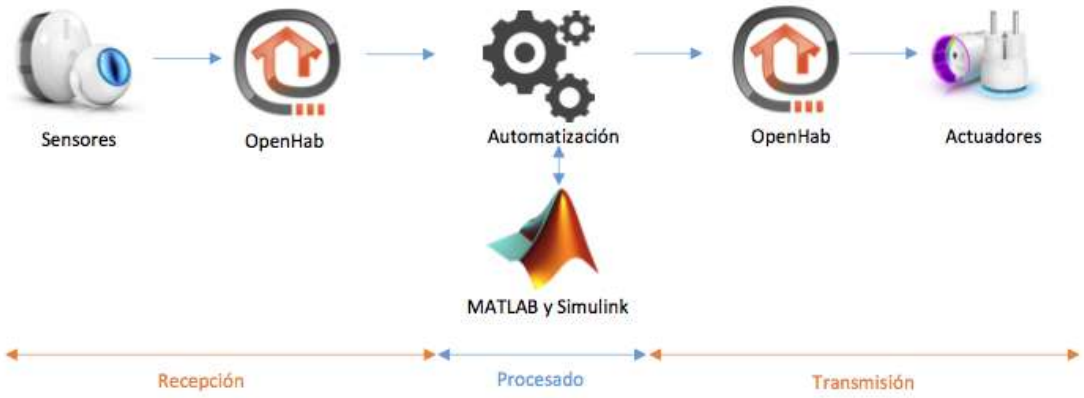


Figura 1.1. Esquema de funcionamiento del sistema de control

1.1.3. Objetivos del proyecto

Para la consecución del proyecto se han definido los siguientes objetivos:

1. Construcción del hardware del centro de control domótico.
2. Controlar múltiples periféricos domóticos de diferentes marcas bajo un mismo sistema de automatización.
3. Aplicación de procesos de control con Matlab sobre diferentes periféricos. Probar que los scripts de Matlab y Simulink cargados en el centro de control funcionan controlando los periféricos del sistema.

Además de los objetivos principales también se definen algunos objetivos secundarios del proyecto:

1. Desarrollo de un pequeño periférico de bajo coste para la implementación en el centro de control. Se pretende crear un dispositivo DIY con el código necesario para ser reconocido y controlado por el centro de control.
2. Comunicación con internet para el control remoto del sistema. Desarrollo del software correspondiente para poder acceder al centro de control desde cualquier plataforma móvil.

1.1.4. Metodología de trabajo

Para la correcta realización de los objetivos propuestos se seguirá el siguiente procedimiento:

- Diseño de un prototipo basado en Raspberry Pi para cumplir con los requerimientos mínimos del sistema.
- Fabricación del prototipo.
- Instalación y comprobación del funcionamiento de la plataforma OpenHab en la
- Raspberry Pi.
- Desarrollo de driver básico de control para el prototipo.
- Test de funcionamiento del prototipo. En este punto se pretende comprobar la viabilidad del proyecto antes de comenzar a fabricar el sistema completo.
- Diseño físico y digital del sistema completo.
- Construcción del centro de control.
- Programación de los drivers del hardware diseñado.
- Pruebas de funcionamiento del hardware.
- Instalación de OpenHab y Matlab en el sistema.
- Programación de la interfaz de control del centro de control domo?tico.
- Pruebas de funcionamiento del dispositivo final.
- Ajustes finales.
- Redacción de la memoria.

2

Estado del Arte

2.1. Estado de la cuestión

2.1.1. El inicio de la domótica

Si tenemos en cuenta sus orígenes radicados en la industria de la distribución eléctrica, la domótica tal cual la conocemos a día de hoy ha evolucionado de forma notable. El comienzo de dicha tecnología podríamos datarla junto con la llegada del protocolo X10[3] en 1975, el cual se basaba en la transmisión de información para el telecontrol de dispositivos basados en corrientes portadoras o Power Line. Este primer protocolo fue ampliamente extendido por Estados Unidos y Europa debido a su sencillez, accesibilidad y por sus capacidades para la implementación de controles sencillos en el hogar.

Con la creciente popularidad del protocolo X10, diferentes empresas del sector eléctrico comenzaron a desarrollar tecnologías para el control domótico, entre las que destacan BatiBus, EIB y EHS. Dichas tecnologías trataron de hacerse con el nuevo y creciente mercado de la domótica, teniendo cada una éxito en diferentes localizaciones de Europa. Sin embargo, no sería hasta finales de la década de 1990 cuando conseguirían extender su influencia al juntar fuerzas creando el actualmente conocido estándar KNX. Este nuevo estándar no tardó en ganar popularidad extendiéndose por todo el mundo y haciéndose hueco en un reticente mercado como es el de la construcción.

Junto con la consolidación del estándar KNX, se desarrolla en Estados Unidos la empresa Lonworks, la cual desarrollaría en 1999 el protocolo LonTalk para el control de redes haciéndose con la gran mayoría del mercado de Estados Unidos. Tanto el estándar LON como KNX ganan rápidamente popularidad debido a la aparición de pequeñas empresas que desarrollarían productos basados en alguno de estos estándares, sirviendo como base para el desarrollo de la domótica que conocemos a día de hoy.

2.1.2. Introducción a la domótica

Para poder comprender en que consiste la domótica no debemos entenderla como una tecnología en concreto, sino como una conjunción de diferentes tecnologías. La integración de dichas tecnologías en el ámbito del hogar constituye lo que entendemos por sistema domótico. Por consiguiente, podemos entender la domótica como una estructura flexible y adaptable a

las necesidades de cada hogar, con la capacidad de integrar las tecnologías más convenientes para satisfacer los diferentes cometidos para los que se imple- menta. Todo sistema domótico, como ya hemos comentado, está formado por diferentes dispositivos independientes que se comunican unos con otros formando una red de comunicaciones. Dicha red puede configurarse adop- tando diferentes arquitecturas dependiendo de las distintas necesidades. No obstante, podemos destacar dos arquitecturas predominantes. En primer lugar, los sistemas centralizados, donde todo el procesado de información se realiza en un mismo dispositivo al que se conectan el resto de los dispositivos, tipo de arquitectura típica de una topología en estrella (Figura 1). Por otro lado, tenemos los sistemas distribuidos, donde el procesado de datos se divide entre los diferentes dispositivos que conforman la red, tipo de sistemas característico de topologías en malla y anillo (figura 2).

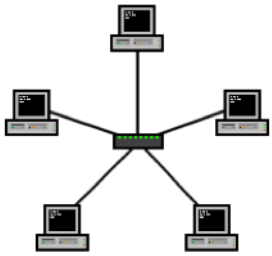


Figura 2.1. Topología en anillo

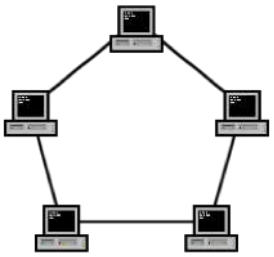


Figura 2.2. Topología en estrella

2.1.3. Sistemas centralizados

Entre las diferentes arquitecturas que puede adoptar un sistema domótico, explicaremos en profundidad los sistemas centralizados por su especial relevancia en el desarrollo del propio proyecto. Los sistemas centralizados se caracterizan por tener un único dispositivo encargado del procesado y distribución de los datos, estando el resto de dispositivos conectados a él a través de diferentes mecanismos de comunicación. A este tipo de dispositivo se le conoce en el ámbito domótica como centro de control. Dicho centro de control actúa como intermediario en todos los procesos a controlar, recibiendo señales de los distintos dispositivos de adquisición de datos, procesando dichas señales y consecuentemente enviando las respuestas pertinentes a los dispositivos de actuación. Es por tanto que dicho centro de control se caracteriza por su capacidad para procesar datos, disponiendo de un software de control dedicado y por su capacidad para establecer comunicaciones con múltiples dispositivos, siendo efectivo manejando tanto el trafico entrante como el saliente.

2.1.4. Medios de comunicación y protocolos

Como hemos comentado anteriormente, las comunicaciones son una parte vital para la implementación de cualquier sistema domótico. Estas comunicaciones se pueden realizar a través de diferentes medios, entre los que destacan los medios guiados o cableados y los aéreos. Dichos medios presentan diferentes ventajas y desventajas, siendo conveniente la implementación de uno u otro en función de las características requeridas en dicha comunicación. Es por este motivo que es muy común el uso de arquitecturas que integren ambos medios en su red de comunicaciones. No obstante, el hecho de que se comuniquen por el mismo medio no es condición suficiente para poder establecer una comunicación efectiva.

Los dispositivos en dicha red deben comunicarse siguiendo el mismo conjunto de normas y reglas para la transmisión de información, definiéndose así la semántica, sintaxis y métodos de sincronización de la comunicación. A este conjunto de reglas y normas se le conoce como protocolos de comunicación [4].

2.1.5. Protocolos en la domótica

En los sistemas domóticos es muy común encontrar dispositivos que usen protocolos de comunicaciones diferentes según las características del medio físico donde se produce la transferencia de información. Algunos de los más relevante hoy en día son:

- **Bluethooth:** Protocolo de comunicación inalámbrica mediante enlace por radiofrecuencia en la banda ISM a 2.4GHz según la norma IEEE 802.15.1. [5]
- **Ethernet:** Protocolo de comunicación por cable según la norma IEEE 802.3. [6]
- **WI-FI:** Estándar de comunicación inalámbrica según la norma IEEE 802.11 para comunicaciones inalámbricas en red local (LAN).
- **Z-wave:** Protocolo de comunicación inalámbrica de bajo consumo. Opera a 868.42 MHz para evitar interferencias con otros protocoles como WI-FI y Bluetooth. [7]
- **Zig-Bee:** Esta?ndar de comunicación inalámbrica según la norma IEEE 802.15.4 para redes de a?rea personal (WPAN). Característico por su baja tasa de transmisión de datos y su alta eficiencia energética. [7]

2.1.6. Openhab

Como hemos visto, la domótica es una composición de múltiples tecnologías todas ellas interconectada en una red que puede presentar a su vez múltiples protocolos y sistemas de comunicación. En respuesta al intento de facilitar los procesos de integración de dispositivos en el hogar nace la iniciativa OpenHAB [8]. Dicha iniciativa consiste en una plataforma de código abierto capaz de conectar diferentes dispositivos independientemente del protocolo de comunicación empleado por cada uno. OpenHab establece por tanto un puente entre dispositivos con tecnologías diferentes, haciendo posible la integración y control de gran cantidad de periféricos independientemente de su fabricante.

2.1.7. MATLAB y Simulink

Hemos hecho hincapié en la relevancia de los sistemas de comunicación en el campo de la domótica, sin embargo, para que las implementaciones domóticas sean efectivas es necesario tener cierto control y capacidad para procesar los datos entrantes. La implementación de estos mecanismos de control es posible gracias a entornos de programación como son MATLAB y Simulink [9], desde los cuales es posible procesar los datos entrantes del sistema y responder a ellos de forma efectiva. Una de las aplicaciones más relevantes de estos sistemas de control en la domótica es la eficiencia energética. Gracias a la gestión inteligente de los recursos energéticos de la casa es posible reducir de forma significativa el consumo energético total del hogar, reduciendo consecuentemente la factura eléctrica.

3

Diseño general

3.1. Diseño general

En este capítulo realizaremos una aproximación global al proyecto. Analizaremos cuales son los requerimientos que debe cumplir el prototipo, así como las características propias que debe tener. Para realizar este análisis tendremos que considerar las limitaciones del propio hardware y software, seleccionando convenientemente los más adecuados para el correcto funcionamiento del sistema. Por otro lado, explicaremos el funcionamiento general del sistema en su conjunto, dando una idea de cómo se interconectan todos los elementos que lo componen.

3.1.1. Características y requerimientos

El proyecto de El Nodo tiene por objetivo principal servir como centro domótico principal en una vivienda particular, siendo por tanto capaz de gestionar eficientemente las comunicaciones con los diferentes periféricos y de procesar los pertinentes datos en tránsito. Por otro lado, el dispositivo debe ser fácil de utilizar por usuarios independientemente de sus conocimientos técnicos. Teniendo en cuenta estas tres dimensiones del proyecto podemos dividir el diseño en tres dimensiones: Comunicaciones, procesado de datos e interacción con el usuario.

A continuación, realizaremos un análisis cualitativo de las características que debe reunir el sistema en sus diferentes dimensiones:

- **Comunicaciones:** El nodo es un sistema basado en su alta flexibilidad en cuanto a protocolos y medios de comunicación. Es por ello que se requiere de hardware capaz de administrar los siguientes protocolos de comunicación.
    - **Bluethooth:** Gran cantidad de dispositivos actualmente utilizan comunicaciones Bluethooth para la transmisión de datos de corto alcance. Las utilidades más relevantes de este protocolo en el sistema es la posibilidad de conectar dispositivos multimedia como reproductores de audio inalámbricos así como la comunicación con teléfonos móviles.
    - **WI-FI:** Ð La capacidad del sistema para conectarse a internet es una característica imprescindible. A través de WI-FI, El Nodo es capaz de conectar con la red local de

la vivienda y controlar todos los dispositivos que estén conectados a la misma red. Además, hace posible la adquisición de datos a través de internet, habilitando el control remoto del sistema desde cualquier dispositivo con acceso a internet.

- **Ethernet:** En adición a las capacidades que aporta el protocolo WI-FI, añadimos el protocolo ethernet, el cual aporta las mismas prestaciones de acceso a internet con un mejor rendimiento y velocidades.

- **Z-Wave:** Ð El protocolo Z-Wave es un protocolo ampliamente extendido en el mundo de la domótica gracias a sus altas prestaciones en cuanto a velocidad de transmisión, alcance y fiabilidad. Existen actualmente en el mercado gran variedad de dispositivos domóticos que utilizan este protocolo, lo que lo convierten en un componente indispensable para El Nodo.

- **Zig-Bee:** Al igual que el protocolo Z-Wave, Zig-Bee tiene un gran nicho de mercado en el mundo de la domótica, principalmente debido a su alto rendimiento energético y fiabilidad.

Figura 3.1. Protocolos requeridos en el sistema.

- **Procesado de datos:** Para la correcta gestión de los datos en tránsito es necesario disponer de tanto software y hardware capaz de soportar rutinas de control específicas. Para ello se requiere:

  - **Software de control:** Software capaz de recibir las señales entrantes desde los periféricos de comunicación, procesarlos mediante scripts y algoritmos de control y finalmente enviarlos para su transmisión nuevamente a los periféricos de comunicaciones.

  - **Procesador:** Se requiere de un procesador capaz de soportar el software seleccionado. Es indispensable que disponga de potencia suficiente para realizar las tareas a una velocidad razonable sin comprometer el funcionamiento de otros procesos que puedan ocurrir en paralelo.

- **Interfaz de usuario:** Uno de los aspectos fundamentales del sistema es la simplicidad y el intuitivo funcionamiento del dispositivo. Para que el usuario sea capaz de controlar el sistema este debe de disponer de los siguientes elementos:
  - **Botones e indicadores:** Para que el usuario interactúe con el dispositivo es necesario que disponga de botones para ejecutar los comandos que se deseen. Además debe contar con algún tipo de indicador lumínico que de información sobre el estado de los diferentes procesos.
  - **Interfaz gráfica:** Con motivo de simplificar las tareas al usuario final es necesario el uso de una interfaz gráfica clara y concisa que guíe al usuario a través de las diferentes funcionalidades del dispositivo.

3.1.2. Selección de componentes y tecnologías

Una vez hemos definido los requerimientos del sistema, debemos asignar el hardware y software necesarios para satisfacer dichas características. En esta sección analizaremos punto por punto el hardware seleccionado para cada una de las funciones del sistema.

3.1.2.1. Procesado de datos

Antes de comenzar con la selección del hardware que usaremos, es importante definir que tecnología usar como centro de procesado principal. Debemos establecer cuál será el cerebro de las operaciones de todo el sistema, ya que este condicionará la elección del resto de componentes.

A la hora de elegir tecnología para la unidad de procesamiento principal tenemos varias posibilidades:

- **FPGA:** ¥ Las FPGAs son chips característicos por ser capaces de realizar tareas a gran velocidad gracias a la capacidad que tienen de programar su propia estructura física. Sin embargo, esta arquitectura presenta algunos problemas importantes para el sistema. En primer lugar, su precio. La implementación de estos chips es extremadamente cara en un sistema de estas características. Por otro lado, su programación es compleja y carece de flexibilidad a la hora de ser reprogramado o actualizado.
- **Arquitectura AVR:** La tecnología AVR es una arquitectura de microcontrolador muy famosa, principalmente por su bajo coste y alta eficacia. Estos microcontroladores son ideales para proyectos que no requieren una alta carga de procesamiento puesto que generalmente se trata de procesadores de 8 bits con un limitado set de instrucciones y velocidades de reloj bajas.
- **Arquitectura PIC:** Similar a la tecnología AVR tenemos los microcontroladores PIC, los cuales reúnen características muy similares a los AVR con algunas diferencias a en su set de instrucciones básicas y en sus ciclos por instrucción.
- **Arquitectura ARM:** La arquitectura ARM ofrece características similares a las de AVR, pero con algunas mejoras en cuanto a velocidad de reloj y bus de datos, el cual suele ser de 32 o 64 bits. Por contrapartida, los microprocesadores ARM tiene un coste superior a la tecnología AVR.

Teniendo en cuenta este análisis cualitativo de las tecnologías más relevantes a usar, podemos empezar a valorar cuál de ellas es la más óptima para implementar en el sistema. En primer lugar, descartamos la implementación de FPGA debido al alto coste y a su dificultad en cuanto a la programación e instalación de software de terceros. A continuación, debemos valorar las diferentes familias de microcontroladores. Para hacer más fácil la diferenciación entre dichas tecnologías y valorar la viabilidad de cada una en el proyecto se han reunido sus características en la siguiente tabla:

Una vez se han analizado las características de las diferentes arquitecturas, concluimos que la arquitectura ARM es la más conveniente debido a su mayor compatibilidad de protocolos y su mayor velocidad y capacidad de procesamiento con un bus de 32bits. Esta tecnología es por norma general más costosa que la PIC y AVR. No obstante, en relación con las prestaciones ofrecidas por este microcontrolador, el rendimiento es muy superior.

	PIC	AVR	ARM
Bus de datos	8/16/32-bit	8/32-bit	32-bit y 64-bit
Protocolos de comunicaciones	PIC, UART, USART, LIN, CAN, Ethernet, SPI, I2S	UART, USART, SPI, I2C, (Algunos AVR soportan CAN, USB, Ethernet)	UART, USART, LIN, I2C, SPI, CAN, USB, Ethernet, I2S, DSP, SAI (serial audio interface), IrDA
Velocidad	4 ciclo de reloj / instrucción	1 ciclo de reloj / instrucción	1 ciclo de reloj / instrucción
Memoria	SRAM, FLASH	Flash, SRAM, EEPROM	Flash, SDRAM, EEPROM
ISA	RISC (Algunas caracteristicas)	RISC	RISC
Arquitectura de la memoria	Arquitectura Harvard	Arquitectura modificada	Arquitectura Harvard modificada
Consumo energético	Bajo	Bajo	Bajo
Familia	PIC16,PIC17, PIC18, PIC24, PIC32	Tiny, Atmega, Xmega	ARMv4,5,6,7 y series
Comunidad	Buena	Muy Buena	Muy Buena
Fabricante	Microchip Average	Atmel	Apple, Nvidia, Qualcomm, Samsung Electronics, and TI etc.
Prestaciones / Precio	Media	Media	Alta
Productos populares	PIC18fXX8, PIC16f88X, PIC32MXX	Atmega8, 16, 32, Arduino	LPC2148, ARM Cortex-MX, etc.

Figura 3.2. Comparativa de las diferentes arquitecturas de microcontroladores.

Existe gran variedad de microcontroladores ARM en el mercado a precios muy competitivos. Sin embargo, a la hora de elegir debemos ser conscientes de la imposibilidad de comprar el microcontrolador por separado, ya que esto implicaría tener que soldarlo en una placa diseñada específicamente para ello. Por suerte, existe en el mercado gran cantidad de placas preparadas con estos microcontroladores y todos los componentes necesarios para su uso inmediato. Algunas de los productos más destacados son: ODroid, C.H.I.P Pro, Orange Pi y Raspberry Pi.

Si nos ceñimos a las especificaciones y precios de cada una de estas placas, encontraremos que todas ellas son viables para nuestro proyecto, no obstante nos decantaremos por usar la Raspberry Pi modelo 3B, por el amplio soporte técnico que tiene así como por su amplia compatibilidad con software de terceros.

Figura 3.3. Placa Raspberry pi modelo 3B

Finalmente, y antes de concluir con la selección de procesador principal para el proyecto, es conveniente destacar la arquitectura Quad Core de Intel como un candidato posible para remplazar la utilización de la Raspberry Pi. En concreto, la familia de placas Up-Core, basadas en el procesador Intel Atom. Esta placa de Intel ofrece unas prestaciones muy superiores en

cuanto a capacidad de procesado de datos en comparación a la Raspberry, aunque por motivos de costes se prescindirá de ella.

3.1.2.2. Comunicaciones

Retomando la discusión iniciada en la sección 2.1 de características y requerimientos, las comunicaciones son una parte fundamental del proyecto. Para satisfacer los requisitos previamente descritos, tenemos que seleccionar componentes teniendo en consideración el procesador seleccionado anteriormente, la Raspberry Pi.

Para realizar esta selección separaremos el análisis en función de los diferentes protocolos:

- **Bluethooth:** La comunicación bluethooth está soportada de forma nativa por la propia placa Raspberry Pi modelo 3B, sin embargo, con motivo de aliviar carga de procesamiento y no saturar determinados puertos serie internos de la Raspberry Pi, añadiremos un módulo HC- 05 por su fácil utilización y bajo coste.
- **WI-FI:** De forma similar a la comunicación Bluethooth, la Raspberry Pi dispone de un módulo WI-Fi integrado. No obstante, también usaremos un módulo extra para realizar funcionalidades adicionales tales como la generación de puntos de enlace Wi-FI configurando dicho módulo como hotspot. Para esta finalidad se ha escogido el módulo ESP8266 por su eficacia y bajo costo.
- **Ethernet:** ¥ El protocolo ethernet también será gestionado por la propia Raspberry Pi, puesto que dispone de un puerto ethernet dedicado a dicho propósito.
- **Z-Wave:** Para dotar a nuestro sistema de capacidad para comunicarse con dispositivos Z-Wave tendremos que hacer uso de un módulo especial. Existe gran variedad de módulos Z-Wave en el mercado, gran parte de ellos en formato USB dongle, sin embargo, para este proyecto resulta de especial relevancia aquellos diseñados especialmente para raspberri Pi, los cuales se conectan al GPIO de la misma. Cumpliendo estas características creemos conveniente elegir el módulo Razberry.
- **Zig-Bee:** ¥ Al igual que ocurre con el protocolo Z-Wave, la Raspberri Pi por si sola carece de hardware para comunicarse mediante protocolo Zig-Bee. Es por este motivo que tendremos que hacer uso de un módulo para implementar dicho protocolo en el sistema. Uno de los módulos más populares en este campo es el modelo Xbee, el cual usaremos debido a su bajo costo y rendimiento.

3.1.2.3. Interfaz de usuario

Para que el sistema sea funcional, no se debe olvidar la importancia de integrar sistemas para la interacción con el usuario. Es necesario establecer un método para que el usuario envié diferentes consignas a nuestro sistema, así como capacidad para que el sistema muestre información relevante al usuario sobre los diferentes procesos.

Con motivo de generar un entorno adecuado para la interacción con el usuario se ha optado por la integración de una interfaz de usuario basada en una pantalla táctil. El uso de una pantalla de estas características nos permite tanto mostrar información como recibirla a través de las capacidades táctiles. La pantalla que se utilizará es la oficial de siete pulgadas compatible

con la Raspberry Pi (Figura: 3.4). La principal ventaja de escoger esta pantalla táctil es su alta compatibilidad con la placa Raspberry Pi, la cual no requiere de ningún tipo de configuración especial.

Figura 3.4. Pantalla táctil 7” oficial de la Raspberry Pi.

Por otro lado, respecto al software se hará uso de tecnologías de programación web para la programación de la interfaz. Entre estas tecnologías destacan HTML, CSS y Javascript por su sencillez y versatilidad multiplataforma.

En resumen, la implementación de la interfaz gráfica constará de dos bloques importantes, el hardware (Pantalla táctil) y el software (Proceso web), cuyo funcionamiento se explicará en la sección 2.3, Diseño de funcionamiento general.

3.1.3. Diseño de funcionamiento general

Una vez definidas las características y asignadas las tecnologías involucradas en el sistema, sólo queda estructurar los diferentes bloques para dar forma a un prototipo funcional. En esta sección haremos un análisis cualitativo del funcionamiento general del sistema, explicando cómo interactúan las diferentes tecnologías entre sí.

Antes de adentrarnos en el funcionamiento en detalle del sistema, debemos comprender como se va estructurar. Para ello, dividiremos conceptualmente el proyecto en cinco bloques, cada uno de ellos encargado de una función específica. Los cinco bloques principales son los siguientes:

- **Interfaz:** Sistema compuesto por una pantalla táctil de siete pulgadas y software para visualización de la interfaz de usuario.
- **Servidor El Nodo:** Proceso que se ejecuta en el procesador principal (Raspberry Pi) encargado de la coordinación de los diferentes bloques del sistema.

- **Gestor de hardware:** Conjunto de sistemas electrónicos externos a la Raspberry Pi para expandir su funcionalidad, como pueden ser sensores, altavoces, micrófonos, placas de expansión, puertos, etc.
- **Servidor Openhab:** Software encargado de la interpretación de las comunicaciones con los dispositivos domoóicos.
- **Proceso MATLAB:** Algoritmos de control implementados en la Raspberry Pi para el control de los diferentes dispositivos conectados.

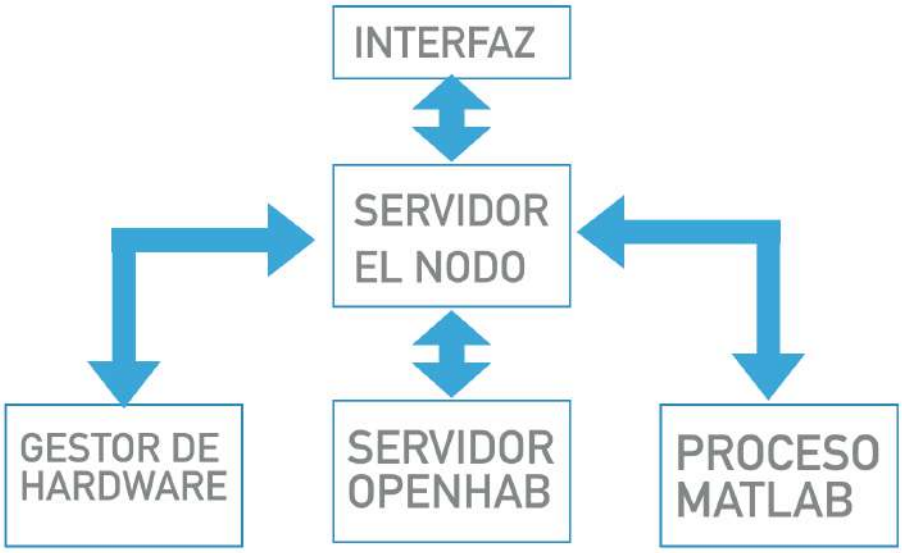


Figura 3.5. Esquema de funcionamiento general

Todos estos bloques están interconectados siguiendo el esquema de la figura 8, donde se aprecia como el servidor de El Nodo es el encargado de la coordinación de todos los demás bloques, pasando toda la información por este proceso. La idea contenida detrás de esta arquitectura, es poder centralizar el código en un mismo proceso, simplificando las tareas a realizar por el resto de bloques. Además, la centralización de los recursos en un único proceso nos asegura la disponibilidad de todos los recursos del sistema en un mismo hilo de ejecución, simplificando por consiguiente las comunicaciones entre bloques.

El servidor de El Nodo consta de un servidor web programado en Javascript con NodeJS. Este servidor se comunicará con el resto de bloques a través de APIs basadas en peticiones del tipo GET y POST, que son bidireccionales, por lo que es capaz tanto enviar comandos a otros bloques como de recibirlos.

Un ejemplo claro de esta comunicación es la que tiene lugar entre la interfaz y el servidor de El Nodo. El bloque de interfaz hace de cliente web mientras que el servidor de El Nodo hace de servidor web. Al arrancar el sistema la interfaz solicita al servidor que le facilite el código a ejecutar para mostrar la página de inicio. Acto seguido, el servidor procesa la petición y envía a la interfaz los documentos necesarios, los cuales son normalmente HTML, CSS y Javascript. Estas peticiones se realizan de la misma manera que lo hacen nuestros ordenadores cada vez que pretendemos acceder a una página web, la única diferencia es que tanto cliente como servidor están alojados en el mismo dispositivo, la Raspberry Pi.

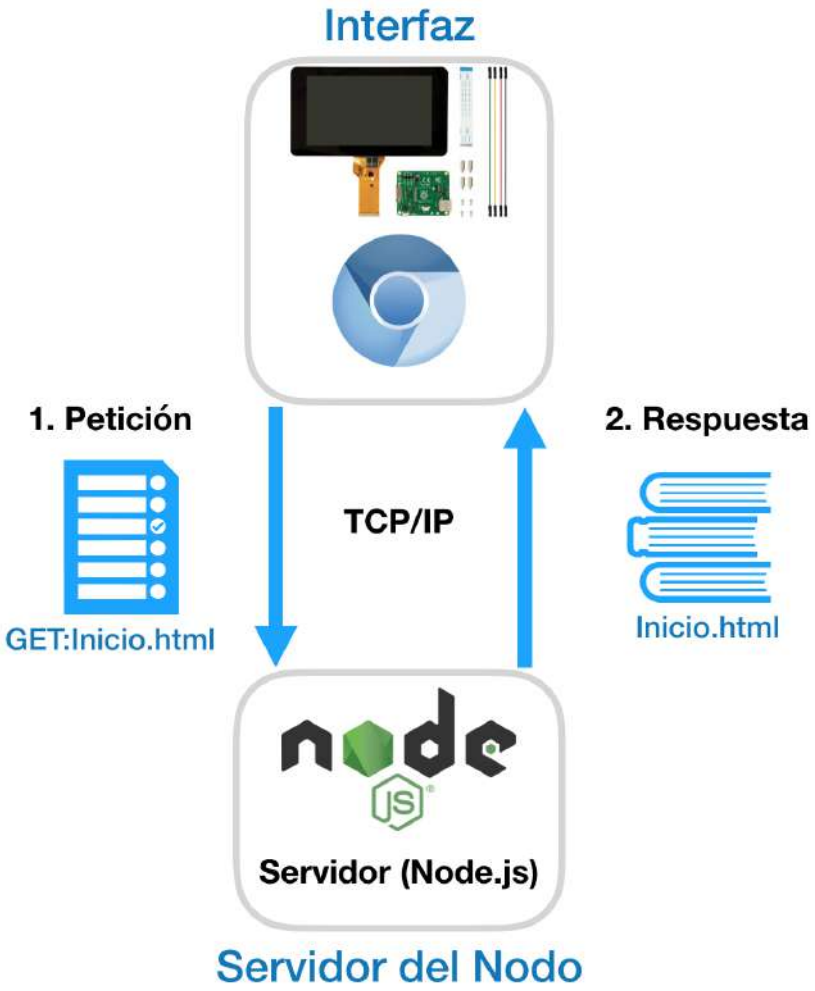


Figura 3.6. Esquema comunicación interfaz

De esta misma manera se comunica el servidor de El Nodo con el servidor de OpenHab y el proceso de MATLAB, con la diferencia de que normalmente la información que se comunica son comandos para su ejecución.

Por otro lado, nos queda cubrir la comunicación entre el servidor de El Nodo y el gestor de hardware, el cual difiere en algunos aspectos del resto. Como ya se ha mencionado, el gestor de hardware es un conjunto de hardware externo a la Raspberry Pi que funciona a modo de periférico para el procesador principal. Todo este hardware está conectado a una placa madre, de la cual se establecen todas las conexiones a los dispositivos internos del sistema. En dicha placa madre está alojado un microcontrolador ATmega 2560, el cual es responsable de la coordinación del hardware del sistema. Todos los comandos enviados desde el servidor de El Nodo alojado en la Raspberry Pi se envían a través de I2C al ATmega, para que éste ejecute la orden haciendo uso de los dispositivos conectados a la placa madre mediante SPI.

Para visualizar el funcionamiento del gestor de hardware y su comunicación con el servidor de El Nodo, pondremos un ejemplo de funcionamiento. A continuación, se muestra el caso en el que se desea enviar un comando a un dispositivo en la casa que se comunica por radiofrecuencia (Figura:10). En primera instancia, la orden se procesa en el servidor de El Nodo, generando el código a trasmitir al gestor de hardware. Una vez interpretado, se envía el comando hacia

la placa madre donde se encuentra el ATmega mediante I2C. Acto seguido, el Atmega reenvía el comando a el transmisor de radiofrecuencia por SPI el cual envía finalmente el comando al dispositivo de la casa.

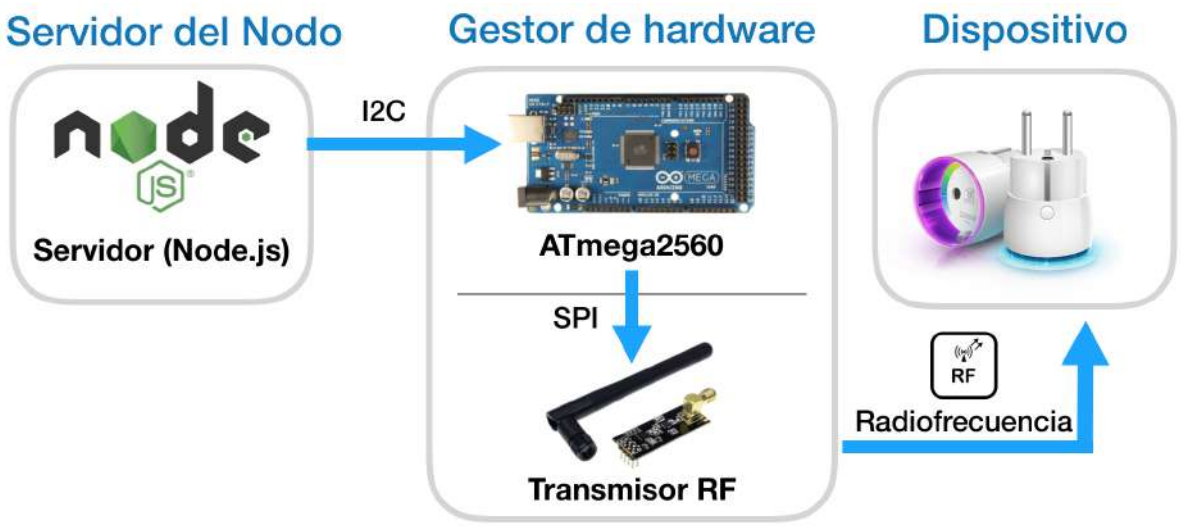


Figura 3.7. Esquema comunicación gestor de hardware

Como se puede observar en el ejemplo expuesto de la comunicación con el gestor de hardware, los bloques están definidos independientemente de si se trata de software o hardware. En el ejemplo se ilustra como el gestor de hardware dispone tanto de su hardware particular, el ATmega como de su propio software, el driver que lo controla. Es por este motivo que podemos decir que el gestor de hardware es un bloque compuesto tanto por software como por hardware. Mismo es el caso del bloque de interfaz, compuesto tanto por pantalla táctil como por el código que conforma la UI. Por el contrario, los bloques de OpenHab, Matlab y el servidor de El Nodo, podemos decir que son meramente software puesto que todos ellos son procesos internos que se ejecutan dentro de la misma Raspberry Pi.

De forma resumida podemos decir que el sistema de control se funda- menta en cinco bloques principales, independientes de hardware y software. Estos bloques se relacionan entre sí mediante diferentes métodos de comuni- cación, teniendo como centro del flujo de datos el servidor de El Nodo. Una vez conocido el funcionamiento y estructura general del sistema, solo queda descomponer cada bloque en sus dimensiones tanto de hardware como de software, tema que trataremos en profundidad en los siguientes capítulos.

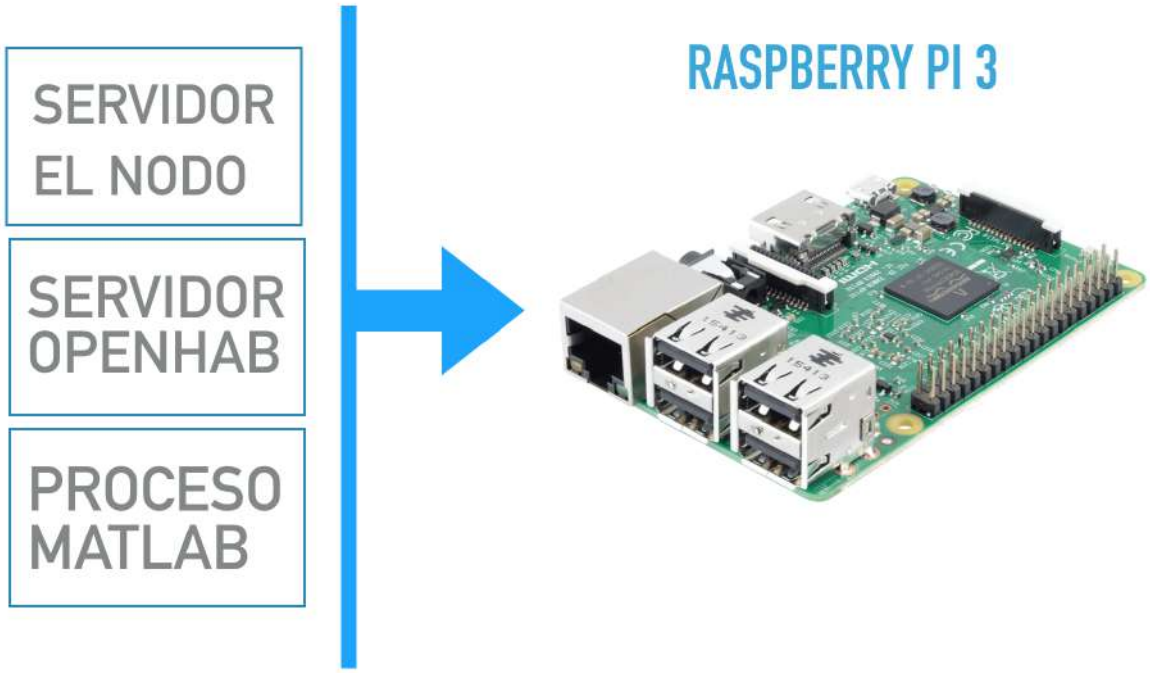


Figura 3.8. Bloques que se ejecutan en la Raspberry Pi

4 Hardware

4.1. Hardware

En este capítulo se verá en detalle el hardware que compondrá el centro de control domótico. Se expondrá el proceso de diseño físico del prototipo, donde se especificarán las características físicas que debe de tener el mismo para asegurar la viabilidad del producto.

En primer lugar, analizaremos aspectos como factor de forma, peso y tamaño. Acto seguido, se realizará el modelado del sistema en 3D, generando sus respectivos planos, definiendo los métodos de montaje y caracterizando cada una de las piezas a usar. Una vez se defina el modelo físico del prototipo, se realizarán los esquemas electrónicos necesarios para montar las diferentes placas del sistema. Junto con el diseño de cada una de las partes, se irá mostrando algunos de los pasos seguidos para el montaje y fabricación de las diferentes piezas que componen el sistema.

4.1.1. Diseño físico del prototipo

Antes de empezar el modelado en 3D, es importante que se tenga una idea clara de cómo será el prototipo a fabricar. Para ello, tendremos que pensar en la forma que debe tener, en el peso y en su tamaño. Estos parámetros de diseño vienen generalmente designados por las características de portabilidad que queramos que tenga el prototipo, así como su manejabilidad y estética.

En un principio, el centro de control domótico está pensado para ser ubicado en el interior de la casa, concretamente, en una habitación de manera semipermanente. Se piensa como un elemento que debe de formar parte del mobiliario de la casa, teniendo importancia el aspecto estético del sistema. El dispositivo debe ser lo suficientemente grande como para que la interacción con el usuario sea cómoda, pero sin que su tamaño imposibilite su reubicación una vez ha sido instalado. Debe disponer de diferentes métodos de captación y visualización de información, como botones y luces indicadoras. De esta forma, podemos enumerar las siguientes características a tener en cuenta para diseñar el prototipo:

- **Tamaño:** Debe ser lo suficientemente grande como para que sea fácil de ver y manipular.
- **Estética:** Formará parte del mobiliario por lo que tendrá que tener una estética cuidada.

- **Ubicación:** Estará ubicado en interiores, donde no deberá enfrentar ninguna adversidad meteorológica. La única protección de la que deberá precisar, es a la propia manipulación del usuario y a la leve afección por el polvo.

- **Interacción con usuario:** Constará con una pantalla táctil de siete pulgadas, altavoces, micrófono, botones e indicadores lumínicos.

Reuniendo estas características se ha realizado un sencillo boceto conceptual del modelo físico del prototipo como así se muestra en la siguiente figura

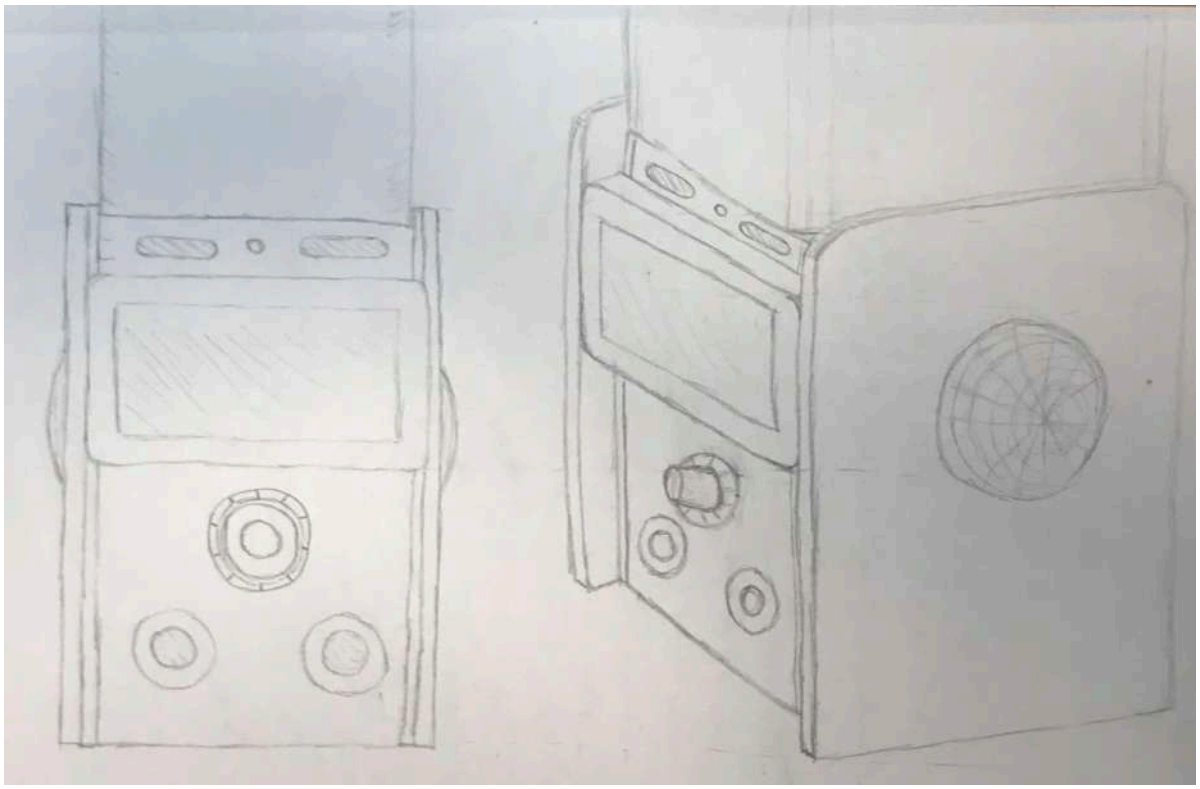


Figura 4.1. Boceto del sistema físico.

Una de las características más destacadas del boceto realizado es el uso de cuerpos planos que forman una estructura cuadrada. Se ha escogido esta apariencia por su fácil construcción. Toda su estructura puede realizarse a partir de láminas de diferentes materiales unidas entre sí por tornillería o adhesivos. Otro aspecto importante del diseño son sus dimensiones. Aunque no se haya acotado el boceto, vemos como todo el dimensionamiento del dispositivo está basado en las dimensiones del componente principal, la pantalla táctil. El ancho de la pantalla táctil que se usará en el prototipo define el ancho del dispositivo. Por otro lado, la altura también se relaciona con las dimensiones de la pantalla. Con la intención de conservar ciertos aspectos estéticos, el dispositivo se divide verticalmente en tres secciones, siendo la central, la pantalla, la más importante. La sección inferior contiene botones y un rotor para la interacción con el usuario, la intermedia, la pantalla y la superior una cúpula donde se pretende alojar indicadores lumínicos a modo de matriz led. Todo el sistema se estructura de forma simétrica teniendo los altavoces a cada lado para mantener una imagen equilibrada del dispositivo.

4.1.2. Modelado y esquemas de montaje

Una vez se ha fijado el diseño del prototipo se da paso al modelado 3D. Para realizar el modelo 3D se hará uso del software de Autodesk Fusion 360. Este programa dispone de herramientas de modelado paramétrico así como de entornos de renderizado, ensamblaje y animación. Como ya habíamos comentado en la sección 3.1, las dimensiones de la pan- talla táctil de la Raspberry Pi serán las que definan las dimensiones del resto del dispositivo. Es por este motivo que antes de comenzar con el modelado del sistema, es necesario dimensionar la propia pantalla táctil e introducir sus medidas en nuestro entorno de modelado.

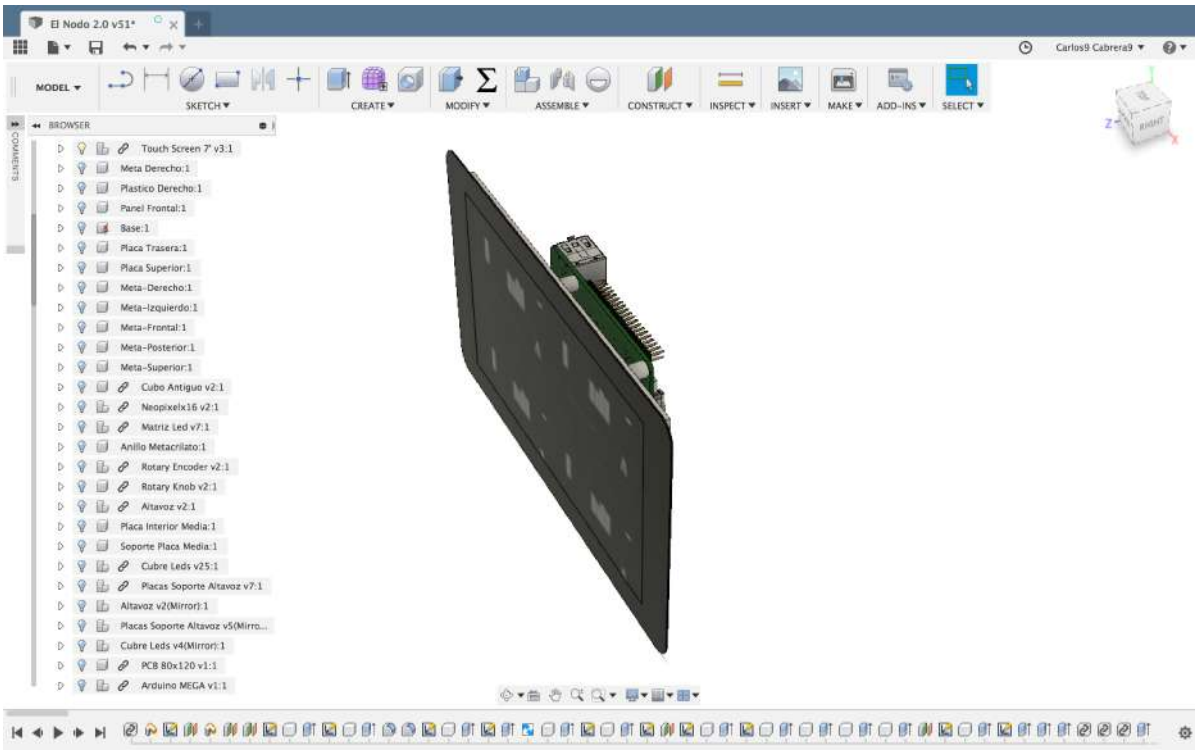


Figura 4.2. Modelo de la pantalla táctil 7".

Aunque las medidas de la pantalla táctil de la Raspberry Pi sea un buen punto de partida para comenzar a modelar el sistema en 3D, todavía tenemos que valorar algunos aspectos esenciales del diseño. Estos aspectos se resumen básicamente en la viabilidad de fabricación y la funcionalidad de la estructura.

En primer lugar, tenemos que tener en cuenta que la fabricación del dispositivo que se diseñe en 3D tiene que ser viable con técnicas tanto económicas como fáciles de implementar. Como ya habíamos mencionado anteriormente, la geometría del dispositivo es fundamentalmente plana para facilitar su posterior proceso de construcción. Este tipo de geometría nos per- mite componer los diferentes cuerpos del sistema con planchas de diferentes materiales, las cuales presentan la ventaja de tener un precio reducido y ser de fácil obtención en el mercado debido a su simple geometría. Además, son fácilmente manipulables puesto que se pueden cortar con facilidad.

Teniendo en cuenta estas consideraciones, se pretende utilizar láminas de madera recortadas con cortadora láser para la construcción de la estructura del sistema. Este método de construcción nos permite realizar un prototipo con materiales baratos, como es la madera

y con un acabado semiprofesional, gracias a la capacidad de realizar geometría compleja en 2D de manera precisa con la cortadora láser.

Sin embargo, a pesar de las grandes ventajas que nos aporta el uso de la madera cortada con láser, la complejidad del proyecto nos impide realizar el proyecto únicamente usando esta técnica. La naturaleza plana de las láminas dificulta realizar geometría compleja como pueden ser volúmenes. Es por este motivo que, para determinadas piezas, tenemos que recurrir a otras técnicas de fabricación como la impresión 3D. Con la impresión en 3D se pretenden suplir las limitaciones de la fabricación laminar dando la posibilidad de generar piezas con geometría compleja sin comprometer la economía del proyecto.

Existe actualmente gran variedad de métodos de impresión 3D, cada uno con sus ventajas e inconvenientes. No obstante, dentro del ámbito de nuestro proyecto solo tienen cabida los métodos más económicos como son el ?Fused deposition modeling? (en adelante FDM) y el ?Stereolithography apparatus? (en adelante SLA). Ambas técnicas de impresión son relevantes en la fabricación de prototipos, presentando pequeñas diferencias en el precio y acabado de las piezas resultantes. En primer lugar, el método FDM es capaz de producir piezas muy económicas sacrificando en cierta medida el acabado de la pieza. Por otro lado, el SLA nos proporciona piezas mucho mejor acabadas con una alta precisión a costa de un precio ligeramente superior. Estas dos tecnologías son perfectamente válidas para el proyecto en cuestión, no obstante, en este caso se optará por utilizar el método FDM por su mayor popularidad y mayor disponibilidad de impresoras de este tipo. El último aspecto a tener en cuenta a lo que fabricación respecta antes de modelar, es el método de ensamblaje. En el caso de El Nodo, se pretende que sea lo más sencillo posible sin comprometer la resistencia estructural del dispositivo. Para ello, optaremos por la tornillería como principal método de ensamblaje, junto con el uso de adhesivos para zonas especialmente delicadas. La geometría plana hace ideal el uso de pares de tornillo y tuerca para unir diferentes placas entre ellas, aportando resistencia suficiente a nuestro proyecto de forma económica y sencilla.

Una vez tenemos en cuenta el método de fabricación a usar, podemos continuar con el modelado en 3D teniendo en cuenta todas las premisas planteadas. Para ello proseguiremos con la creación de los elementos estructurales principales. Estos consisten principalmente en la base del dispositivo y en dos marcos de madera encargados de aportar resistencia estructural al sistema. Para unir entre sí estos tres elementos usaremos cuatro pares de tornillos y tuercas M4 como se ilustra en la figura 4.3.

Figura 4.3. Modelo 3D de la estructura.

El siguiente paso en el diseño del modelo será crear la parte frontal del dispositivo. Esta cara frontal mantiene la pantalla táctil y el botón de la parte inferior en su sitio, además de aportar elementos estíticos al dispositivo. Para conseguir este cometido el diseño constará de tres piezas diferenciadas: Escuadras de sujeción para la pantalla, marco de sujeción del dial y cubierta frontal. (Figura: 4.4)

(a) Escuadras de sujeción para la pantalla. **(b)** Marcho de sujeción del dial. **(c)** Cubierta frontal.

Figura 4.4. Montaje y piezas parte frontal.

La siguiente parte a diseñar son los laterales del modelo. En esta zona se instalarán los altavoces y los embellecedores laterales. El diseño de los laterales consta principalmente de seis piezas: Escuadras de sujeción de los altavoces, marco de sujeción de los altavoces, altavoces, placas separadoras laterales, embellecedores laterales y cubiertas laterales. (Figura: 4.5)

(a) Escuadras de sujeción de los altavoces.

(b) Marco de sujeción de los altavoces.

(c) Altavoces.

(d) Placas separadoras laterales.

(e) Embellecedores laterales.

(f) Cubiertas laterales.

Figura 4.5. Montaje y piezas parte lateral.

UUna vez tenemos modelado los laterales procederemos a modelar y distribuir el espacio interior. A la hora de modelar el interior tenemos que tener en cuenta las dimensiones de los componentes electrónicos más significativos que vamos a usar. En primer lugar, tendremos que dividir el espacio interior en tres partes diferenciadas.

La primera parte es la zona de alimentación, aquí irá alojada toda la electrónica de potencia encargada de alimentar al resto del sistema. Esta zona de alimentación irá ubicada en la parte inferior del sistema principal- mente por su considerable peso con respecto al resto de componentes, ba- jando consecuentemente el centro de masas del dispositivo. Además, colocar la alimentación en la parte inferior nos permite situar fácilmente la toma de corriente cerca de la base.

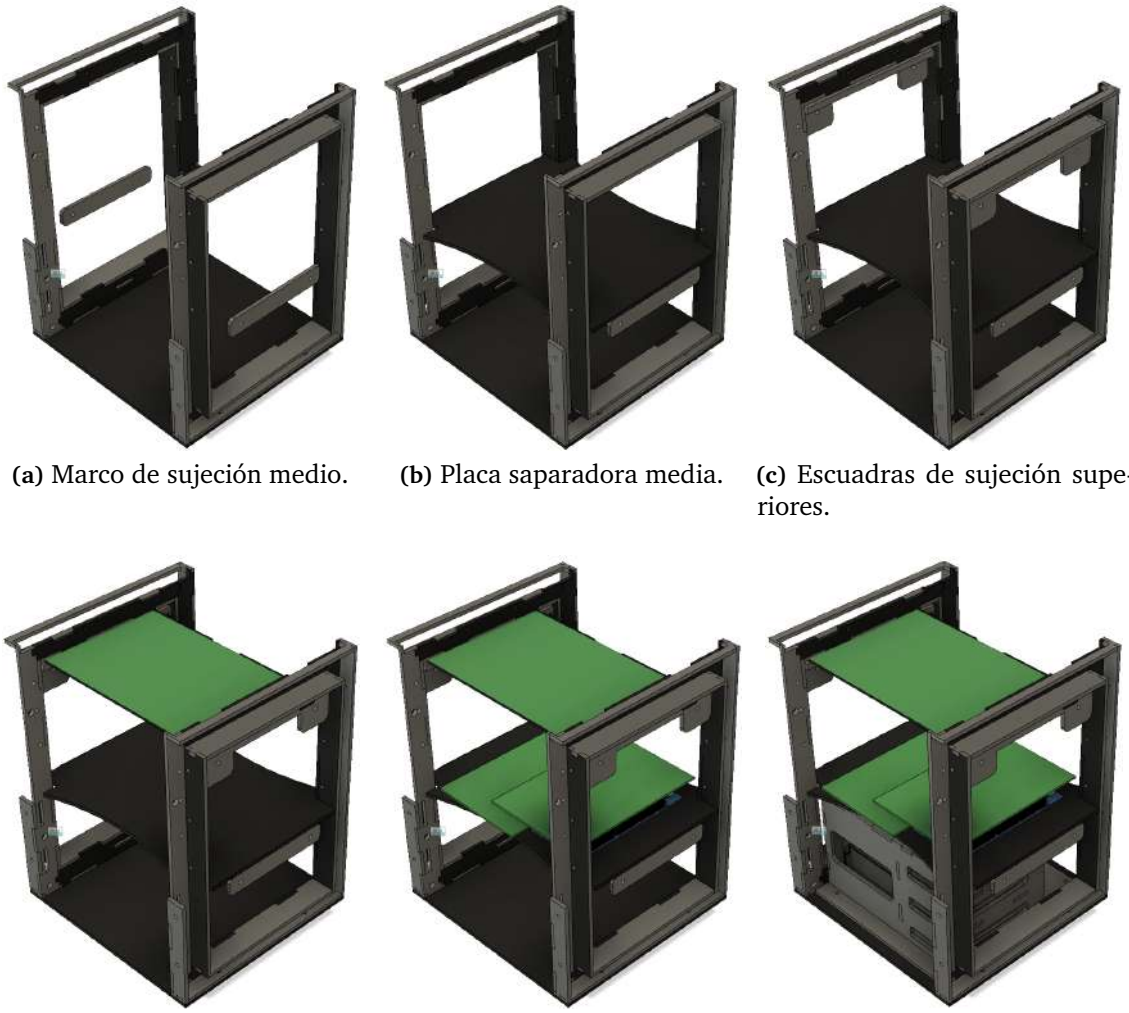
La segunda zona estará dedicada a las comunicaciones. Las placas de comunicaciones son especialmente susceptibles a interferencias electromagnéticas por lo que es necesario que estén asiladas de la electrónica de potencia. Para disminuir el efecto de dichas interferencias, las placas de comunicaciones se situarán en la parte superior del dispositivo. Esta ubicación presenta una ventaja adicional, ya que es ideal para la colocación de antenas que mejoran la recepción y transmisión de señales.

Finalmente, la zona intermedia estará reservada para el resto de componentes, como son la placa madre, Raspberry Pi, placas de expansión, etc. La ubicación de la placa madre en la parte central nos facilita la colocación de puertos en la parte posterior del dispositivo, así como acceso directo para conectar todos los periféricos.

Para implementar esta división en tres zonas de la parte interior del sistema se modelarán tres piezas importantes: Marco de sujeción medio, placa separadora media y escuadras de sujeción superiores.

Como ya hemos comentado, la fuente de alimentación es un componente muy importante a tener en cuenta para el modelado del sistema debido a su gran volumen y peso con respecto al resto de los componentes electrónicos. Es preciso tener en mente que tipo de fuente utilizar para modelar el sistema de ventilación necesario, así como su dimensionado.

Para poder alimentar a todos los periféricos del sistema se requiere tres niveles de voltaje en corriente continua: 12V, 5V y 3.3V. Para conseguir dichos voltajes primero tenemos que transformar los 230V AC de la toma de corriente a un valor manejable DC, para su posterior transformación. Para ello, utilizaremos dos transformadores AC/DC que nos permite tener 24DC filtrados. Estos 24V serás posteriormente transformados por tres buck- converters a los niveles de voltaje deseados.



(a) Marco de sujeción medio. **(b)** Placa saparadora media. **(c)** Escuadras de sujeción superiores.

(d) Montaje placas de comunicaciones. **(e)** Montaje placa madre. **(f)** Montaje fuente de alimentación.

Figura 4.6. Montaje y piezas parte lateral.

Con el objetivo de modelar un recubrimiento para todos estos dispositivos se ha modelado la fuente de alimentación de la figura 21. Como se puede apreciar en esta figura, la carcasa dispone de dos compartimentos en el lateral derecho para los trasformadores AC/DC y tres para los transformadores DC/DC. Todos estos compartimentos están diseñados con las dimensiones precisas para integrarse con el resto del sistema. Además, se han añadido orificios de ventilación acorde con el resto del dispositivo.

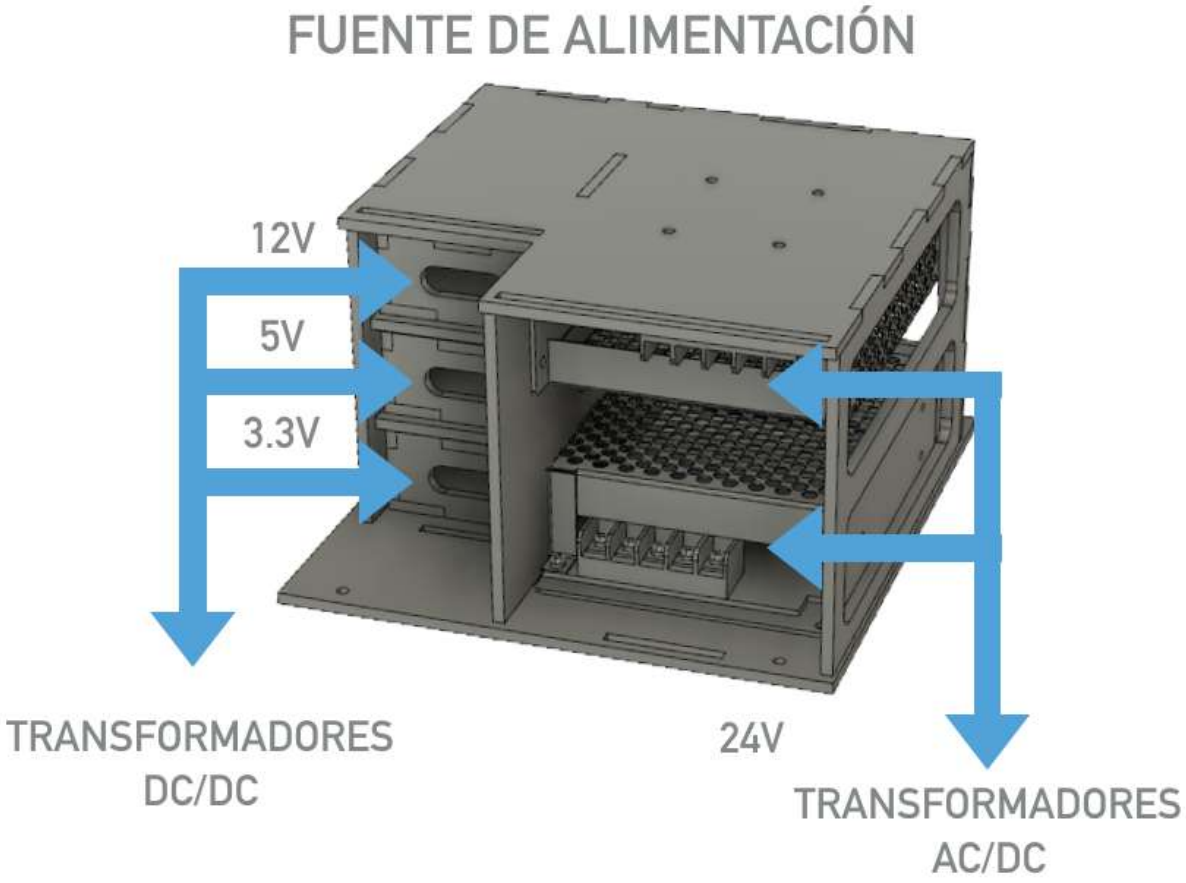


Figura 4.7. Modelo de la fuente de alimentación.

Después de haber diseñado la distribución interior, podemos diseñar la parte superior del dispositivo. En esta parte irá alojado el cubo led, que separa el interior del sistema con el exterior. Esta parte consta de tres piezas principales: La placa separadora superior, la cubierta superior y el cubo de metacrilato. (Figura 4.8)

(a) Placa separadora superior. **(b)** Cubierta superior. **(c)** Cubo de metacrilato.

Figura 4.8. Montaje y piezas de la parte superior.

Finalmente, es necesario diseñar la parte posterior. Esta parte consta simplemente de una cubierta trasera que se atornilla a los marcos estructurales. Esta cubierta trasera necesita agujeros para los diferentes puertos que se utilizarán, así como una ranura para la alimentación. (Figura 4.9)

Figura 4.9. Modelo de la cubierta posterior.

Como resultado final, ensamblando todas las piezas anteriormente modeladas obtenemos el modelo final:

(a) Montaje sin cubiertas ni pantalla.

(b) Montaje sin cubiertas.

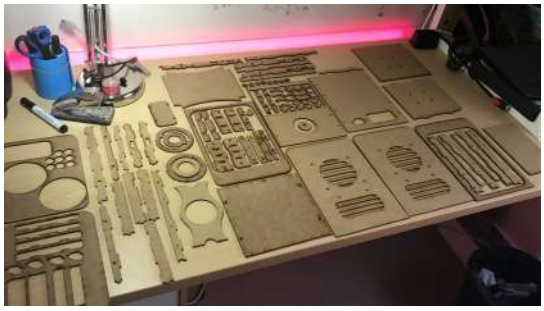
(c) Montaje completo de El Nodo.

Figura 4.10. Montaje completo de El Nodo.

Figura 4.11. Render del modelo final.

4.1.3. Montaje y fabricación

Una vez tenemos listo el modelo 3D con todas las medidas y junto con los componentes electrónicos, solo falta fabricar las piezas y montarlas según el procedimiento anteriormente descrito. Para comenzar, se han cortado las piezas principales con la cortadora láser en madera DM de 3mm de espesor, así como las piezas de metacrilato de la paste superior y lateral.



(a) Piezas de madera DM.

(b) Piezas de metacrilato.

Figura 4.12

Con las piezas ya cortadas, conformamos las piezas finales uniendo los fragmentos con cola blanca y tornilleria para, a continuación, comprobar que las dimensiones son las correctas:

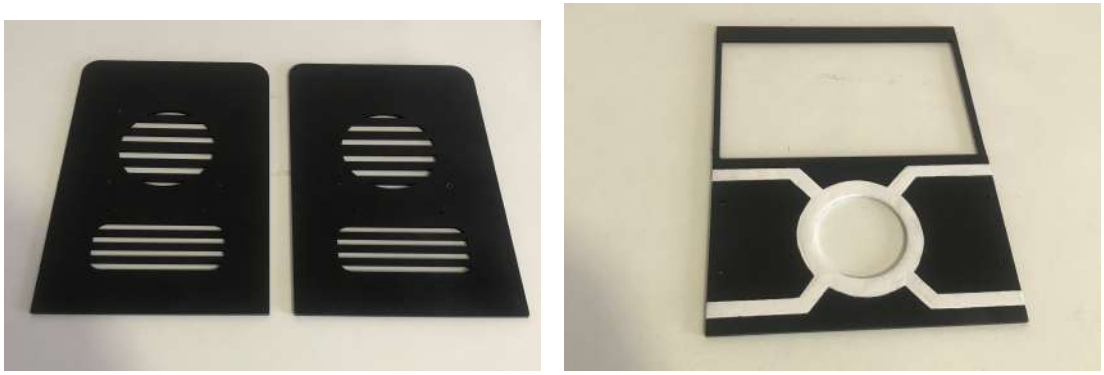
(a) Montaje del marco de madera

(b) Montaje completo de madera.

Figura 4.13

Una vez se ha comprobado que las piezas de madera y metacrilato tienen las dimensiones correctas se procede a aplicar una capa de pintura para darle el acabado superficial deseado:

Una vez están cortadas y ensambladas las piezas de madera y metacrilato, se procederá a imprimir en 3D algunas de las piezas de geometría más compleja. Estas piezas se han impreso en una impresora Prusa i3 usando como material de extrusión ABS negro por su bajo coste y la alta resistencia .



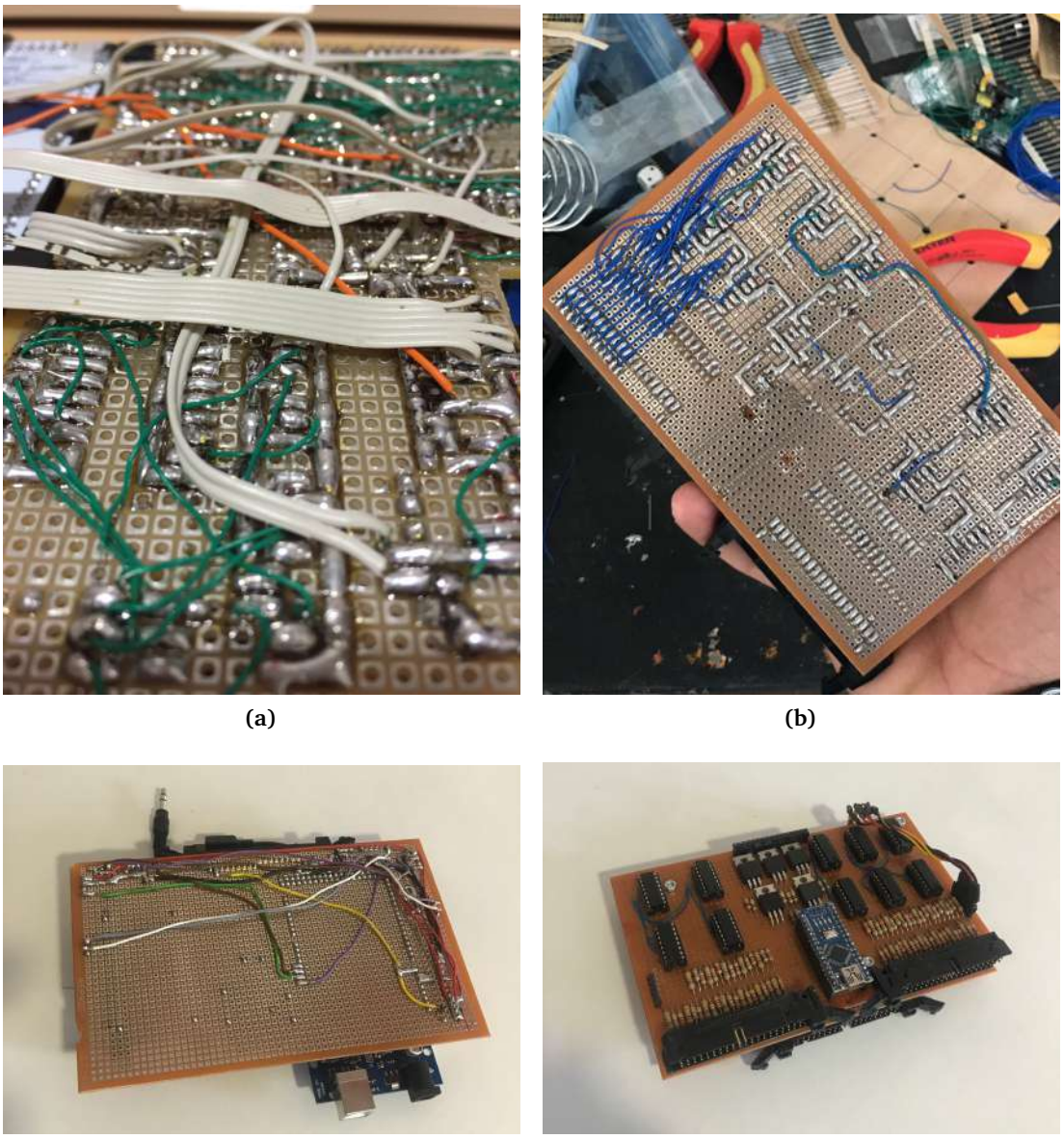
(a) Pintura cubiertas laterales. **(b)** Pintura panel frontal.

Figura 4.14

Figura 4.15. Piezas impresas en ABS.

Con todas estas piezas damos por concluida la fabricación de los elementos estructurales del sistema y comenzamos la fabricación y montaje de los componentes electrónicos. Para ello se han soldado los componentes mostrados en los esquemas electrónicos de la sección anterior en placas perforadas especiales para soldadura con estaño.

Junto con el ensamblaje de las placas electrónicas se ha construido la matriz de leds RGB, usando una pieza de madera perforada como plantilla par distribuir estos leds de forma uniforme. Primero se han conectado formando los pisos del cubo agrupando 25 leds para más adelante soldar cada piso a una misma linea de conexión vertical que se incrusta en la base del cubo.



(a) **(b)**

(c) **(d)**

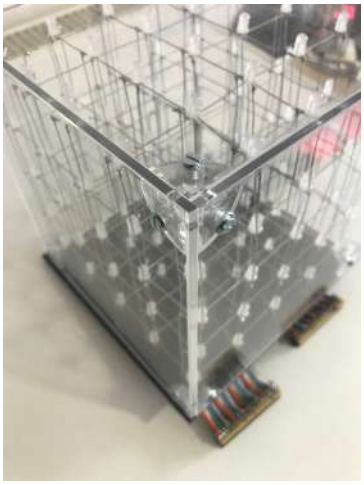
Figura 4.16. Soldadura de placas electrónicas.

Llegado este punto, todas las piezas están listas para ser ensambladas entre sí para conformar el dispositivo final. Para realizar este ensamblaje se requiere de tornillería de métricas 3, 4 y 5 exclusivamente, así como las tuercas y arandelas correspondientes. Para dar una idea de las piezas que componen el sistema completo, en la siguiente figura se agrupan todos los elementos antes de ser montados:

Finalmente realizamos el montaje final del sistema:

(a) Ensamblaje piso.

(b) Ensamblaje cubo.

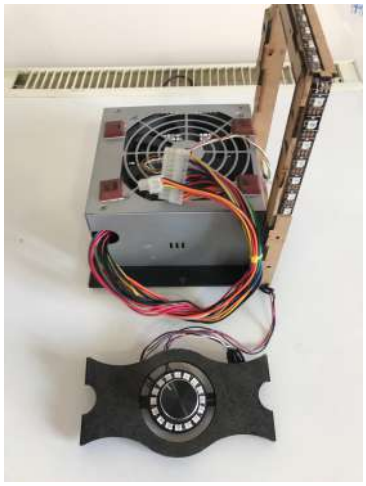


(c) Cubo completo.

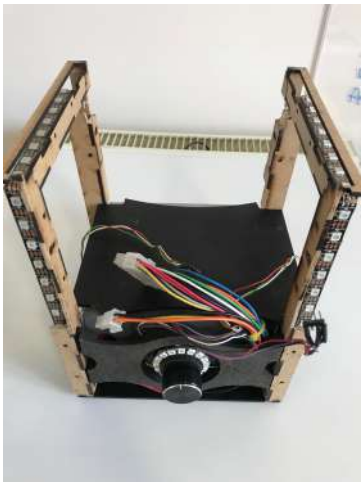
Figura 4.17. Montaje cubo led.

Figura 4.18. Piezas de El Nodo.

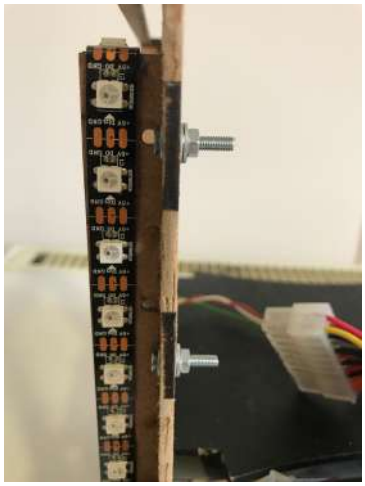
(a) Base y fuente.



(b) Marcos laterales.

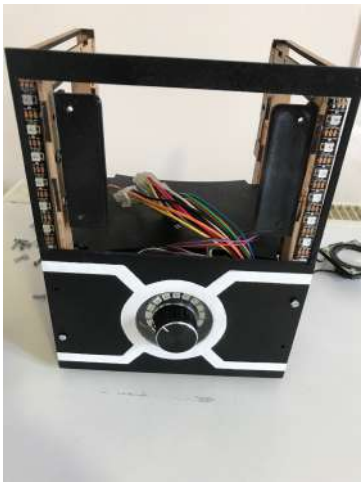


(c) Dial y separador.



(d) Tornilleria marco.

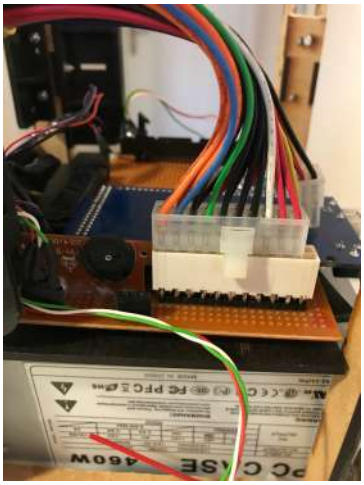
(e) Escuadras laterales.



(f) Panel frontal.

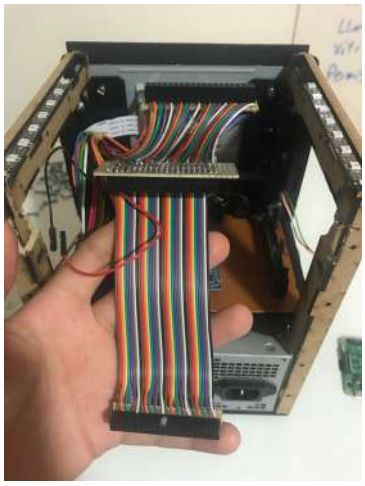
(g) Placa madre.

(h) Conexión leds.



(i) Alimentación.

Figura 4.19



(j) Bus de comunicación.

(k) Altavoces laterales.

(l) Puertos traseros.

(m) Placa trasera.

(n) Placas laterales.

(ñ) Comunicaciones.

(o) Pantalla tactil.

(p) Cubo led.

(q) Nodo ensamblado.

Figura 4.19. Montaje general de El Nodo.

5 Software

5.1. Software

Llegados a este punto es hora de hablar de la otra dimensión del proyecto, el Software. Como cabe esperar el software es un elemento fundamental en el desarrollo del proyecto, siendo este mismo el que aporte al hardware la capacidad para realizar todas las funcionalidades propuestas. Debido a la complejidad del proyecto, el software implementado en el sistema no se limita a un único proceso o código, sino que se trata de una composición de diferentes procesos que se entrelazan entre si para gestionar el sistema en su conjunto. Al igual que con el hardware, el software se compone de tecnologías diversas, cada una de estas especificas para acometer las diferentes tareas que se requieran.
Para explicar con detalle cada uno de los procesos que tienen lugar en el sistema primero realizaremos una pequeña descripción de la estructura general del software para después atender cada una de las instancias por separado de forma más exhaustiva.

5.1.1. Estructura general del software

Como habíamos visto en el capitulo 3.1 de diseño general, el sistema se puede dividir en cinco grupos diferenciados: La interfaz, el servidor de El Nodo, el servidos Openhab, el proceso Matlab y el gestor de hardware. Esta división es extrapolable al ambito del software con la inclusión de un nuevo elemento externo al dispositivo, la nube.

Uno de los requisitos a la hora de diseñar El Nodo es su capacidad para establecer comunicaciones con todo tipo de dispositivos de forma cómoda y flexible. Teniendo este aspecto en cuenta se ha modelado todo el sistema con la idea de generar un software basado en tecnología web. Este tipo de tecnología nos permite expandir la funcionalidad de El Nodo mas allá del propio dispositivo, siendo accesible desde cualquier otro dispositivo con acceso a Internet como ordenadores, tablets e incluso móviles.

El dispositivo diseñado esta pensado para permanecer en el hogar en una posición fija, es por ello que la portabilidad de su funcionalidad reside en su conectividad. Todo el sistema esta pensado para ser controlado remotamente desde cualquier dispositivo móvil con acceso a Internet. Este proceso de conexión entre El Nodo e Internet se realiza a través de La Nube.

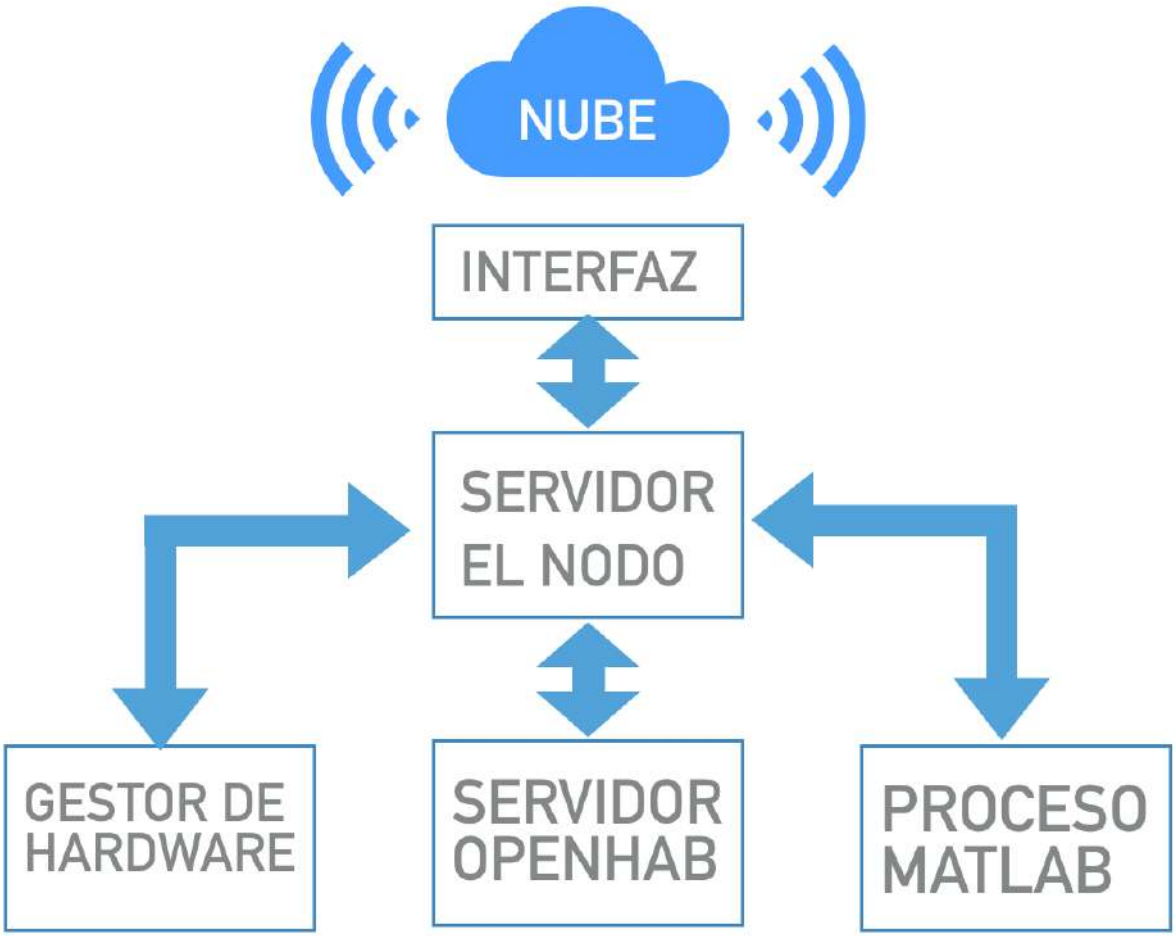


Figura 5.1. Esquema estructura general del software.

En esta sección explicaremos brevemente que tenologia se ha emplado para realizar cada uno de los bloque en cuestión, así como describir su funcionalidad a grandes rasgos, para más adelante concretar.

- **Servidor de El Nodo:**
  El servidor de el Nodo es el proceso principal del sistema. Su función es la de coordinar todos los demás bloques en el sistema, así como gestionar la información que recibe para almacenarla en una base de datos si procede. Como su propio nombre indica se trata de un proceso de servidor, es por ello que su funcionamiento está basado en la respuesta ante eventos que puedan ocurrir en el sistema. Estos eventos se activan generalmente por la llegada de información por alguno de los puertos habilitados. Cuando otro bloque envía información al servidor de El Nodo este activa un evento de escucha y respuesta para atender a la petición de dicho bloque. Estas peticiones pueden ser simplemente reenviadas a otro modulo del sistema o ejecutar alguna rutina en especial. Cada petición que se realiza se procesa de forma independiente y de forma asíncrona, con el objetivo de poder atender a múltiples peticiones al mismo tiempo.

- **Servidor de Openhab:**
  El servidor de Openhab es un proceso basado íntegramente en la iniciativa opensource de Openhab. Openhab (open Home Automation Bus) es un programa multiplataforma para la gestión de dispositivos domóticos que integra una gran variedad de protocolos empleados

en el mundo de la domotica independientemente de la tecnologia que se use. Openhab cuenta con una comunidad muy activa de desarrolladores que se dedican a dar soporte para el control de multitud de dispositivos comerciales. En el sistema Openhab tiene una función de interprete para las comunicaciones con los dispositivos que se conecten a el, estableciendo comunicación con el resto del sistema mediante un servidor http que el propio Openhab ejecuta en segundo plano.

- **Proceso de Matlab:**
  Al igual que Openhab, el proceso de Matlab es un proceso creado por terceros, en este caso se trata de un software desarrollado por Matworks. Este software permite ejecutar modelos de Matlab y Simulink en la Raspberry Pi de forma autónoma, teniendo acceso al hardware de El Nodo gracias a la implementación de comunicaciones con el servidor de El Nodo mediante protocolo TCP/IP. El uso de Matlab y Simulink en el sistema expande las opciones de programación de rutinas y modelos de control para los dispositivos de la casa. Permite realizar controles avanzados para la optimización energética de la casa, especialmente en el caso de la climatización.

- **Gestor de hardware:**
  Por otro lado el gestor de hardware es el encargado de controlar los diferentes perifericos internos de los que El Nodo dispone. Dicho gestor en lo que a software respecta cabe destacar dos drivers importantes. El primero es el coordinador, este driver es el que se ejecuta en la placa madre del sistema y su función principal es la de comunicarse con todos los perifericos internos y procesar las señales para posteriormente transmitirlas al procesador principal. En segundo lugar tenemos el driver de control de la matriz led. este driver gestiona las animaciones de la matriz de leds situada en la parte superior del dispositivo. Este driver esta conectado al coordinador de la placa madre para recibir los comandos pertinentes a que información mostrar en la matriz. La comunicación entre el gestor de hardware y el servidor de el nodo se realiza a través del protocolo SPI.

- **Interfaz:**
  Para poder interactuar con el sistema de forma fácil y sencilla es preciso contar con una interfaz gráfica adecuada para el sistema. Esta interfaz ha sido programada utilizando lenguajes de programación orientados a la web. La tecnología web presenta algunas claras ventajas frente a otras alternativas como Java o C a la hora de realizar la interfaz gráfica. La primera es su fácil programación gracias a la utilización de un motor de navegación, el cual interpreta el código con una gran robustez ante los errores. La segunda razón es su alta portabilidad, siendo el mismo código ejecutable en cualquier otro dispositivo capaz de ejecutar un navegador web. Ademas, al tratarse de un lenguaje de programación orientado a web, la implementación de la comunicación con el servidor de El Nodo es trivial mediante peticiones Http.

- **La nube:**
  La Nube, permite al usuario controlar y visualizar toda la información que manipula El Nodo teniendo acceso directo a toda su funcionalidad. Este acceso se produce mediante la constante comunicación que se genera entre el servidor de El Nodo y el dominio web. Esta comunicación se produce a través del protocolo http para enviar los diferentes comandos y datos de un lado a otro. Es por tanto que gracias a la implemantación de la nube es posible establecer comunicaciones con el sistema desde cualquier lugar del mundo, siendo este mecanismo un Gateway para controlar el sistema desde Internet. Este tipo de tecnologia presenta un gran potencial para extender la funcionalidad del sistema, sin

embargo tambien presenta una gran amenaza por problemas de intrusión. Es por este motivo que es necesario aplicar sistemas de seguridad avanzados para prohibir el acceso a usuarios no autorizados.

Como hemos visto cada bloque del sistema tiene su proposito y esta programado con lenguajes de programación diferentes para cumplir su cometido de la forma mas eficiente posible. A continuación se muestra una pequeña figura que muestra los lenguajes y tecnologias empleados en cada caso:

Figura 5.2. Lenguajes y tecnologías empleados.

5.1.2. Configuración del sistema operativo

Comenzando por el propio arranque del sistema es importante destacar algunas modificaciones que se han hecho al sistema operativo para que se ejecuten determinadas rutinas en el arranque asi como la propia sustitución de los elementos de

5.1.3. Servidor de el Nodo

Para empezar a adentrarnos en el funcionamiento del software de El Nodo empezaremos por uno de los bloques más importantes del sistema, el servidor de El Nodo. Este proceso es el encargado de gestionar el resto de bloques del sistema, haciendo de mediador en las comunicaciones tanto internas como externas. La implementación de este servicio nos ofrece la posibilidad de extender la funcionalidad de programas de terceros como son Matlab y Openhab haciendo de puente entre estos y el resto del sistema. Además es capaz de gestionar elementos como son la interfaz gráfica y las comunicaciones con el gestor de hardware haciendo del sistema un conjunto de procesos centralizados en este servidor.

Este servidor se trata de un proceso local que se ejecuta en el procesador de la Raspberry Pi bajo permisos de administrador, siendo por tanto capaz de generar comandos de sistema para realizar modificaciones en el comportamiento del sistema operativo linux instalado en la

Raspberry. Estos permisos permiten a el servidor de El Nodo obtener información de sistema, cambiar la configuración e incluso modificar y guardar archivos internos.

Existen varias tecnologías posibles para integrar este servicio en la Raspberry Pi, no obstante se ha optado por implementar un servidor Node.js por su gran versatilidad y compatibilidad multiplataforma. Node.js es un servidor basado en el lenguaje de programación Javascript, lenguaje ampliamente utilizado en el mundo del desarrollo web por su sencillez y alta funcionalidad. este lenguaje nos permite construir multiples servidores en un mismo proceso, siendo capaz de atender a peticiones con diferentes tecnologias como http o tcp/ip sin necesidad de ejecutar multiples instancias del servidor.

En conjunción con la capacidad de Node.js para habilitar los servidores destaca por otro lado la gran cantidad de módulos de extensión de los que dispone. En este proyecto se hará uso de varios modulos de extensión, no obstante es importante destacar el framework Electron. Electron es un proceso que integra la ejecución de un servidor Node.js con el navegador Chromium, haciendo posible generar una aplicación visual basada en el binomio servidor-cliente típico de la programación web. De este modo podemos generar una interfaz gráfica que se integre perfectamente con el servidor de El Nodo.

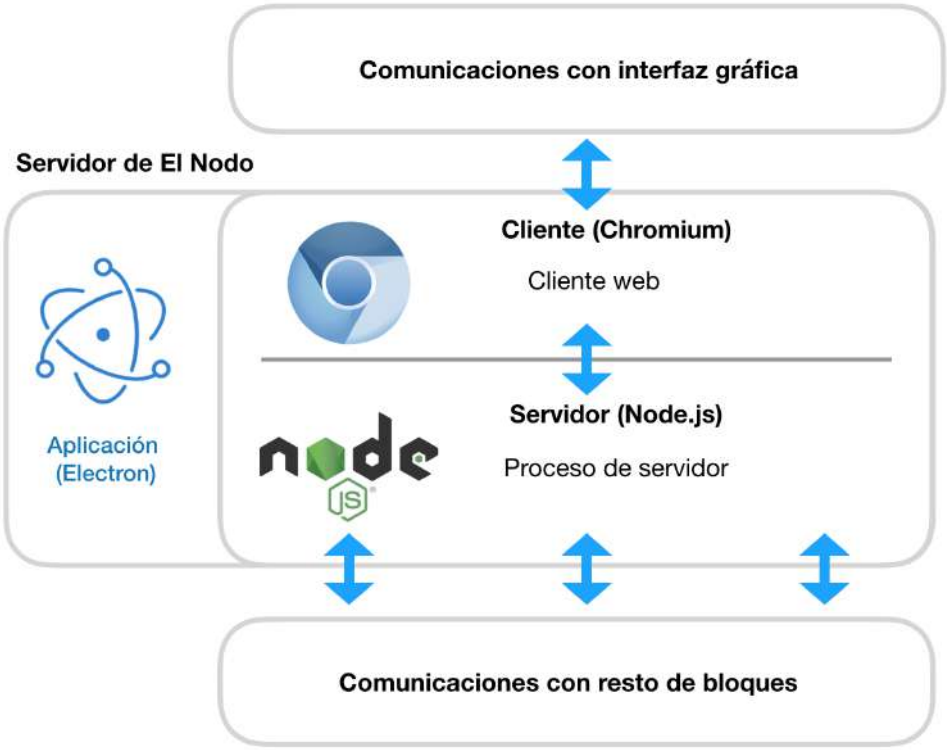


Figura 5.3. Conexión de los módulos de el servidor de El Nodo

Como adición a la funcionalidad que nos aporta la implementación del framework de electron se la suma la integración de una base de datos para la manipulación de la información del sistema. La base de datos empleada es Mongo DB, escogida por sus caracterirticas de base de datos no relacional y por su gran compatibilidad y rendimiento junto con Node.js. Finalmente, para agregar una capa extra de compatibilidad con el sistema se integran procesos desarrollados en C++ que dotan a el bloque de servidor de El Nodo de una mayor capacidad para manipular

el sistema de archivos y ajustes internos de Linux.

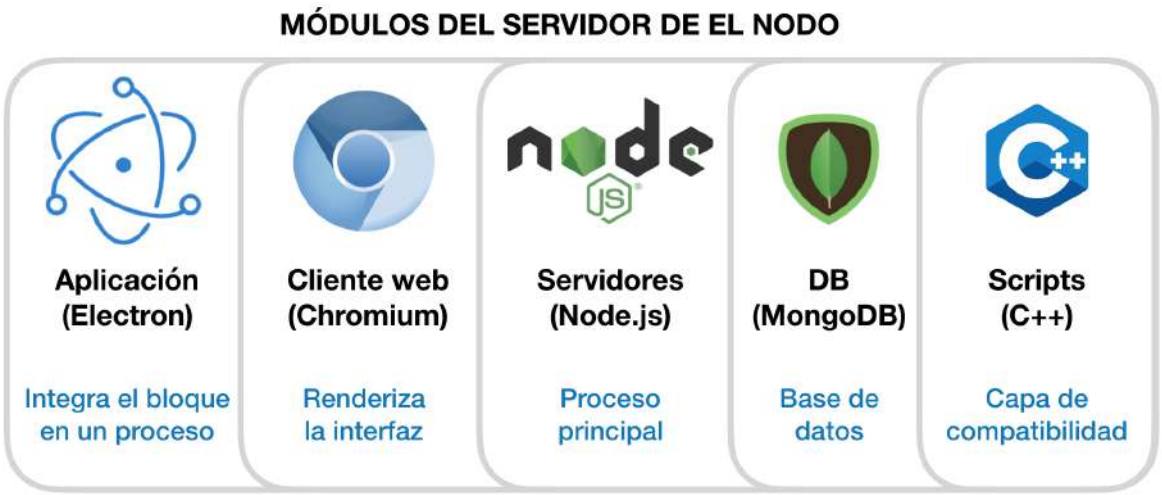


Figura 5.4. Módulos del servidor de El Nodo

Una vez tenemos claro los módulos de los que esta compuesto el bloque del servidor queda diseñar como se realizarán las comunicaciones con los distintos bloques. En primer lugar tenemos el servidor Openhab, el cual como ya comentabamos dispone de un pequeño servicio HTTP con el que podemos interactuar a través de una Rest API. Para enviar solicitudes a este servicio usaremos peticiones HTTP tales como POST o GET. Este tipo de peticiones tambien nos será utili para establecer conexión con la interfaz grafica y con la nube, al tratarse ambas de procesos basados en el protocolo HTTP. Otra conexión importante es la que tiene lugar con el proceso de Matlab y Simulink. Para realizar esta comunicación se usara el protocolo TCP y UDP, ya que se trata de un protocolo de facil implementación en modelos de Matworks. Por último, la comunicación con el gestor de hardware se realizará a traves de los pines de la Raspberry Pi, más especificamente el puerto SPI.

Llegados a este punto solo nos queda explicar por encima el código empleado en este proceso para habilitar todas las funcionalidades descritas hasta el momento. Se tratará de aportar una idea general sobre el funcionamiento general del programa, sin embargo por temas de extensión solo se mostrarán las partes que se consideran más relevantes.

Comenzando por la inicialización del programa, las primeras lineas de código estan encargadas de incluir algunos de los módulos de los que dispone el servidor. Estos módulos son la aplicación electron, encargada de ejecutar el cliente de chromium, el modulo HTTP, especifico para realizar las peticiones POST y GET en el sistema y el módulo NET, cuya funcionalidad es la de permitir abrir comunicaciones tanto TCP como UDP. Nótese tambien que se incluyen paquetes como url, path y querystring. Estos paquetes serviran de soporte para realizar algunas rutinas de forma más sencilla.

Tras incluir todos los módulos necesarios declaramos el objeto encargado de contener la interfaz gráfica llamado mainWindow (ventana principal). Se define a su vez un constructor par adicho objeto donde se especificará la configuración de dicha ventana así como la ruta para

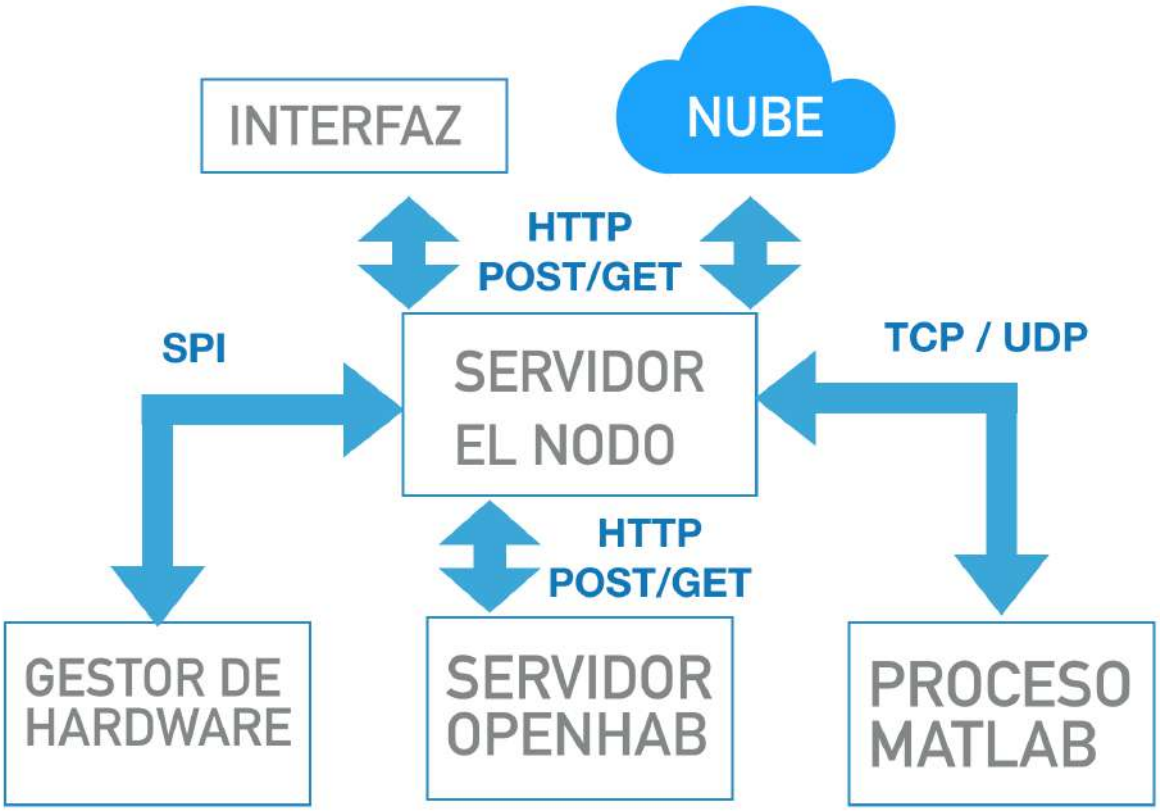


Figura 5.5. Comunicaciones servidor de El Nodo.

```
// Definiciones

const electron = require('electron')
const path = require('path')
const url = require('url')
const net = require('net');
const http = require('http');
const querystring = require('querystring');
```

Figura 5.6. Declaración de módulos.

el sistemade archivos web que suinistrarán a la interfaz.

Para manipular el objeto de ventana principal se crearán eventos especiales para la llamada a los diferentes metodos del objeto. Algunos de estos métodos son el contructor, llamado cuando la aplicación electron este lista, el cierre del proceso cuando no queden ventanas abiertas y su activación mientras el programa este ejecutándose.

En este punto tenemos creada la aplicación responsable de porcesar la interfaz gráfica, así como algunos eventos de control asociados a la misma. Sin embargo todavia esta por establecerse las conexiones pertinentes con Openhab, Matlab y Simulink y la nube.

Comenzaremos por dar soporte al proceso de Matlab habilitando un servidor TCP en el puerto 1337. La conexión se establece dentro del mismo dispositivo por tanto la ip del proceso

```
let mainWindow

function createWindow () {

  // Crea la interfaz principal
  mainWindow = new BrowserWindow({width: 800, height: 480})

  // Carga es servidor de archivos de la interfaz -- (Carga index.html)

  mainWindow.loadURL(url.format({
    pathname: path.join(__dirname, 'Aplicacion/index.html'),
    protocol: 'file:',
    slashes: true
  }))

  // Open the DevTools.
  // mainWindow.webContents.openDevTools()

  // Evento de cierre de aplicacion
  mainWindow.on('closed', function () {

    mainWindow = null
  })
}
```

Figura 5.7. Objeto de ventana principal y su constructor.

```
//Evento de inicio de la aplicacion
app.on('ready', createWindow)

// Cerrar proceso cuando todas las ventanas esten cerradas.
app.on('window-all-closed', function () {
  if (process.platform !== 'darwin') {
    app.quit()
  }
})

app.on('activate', function () {
  if (mainWindow === null) {
    createWindow()
  }
})
```

Figura 5.8. Declaración de módulos.

se Matlab y Simulink será la local "127.0.0.1". La conexión se declara al crear el objeto especial "net.Socket().ªsignada a la variable "simulinkTCP", siendo por tanto dicha variable el manejador para la conexión entre ambos procesos.

Este servidor atenderá las peticiones que se envíen desde los modelos de Matlab y Simulink mediante el evento de escucha "server.listen", recibiendo los datos en una variable "dataz

procesados dentro de la propia función del socket del servidor creado.

```
//************** Inicio Servidor TCP ******************
var simulinkTCP = new net.Socket();
simulinkTCP.connect(8081, ipSimulink, function() {
    console.log('Conectado a Simulink');
    simulinkTCP.write("OK");
  });

simulinkTCP.on('data', function(data) {
    console.log('Recibido: ' + data);
  });

  simulinkTCP.on('close', function() {
    console.log('Cerrando conexion');
  });

//---------------Escucha-------------------------------
var server = net.createServer(function(socket) {

  //socket.pipe(socket);
  //simulinkTCP = socket;
  // Recepcion de mensajes
  socket.on('data', function (data) {
    //Recepcion
    console.log("Recibido: ");
    console.log(data.toString());
    //Respuesta
    console.log("Respuesta: ");

    if(data.toString()=="1"){
      console.log("OK, Encendiendo lampara");
      comando("ON");
    }else{
    if(data.toString()=="0"){
      console.log("OK, Apagando lampara");
      comando("OFF");
    }else{
      console.log("COMANDO - ERROR");
    }}

  });

});
server.listen(1337);
console.log("Servidor TCP escuchando en puerto 1337");
```

Figura 5.9. Declaración del servidor TCP para el proceso Matlab.

Por otro lado, el servidor TCP también puede enviar datos a los modelos de Matlab y Simulink gracias a la implementación de funciones de transmisión de datos, las cuales escriben en el socket que contenido en la variable "simulinkTCP"mediante el método "simulinkTCP.write();".

```
//---------------Transmite---------------------------------
function transmitirSimulink(mensaje){
    console.log('Enviando a Simulink: '+mensaje);
    simulinkTCP.write(mensaje);
}

function transmitirSimulink2(mensaje){
    simulinkTCP.connect(8081,ipSimulink, function() {
    console.log('Enviando a Simulink: '+mensaje);
    simulinkTCP.write(mensaje);
  });

  simulinkTCP.on('data', function(data) {
    console.log('Received: ' + data);
     // kill client after server's response
      simulinkTCP.destroy();
  });

  simulinkTCP.on('close', function() {
    console.log('Connection closed');
  });
}
```

Figura 5.10. Funciones de transmisión de datos al proceso Matlab.

Finalmente, queda por implementar la conexión con Openhab y la nube. Como se comentó anteriormente, esta conexión se realizará mediante el protocolo HTTP y peticiones de carácter GET y POST.

De forma análoga a como se realizo el servidor TCP el servidor HTTP atenderá a las peticiones entrantes mediante el método "listen()", abriendo en este caso el puerto 8000. Respecto a la dirección IP de los componentes, Openhab se encuentra en la dirección local "127.0.0.1"puesto que se trata de un proceso más que se ejecuta en la propia Raspberry, sin embargo la nube se ubica en un dominio externo, siendo más delicada la conexión con el mismo por temas de seguridad. Una opción para la conexión con la nube sería abrir el puerto a cualquier dirección IP, sin embargo esta flexibilidad expone al sistema a ser posiblemente controlado por terceros sin autentificación. Para realizar la conexión sin comprometer la seguridad del dispositivo se han tenido que implementar algunos metodos de autentificación, los cuales serán explicados posteriormente.

Además de atender a las peticiones entrantes el servidor tiene que ser capaz de generar sus propias peticiones GET y POST. Estos dos tipos de peticiones se realizan mediante la funcion .enviarGET(peticion);"para paticiones GET y la funcion .enviarPOST(peticion);"para la función POST.

En las siguientes figuras (5.12 y 5.13) se muestra un ejemplo de estas funciones para la petición GET a Openhab solicitando el estado de un interruptor y la petición POST cambiando el estado de dicho interruptor.

```
//*************** Inicio Servidor HTTP ******************
http.createServer((request, response) => {
  const { headers, method, url } = request;
  let body = [];
  request.on('error', (err) => {
    console.error(err);
  }).on('data', (chunk) => {
    body.push(chunk);
  }).on('end', () => {
    body = Buffer.concat(body).toString();
    console.log(body);
    comandoInterfaz(body);
    // Inicio Respuesta
    response.on('error', (err) => {
      console.error(err);
    });
    response.statusCode = 200;
    response.setHeader('Content-Type', 'application/json');

    const responseBody = { headers, method, url, body };

    response.write(JSON.stringify(responseBody));
    response.end();

  });
}).listen(8000,ipInterfaz);

console.log("Servidor HTTP escuchando en puerto 8000");
//*************** Final Servidor HTTP ******************
```

Figura 5.11. Inicialización del servidor HTTP.

5.1.4. Proceso MATLAB y Simulink

Una de las funcionalidades más destacadas del proyecto es el hecho de poder ejecutar complejos algoritmos de control para la gestión de los diferentes dispositivos del hogar. Esta funcionalidad se debe principalmente a la integración de Matlab y Simulink en el procesador principal de El Nodo. Matlab y Simulink representan un proceso más que se ejecuta en paralelo junto con el servidor de El Nodo, estando en segundo plano desde el arranque. Cada modelo o codigo generado por Matlab y Simulink puede ser cargado dentro de El Nodo mediante conexión SSH con el ordenador para que se ejecute de forma autonoma o incluso para ser ejecutado externamente para la monitorización de las variables internas del modelo mientras se ejecuta desde el ordenador.

Los modelos cargados en El Nodo son gestionados por el servidor de El Nodo, siendo este mismo capaz de arrancar y parar un modelo en especifico. Esta forma de gestionar los modelos permite a su vez la creación de multiples instancias del proceso de Matlab, siendo posible ejecutar varios scripts al mismo tiempo.

Dicho proceso tiene acceso prácticamente integro al hardware del resto del sistema gracias a la interacción con el servidor de El Nodo, el cual gestiona las peticiones de información ejercidas por Matlab actuando segun corresponda con la petición. Ejemplos de estas peticiones pueden

```
//*************** Peticion GET ************************

function enviarGET(peticion) {
  // Build the post string from an object
  /*
  var post_data = querystring.stringify({
    'compilation_level' : 'ADVANCED_OPTIMIZATIONS',
      'output_format': 'json',
      'output_info': 'compiled_code',
        'warning_level' : 'QUIET',
        'js_code' : peticion
  });
 */

  var post_data = peticion;
  // An object of options to indicate where to post to
  var post_options = {
      host: '172.20.1.234',
      port: '8080',
      path: '/rest/items/zwave_device_3417dab8_node2_switch_binary',
      method: 'GET',
      headers: {
          'Content-Type': 'text/plain',
          'Content-Length': Buffer.byteLength(post_data)
      }
  };

  // Set up the request
  var post_req = http.request(post_options, function(res) {
      res.setEncoding('utf8');
      res.on('data', function (chunk) {
          console.log('Respuesta Openhab: ' + chunk);
      });
  });

  // post the data
  post_req.write(post_data);
  post_req.end();

}
```

Figura 5.12. Función enviarGET().

ser la optención de variables importantes como puede ser la temperatura de una habitación o el estado de un actuador instalado en la casa, además de poder solicitar que se manipule el estado de algun dispositivo de la casa.

Estas características aportan a los modelos realizados en Matlab y Simulink la posibilidad de interactuar con todos los dispositivos de la casa de forma rapida y sencilla, extendiendo la funcionalidad de tanto el dispositivo como del propio software de Matworks.

```
//*************** Peticion POST ************************

function enviarPOST(peticion) {
  // Build the post string from an object
  /*
  var post_data = querystring.stringify({
    'compilation_level' : 'ADVANCED_OPTIMIZATIONS',
      'output_format': 'json',
      'output_info': 'compiled_code',
        'warning_level' : 'QUIET',
        'js_code' : peticion
  });
 */
  var post_data = peticion;
  // An object of options to indicate where to post to
  var post_options = {
      host: '172.20.1.234',
      port: '8080',
      path: '/rest/items/zwave_device_3417dab8_node2_switch_binary',
      method: 'POST',
      headers: {
          'Content-Type': 'text/plain',
          'Content-Length': Buffer.byteLength(post_data)
      }
  };

  // Set up the request
  var post_req = http.request(post_options, function(res) {
      res.setEncoding('utf8');
      res.on('data', function (chunk) {
          console.log('Respuesta Openhab: ' + chunk);
      });
  });

  // post the data
  post_req.write(post_data);
  post_req.end();

}
```

Figura 5.13. Función enviarPOST().

La implementación de modelos en el sistema se hace especialmente sencilla gracias a la implicación del servidor de El Nodo, puesto que este mismo hace de API para todos los modelos de Matlab y Simulink. No obstante, como toda API para poder usarse correctamente existen algunas reglas que se deben seguir. Para ilustrar como debe de estructurarse un modelo de Matlab para interactuar con el hardware de El Nodo, pondremos un ejemplo de un modelo sencillo que encendera y apagará una lampara de acuerdo a una onda cuadrada de cinco segundo de ancho de pulso.

En primer lugar, es importante que definamos en el modelo una referencia temporal que nos asegure que el tiempo del modelo sea tiempo real transcurrido, para ello utilizaremos un

bloque llamado "Set Pace".

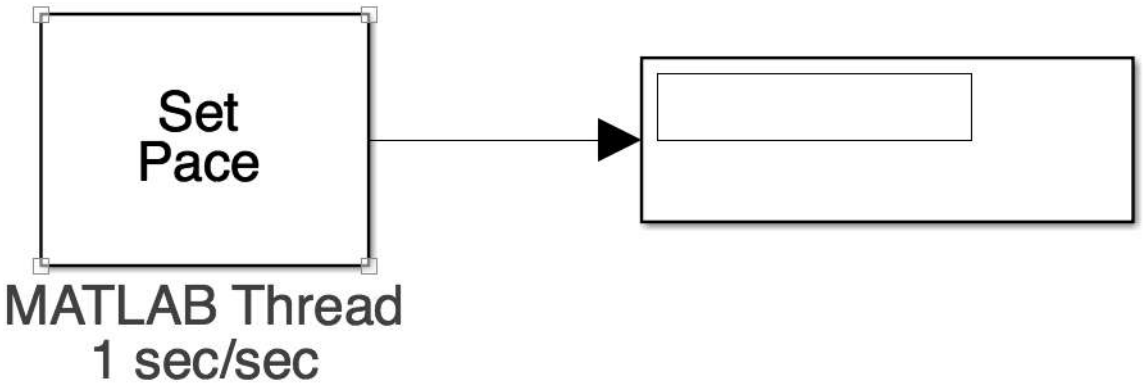
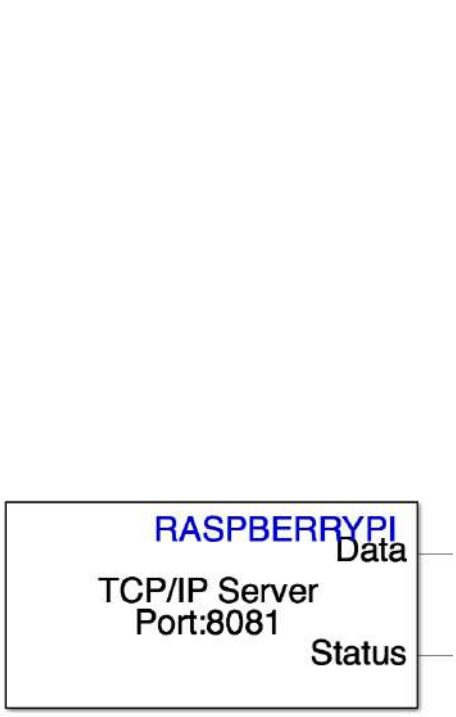
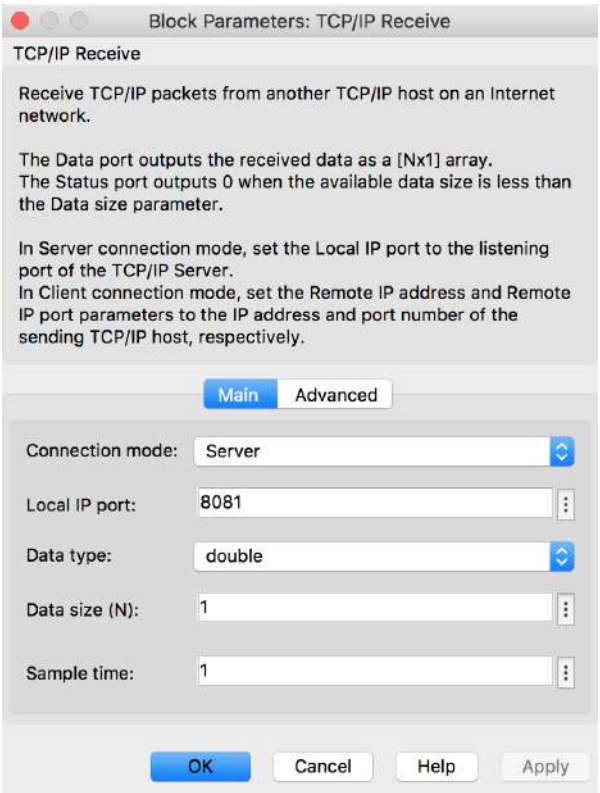


Figura 5.14. Aplicación del timpo real en el modelo

Además, el modelo debe contener un bloque que abra conexión para que el servidor de El Nodo se pueda conectar. Esta conexión se debe abrir habilitando la recepción de mensajes TCP en la IP local por el puerto 8081 usando el bloque de recepción TCP especial de la Raspberry Pi.



(a) Bloque de recepción TCP.



(b) Configuración del bloque TCP Receive.

Figura 5.15. Habilitar recepción TCP.

Una vez arranque el modelo en el sistema, el servidor de El Nodo detectará la conexión existente y se enlazará con el modelo, estando atento a la recepción de solicitudes por parte del modelo.

Por otro lado, el modelo requiero tambien de un bloque de transmisión TCP para poder realizar solicitudes al servidor de El Nodo. Para ello se usara el bloque de TCP Send configurado como cliente en el puerto 1337 e IP local ("127.0.0.1").

(a) Bloque de recepción TCP. **(b)** Configuración del bloque TCP Send.

Figura 5.16. Habilitar recepción TCP.

Ahora que tenemos las vías necesarias para tanto transmitir como recibir datos del servidor de El Nodo, podemos comenzar a ejemplificar un modelo real a modo ilustrativo. Lo primero que se debe de tener en cuenta es la manera que tiene el servidor de El Nodo de recibir y transmitir los comando. Estos comandos son enviados y recibidos mediante un codigo numerico que se establece en la API del sistema.Esta API contiene un listado de los comandos que se pueden ejecutar junto con su código asignado, siendo esta lista accesible desde la interfaz gráfica.

Es por este motivo que a la hora de crear cada modelo es necesario verificar los comandos que se pueden emplear ya que estos son susceptible de cambiar al conectar nuevos dispositivos al sistema. En este ejemplo se intenta controlar una lámpara para encenerla y apagarla según una onda cuadrada, para ello se usan los comando 0 y 1 para el encendido y apagado respectivamente.

Si ejecutamos este modelo en El Nodo, conseguiremos encender y apagar una lámpara de forma intermitente con un periodo constante. Sin embargo, con este modelo solo hemos probado el envio de comandos al servidor de El Nodo, falta comprobar que el modelo es capaz de responder a comando enviados desde el propio servidor de El Nodo. Para realizar esta demostración se conectará el bloque de recepción TCP a un selector que procese que comando se ha recibido y poder ejecutar una acción en respuesta. En este caso usaremos el comando 48 y 49, específicos para detener y arrancar el modelo.

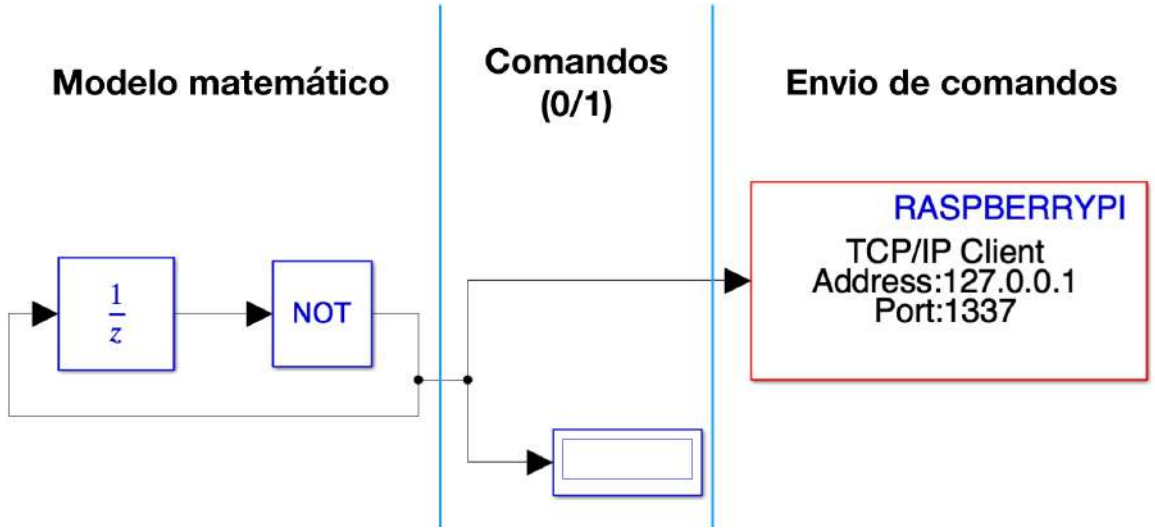


Figura 5.17. Habilitar escucha TCP

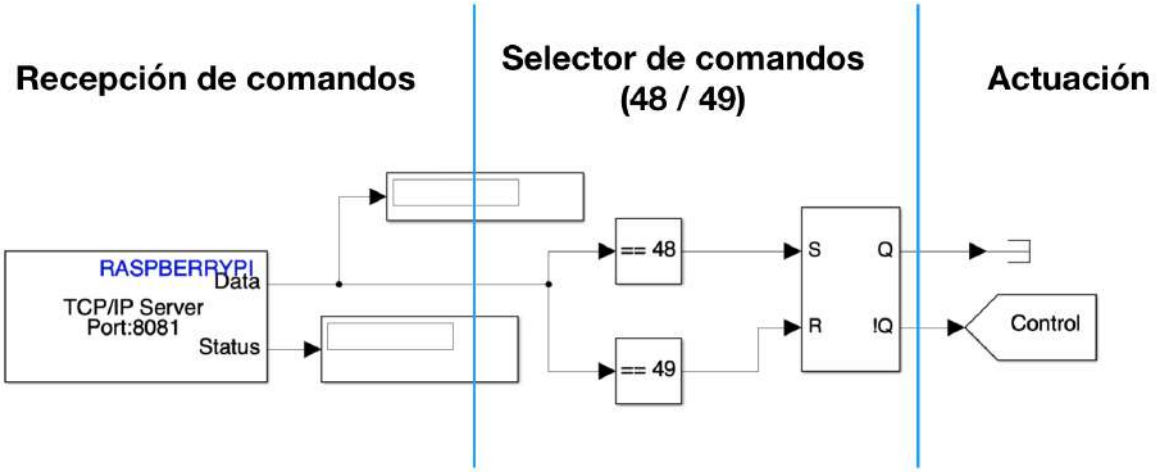


Figura 5.18. Procesado de comandos desde El Nodo.

Como se puede apreciar en la figura 5.18, la etapa de actuación del modelo pone a 1 la variable control cuando el comando recibido es 49 y a 0 cuando el comando es 48. Esta variable la podemos usar en el modelo del control de la lampara de la figura 5.17 para parar o arrancar el modelo.

Finalmente, con las modificaciones realizadas al modelo, Simulink es capaz de encender y apagar una lampara intermitentemente y responder a las peticiones realizadas desde el servidor de El Nodo, activando y desactivando el control del modelo sobre la lampara. Este modelo es un ejemplo de lo que se puede realizar con la integración de Matlab y Simulink en el sistema, ilustrando que elementos debe de tener todo modelo implementado en el sistema. En la siguiente figura (figura 5.20)se muestra la consola de el servidor de El Nodo mientras se ejecuta el modelo diseñado.

5.1.5. Gestor de hardware

Por el momento hemos hablado de bloques cuya ejecución estaba limitada al procesador principal de El Nodo, la Raspberry Pi, dejando de lado el resto de hardware integrado en el dispositivo. Para que los diferentes bloques que componen el sistema sean capaces de usar

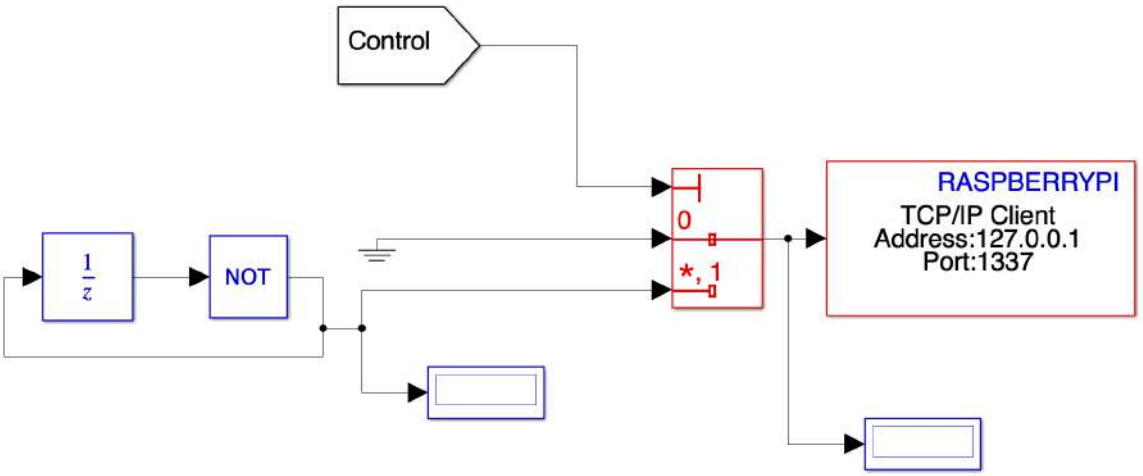


Figura 5.19. Modelo modificado par interrumpir la ejecución.

```
Servidor TCP escuchando en puerto 1337
Servidor HTTP escuchando en puerto 8000
Recibido:
1
Respuesta:
OK, Encendiendo lampara
26
Recibido:
0
Respuesta:
OK, Apagando lampara
26
Recibido:
1
Respuesta:
OK, Encendiendo lampara
26
Recibido:
0
Respuesta:
OK, Apagando lampara
26
Recibido:
1
Respuesta:
OK, Encendiendo lampara
26
Recibido:
0
```

Figura 5.20. Console log servidor El Nodo

el hardware del que dispone El Nodo es necesario un sistema que administre los recursos y conecte los diferentes periféricos. El bloque encargado de esta función es el gestor de hardware.

Como su nombre indica el gestor de hardware es el encargado de gestionar el hardware disponible y hacerlo accesible para el resto de bloques y procesos del sistema, expandiendo así la funcionalidad del resto de procesos. Gracias a este gestor el servidor de El Nodo es capaz de utilizar recursos como altavoces, micrófonos, dispositivos de transmisión de datos e incluso la iluminación del dispositivo. Todo este proceso se gestiona principalmente gracias a la inclusión de la placa madre del dispositivo, la cual como habíamos comentado en el apartado de hardware, conectan todos los periféricos con el GPIO de la Raspberry Pi, donde se ejecutan el resto de procesos.

El software involucrado con la administración de dichos recursos se ejecuta en el ATmega2560 del Arduino Mega situado en la placa madre. Este procesador interpreta las instrucciones recibidas desde el servidor de El Nodo a través del bus de comunicaciones I2C integrado tanto en la Raspberry Pi como en el Arduino Mega, actuando convenientemente sobre el hardware del sistema.

Mediante este procedimiento se habilita a procesos como son Openhab, Matlab y Simulink a interactuar con las luces del dispositivo, enviar señales mediante radiofrecuencia, generar animaciones en la matriz de leds, atender al accionamiento del dial integrado o incluso interactuar con hardware adicional instalado en las placas de expansión compatibles con El Nodo.

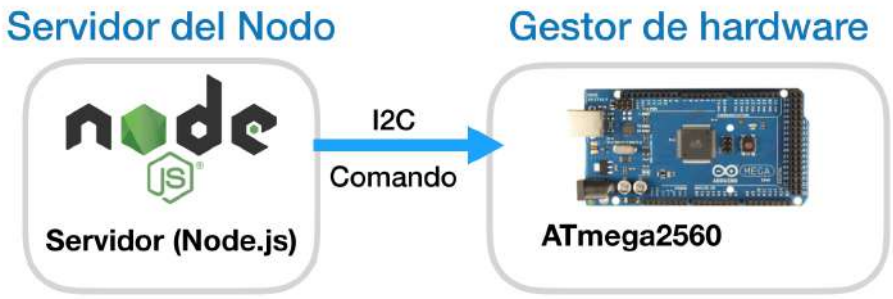


Figura 5.21. Comunicación entre el servidor de El Nodo y el gestor de hardware.

Ademas de el software de gestión instalado en la placa madre, es de especial relevancia el driver de la matriz de leds RGB 5x5x5 situado en la parte superior del dispositivo. Dicho driver controla por completo la iluminación de dicha matriz teniendo almacenadas todas las animaciones preprogramadas. Este driver al igual que el gestor de hardware esta ejecutado desde un Arduino, en este caso un Arduino Nano.

El driver de la matriz de leds se comunica con el gestor de hardware mediante el protocolo I2C, el cual le indica que animaciones ejecutar, teniendo como parametros los colores y las coordenadas de los leds a encender. Por motivos de extensión del documento el funcionamiento de dicho driver se explicará en un Anexo a parte.

5.1.6. Interfaz gráfica

Al tratarse de un dispositivo estático que pretende formar parte del hogar es preciso que este dispositivo disponga de capacidad para interactuar con los miembros de la casa de forma sencilla y cómoda. Con este pensamiento en mente se ha instalado una pantalla de siete pulgadas táctil en la parte frontal del sistema. Ahora bien, sin interfaz gráfica la pantalla carece de sentido, es por ello que uno de los bloques principales del sistema es su interfaz de usuario (UI).

La interfaz de usuario esta basada en tecnologia puramente web, estando programada en HTML, CSS y Javascript. Estas tecnologias noc permiten, como ya comentamos con anterioridad, ser facilmente ejecutados por un navegador web que corre en segundo plano en la Raspberry

Pi. En este caso el navegador utilizado es Chromium, ya que se trata de un navegador opensource con caracteristicas similares a Google Chrome aunque de menor potencia, ideal para procesadores como el de la Raspberry Pi por su ligereza.

Chromium se ejecuta en segundo plano con el arranque de el servidor de El Nodo, el cual carga todos los archivos web en el navegador,mostrando la primera pantalla de carga hasta que finalmente una vez los archivos están cargados se muestra la pantalla de aplicaciones.

Figura 5.22. Pantalla de carga.

Figura 5.23. Pantalla de aplicaciones.

La estructura de la interfaz se ha pensado para ser lo más sencilla e intuitiva posible teniendo tan solo tres pantallas principales: Pantalla de aplicaciones, pantalla de habitaciones y pantalla de control. En primer lugar, al encender el dispositivo se muestra la pantalla de carga a la espera de que los procesos internos de arranque del sistema finalicen. Tras una ligera demora se da el control al usuario mostrando la primera pantalla ineractiva, la pantalla de aplicaciones. Junto desde esta pantalla se puede navegar al resto de la interfaz a tan solo deslizando el dedo a la derecha a modo de slide, cambiando la pantalla alternativamente entre la pantalla de control la de habitaciones y la de aplicaciones.

Figura 5.24. Estructura de la interfaz.

Desde la pantalla de aplicaciones podemos acceder la gran mayoria de la funcionalidad que ofrece El Nodo, funcionalidad la cual se encuentra encapsulada en forma de pequeñas aplicaciones, cada una con un proposito concreto. En este punto del desarrollo se han implementado las siguientes aplicaciones:

- **Reloj (En Desarrollo):** Desde esta pequeña aplicación podemos establecer alarmas para nuestro dispositivo, asi como generar eventos temporales para ciertas rutinas de la casa.
- **Calendario:** Sincroniza el calendario de Google Calendar de una cuanta de gmail para mostrar el calendario personal del usuario.
- **Red (En Desarrollo):** Aplicación para navegación por internet.
- **Cloud (En Desarrollo):** Gestor del sistema de almacenado de datos en la nube de El Nodo.
- **Microfono (En Desarrollo):** Acceso al control por voz y grabación de voz a través del microfono integrado de El Nodo.

- **Radio:** Sistema de radio web de El Nodo. Conecta con sevicios de distribución de emisoras de radio por internet.
- **Imagenes (En Desarrollo):** Gestor y visualizador de imagenes.
- **Videos (En Desarrollo):** Gestor y visualizador de videos.
- **Archivos (En Desarrollo):** Gestor de archivos almacenados en El Nodo.
- **Rastreador:** Aplicación de rastreo de dispositivos domoticos para su inclusión en la red de El Nodo. Permite realizar un barrido de los dispositivos compatibles en rango para su control.
- **Dispositivos:** Permite controlar todos aquellos dispositivos previamente conectados al sistema.
- **Spotify:** Habilita la aplicación web de spotify en el dispositivo.
- **Ajustes:** Permite configurar el dispositivos para ajustarlo a las necesidades y criterios del usuario.

Prosiguiendo con la pantalla de control, esta es accesible desde la pantalla de aplicaciones haciendo slide con el dedo. Dicha pantalla esta enfocado a la implementación de sistemas de control avanzados, como son los modelos de Matlab y Simulink. Desde estas aplicaciones podemos habilitar y desabilitar scripts y modelos de Matlab y Simulink previamente cargados en el sistema. Estos modelos se ejecutan en segundo plano, monitorizando el estado de los dispositivos continuamente y actuando sobre ellos si procede.

Figura 5.25. Pantalla de control.

Finalmente comentar brevemente la pantalla de habitaciones. Esta pantalla pretende agrupar la funcionalidad de los diferentes dispositivos conectados a El Nodo entre las diferentes habitaciones o zonas en la que se divide la casa para así poder gestionar de forma rápida

e intuitiva los distintos dispositivos.

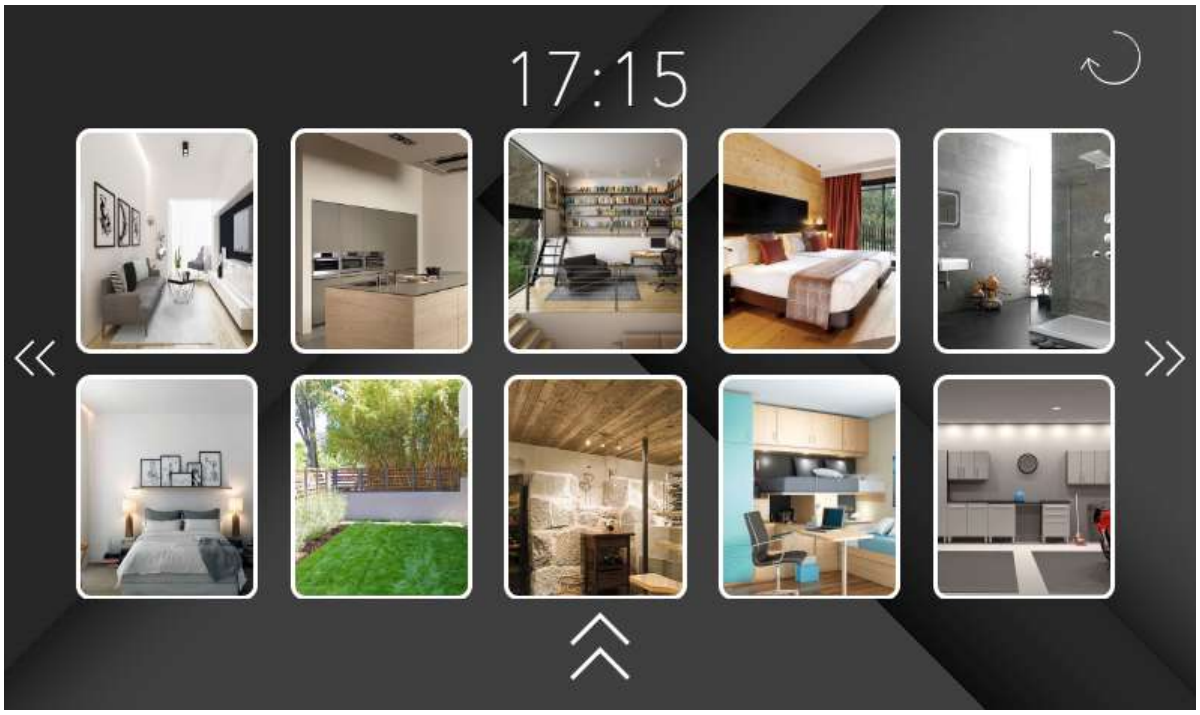


Figura 5.26. Pantalla de habitaciones.

Como habíamos mencionado en la sección del servidor de El Nodo, la interfaz está integrada mediante una aplicación Electron.js, la cual integra el servidor con un cliente basado en el navegador Chromium. Esta aplicación se ejecuta en el arranque del sistema, una vez el sistema operativo Linux esta cargado, estableciendo conexión inmediata con el servidor de El Nodo y el resto de componentes del sistema.

Al tratarse de una aplicación Electron, dicha aplicación se ha programado con lenguajes de programación web, más específicamente HTML, CSS y Javascript. Estos lenguajes son especialmente relevantes por su sencilla programación y versatilidad a la hora de implementarlo en el sistema. Dicho código fuente esta dividido en un archivo principal HTML que se encarga de dar estructura a los elementos principales de la pantalla, un archivo único CSS para generar los estilos y estética de todos los componentes, y finalmente un script general en Javascript para atender a eventos especiales.

Para la el desarrollo de las aplicaciones, el procedimiento es el mismo, estando programadas en HTML, CSS y Javascript. Sin embargo a diferencia de la interfaz que presenta archivos de código comunes, cada aplicación esta compuesto por tres archivos de código fuente de las tres tecnologías descritas de forma independiente. Con esta división conseguimos separar el desarrollo de la propia interfaz con las aplicaciones, siendo posible re programar el comportamiento de cada aplicación sin afectar al funcionamiento del resto de la interfaz.

Finalmente, solo queda hablar brevemente de como se comunica la interfaz con el resto del sistema. Dicha comunicación se produce a través de simples peticiones web a el servidor de El Nodo mediante POST y GET. Estas peticiones se generan a mediante la implementación de

script escritos en Javascript que controlan dichos eventos.

```
function enviarPeticionGET(path){

var options = {
  host: "172.0.0.1",
  port: 8080,
  path: path,
  method: 'GET'
};

http.request(options, function(res) {
  console.log('STATUS: ' + res.statusCode);
  console.log('HEADERS: ' + JSON.stringify(res.headers));
  res.setEncoding('utf8');
  res.on('data', function (chunk) {
    console.log('BODY: ' + chunk);
  });
}).end();
}
```

Figura 5.27. Elemplo de petición GET.

```
function enviarPeticionPOST(peticion){
  xhttp.open("POST", "http://127.0.0.1:8000", true);
  xhttp.setRequestHeader("Content-type", "text/plain");
  xhttp.send(peticion);
}
```

Figura 5.28. Ejempli de petición POST.

6
Resultados

El objetivo de este proyecto desde su inicio es la realización y puesta en marcha de un sistema de control domótico. Este sistema estaría compuesto por diversas tecnologias capaces de dotar al sistema de una amplia funcionalidad tanto para la conectividad de dispositivos como para su pertinente control y gestión. En esta sección se analizará el desarrollo del proyecto en su conjunto siguiendo punto por punto los objetivos que previamente se marcaron en la sección **??**.

6.1. Objetivos principales del proyecto

En un primer lugar, se definió como objetivo la propia construcción física del centro de control, integrando tanto los componentes mecánicos como los electrónicos. Este sistema debía de presentar ciertos requerimientos en cuanto a capacidad de procesamiento, versatilidad y estética, los cuales fueron previamente definidos en la sección de diseño del hardware A.2.

Si bien es complicado cuantificar el grado de éxito en la realización de este objetivo, podemos consideras el objetivo como logrado satisfactoriamente. El sistema se ha conseguido fabricar replicando casi de forma exacta a lo planificado en el modelado 3D y en los esquemas teóricos planteado con anterioridad, reuniendo por tanto la capacidad de procesamiento y la versatilidad del sistema para la que fue diseñado.

El sistema físico presenta una gran robustez en la estructura siendo resistente a posibles traslados y a la manipulación de los usuarios. La realización de las placas electrónicas, a pesar de estar soldadas a mano, no presentan problemas de interferencias ni de pérdidas. Como añadido, es fundamental mencionar la modularidad del sistema, que es fácilmente desmontable en caso de tener que ser transportado o en caso de avería en alguno de sus componentes.

Otro aspecto importante a valorar es el acabado del dispositivo. Ciertamente, la estética es un factor subjetivo, no obstante se puede concluir que la calidad de las piezas es profesional y elegante, ajustándose lo más posible al concepto que se bocetó en primera instancia (figura 4.1). La utilización de técnicas como el corte por láser y la impresión 3D garantizan la simetría y exactitud en las métricas del sistema, siendo relativamente sencilla la producción en serie de dicho aparato.

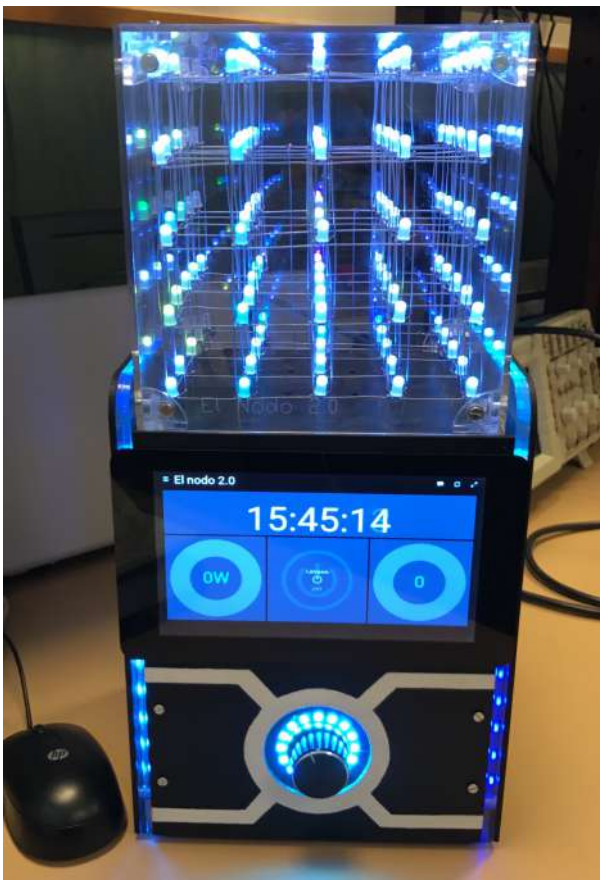


Figura 6.1. Sistema físico El Nodo.

Finalmente en lo que a la integración del sistema respecta, podemos concluir que a pesar de componer diferentes tecnologías, todas se han conseguido integrar dentro de un mismo ecosistema de manera satisfactoria, estableciendo enlaces entre los diferentes módulos ya sea por vía software o hardware.

Para analizar más exhaustivamente el grado de exito en la integración del sistema en su conjunto a continuación se nombrarán algunos de los hitos logrados para la realización del dispositivo:

- **Diseño y selección de componentes:** En primer lugar se ha hecho un estudio de los componentes necesarios para aportar al sistema de la funcionalidad necesaria, seguido de su posterior diseño y planificación para su implementación. Este porceso lo podemos ver reflejado en el capitiulo **??**, diseño general.

- **Fabricación y montaje del sistema:** Una vez claro los componentes a usar y su forma de ensamblado se ha procedido a la fabricación del dispositivo. Se han cortado los diferentes elementos de chapas de madera y metacrilato, así como impreso piezas de geometría más compleja en 3D. Todas estas piezas en conjunción con los componentes electrónicos se han ensamblado para conformar el sistema físico final. La fabricación del dispositivo se cubrió en el capitulo A.2, hardware.

- **Configuración del sistema operativo:** El sistema integra como procesador principal una Raspberry Pi, con el sistema operativo Raspbian instalado por defecto. Este sistema operativo se trata de una distribución de Linux con ciertas modificaciones por razones de compatibilidad con el hardware de la placa. Para poder alojar nuestro software en dicho sistema operativo se ha tenido que configurar dicha distribución Linux acorde con las necesidades del proyecto. Una de las modificaciones más importantes es la habilitación de los diferentes puertos de comunicaciones de se usan para comunicar los diferentes módulos y el arranque automático de la interfaz gráfica y lo procesos programados para

El Nodo.

- **Instalación de Openhab:** Uno de los elementos más importantes del proyecto es el módulo de Openhab. Para poder instalarlo en el sistema se siguió el procedimiento de instalación estándar y configurando el programa para ser ejecutado junto con el arranque del sistema. Este proceso corre en paralelo con el resto de procesos de El Nodo, como son el servidor de El Nodo y los procesos de MATLAB y Simulink. Dicho programa se desarrollo en detalle en el capitulo **??**, servidor Openhab.

- **Instalación de MATLAB y Simulink:** Para las rutinas de control avanzado se escogio MATLAB y Simulink debido a su gran potencia y versatilidad así como por su compatibilidad con la plataforma Raspberry Pi. Este software se instalo manualmente y con una configuración especifica para el sistema en cuestión para una mayor compatibilidad con el resto de módulos. Este proceso se desarrollo en detalle en el capitulo 5.1.4, proceso MATLAB y Simulink.

- **Programación del servidor de El Nodo:** Para coordinar todos los elementos del sistema se requiere de un proceso principal que sirva de gestor entre componentes. Para ello se ha programado el servidor de El Nodo,este proceso encagado esta encargado de regular la actividad de los diferentes procesos que transcurren el el dispositivo. Este módulo se desarrollo en detalle en el capitulo 5.1.3, servidor de El Nodo.

- **Implementación de la interfaz gráfica:** Tras desarrollar el servidor e El Nodo, se implementa la interfaz gráfica a modo de cliente web sobre el servidor ya desarrollado. Esta interfaz hace de puente entre el usuario y el dispositivo. El desarrollo de la interfaz se recopila en el capitulo **??**

- **Integración del gestor de hardware:** El último bloque que queda por realizar en el sistema es el gestor de hardware. Este gestor consiste en la implementación de un placa madre que conecta el procesador principal (Raspberry Pi) con el resto de dispositivos del sistema y es el encargado de procesar las ordenes provenientes del servidor de El Nodo y Este bloque se describió en el capitulo 5.1.5, gestor de hardware.

- **Intercomunicación entre módulos:** Un hito importante en la integración del sistema es la transmisión de mensajes entre procesos dentro del sistema. Llegados a este punto el servidor de El Nodo es capaz de recibir mensajes provenientes del resto de módulos y responder convenientemente.

Todos estos pasos seguidos han conllevado a la realización del sistema tal y como se encuentra en este punto final de desarrollo, punto en el cual podemos considerar al sistema de plenamente funcional y operativo. Este desarrollo pretende servir como prueba de concepto mostrando que un sistema de estas características es completamente viable tanto a nivel teórico como práctico.

No obstante, aunque el sistema haya probado un buen funcionamiento y rendimiento en su funcionalidad básica todavía se trata de un proyecto lejos de poder atender cualquier nicho de mercado en la sociedad actual. El sistema tal cual su estado de desarrollo carece de algunas funcionalidades básicas para poder considerarse un sistema de control completo, uno de los más importantes es la implementación de sistemas de seguridad avanzados para garantizar la privacidad en el hogar.

Según los objetivos propuestos para este proyecto, este no se limita a la simple implemtentaicon e integracion del centro de control domótico, sino que se pretende poder realizar un control de multiples perfiericos a modo de prueba del sistema. Estos dispositivos se pretende que sean de diferentes marcas y utilicen diferentes tecnologías, para así poder verificar el funcionamiento, tanto de las comunicaciones, como del propio software de control. Para comprobar que el sistema cumple con el objetivo prouesto, se han utilizado dos periféricos de la marca fibaro, y se les ha aplicado un control básico en lazo cerrado, procesado desde madlab y comunicándose mediante tecnología Z-Wave.

Dicha prueba consiste en el encendido y apagado de una lámpara en función de una temperatura crítica registrada a través de un sensor de temperatura. En primer lugar, el modelo de madlab y simulink reconocen los periféricos tanto de actuacion como de recepción y comienza a ejecutar el proceso de control. Una vez enlazados los periféricos, el modelo registra la temperatura y verifica su estado en relación a la consigna. Si se considera que la temperatura es suficiente como para activar el encendido de la lámpara, envía una señal al actuador oswitch.

Con esta pequeña prueba, se han comprobado los siguientes puntos:

- **funcionamiento del modelo de control:** En este ejemplo, el proceso de madlab y simulink ha realizado las tareas de control programadas, designando en cada momento la actuación del interruptor, en función de los datos recibidos por el sensor. Este comportamiento sirve como demostración de que los modelos y scripts programados en madlab pueden ser procesados y ejecutados sin problemas en el sistema.

- **funcionamiento del servidor openhub:** Como ya comentábamos, el módulo openhub es el encargado de gestionar las comunicaciones con los periféricos, resolviendo cualquier problema de compatibilidad de protocolos que pueda surgir. En el caso del ejemplo, openhub ha hecho uso del control Z-Wave instalado en El Nodo para enviar los pertinentes comandos a los periféricos en cuestión. Por otro lado, se prueba la exitosa comunicación entre módulos, pues ésta es fluida a lo largo de todo el proceso y sin aparentes conflictos o corrupción de datos.

- **funcionamiento de la interfaz:** Todo este proceso ha sido ejecutado a través de la interfaz de El Nodo, siendo ésta la encargada de activar y desactivar todo el modelo. En la interfaz gráfica se ha ubicado un botón de activación, el cual envía tras ser pulsado un comando de activación al servidor de El Nodo. La intervención de la interfaz en este proceso prueba la comunicación también existente entre la propia interfaz y el resto de módulos del sistema.

Como se comprueba en el ejemplo, damos por realizado el segundo objetivo del proyecto, el control de múltiples periféricos bajo el sistema de El Nodo. No obstante, y a pesar de lo propuesto inicialmente en los objetivos, sólo se han usado periféricos de la marca fibaro. Esta decisión se debe a la imposibilidad de realizar pruebas con dispositivos de otras marcas, sin embargo, al usarse el servidor de openhub, y ser este un software cuyo funcionamiento está probado, podemos extrapolar una compatibilidad semejante con dispositivos de otras marcas.

6.2. Objetivos secundarios del proyecto

En cuanto a los objetivos secundarios del proyecto, se ha planteado establecer comunicación con internet para el control remoto del sistema. Debido a las características intrínsecas del dispositivo, podemos considerar que este objetivo está parcialmente cumplido, puesto que de por sí, la estructura interna del software está abierta a comunicaciones entrantes desde la red. Estas comunicaciones están habilitadas gracias a la implementación del servidor de El Nodo, el cual es capaz de recibir peticiones HTTP, desde cualquier dominio o dispositivo perteneciente a la red. Aunque no se ha terminado de programar un dominio en internet específico para el control remoto de El Nodo, es posible realizar peticiones y enviar comandos desde la propia red local.

A

Presupuesto

A.1. Materiales

En esta sección se muestra el desglose de materiales empleados en la fabricación de El Nodo, mostrando la cantidad usada de cada elemnte asi como su coste unitario. Estos materiales se han dividido en componentes mecánicos y electronicos con el obetivo de separar lo más posible el origen de los gastos de fabricación. Elementos como herramientas no se contabilizarán en este presupuesto, pues se pretende dar una aproximación al valor individual del dispositivo.

A.1.1. Componentes Mecánicos

Tabla A.1. Sumas parciales: Componentes mecánicos

Componente	Cantidad	Precio Unitario (€)	Coste Parcial (€)
Tornillos M4 x 16	10	0,3	3
Tornillo M3 x 15	50	0,3	15
Tornillo M3 x 10	20	0,3	6
Madera DM 600x600x3	5	3	15
Metacrilato 600x600x3	2	20	40
		TOTAL	110,40

A.1.2. Componentes electrónicos

Tabla A.2. Sumas parciales: componentes hardware

Componente	Cantidad	Precio Unitario (€)	Coste Parcial (€)
Raspberry Pi 3B	1	32,79	32,79
TouchScreen 7’	1	57,70	57,70
Fuente de alimentación 12V, 50W	1	9,49	9,49
Altvoces Omega 6W	1	7,95	7,95
Tiras Led Adresables	1	5,46	5,46
Arduino Mega 2560	1	10,80	10,80
Buck Converter	5	3,83	19,15
Led RGB 5mm Difuso	70	0,4	28,00
Led Ring	1	7,50	7,50
Microfono USB	1	3,31	3,31
Esp8266	1	5	5
USB Hub	1	4,80	4,80
Antena	1	5,16	5,16
Rotory encoder	1	52,10	2,10
AC Socket	1	1	1
NRF24l01	1	6,50	6,50
HC-05 Bluetooth	1	3,90	3,90
		TOTAL	210,61

A.1.3. Transporte

Algunos de los materiales anteriormente mencionados han sido comprados por Internet, lo que supone en la mayoría de casos un coste adicional por transporte. Este coste adicional se contempla en la siguiente tabla:

Tabla A.3. Sumas parciales: Coste de transporte

Componente	Cantidad	Precio Unitario (€)	Coste Parcial (€)
Raspberry Pi Model 3B+	1	4,50	4,50
TouchScreen 7’	1	4,50	4,50
Arduino Mega 2560	1	1,50	1,50
Led RGB 5mm Difuso	70	0,03	2,10
Led Ring	1	1	1
Microfono USB	1	1,62	1,62
Esp8266	1	1	1
USB Hub	1	3,30	3,30
Antena	1	1,29	1,29
NRF24l01	1	0,99	0,99
HC-05 Bluetooth	1	0,75	0,75
		TOTAL	15,22

A.2. Mano de obra

A continuación se exponen los gastos referentes a la mano de obra o servicios contratados para la fabricación del dispositivo final:

Tabla A.4. Sumas parciales: Servicios contratados

Servicio	Cantidad	Precio Unitario (€)	Coste Parcial (€)
Corte laser madera DM	2	10	20
Corte laser metacrilato	2	20	40
Impresión 3D ABS	1	30	30
Impresión 3D PLA	1	15	15
		TOTAL	155

A.3. Presupuesto general

Tabla A.5. Presupuesto general

Elemento	Suma Parcial
Componentes mecánicos	110,40
Componentes electrónicos	210,61
Transporte de mercancías	15,22
Servicios y mano de obra	155
TOTAL	491,23

Bibliografía

[1] H. Yin, F. Yang, and J. Li, "Multi-objective optimization design of a carbon fiber reinforced composite upper arm," 2017.

[2] F. Jiang, J. Zhao, A. K. Kota, N. Xi, M. W. Mutka, and L. Xiao, "A Miniature Water Surface Jumping Robot," *IEEE Robotics and Automation Letters*, vol. 2, pp. 1272 – 1279, 2017.

[3] S. yeol Yoo, B.-H. Jun, H. Shim, P.-M. Lee, and B. Kim, "Finite element analysis of carbon fiber reinforced plastic body frame for seabed robot, Crabster200," 2013.

[4] J. Franch and J. Rodriguez-Fortun, "Control and trajectory generation of an ackerman vehicle by dynamic linearization," 2009.

[5] W. Ren, Q. Gu, D. xin He, and J. Zhao, "A car-like mobile robot design and implementation," 2012.

[6] G. Slemon and A. Straughen, *Electric Machines*. Addison-Wesley, 1980.

[7] J. Gonçalves, J. Lima, P. J. Costa, and A. P. Moreira, "Modeling and simulation of the emg30 geared motor with encoder resorting to simtwo: The official robot factory simulator," July 2013.

[8] M. Stanley, J. Lee, and A. Spanias, *Sensor Analysis for the Internet of Things*. Morgan & Claypool, 2018.

[9] H. Chao and Y. Chen, *Attitude Estimation Using Low-Cost IMUs for Small Unmanned Aerial Vehicles*. IEEE, 2012.

[10] J. R. Brauer, *Hall Effect and Magnetoresistive Sensors*. IEEE, 2006.

[11] Z. Wang and L. Wu, "Automated extraction of building geometric features from raw lidar data," in *2009 IEEE International Geoscience and Remote Sensing Symposium*, vol. 2, pp. II–436–II–439, July 2009.

[12] S. Verghese, "Self-driving cars and lidar," in *2017 Conference on Lasers and Electro-Optics (CLEO)*, pp. 1–1, May 2017.

[13] M. Aldibaja, N. Suganuma, and K. Yoneda, "LIDAR-data accumulation strategy to generate high definition maps for autonomous vehicles," 2017.

[14] J. de Jesús Rico Jimenez, J. J. G. Barbosa, J. B. H. Ramos, F. J. O. Rodríguez, D.-E. H. Garcia, and R. Gonzalez-Barbosa, "Digitalizacion del entorno a partir de un LIDAR HDL-64E," *Nexo*, 2012.

[15] C. Huihai, L. Shuqiang, and Z. Yingsheng, "An obstacle detection algorithm used sequential sonar data for Autonomous Land Vehicle," 2011.

[16] A. K. Maini, *Radar Fundamentals*. Wiley, 2016.

[17] O. Boric-Lubecke, V. M. Lubecke, A. D. Droitcour, B.-K. Park, and A. Singh, *Radar Principles*. IEEE, 2016.

[18] A. K. Maini, *Military Laser Systems*. Wiley, 2016.

[19] D. A. Pomerleau, "ALVINN: An Autonomous Land Vehicle In a Neural Network," *Carnegie Mellon University*, 1989.

[20] N. Patel, A. Choromanska, P. Krishnamurthy, and F. Khorrami, "Sensor modality fusion with CNNs for UGV autonomous driving in indoor environments," 2017.

[21] F. Caron, E. Duflos, and P. Vanheeghe, "Introduction of contextual information in a multisensor EKF for autonomous land vehicle positioning," 2005.

[22] P. J. Hargrave, "A tutorial introduction to kalman filtering," in *IEE Colloquium on Kalman Filters: Introduction, Applications and Future Developments*, pp. 1/1–1/6, Feb 1989.

[23] J. Lacambre, M. Narozny, and M. Duplaquet, "The enriched sigma point kalman filter an adaptation of the unscented kalman filter for navigation applications," in *Proceedings of the 16th International Conference on Information Fusion*, pp. 1813–1818, July 2013.

[24] H. Durrant-Whyte, N. Roy, and P. Abbeel, *A Serial Approach to Handling High-Dimensional Measurements in the Sigma-Point Kalman Filter*. MITP, 2012.

[25] Dutton, Thompson, and Barraclough, *The Art of Control Engineering*. Addison-Wesley, 1997.

[26] N.S.Nise, *Control Systems Engineering*. John Wiley and Sons, 6 ed., 2011.

[27] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng, "Path planning for autonomous vehicles using model predictive control," 2017.

[28] R. H. Brown, S. C. Schneider, and M. G. Mulligan, "Analysis of algorithms for velocity estimation from discrete position versus time data," *IEEE Transactions on Industrial Electronics*, vol. 39, pp. 11–19, Feb 1992.

[29] M. Faccio, P. Grande, F. Parasiliti, R. Petrella, and M. Tursini, "An embedded system for position and speed measurement adopting incremental encoders," in *Conference Record of the 2004 IEEE Industry Applications Conference, 2004. 39th IAS Annual Meeting.*, vol. 2, pp. 1192–1199 vol.2, Oct 2004.

[30] A. Romero, A. J. Muñoz-Ramírez, and J. M. G. de Gabriel, "Realimentación de velocidad con encoders de baja resolución en simulink," 2015.

[31] J. Parada Puig and P. A Lischinsky, "Mínimos cuadrados recursivos para el control adaptativo a tiempo real de un péndulo," pp. EC85–EC91, 01 2010.

[32] X. Chen, "Recursive least-squares method with membership functions," in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*, vol. 3, pp. 1962–1966 vol.3, Aug 2004.

[33] W. Xu and F. Liu, "Recursive algorithm of generalized least squares estimator," in *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, vol. 3, pp. 487–490, Feb 2010.

[34] Q. Chen, Y. Gu, and F. Ding, "Data filtering based recursive least squares estimation algorithm for a class of wiener nonlinear systems," in *Proceeding of the 11th World Congress on Intelligent Control and Automation*, pp. 1848–1852, June 2014.

[35] S. Gurin, "Algoritmos de ordenación," 2004.

[36] Q. Jia, M. Wang, S. Liu, J. Ge, and C. Gu, "Research and development of mecanum-wheeled omnidirectional mobile robot implemented by multiple control methods," in *2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pp. 1–4, Nov 2016.

[37] Y. Liu, H. Li, L. Ding, L. Liu, T. Liu, J. Wang, and H. Gao, "An omnidirectional mobile operating robot based on mecanum wheel," in *2017 2nd International Conference on Advanced Robotics and Mechatronics (ICARM)*, pp. 468–473, Aug 2017.

[38] D. Peña Sánchez de Rivera, *Deducción de distribuciones: el método de Monte Carlo*. Alianza Editorial, 2001.

[39] T. M. Mitchell, *Machine Learning*. McGraw Hill, 1997.