UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DESARROLLO DE CONTROL MEDIANTE DISPOSITIVO MÓVIL DE UN SISTEMA DOMÓTICO

AUTOR: Borja Martínez Martínez

TUTOR: Joaquín Gracia Morán

Curso Académico: 2016-17

ÍNDICE

DOCUMENTOS CONTENIDOS EN EL TFG

- Memoria
- Pliego de condiciones
- Presupuesto

ÍNDICE DE LA MEMORIA

1. Objetivos y Alcance del Proyecto ……. 6

2.Introducción …….. 11

3. Análisis de alternativas para distintos elementos …….. 14

4. Desarrollo de las aplicaciones …….. 19

5. Envío y recepción de datos del sistema domótico …… 21

6.Métodos y conexiones electrónicas Arduino …….. 28

7. Bibliografía …….. 33

ÍNDICE DEL PLIEGO DE CONDICIONES

1. Condiciones legales del software utilizado …….. 36

2. Requisitos Técnicos requeridos por el software empleado …….. 37

3. Plazos …… 39

ÍNDICE DEL PRESUPUESTO

1. Presupuesto de ejecución por contrata …….. 42

2. Presupuesto de ejecución material por capítulos …….. 43

3. Estado de mediciones …….. 46

4. Precios descompuestos …… 48

5. Justificación de precios unitarios …….. 54

AGRADECIMIENTOS

Quiero aprovechar el momento para agradecer el apoyo de mi familia, sin la cual no habría sido posible realizar todo esto, también agradecer a mi tutor todo el apoyo, ayuda y consejos dados durante la realización de dicho proyecto y finalmente a mis compañeros cuya compañía y apoyo ha sido de gran utilidad para seguir con todo.

RESUMEN

Con el auge de los dispositivos móviles y su facilidad para conectarse a una red de comunicaciones, es posible realizar acciones de control desde cualquier lugar y de una forma relativamente sencilla. Un ejemplo de esto son las casas domotizadas.

El presente TFG permite controlar diferentes funciones de un sistema domótico mediante una aplicación Android, controlando elementos tales como luces, persianas o puertas, así como recibir información de la vivienda. Este TFG consta de 3 partes:

- Aplicación Android: Instalada en un dispositivo móvil y que nos permite enviar las órdenes a los diferentes sistemas o dispositivos mediante una interfaz gráfica y de fácil comprensión.
- Servidor Web: Se encarga de recibir la orden de la aplicación y redirigirla al actuador correspondiente, permitiéndonos así tener varios dispositivos en el mismo sistema.
- Actuador, implementado con Arduino, y que se divide en dos partes:
  - Programación de Arduino: Que permite ejecutar distintas tareas dependiendo de la orden que le llegue.
  - Diseño electrónico: Se basa en la correcta estructuración de los componentes electrónicos (relés, motores, leds, etc.) y del circuito (cables, divisores de potencia, etc.) para su correcto funcionamiento.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

I. MEMORIA

Curso Académico: 2016-17

ÍNDICE DE LA MEMORIA

1. Objetivos y alcance del proyecto 6

1.1 Establecimiento de los objetivos 6

1.2 Definición de alcance 6

1.3 Justificación técnica 7

1.4 Justificación académica 7

1.5 Estructura de descomposición del trabajo 9

1.6. Planificación de tareas 10

2.Introducción 11

2.1 Motivación 11

2.2 Funcionamiento 11

3. Análisis de alternativas para distintos elementos 14

3.1 Métodos de conexión entre la aplicación y microcontrolador 14

3.2 Valoración y elección del método 16

3.3 Microcontroladores 17

4. Desarrollo de las aplicaciones 19

4.1 Aplicación Android 19

4.2 Código Arduino 20

5. Envío y recepción de datos del sistema domótico 21

5.1 Envío de datos de la aplicación Android 21

5.2 Servidor para el redireccionamiento de datos 22

5.3 Envío y recepción de datos en Arduino 23

5.4 Envío de datos por Bluetooth en Android 25

5.5 Recepción de datos por Bluetooth en Arduino 26

5.6 Guardad y extracción de datos en le EEPROM 27

6. Métodos y conexiones electrónicas Arduino 28

6.1 Conexión de componentes a Arduino 28

6.2 Control de luces 29

6.3 Control de persiana 29

6.4 Control de puertas corredizas 31

7. Bibliografía 33

1. Objetivos y alcance del proyecto

1.1 Establecimiento de los objetivos

Objetivo Principal: Crear un sistema domótico que permita controlar diferentes elementos de una vivienda mediante una aplicación para Android, incluyendo tanto la creación de la aplicación como la conexión entre el dispositivo Android y el microcontrolador, así como el diseño de los diferentes elementos electrónicos para el correcto funcionamiento de los elementos a controlar.

Objetivos parciales:

1- Permitir el uso de varios sensores y actuadores simultáneamente para así evitar tener un nodo central de control.

2- Permitir varios usuarios.

3- Dar facilidad para la implementación de nuevas funcionalidades, así como la instalación de nuevos sistemas.

4- Conseguir un sistema de fácil instalación.

5- Minimizar el tráfico y consumo del servidor para que cualquier pequeño sistema pueda ejecutarlo.

1.2 Alcance del proyecto

El objeto de este TFG es diseñar un sistema domótico junto a una aplicación con una interfaz sencilla para que pueda ser utilizada por cualquier persona sin conocimientos informáticos.

En primer lugar, se necesitará realizar la conexión entre la Aplicación Android y los microcontroladores, siendo éste uno de los pilares del proyecto. Esta conexión se realizará en varios pasos:

1- Desde la aplicación utilizaremos el servicio Volley para el envío de datos por Internet desde el dispositivo Android.
2- Los datos llegarán luego a un Servidor Web que se encargará de redirigirlos al microcontrolador correspondiente.
3- Ejecución de la orden recibida.

Una vez completada la conexión entre los distintos elementos se realizará la implementación de las órdenes tanto en la aplicación como en los microcontroladores, así como los métodos que llevará a cabo dependiendo de la orden que le llegue.

Como tercer paso, y no menos importante, se diseñarán los circuitos conectados al microcontrolador, los cuales permitirán el funcionamiento de los diferentes componentes que ejecutarán la acción que queremos realizar en la vivienda.

Además, también veremos el por qué se ha realizado de la manera mencionada y qué ventajas nos aporta frente a otras alternativas, así como algunas funcionalidades para un posible trabajo futuro de este proyecto, las cuales permitirían aumentar considerablemente el alcance del mismo pero que quedan fuera del alcance de este TFG.

1.3 Justificación Técnica

La comodidad y ventajas que aporta el uso de un sistema domótico hoy en día permite a este proyecto tener cabida en la mayoría de viviendas, así como incluso en edificios públicos o industriales los cuales requieren de un "protocolo" de encendido de sus respectivos elementos, tales como maquinaria, luces, ascensores, persianas, etc., permitiendo un ahorro considerable de tiempo y esfuerzo.

También aporta una mayor eficiencia tanto energética como de tiempo, ya que estos sistemas automatizados pueden disponer de cierta inteligencia para saber cuándo es el momento oportuno para encender las luces o abrir una persiana, así como cuando apagarlas.

Otra ventaja es que el proyecto llevado a cabo es de bajo coste respecto a sus competidores, ya que se lleva a cabo con elementos de bajo coste y minimizando y aprovechando recursos ya instalados en los edificios.

1.4 Justificación Académica

Dado que el proyecto desarrollado se realiza para la obtención del título Grado en Ingeniería en Tecnologías Industriales, éste tiene un claro carácter académico. Durante la realización del proyecto se han puesto en práctica conocimientos adquiridos durante la titulación, tanto de la parte informática en programación como de la parte eléctrica y electrónica.

En cuanto a la motivación personal del alumno, cabe destacar el gran interés mostrado por la informática y por los sistemas electrónicos y de control y, en particular, por las técnicas que dotan a los distintos sistemas (en este caso un sistema domótico).

Además, dada la previa experiencia con el lenguaje de programación JAVA y la realización del curso de CFP "TALLER DE DESARROLLO DE SISTEMAS DOMÓTICOS BASADOS EN ARDUINO) ha resultado ser un incentivo adicional para llevar a cabo dicho proyecto. Por último, y no menos importante, la posibilidad de poder llevar el trabajo realizado a un caso práctico, ha sido el motivo decisivo para escoger el presente trabajo.

La labor llevada a cabo en este trabajo se justifica atendiendo al interés que surge de dos ámbitos distintos: el académico y el industrial.

En cuanto al interés académico, se han podido poner en práctica distintos conocimientos adquiridos en diversas asignaturas a lo largo del grado, así como también se han adquirido y

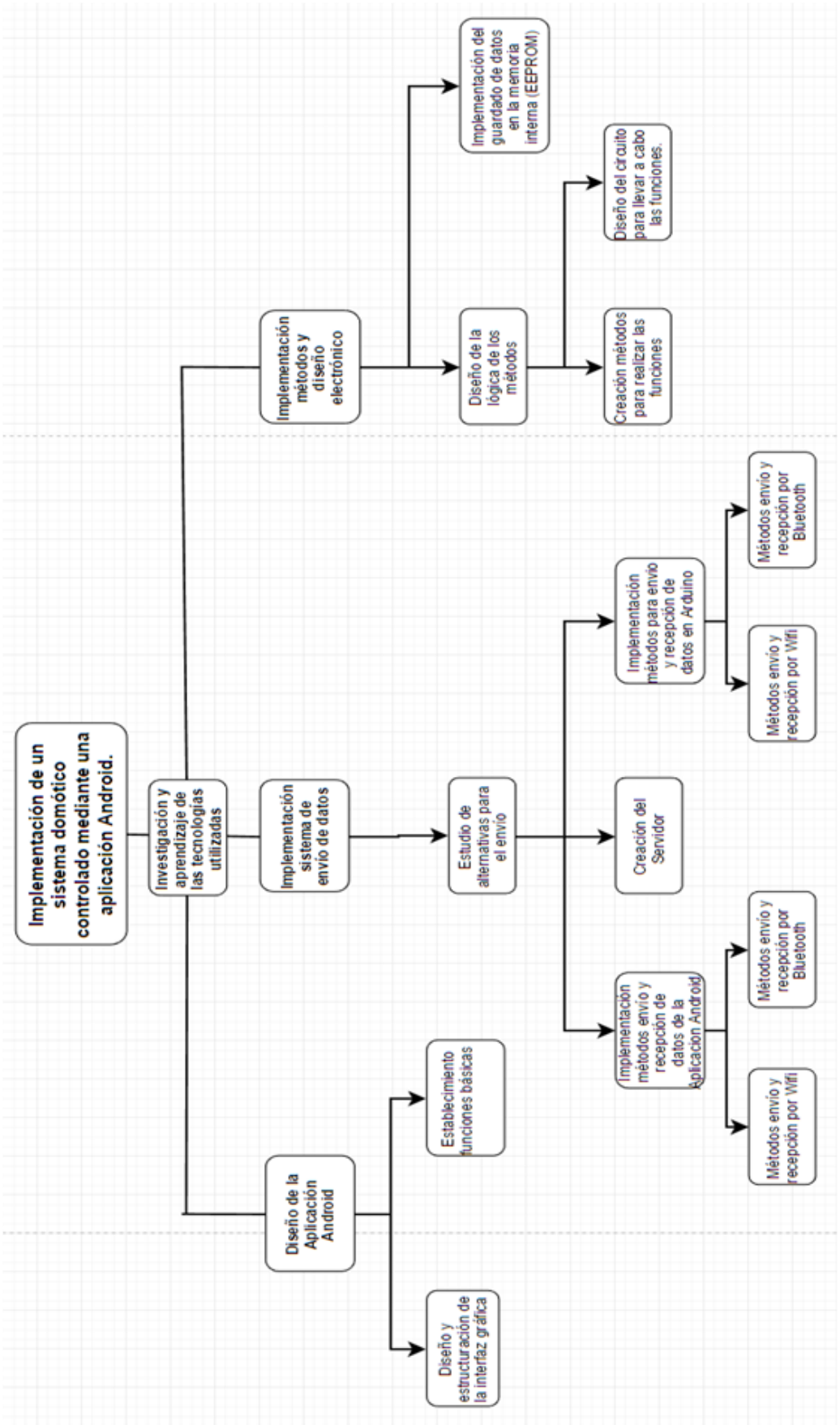
consolidado otros nuevos. Para obtener estos últimos ha sido necesaria, además, la investigación y puesta al día por parte del alumno mediante la consulta de múltiples fuentes:

- Para el acceso a los **actuadores y sensores** del sistema domótico desde la interfaz del programa ha resultado de gran utilidad disponer de las nociones de programación a bajo nivel aprendidas en Informática Industrial, Tecnología Electrónica e Informática.
- Durante el **diseño del controlador** encargado de la automatización de persianas y puertas, ha sido esencial contar con el conocimiento y las habilidades adquiridas en las asignaturas Sistemas Automáticos y Tecnología Automática.
- En el ámbito de los **circuitos electrónicos y electrónica**, ha resultado de gran utilidad contar con los conocimientos obtenidos en las asignaturas Sistemas electrónicos, teoría de circuitos y tecnología eléctrica.
- Para la implementación de la **comunicación con el servidor** han sido de gran ayuda los conocimientos obtenidos en la asignatura de Internet y Servicios en Red.

Respecto al interés industrial, los proyectos de domótica y automatización tienen una gran importancia dentro de dicho ámbito ya que son utilizados en prácticamente todas las industrias para el ahorro y la obtención de un mayor rendimiento con un menor esfuerzo.

Por otro lado, las distintas técnicas empleadas y los diseños desarrollados permiten su uso para cualquier persona no cualificada y la facilidad para la adición de nuevas funcionalidades y la ampliación de dicho sistema.

1.5 Estructura del trabajo



1.6 Planificación de las tareas

Fase	Código	Tarea	Duración(h)	Precedencias
Investigación y aprendizaje	1.1	Investigación tecnologías	9	-
	1.2	Aprendizaje Java	70	-
	1.3	Aprendizaje Arduino	40	-
	1.4	Aprendizaje PHP	12	-
Diseño de la aplicación Android	2.1	Diseño y estructuración de la interfaz gráfica	11	1.2
	2.2	Establecimiento de las funciones básicas	6	2.1
Implementación del sistema de envío de datos	3.1	Estudio de Alternativas para el envío	10	Fase 1
	3.2	Implementación métodos envío y recepción de datos de la Aplicación Android por Wifi	37	2.2 + 3.1
	3.3	Implementación métodos envío y recepción de datos de la Aplicación Android por Bluetooth	31	2.2 + 3.1
	3.4	Creación del Servidor	16	1.4 + 3.1
	3.5	Implementación métodos envío y recepción de datos en Arduino por Wifi	28	
	3.6	Implementación métodos envío y recepción de datos en Arduino por Bluetooth	29	
Implementación métodos y diseño electrónico	4.1	Creación y diseño de los métodos para llevar a cabo las funciones	16	1.3
	4.2	Diseño del circuito para realizar las funciones	8	4.1
	4.3	Implementación del guardado de datos en la memoria interna (EEPROM)	19	3.6

Cabe destacar que la duración del proyecto es superior a la correspondiente de las 300h. Esto se debe al gran número de pruebas realizadas, así como al tiempo dedicado a estudiar la mejor forma de realizar las diferentes tareas correctamente.

2. INTRODUCCIÓN

2.1 Motivación

Se denomina domótica a los sistemas capaces de automatizar una vivienda o edificación de cualquier tipo, aportando servicios tanto de gestión energética como de seguridad, bienestar o comunicación, y que pueden estar integrados por medio de redes interiores y exteriores de comunicación, cableadas o inalámbricas, y cuyo control goza de cierta ubicuidad.

Hoy en día, los dispositivos móviles y sus aplicaciones están en pleno auge, estando disponible cada vez para más usuarios. Además, su facilidad de manejo y capacidad de procesamiento ha aumentado considerablemente a lo largo de los últimos años.

A pesar de que ya existen sistemas domóticos bastante completos y consolidados, estos son caros y requieren una preinstalación en la vivienda de los elementos a controlar por el sistema. Debido a esto, el presente proyecto se centrará en facilitar la instalación del sistema domótico en cualquier vivienda, exceptuando los elementos que requieren un diseño concreto, como por ejemplo una persiana, la cual debería ser eléctrica para poder ser controlada por nuestro sistema, así como en facilitar también la implementación nuevos tipos de funcionalidades.

Otro factor importante en los sistemas domóticos actuales es que la mayoría cuentan con un servidor central de control, es decir, una pantalla táctil fija o Tablet que es la que se encarga del procesamiento de órdenes y del envío de éstas para así controlar desde ahí la totalidad de la vivienda. En este TFG se ha desarrollado una aplicación que puede ser usada por cualquier dispositivo Android.

2.2 Funcionamiento

En este punto se va a explicar el funcionamiento básico de la aplicación para control del sistema desarrollada para dispositivos Android.

La pantalla inicial de la aplicación se muestra en la Imagen1. Una vez aquí será necesario (si no lo estuviese ya) introducir el Arduino en nuestra red wifi. Esto se realizará mediante bluetooth. El proceso ya viene implementado en la aplicación con la excepción de que primero se ha de vincular nuestro móvil con el bluetooth del Arduino. Para ello, simplemente se accede al bluetooth del dispositivo móvil > añadir dispositivo > y se selecciona el Arduino en concreto. Más adelante se explicará la razón de este paso..

Una vez vinculados con el Arduino mediante el Bluetooth, para añadir el Arduino a nuestra red Wifi se pulsa el botón “Añadir Red Wifi” (Imagen 1), mostrándose la pantalla de la Imagen 2.

En esta segunda pantalla (Imagen 2), aparecerá una lista con los dispositivos vinculados al Bluetooth de nuestro móvil, y de ahí se puede seleccionar el Arduino que queremos conectar a nuestra red Wifi. Cada Arduino tendrá un nombre asignado a su habitación correspondiente para evitar confusiones. El último paso se muestra en la Imagen 3. En esta pantalla se permite la introducción de los datos correspondientes a la SSID (*Service Set Identifier* o nombre de la red Wifi) y la contraseña de la red Wifi. La aplicación enviará los datos al Arduino y nos devolverá al menú principal.

De esta forma, solo queda reiniciar el Arduino para que se conecte con la red Wifi, con lo que se finalizaría su configuración.

Una vez configurado el Arduino y conectado a la red Wifi, ya se puede acceder a la habitación correspondiente, tal y como se muestra en la Imagen 1. Una vez seleccionada la estancia a controlar, se llegará a la siguiente pantalla (Imagen 4), en la cual podemos realizar distintas acciones.

En la parte superior de la pantalla (ver Imagen 3) aparecerá la temperatura y humedad de la habitación correspondiente proporcionadas por el Arduino.

Para cada habitación se han implementado tres acciones básicas, aparte de la lectura de datos:

- Encendido y apagado de las luces de la habitación (pulsando el switch cambiará el estado de las luces).
- Control de la puerta, que vendrá implementado solo en el caso de que se disponga de una puerta corrediza. En este caso, se permite tanto abrirla, como detenerla en el estado correspondiente o cerrarla.
- Control de la persiana, el cual permite realizar acciones similares a la de la puerta: subirla, bajarla o pararla en el estado que queramos.

Imagen 1

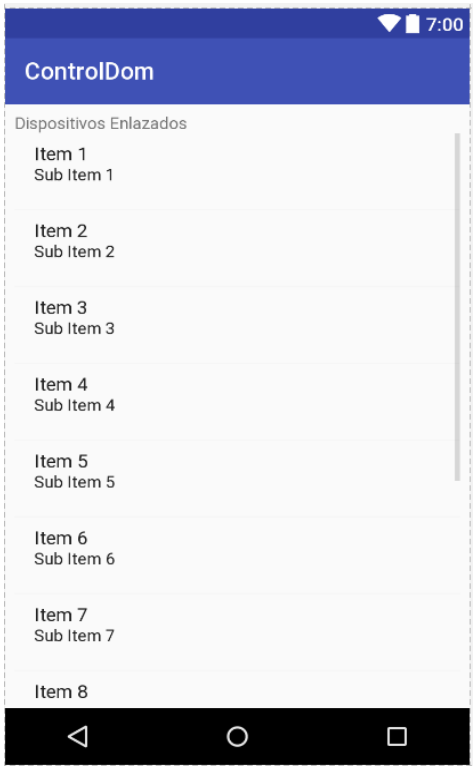


Imagen 2

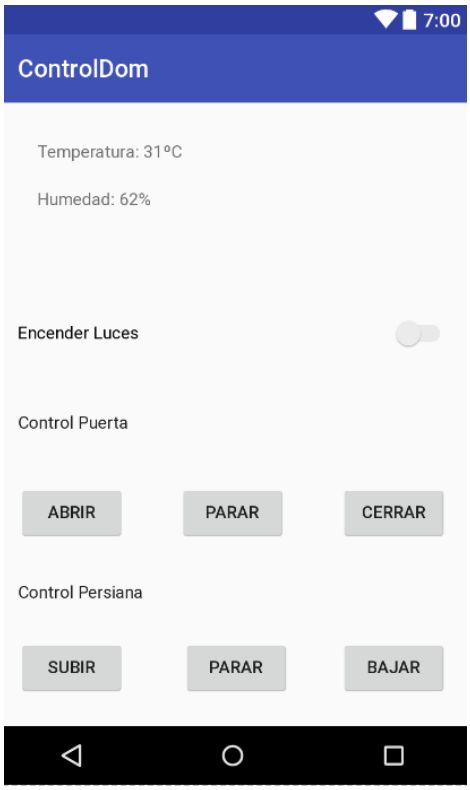


Imagen 3

Imagen 4

El envío de datos de la red Wifi al Arduino se realiza mediante la aplicación. Mediante el Bluetooth del dispositivo, enviamos todos los datos al Arduino, el cual los recibe por su propio dispositivo Bluetooth, leyéndolos y almacenándolos en su memoria interna (EEPROM) para disponer de ellos cada vez que se inicie y así evitar la pérdida de dichos datos al apagarse.

El envío de órdenes al Arduino también se realiza mediante la aplicación Android, donde al pulsar en el botón correspondiente, la aplicación envía una solicitud a una dirección IP en concreto.

Esta dirección es la del servidor, el cual se encarga de procesar esta información y enviarle la orden seleccionada al Arduino correspondiente, mediante un sistema de servicios y envio de datos creado en PHP con petición cURL. Finalmente, la orden será procesada por el Arduino y ejecutará su acción correspondiente.

El Arduino procesará la orden recibida, activando los actuadores necesarios (motores, sensores, leds, relés, etc.) para ejecutar dicha orden.

3. Análisis de alternativas para los distintos elementos

3.1 Métodos de conexión entre la aplicación y microcontrolador

Principalmente se disponen de tres formas de permitir el envío de datos entre la Aplicación Android y el microcontrolador: Infrarrojos, Bluetooth y Wifi.

Se va a hacer un estudio de las ventajas e inconvenientes de cada uno para evaluar cuál sería el más apropiado a implementar en el proyecto.

- **Infrarrojos:** Los infrarrojos son un tipo de radiación electromagnética y térmica de mayor longitud que la onda de luz visible. Gracias a la tecnología de hoy en día, son un método de envió de información bastante utilizado, sobre todo para órdenes sencillas, ya que se basa en un parpadeo muy rápido el cual es traducido por un sensor para recibir el mensaje.

    - Ventajas:
        - La principal ventaja de los infrarrojos es su sencillez, ya que simplemente implica enviar una señal que cuando es captada por el sensor ejecuta una orden predefinida por el microcontrolador.
        - Tienen un bajo coste tanto los sensores como los emisores.
        - Alta disponibilidad.

    - Inconvenientes:
        - Tienen un rango limitado. Un emisor de largo alcance llegaría a los 8m, por lo cual habría que estar cerca de lo que se quiere controlar o se necesitaría algún tipo de repetidores de señal que enviara ésta a lo largo de la casa hasta su destino correspondiente.
        - Muy baja seguridad, ya que cualquier persona con un emisor de infrarrojos podría enviar una señal al dispositivo y controlarlo.
        - Son buenos para el envío de mensajes cortos, pero a medida que aumenta la longitud del mensaje hay mayor retraso en el envío y más posibilidades de perder parte de la señal y tener que volver a enviarla.
        - Son los mismos para todos, por lo que es posible que aparezcan interferencias por otros dispositivos que envíen la misma señal.

- **Bluetooth:** El bluetooth es una especificación industrial para redes inalámbricas de área personal que posibilita la transmisión de voz y datos entre los diferentes dispositivos mediante un enlace por radiofrecuencia. El bluetooth está actualmente muy extendido, ya que permite el envío de grandes cantidades de datos entre dos dispositivos a buena velocidad.

    - Ventajas:
        - Permite el envío de grandes cantidades de datos en un corto periodo de tiempo.
        - Aporta cierta seguridad al tener una contraseña para conectar el dispositivo.
        - Disponible en la mayoría de dispositivos móviles.
    - Inconvenientes:
        - Solo permite la conexión de un dispositivo a la vez.
        - A pesar de tener mayor distancia que los infrarrojos, sigue siendo insuficiente para una vivienda, a excepción de los emisores de clase 1 (alta potencia), los cuales implican un alto coste respecto y no se dispone de ellos en los dispositivos móviles.
        - Aunque disponga de seguridad, ésta es fácilmente vulnerable ya que solo se basa en una contraseña de 4-5 dígitos.
        - Tiene mayor dificultad que los infrarrojos, ya que requiere una instalación de dispositivos bluetooth en los microcontroladores, así como su configuración.

- **Wifi:** Es un mecanismo de conexión de dispositivos electrónicos de forma inalámbrica. Los dispositivos habilitados con Wifi pueden conectarse a internet a través de un punto de acceso de red inalámbrica. Hoy en día el Wifi es una tecnología disponible en la mayoría de los hogares, así como edificios públicos e incluso tiendas.

    - Ventajas:
        - El Wifi permite la conexión de varios dispositivos al mismo tiempo a la red.
        - Mayor alcance de señal. En espacios cerrados suele rondar los 20m, pero en espacios abiertos es aún mayor. Además se puede

emplear el uso de repetidores para aumentar el alcance de la señal.

- Aporta una buena seguridad que viene incorporada en el propio Wifi, donde se necesita una contraseña para poder acceder, utilizándose un protocolo de cifrado.

- Puede conectarse a internet e incluso podríamos acceder a nuestro sistema a través del mismo.

- Inconvenientes:
  - tener varios dispositivos conectados simultáneamente, se necesita de una redirección de datos para saber a quién le enviamos la orden.

  - Mayor complejidad de conexión que los métodos anteriores.

  - Supone un coste de mantenimiento mensual.

  - Necesidad de identificación de todos los elementos en la red.

3.2 Valoración y elección del método

Tomando en cuenta las principales ventajas e inconvenientes de los anteriores métodos, se descarta la opción de infrarrojos debido a su facilidad para ser controlado por cualquier dispositivo similar, ya que requeriría una señal distinta para cada hogar y cada elemento, y también por su limitación tanto de alcance como de saturación de señal en el caso que se utilicen varios dispositivos simultáneamente.

Comparando entonces las opciones de Bluetooth y Wifi, se ha optado por el Wifi debido al requisito de poder tener varios dispositivos conectados simultáneamente a pesar de su mayor complejidad, ya que con el Bluetooth para cambiar de dispositivo se requeriría cortar la conexión manualmente para luego establecerla con el nuevo dispositivo, siempre y cuando esté dentro del alcance.

Se opta pues por la conexión Wifi para el envío de datos entre la Aplicación móvil y el microcontrolador.

A pesar de ello, seguimos teniendo ciertos inconvenientes como son el redireccionamiento de datos y la identificación del dispositivo en la red Wifi.

Para el redireccionamiento de datos se han encontrado dos formas sencillas que se adaptan a nuestro proyecto.

- La primera de ellas se basa en una petición realizada directamente por el dispositivo Android a la dirección IP correspondiente a cada microcontrolador.
- La segunda sería implementar un servidor como intermediario, el cual se encargará de recibir la petición del dispositivo móvil y reenviarla al microcontrolador.

Otro factor influyente que, aunque no se haya implementado en este proyecto, ha ayudado a la elección del método es que al servidor Web se le puede asignar una dirección IP pública para poder acceder al mismo desde internet, pudiendo así controlar el sistema domótico desde cualquier lugar con acceso a internet. Sin embargo, esta función no se ha implementado porque requeriría de un proceso de autentificación externa para poder identificar al usuario y evitar el uso del mismo por cualquiera ajeno a la vivienda, así como de una implementación de seguridad informática al servidor para evitar los posibles ataques de terceros, conocimientos los cuales están fuera del alcance de este proyecto, así como de la formación académica recibida por el alumno.

Se necesita poder introducir el microcontrolador en la red Wifi evitando tener que pre-configurarlo vía programación en código. Se tiene pues que encontrar un método sencillo y eficaz para el envío del SSID y de la contraseña del Wifi desde la aplicación al microcontrolador.

No podemos utilizar el propio Wifi ya que se necesita una contraseña para poder establecer la conexión, por lo que se ha optado por el envío de los anteriores datos (SSID y contraseña) vía Bluetooth hasta la placa correspondiente, la cual los almacenará en su memoria interna para su posterior uso y así evitar su pérdida al reinicio de la misma.

Se realiza de esta manera también teniendo en cuenta la posibilidad del cambio de la SSID o contraseña de la red Wifi a lo largo del tiempo, y evitar tener que programar los microcontroladores cada vez que se realizase dicha acción. Por la parte de seguridad no habría ningún problema, ya que desde el bluetooth no se puede controlar nada del dispositivo excepto quitarlo del Wifi cambiando la red por otra.

3.3 Microcontroladores

Un microcontrolador es un circuito integrado programable capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de un ordenador: unidad central de procesamiento, memoria y periféricos de entrada/salida.

Hay gran cantidad de tipos de microcontroladores en el mercado. Los más comunes son la Raspberry pi, Arduino, los PIC (Peripheral Interface Controller), etc.

En este proyecto se ha optado por el Arduino ya que junto con la Raspberry pi son los más extendidos, por lo que es sencillo encontrar numerosas librerías que aportan una gran cantidad de funciones preconfiguradas, lo que nos facilita la implementación de los métodos necesarios para llevar a cabo este proyecto.

Los microcontroladores tienen tres zonas de programación principales:

- La inicial, donde se llama a las librerías y se crean las variables para el proyecto.
- El Setup, donde se implementan los métodos a los que se llama cuando se enciende el Microcontrolador y sólo se ejecutan una vez por ciclo de encendido.

- El Loop, que es la parte que está en constante procesamiento y se repite continuamente mientras el microcontrolador esté encendido.

4. Desarrollo de las aplicaciones

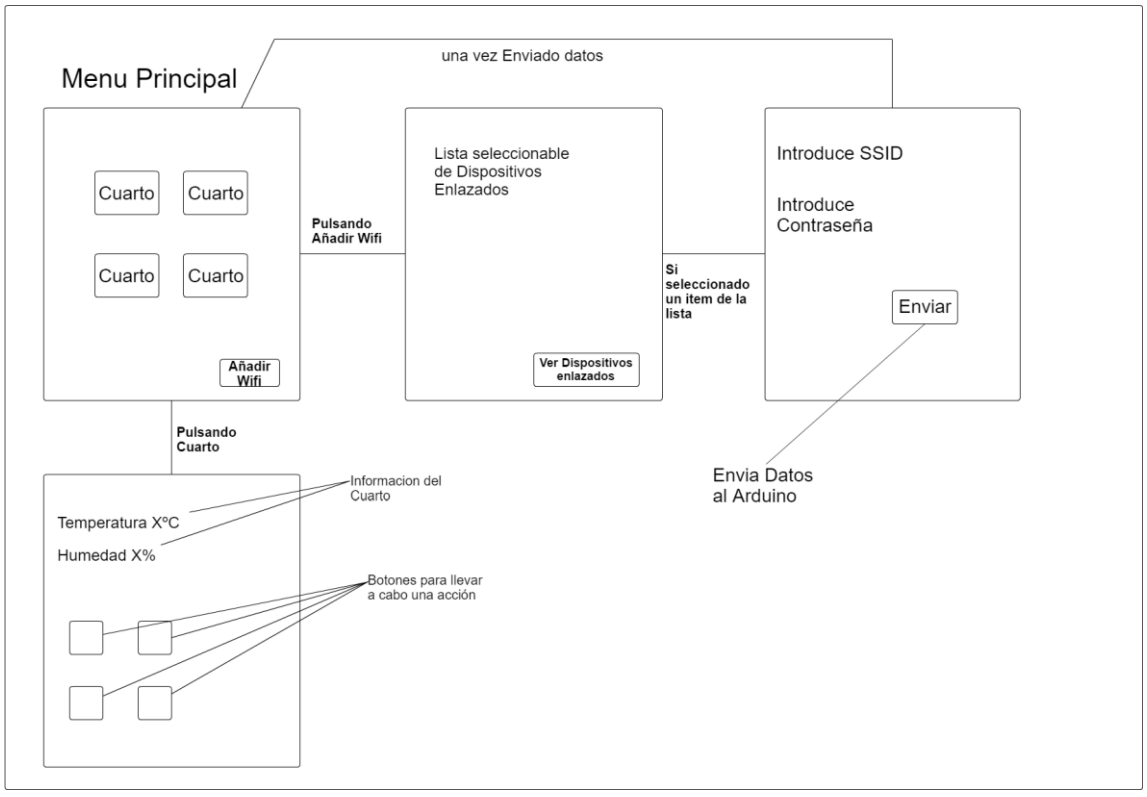
4.1 Aplicación Android

La aplicación Android desarrollada en este TFG se ha diseñado mediante Android Studio. Este entorno gráfico de programación ha sido creado por Google para el desarrollo de las mismas. El lenguaje de programación utilizado es JAVA.

Vamos a proceder a la explicación del funcionamiento de la aplicación. Al poner en marcha la aplicación, aparece el menú principal. Desde aquí se puede:

- Acceder a un cuarto, donde aparecerá una pantalla que muestra la temperatura y humedad del mismo junto con una serie de botones encargados de enviar los datos para realizar una acción en concreto.
- Añadir un Arduino a la red Wifi, el cual nos llevará a una nueva pantalla donde nos mostrará los dispositivos que están vinculados al Bluetooth del dispositivo móvil y nos permitirá seleccionar a cuál queremos enviar la información.

La aplicación se ha diseñado de manera simple e intuitiva para que pueda ser usada por cualquier persona sin conocimientos de la materia. En el diagrama siguiente se puede ver un esquema de su funcionamiento.



Para realizar la conexión entre los diferentes componentes se ha optado por la opción de envío de datos mediante Wifi, donde llegarán a un servidor Web que se encargará de redireccionarlos al Arduino correspondiente, el cual ejecutará la orden recibida.

Estos datos se enviarán en formato de “string”, el cual debe contener varios elementos:

1- La dirección IP del Servidor al cual se envían.
2- La dirección IP del Arduino para señalar el destino.
3- La orden enviada a ejecutar.

Por otro lado, y tal y como se ha comentado previamente, para enviar los datos de usuario y la contraseña de la red Wifi, se seleccionaré el dispositivo que tiene que recibir los datos y éstos se enviarán vía Bluetooth.

4.2 Código Arduino

La programación del Arduino se ha realizado mediante el IDE (*Integrated Development Environment* o Entorno de Desarrollo Integrado) oficial del mismo. Utiliza un lenguaje de programación basado en C++.

Se han implementado diferentes funciones para cada una de las funcionalidades que puede realizar el Arduino. La acción a ejecutar dependerá de los datos que envíe la aplicación Android. Estos datos pasarán antes por el servidor. Más adelante se verán con detalle los diferentes métodos y acciones realizados.

5. Envío y recepción de datos del sistema domótico

5.1 Envío de Datos de la Aplicación Android

En este caso lo que se busca es el envío de una petición a una dirección IP (el servidor) con los datos correspondientes tanto del destino como la orden a ejecutar por el microcontrolador correspondiente.

Se va a utilizar un recurso denominado Volley, que consiste en una librería desarrollada por Google para optimizar el envío de peticiones http desde las aplicaciones Android hacia servidores externos. Este componente actúa como una interfaz de alto nivel, ahorrando así la necesidad de la administración de hilos y procesos como el parsing. El parsing o parser podría ser definido como un programa que analiza una porción de texto para determinar su estructura lógica: la fase de parsing en un compilador toma el texto de un programa y produce un árbol sintáctico que representa la estructura del programa.

Un aspecto a tener en cuenta es la inclusión de la correspondiente librería en el proyecto. Hay varias formas de hacerlo, como clonar su repositorio GIT (almacén donde se guardan librerías) o utilizar uno ya clonado. En este TFG se ha optado por la forma más sencilla, que es añadir el enlace del repositorio ya clonado.

Como se ha visto en el funcionamiento de la Aplicación, solamente van a enviarse datos al Arduino una vez esté seleccionada la habitación que se quiere controlar, por lo que se iniciara el Volley en esas clases sin necesidad de hacerlo en el menú principal.

Para utilizar el servicio Volley hay que inicializar una cola (*Request Queue*) en la aplicación y después se asignará a cada botón una petición diferente dependiendo de su función.

Como punto inicial, y para conocer los datos de temperatura y humedad del cuarto cuando, se debe realizar una petición automática al entrar en dicha estancia.

Aquí podemos ver un pequeño ejemplo de petición:

```
BaPersiana.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        StringRequest BaPersiana = new StringRequest(url, new
Listener<String>() {
            @Override
            public void onResponse(String response) {

            }
        }, new ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                msg("Error de conexion con el servidor, vuelva a
intentarlo");

            }
        });
    }
})
```

El código anterior hace referencia al botón bajar persiana. Sin embargo, aún se desconoce el formato en el que se van a enviar los datos, que vendrá dentro de la variable “url”. Estos datos son procesados por el servidor antes de ser enviado al microcontrolador correspondiente.

5.2 Servidor para redireccionamiento de datos

El siguiente paso consiste en la la creación e instalación del servidor que redireccionará los datos de la aplicación al Arduino y del Arduino a la aplicación.

El principal motivo para utilizar un Servidor es porque permite el uso de múltiples dispositivos para el control domótico, ya que puede distinguir quien ha enviado la petición y a dónde, enviar dicha petición y traer de vuelta la respuesta (si la hubiese) al dispositivo inicial. También se encarga de leer y traducir la petición para simplificar el trabajo al Arduino y recibir así directamente los datos que necesita, evitando la sobrecarga del microcontrolador.

En este TFG se ha instalado un servidor con servicios de PHP y Apache, ya que son los requeridos para poder crear un servidor web (basado en Apache) y crear un programa para el envío y redireccionamiento de datos mediante código PHP, el cual está diseñado para el desarrollo web de contenido dinámico.

El código del servidor se ha desarrollado mediante PHP con una petición de cURL, el cual es un proyecto de software consistente en una biblioteca y un intérprete de comandos orientado a la transferencia de archivos. El principal propósito y uso para cURL es automatizar transferencias de archivos, lo cual se adapta perfectamente al nuestro proyecto, ya que además soporta, entre otros, los protocolos HTTP y HTTPS, los cuales vamos a utilizar.

En este proyecto se ha desarrollado un código PHP para identificar los datos del dispositivo que accede al Servidor y enviar la petición al Arduino correspondiente, y cURL para la obtención de datos y el envío de éstos a la aplicación nuevamente (en el caso de que los haya).

Mediante el código PHP se identifica la dirección IP del Arduino, así como los datos dentro de la petición. Para ello, el envío de datos será de la siguiente forma: http://192.168.1.36/server.php/?ip=XXX&datos=YYY . Este será el formato de la dirección introducida en la variable “url” vista anteriormente, añadiendo la dirección IP y los datos correspondientes a la acción a realizar.

La dirección de envío consta de diferentes partes. La primera parte (“192.168.1.36”) es la dirección IP del servidor, que es la donde se establece la conexión con entre el dispositivo móvil y el servidor. “server.php” es el archivo donde está implementado el código PHP y se realiza la petición cURL.

La segunda parte está formada por unos identificadores “ip=” y “datos=” y el separador “&”. El identificador “ip=” contiene la dirección del Arduino al que vamos a conectarnos (señalizada con XXX) y el identificador “datos=” contiene el mensaje u orden enviada al Arduino (señalizada con YYY). De esta manera, en el código PHP lo que se realiza es la obtención de la información aportada a partir del identificador y el posterior envío de dichos datos a la

dirección IP mediante cURL, el cual permite también la recepción de datos de dicha dirección para su posterior uso en la aplicación Android en el caso de que se requieran.

Este no es el único método de envío y recepción de datos de un dispositivo a otro, pero sí el más sencillo y que mejor se adapta a lo requerido. En principio, no interesa un envío continuo de datos, ni su almacenamiento, debido a que éstos datos son generalmente para el momento, tanto el saber a qué temperatura está la habitación, como abrir una persiana o encender una luz. DE esta forma, se tiene también un consumo muy reducido del servidor, por lo que cualquier pequeño sistema, incluso un Arduino o una Raspberry Pi, podrían realizar las funciones de éste. Este hecho permite un gran ahorro, ya que grandes servidores pueden suponer un costo importante, tanto en compra como en mantenimiento. Además, también podemos independizar el tipo de dispositivo que envía la petición (teléfono, Tablet), ya que el propio cURL se encarga de obtener la información del Arduino y devolverla al dispositivo que la realizó.

Otra gran ventaja de este servidor es que permite la conexión desde el exterior de la red, ya que cURL se encarga de devolver los datos, y el código PHP de recibir la información y traducirla, por lo que podríamos controlar el sistema desde cualquier lugar mediante la conexión al servidor por internet y así poder realizar acciones a distancia, como cerrar las persianas, apagar las luces o incluso conectar la calefacción/refrigeración y programar su encendido desde cualquier lugar. Este caso no está habilitado debido al principal problema de la seguridad informática y la identificación de usuario, ya que para evitar la conexión de otras personas ajenas al servidor se han de tomar medidas de identificación de usuario para evitar que se conecte alguien que no esté autorizado, y la necesidad de seguridad informática para prevenir ataques al servidor, trabajo que está fuera del ámbito de este proyecto.

Un pequeño inconveniente de esta forma de acceso a los diferentes microcontroladores es que cada Arduino ha de tener una dirección IP fija dentro de la red por lo que si una de las seleccionadas en el proyecto ya estuviese siendo utilizada por otro dispositivo, se debería cambiar la dirección IP tanto dentro del código de Arduino como en la habitación de la aplicación Android. Cabe añadir que por defecto vienen dadas unas IP que no suelen utilizarse en una vivienda, pero siempre puede haber excepciones.

5.3 Envío y recepción de datos en Arduino

Una vez resuelto el envío de datos desde la aplicación Android, se va a proceder a la implementación de la recepción de los datos por parte del Arduino y el envío de la respuesta a la petición recibida.

Para realizar dicha acción vamos a utilizar el módulo Wifi ESP8266 para Arduino, el cual ha de conectarse al microcontrolador o puede venir integrado, como ocurre en este caso.

El primer paso a realizar es añadir las librerías correspondientes al módulo ESP8266 al proyecto de Arduino. Una vez añadidas, ya se puede proceder a la inicialización del Wifi y configurarlo para asignarle al Arduino la IP deseada, que será la utilizada luego para el envío de

datos. En este caso se asigna la IP 192.168.1.40 al primer Arduino y se va incrementado en uno a medida que se añaden nuevos microcontroladores.

Una vez establecida la conexión, se definen qué datos realizarán determinadas funcionalidades. Mediante la función Server.on se asigna qué dato activará un método en concreto. Por ejemplo, para encender la luz se utiliza el "String" "EncenderLuz", el cual se llamará al método encender_luz.

```
server.on("EncenderLuz",encender_luz);
```

La escucha del Arduino al servidor se mantiene todo el rato abierta mediante la función server.handleClient en el bucle (loop).

```
server.handleClient();
```

El siguiente paso es el envío de datos desde el Arduino. Como se ha visto en este proyecto, los únicos datos que envía el Arduino a nuestra a aplicación van a ser la temperatura y la humedad de la habitación en cuestión, ya que el resto de órdenes no requieren de un envío de información por parte de éste. A continuación se puede ver el código utilizado para el envío de la temperatura desde el sensor hasta la aplicación Android. Más adelante se explicará cómo se ha realizado dicho proceso, el cual puede ser extrapolado a cualquier tipo de dato o información que se quiera enviar.

```
void nuevatemperatura(){
  Serial.println(dht.readTemperature());
  Serial.println(dht.readHumidity());
  int T = int(dht.readTemperature());
  int H = int(dht.readHumidity());
  String t1 = String(T)+ "&" + String(H);
  server.send(200,"text/plain",t1);
  Serial.println(t1);
```

Para el control de la humedad y la temperatura se ha utilizado un tipo de sensor de humedad y temperatura DHT11.

Dentro del código anterior, los métodos Serial.println son meramente informativos, y sirven mostrar la temperatura en la consola de Arduino, con lo que podemos comprobar que los sensores funcionan correctamente.

Después se procede a la lectura y guardado de la temperatura y humedad mediante las funciones dht.readTemperature y dht.readHumidity y a su posterior agrupamiento en un "String" separados por un elemento diferenciador "&".

Finalmente se envían los datos mediante la función server.send, donde el 200 es un código que indica que todo es correcto, "text/plain" es el tipo de texto y la t1 es el texto a enviar.

Esta información llegará a la Aplicación la cual, mediante el método correspondiente , dividirá el mensaje a partir del diferenciador y guardará los datos para después mostrarlos por pantalla.

5.4 Envío de datos por Bluetooth Android

Se ha visto anteriormente que el envío de la identificación del usuario y la contraseña asociada para conectar el Arduino con la red Wifi se realiza mediante conexión bluetooth entre el dispositivo Android y el Arduino.

El envío de datos por bluetooth viene dado por las variables bluetoothsocket y bluetoothadapter, ya implementadas en el código de Android. También se ha de añadir el UUID del dispositivo bluetooth, que siempre es el mismo: 00001101-0000-1000-8000-00805F9B34FB.

El bluetoothAdapter, como su nombre indica, hace referencia al adaptador de bluetooth del dispositivo, y nos permite realizar tareas como la obtención de los dispositivos bluetooth vinculados al móvil.

El bluetoothsocket hace referencia a un canal de conexión para el bluetooth y se encarga de administrar la conexión.

El UUID es código de identificación único para el dispositivo Bluetooth. En los dispositivos Android es siempre el mismo.

El primer paso es crear la vista de los dispositivos enlazados, añadiendo para ello un Arraylist. Este elemento es una lista que mostrará una cierta cantidad de datos. Este campo será donde aparezcan los dispositivos bluetooth enlazados con el dispositivo móvil. Una vez creado, se inicializa el bluetoothadapter por defecto del dispositivo móvil, ya que viene integrado en la gran mayoría de móviles. Por último, se comprueba que el bluetooth esté activo. En caso de que no lo estuviese, se solicitará al usuario la activación del mismo.

Una vez esté activado el bluetooth del dispositivo móvil, se muestran los dispositivos vinculados al pulsar el botón de “Dispositivos Enlazados”. Al pulsar uno de los elementos de la lista (dispositivos bluetooth vinculados), nos conectaremos con él. Esta parte se realiza leyendo la dirección MAC de dicho dispositivo.

La dirección MAC se corresponde con la dirección física del dispositivo y es única para cada uno. Se obtiene con los últimos 17 caracteres dados por el dispositivo vinculado que aparecen en la Arraylist.

Una vez tenemos los datos del dispositivo enlazado, se guarda la dirección y procedemos a entrar en la siguiente pantalla, donde la aplicación nos pedirá la SSID o nombre de usuario de la red Wifi y su contraseña para su envío al Arduino. Para ello se procede a la conexión con el dispositivo bluetooth seleccionado anteriormente accediendo primero a la dirección antes guardada y comprobando que está disponible, y después estableciendo la conexión.

Para indicar al Arduino que se van a enviar datos acerca de la identificación de la Wifi y la contraseña, se ha implementado el envío de un “string” al inicializar la conexión. De esta manera, el dispositivo Andriod guardará dichos valores.

Una vez conectado el microcontrolador se realiza el envío de los datos escritos en los campos de texto correspondientes. Para ello y como han de enviarse en un mismo mensaje, primero se comprueba cuantos caracteres tiene cada uno y después se crea un "String" que estará formado de la siguiente forma: "XnombredeusuarioYcontraseña" donde X es el número de caracteres del identificador del usuario , "nombredeusurario" es, como su nombre indica, el nombre de usuario de la red Wifi introducido en el campo de texto correspondiente, Y indica el número de caracteres de la contraseña y "contraseña" es el valor de ésta. Se ha decidido añadir el número de caracteres para aumentar la sencillez de su posterior desglosamiento en dos cadenas diferentes dentro de Arduino, ya que han de guardarse independientemente.

5.5 Recepción de datos por Bluetooth en Arduino

La recepción de datos por parte de Arduino se realiza a través de los puertos Rx y Tx a los que está conectado el dispositivo bluetooth. Dichos puertos son los puertos serie genéricos de Arduino, y se utilizan para el envío y recepción de datos por el Arduino. Suelen utilizarse para los dispositivos bluetooth, Wifi o la conexión directa con otros Arduinos. Utilizan el bus I2C para el intercambio de información.

El bluetooth está conectado pues a los pines Rx y Tx. Por estos pines llegarán los datos y se podrán leer a través del bus serie de Arduino. Como se ha visto en el punto anterior, primero llegará un mensaje (en este caso se ha identificado como "NuevaWifi"), lo cual indicará que se van a enviar los datos para el cambio de conexión a una nueva red Wifi. Para ello se implementa al Arduino en modo escucha para el bus serie de la siguiente manera:

```
if (Serial.available()>0)
{
  string = "";
}
while (Serial.available()>0)
{
  command = ((byte)Serial.read());
  if (command == ':')
  {
    break;
  }
  else
  {
    string += command;
  }
  delay(10);
}
```

Mientras el Arduino esté disponible se mantendrá en escucha, y una vez los datos comiencen a llegar, se leerán y se añadirán uno por uno a un string. Si en este string se recibe el dato

“NuevaWifi”, se habilitará de nuevo el modo escucha y se guardarán los datos en un nuevo string para su guardado en la EEPROM del Arduino.

5.6 Guardado y extracción de datos en la EEPROM

La EEPROM es la memoria física no volátil del microcontrolador Arduino. Cuando el microcontrolador inicia un programa, sus variables son siempre borradas e inicializadas con un determinado valor inicial. Es decir, cada vez que el Arduino se resetea, olvida todo lo anterior y vuelve a ejecutar el programa desde el principio.

En este caso estas acciones no suponen ningún problema, excepto para el usuario y la contraseña del Wifi ya que son datos que el Arduino no lleva programados en el código debido a que cada vivienda tiene un usuario y contraseña diferentes. Por lo tanto, son unos datos que posiblemente sean cambiados cada cierto tiempo.

Así pues, se ha implementado el guardado de los archivos en la memoria EEPROM. Esta memoria funciona bit a bit, es decir, que guarda un dato en cada celda, por lo que no se puede almacenar directamente todo en una variable. Debido a este hecho, se ha implementado la adición de un número en el envío del nombre de usuario y de la contraseña, para así conocer cuantas casillas o celdas hemos de leer de la EEPROM para extraer solamente el dato requerido y evitar fallos debido a datos de diferente longitud.

Se debe leer dos veces. La primera lectura desde la casilla 1 hasta la longitud del tamaño del identificador del usuario (dato proporcionado por la casilla 0). La segunda lectura se realiza desde la casilla a la que se quedó anteriormente más una (debido al segundo número) hasta el total del mensaje, quedando el código de la siguiente manera:

```
void extraccion_datos() {
  int a = EEPROM.read(0);
   for (i=0;i<=a; i++ ){
      ssid += WyC[i];
    }
    i++
    int b = String (EEPROM.read(i)).toInt;
    for (j=i++;j<=(b+i+1);j++{
      password += WyC[j];
     }
}
```

Para el guardado simplemente se vuelcan los datos en la memoria.

6. Métodos y conexiones electrónicas Arduino

6.1 Conexión de componentes a Arduino

La conexión de los diferentes componentes a Arduino se realizará de la siguiente manera:

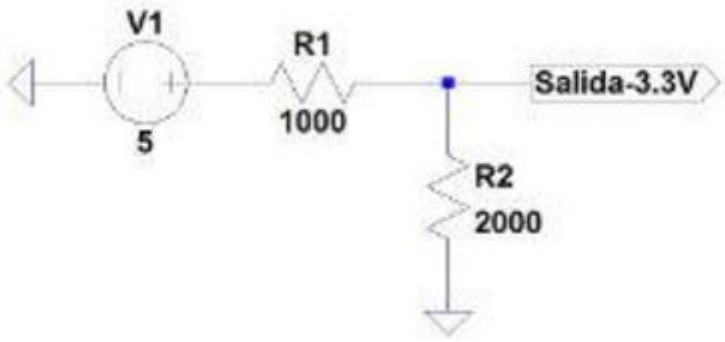
- El encendido de luces tendrá asignado un pin de salida, el cual se encargará de enviar la señal que activará o desactivará el encendido de las mismas. Se explica el funcionamiento de dicho circuito en el apartado 6.2.
- La apertura de la persiana tendrá asignados 4 pines (su funcionamiento se explica en el punto 6.3):
  - Dos de ellos son pines de salida, que son los encargados de enviar la señal al circuito para controlar el sentido de la corriente para la subida o bajada de la persiana.
  - Los otros dos son pines de entrada, conectados a los sensores que se encargan de recibir la señal correspondiente cuando la persiana haya llegado a su límite.
- La apertura de la puerta es igual al de la persiana, utilizando otro pin distinto. Su funcionamiento se explica en el punto 6.4.

Finalmente queda la conexión al dispositivo Bluetooth, el cual tiene el problema de que funciona a 3,3V mientras que el Arduino emite la señal a 5V, por lo que va a ser necesario añadir un divisor de tensión para evitar daños en el mismo.

El divisor de tensión consiste en colocar dos resistencias: la primera en serie con el dispositivo que queremos conectar, y la segunda en paralelo a tierra. Para calcular la tensión de salida, se tiene la siguiente ecuación:

$$V_{\text{out}} = \frac{R_2}{R_1 + R_2} \cdot V_{\text{in}}$$

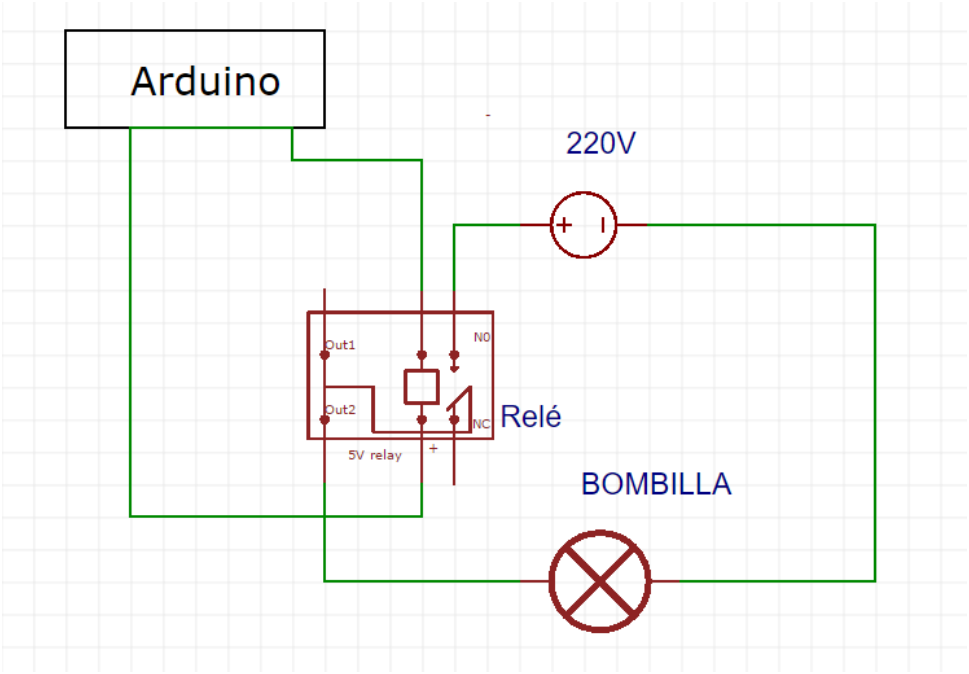
Resolviendo la ecuación anterior, y sabiendo que disponemos de resistencias de 1kΩ obtendríamos que R2 sería de 2kΩ quedando así el circuito:



6.2 Control de luces

La implementación del encendido de las luces se ha realizado mediante dos métodos sencillos, uno para el encendido y otro para el apagado de la luz en cuestión, dependiendo de la orden que llegue del servidor. Cuando le llega la orden de encendido de luces, el sistema ejecutará un método que enviará una señal de voltaje por el pin correspondiente del Arduino, el cual además está conectado a un relé. Esta señal cambiará el estado del relé el cual permitirá el paso de la corriente por el circuito y encenderá la luz correspondiente.

En el caso de que le llegue la orden de apagado, la señal enviada al relé volverá a un valor nulo, y en consecuencia el relé cerrara el circuito y se apagará la luz correspondiente. Un esquema de este funcionamiento se puede ver a continuación:



6.3 Control de persiana

De forma análoga al ejemplo anterior, dependiendo de la orden que nos llegue se va a ejecutar la subida, parada o bajada de la persiana de la habitación.

Primero se va a explicar cómo funciona la persiana y después se procederá a la implementación de los métodos y su posterior instalación.

La persiana constará de 3 ordenes básicas: subida, parada y bajada, así como de 3 estados de persiana: Arriba, Abajo y Parada.

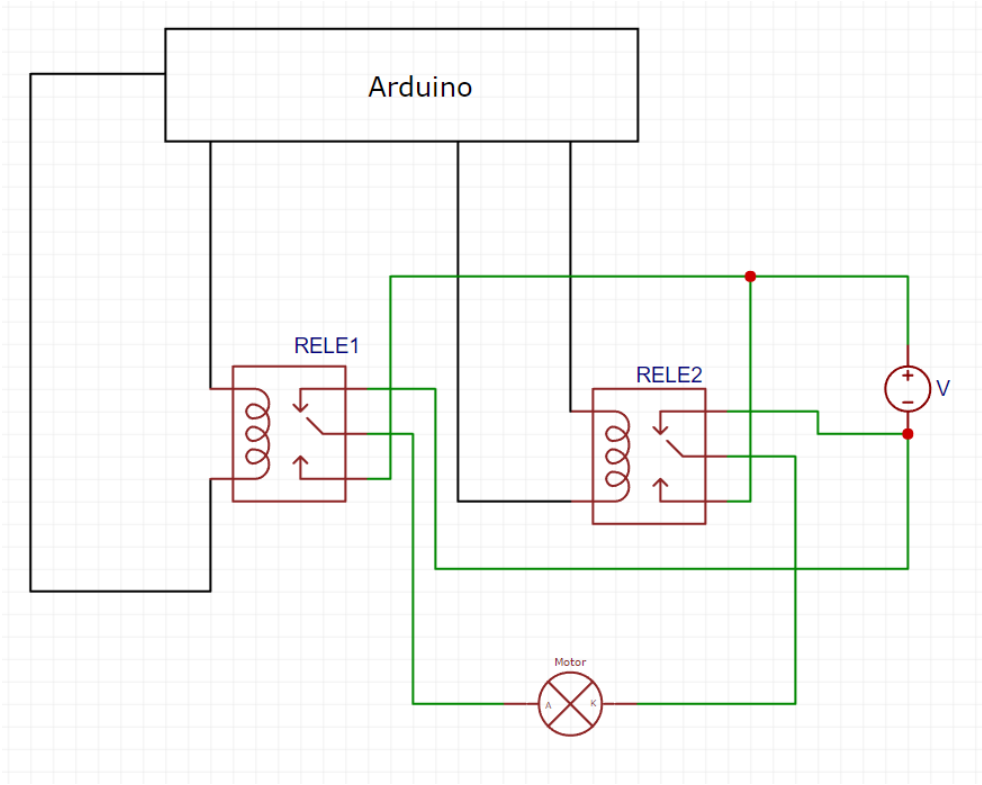
Cuando la persiana está en subida se ha de tener en cuenta que tiene un límite máximo cuando está arriba del todo. Para la medición de dicho límite se ha optado por utilizar un sensor de campos magnéticos, el cual se verá alterado y mandará una señal al Arduino cuando se forme un campo en sus inmediaciones.

Para ello se ha instalado un imán en el extremo inferior de la persiana, y otros dos en la parte superior e inferior de la ventana junto con el sensor magnético, por lo que cuando el imán de

la parte inferior de la persiana se alinee junto con el de la parte superior, se creará un campo magnético y nos enviará la señal de que ha llegado al límite superior para así forzar la parada de la misma. De esta forma, también se ha implementado que el método de subida no pueda ser ejecutado mientras el sensor superior esté activo, ya que implicaría un pequeño movimiento de la persiana debido al retardo en el envío de la señal, y ejecutar esta acción repetidas veces ocasionaría una subida por encima del límite, lo que provocaría que, al no recibir más dicha señal, siguiese subiendo hasta que se le indique lo contrario, con la posible avería que causaría esta acción.

En el caso de bajada, la acción a realizar sería similar, teniendo un sensor diferente para que no se pueda seguir bajando la persiana cuando ya está abajo.

La instalación eléctrica vendría dada por dos relés conectados con un motor de corriente continua, el cual cambiará de sentido dependiendo del relé activo. El valor de los relés cambiará según la polaridad del circuito o se parará si ambos están desconectados. En este caso el Arduino se encarga de enviar dos señales diferentes dependiendo de la orden recibida, y cada señal enviará una señal eléctrica al pin correspondiente al relé que ejecuta dicha acción.



Como se puede comprobar en el circuito anterior, si los dos relés están desconectados, el motor permanecerá inactivo. Si el Relé 1 está activo, la corriente circulará en un sentido y si el Relé 2 se activa, el sentido de la corriente será contrario. Nótese que en el caso de que los dos relés estuviesen activados se produciría un cortocircuito por lo que se ha implementado un sistema para evitar la posible conexión de ambos relés a la vez.

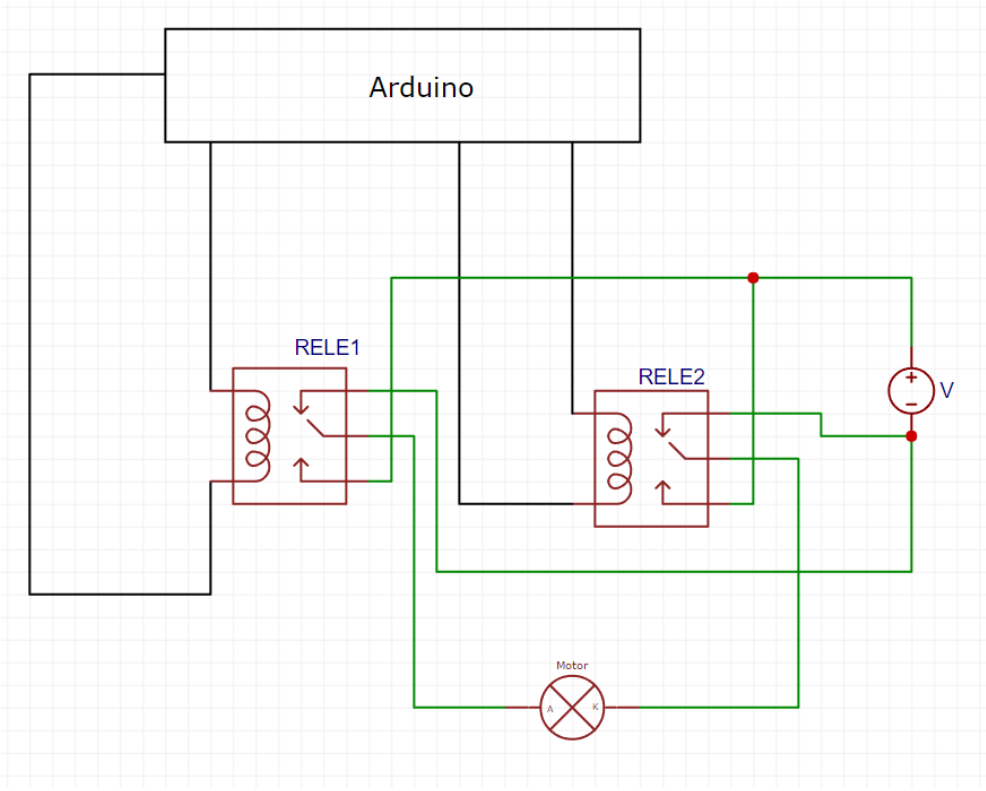
No hay ninguna acción que permita que ambos estén conectados, pero puede darse el caso de que un relé se conecte mientras el otro se cierre y de lugar a cortocircuito por un breve periodo de tiempo, por lo que se ha añadido que cada vez que activemos un relé el otro se

cerrará antes, y habrá una pequeña demora antes de enviar la señal al segundo para su activación.

6.4 Control de puertas corredizas

El método de control de puertas corredizas tiene una mayor similitud con el de la persiana, ya que su funcionamiento es similar. Se tienen 3 estados de puertas: Abierta, Cerrada o Parada, así como 3 órdenes básicas: Abrir, Cerrar y Parar.

El funcionamiento es similar, ya que se han instalado también sensores magnéticos en el extremo inicial de la puerta y dos en la parte inicial y final del marco de esta, para cuando dicha puerta llegue a un extremo (abierto o cerrado) se active el sensor de campos y envíe la señal de parada al Arduino.



Como se puede comprobar el funcionamiento de la electrónica es el mismo que el de la persiana. La única diferencia radica en que el motor, en lugar de ir conectado a un eje para su giro, va conectado a una cremallera para crear el movimiento horizontal que permitirá la apertura de la puerta corrediza.

También se puede añadir, para comodidad del usuario, un sensor de proximidad por ultrasonidos para la detección de un objeto cercano a la puerta. Así, cuando un usuario se acerque a la misma, la puerta se abrirá automáticamente y se cerrará cuando no detecte ningún objeto o persona cercana.

Este sensor funciona mediante el envío de ultrasonidos u ondas mecánicas, las cuales tardan un tiempo en golpear el destino y regresar al sensor. Conociendo la velocidad del sonido se puede calcular la distancia recorrida y así activar la puerta cuando haya un objeto a cierta

distancia de la misma. Por suerte estas operaciones ya vienen realizadas por la librería de Arduino de dicho sensor y no sería necesario calcularlo, simplemente se programa directamente con la distancia en cm.

7. BIBLIOGRAFIA

Arduino:

- CFP de la UPV: Taller de desarrollo de sistemas domóticos basados en Arduino.
- https://www.arduino.cc/ para el aprendizaje de que hacen y las funciones de cada uno de los componentes y la descarga del IDE de Arduino.
- http://www.educachip.com/como-usar-la-memoria-eeprom-de-arduino/
- http://circuits.io/ Para la realización de pruebas de circuitos
- Enrique Hernandez Orallo, Jose Hernandez Orallo; Programación en C++. Editorial Parainfo, S.A. (1993)

Android

- Para el aprendizaje de volley: http://gpmess.com/blog/2014/05/28/volley-usando-webservices-en-android-de-manera-sencilla/
- https://androidstudiofaqs.com/tutoriales
- Temario asignatura "soluciones informáticas para dispositivos móviles" de la escuela ETSIINF de la UPV
- https://developer.android.com/guide/index.html
- Pedro Manuel Cuenca Jiménez; Programación en JAVA. Editorial ANAYA, (1997)
- Microsoft; Arquitectura de aplicaciones para .NET. Diseño de aplicaciones y servicios. Editorial McGrawHill (2003)

Para la realización de circuitos eléctricos para la memoria se utilizó: https://easyeda.com/es

Para el diseño de los diagramas se ha utilizado: https://cacoo.com/diagrams/

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

II. PLIEGO DE CONDICIONES

Curso Académico: 2016-17

ÍNDICE DEL PLIEGO DE CONDICIONES

1. Condiciones legales del software utilizado 36

.1 Android Studio 36

1.2 Arduino IDE 36

2. Requisitos Técnicos requeridos por el software empleado 37

2.1 Android Studio 37

2.2 Arduino IDE 38

3. Plazos 39

1. Condiciones legales del software utilizado

1.1 Android Studio

Las condiciones legales que rigen el uso del entorno de programación de Android Studio pueden encontrarse en el apartado designado a tal efecto en la página web del Android Studio, siendo el enlace de dicha página el siguiente: https://developer.android.com/studio/terms.html

El Android Studio es freeware y está proporcionado por Google. Por lo tanto, las restricciones legales que le son aplicables son escasas. En cuanto a su uso, no existe ninguna restricción relevante, puesto que el programa puede utilizarse y distribuirse libremente, si bien la distribución queda restringida por el hecho de que solo puede distribuirse el programa completo sin realizar modificaciones sobre él y sin cobrar nada por ello. En consecuencia, el trabajo desarrollado con este entorno de programación podría llegar a comercializarse sin que ello suponga una violación de los términos de la licencia.

Cabe destacar que para comercializar una aplicación en Android y subirla a GooglePlay es necesario tener una licencia de desarrollador de Android, licencia que puede ser comprada a Google desde una cuenta de Gmail.

1.2 Arduino IDE

El IDE de Arduino es un software libre proporcionado por el propio Arduino. Las restricciones legales son nulas, por lo que puede ser modificado en libertad y utilizado para lo que se crea conveniente. En consecuencia, el trabajo desarrollado con este entorno de programación puede comercializarse sin que ello suponga una violación de términos legales.

Se pueden leer los términos de uso en la parte inferior de la página de Arduino a partir de este enlace: https://www.arduino.cc/en/Main/Software

Nótese que al dar total libertad para el uso de dicho software, no se hacen responsables de los problemas que puedan surgir con la utilización de éste, sean del tipo que sean.

2. Requisitos técnicos requeridos por el software empleado

2.1 Android Studio

En la página web de Android Studio se pueden ver los requisitos mínimos que se exigen para su correcto funcionamiento:

Windows

- Microsoft® Windows® 7/8/10 (32 o 64 bits).
- 3 GB de memoria RAM como mínimo (se recomiendan 8), más 1 GB para el emulador de Android.
- 2 GB de espacio en disco disponible como mínimo
  (se recomiendan 4); 500 MB para el IDE + 1,5 GB para Android SDK y la imagen de sistema del emulador.
- Resolución de pantalla mínima de 1280 x 800.
- Para el emulador acelerado: Sistema operativo de 64 bits y procesador Intel® compatible con Intel® VT-x, Intel® EM64T (Intel® 64) y la funcionalidad Execute Disable (XD) Bit.

Mac

- Mac® OS X® 10.10 (Yosemite) o versiones posteriores, hasta la 10.12 (macOS Sierra).
- 3 GB de memoria RAM como mínimo (se recomiendan 8), más 1 GB para el emulador de Android.
- 2 GB de espacio en disco disponible como mínimo
  (se recomiendan 4); 500 MB para el IDE + 1,5 GB para Android SDK y la imagen de sistema del emulador.
- Resolución de pantalla mínima de 1280 x 800.

Linux

- GNOME o KDE de escritorio.

Pruebas realizadas en Ubuntu® 12.04, Precise Pangolin (distribución de 64 bits capaz de ejecutar aplicaciones de 32 bits).

- Distribución de 64 bits capaz de ejecutar aplicaciones de 32 bits.
- GNU C Library (glibc) 2.19 o versiones posteriores.
- 3 GB de memoria RAM como mínimo (se recomiendan 8), más 1 GB para el emulador de Android.
- 2 GB de espacio en disco disponible como mínimo
  (se recomiendan 4); 500 MB para el IDE + 1,5 GB para Android SDK y la imagen de sistema del emulador.
- Resolución de pantalla mínima de 1280 x 800.
- Para el emulador acelerado: Procesador Intel® compatible con Intel® VT-x, Intel® EM64T (Intel® 64) y la funcionalidad Execute Disable (XD) Bit, o procesador AMD compatible con AMD Virtualization™ (AMD-V™)

El emulador lo que permite es la simulación de un dispositivo Android virtual desde el ordenador en el cual se puede instalar y probar la aplicación que se está diseñando sin necesidad de un dispositivo Android externo.

También cabe destacar que la función de auto detección de errores de Android Studio implica un alto consumo, por lo que estos requisitos son para la Aplicación con dicha función desactivada.

2.2 Arduino IDE

El IDE de Arduino tienes unos requisitos equivalentes a la máquina virtual de java 8, por lo que son requisitos muy bajos que cualquier equipo de hoy en día es capaz de alcanzar. Desde la página oficial de java podemos ver que son los siguientes:

Windows

- Windows 10 (8u51 y superiores)
- Windows 8.x (escritorio)
- Windows 7 SP1
- Windows Vista SP2
- Windows Server 2008 R2 SP1 (64 bits)
- Windows Server 2012 y 2012 R2 (64 bits)
- RAM: 128 MB
- Espacio en disco: 124 MB para JRE; 2 MB para Java Update
- Procesador: Mínimo Pentium 2 a 266 MHz
- Exploradores: Internet Explorer 9 y superior, Firefox

Mac OS X

- Mac con Intel que ejecuta Mac OS X 10.8.3+, 10.9+
- Privilegios de administrador para la instalación
- Explorador de 64 bits

Se requiere un explorador de 64 bits (Safari, por ejemplo) para ejecutar Oracle Java en Mac.

Linux

- Oracle Linux 5.5+[1]
- Oracle Linux 6.x (32 bits), 6.x (64 bits)[2]
- Oracle Linux 7.x (64 bits)[2] (8u20 y superiores)
- Red Hat Enterprise Linux 5.5+[1], 6.x (32 bits), 6.x (64 bits)[2]
- Red Hat Enterprise Linux 7.x (64 bits)[2] (8u20 y superiores)
- Suse Linux Enterprise Server 10 SP2+, 11.x
- Suse Linux Enterprise Server 12.x (64 bits)[2] (8u31 y superiores)
- Ubuntu Linux 12.04 LTS, 13.x
- Ubuntu Linux 14.x (8u25 y superiores)
- Ubuntu Linux 15.04 (8u45 y superiores)
- Ubuntu Linux 15.10 (8u65 y superiores)
- Exploradores: Firefox

3. Plazos

Desde el inicio del proyecto se establece un plazo máximo de 2 semanas para el estudio de las alternativas y la selección de los métodos de conexión entre los dispositivos.

El diseño de la interfaz de la aplicación Android tendrá lugar en el plazo de 1 semana, junto con las funciones básicas del mismo.

La implementación de los métodos de conexión entre dispositivos, envío de datos y las pruebas necesarias para su correcto funcionamiento se realizará en un plazo de 3 semanas

Finalmente el diseño de los circuitos y la implementación de los métodos en Arduino y su correspondiente lógica serán llevadas a cabo en un plazo de 2 semanas y media.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

III. PRESUPUESTO

Curso Académico: 2016-17

INDICE DEL PRESUPUESTO

1. Presupuesto de ejecución por contrata 42

2. Presupuesto de ejecución material por capítulos 43

3. Estado de mediciones 46

4. Precios descompuestos 48

5. Justificación de precios unitarios 54

1. PRESUPUESTO DE EJECUCIÓN POR CONTRATA

Capítulo	Importe (€)
Capítulo 1 Investigación y aprendizaje	379,413
Capítulo 2 Diseño de la aplicación Android	443,003
Capítulo 3 Implementación del sistema de envío de datos	5228,8126
Capítulo 4 Implementación métodos y diseño electrónico	1567,909
Costes Materiales por vivienda	142
Presupuesto de ejecución Material	7761,1376
5% gastos generales	388,05688
Suma	8149,19448
21% IVA	1711,330841
Presupuesto de ejecución por contrata	9860,525321

Asciende el presupuesto de ejecución por Contrata a la expresada cantidad de NUEVE MIL OCHOCIENTOS SESENTA EUROS CON CINCUENTA Y DOS CÉNTIMOS (9.860,52).

2. PRESPUESTO DE EJECUCIÓN MATERIAL POR CAPITULOS

Presupuesto Parcial Nº1 Investigación tecnologías.

Núm.	Código	Ud.	Denominación	Cantidad	Precio(€/h)	Total (€)
1,1	Hor1	h	Investigación tecnologias	9	42,157	379,413

Total Presupuesto Parcial nº1 = 379,413€

Presupuesto Parcial nº2 Diseño de la aplicación Android.

Núm.	Código	Ud.	Denominación	Cantidad	Precio(€/h)	Total (€)
2.1	Hor2	h	Diseño y estructuración de la interfaz gráfica	11	26,059	286,649
2.2	Hor3	h	Establecimiento de las funciones básicas	6	26,059	156,354

Total Presupuesto Parcial nº2 = 443,003€

Presupuesto Parcial nº3 Implementación del sistema de envío de datos.

Núm.	Código	Ud.	Denominación	Cantidad	Precio(€/h)	Total (€)
3.1	Hor4	h	Estudio de Alternativas para el envío	10	31,261	312,61
3.2	Hor5	h	Implementación métodos envío y recepción de datos de la Aplicación Android por Wifi	37	33,3418	1233,6466
3.3	Hor6	h	Implementación métodos envío y recepción de datos de la Aplicación Android por Bluetooth	31	42,157	1306,867
3.4	Hor7	h	Creación del Servidor	16	33,3418	533,4688
3.5	Hor8	h	Implementación métodos envío y recepción de datos en Arduino por Wifi	28	31,261	875,308
3.6	Hor9	h	Implementación métodos envío y recepción de datos en Arduino por Bluetooth	29	33,3418	966,9122

Total Presupuesto Parcial nº3 = 5228,8126€

Presupuesto Parcial nº4 Implementación métodos y diseño electrónico.

Núm.	Código	Ud.	Denominación	Cantidad	Precio(€/h)	Total (€)
4.1	Hor10	h	Creación y diseño de los métodos para llevar a cabo las funciones	16	36,463	583,408
4.2	Hor11	h	Diseño del circuito para realizar las funciones	8	36,463	291,704
4.3	Hor12	h	Implementación del guardado de datos en la memoria interna (EEPROM)	19	36,463	692,797

Total Presupuesto Parcial nº4 = 1567,909€

3. ESTADO DE MEDICIONES

Presupuesto Parcial Nº1 Investigación tecnologías.

N.º	Ud.	Descripción	Medición
1.1	h	Hora de trabajo dedicada a la investigación de las tecnologías a utilizar	9
		Total h…….:	**9**

Presupuesto Parcial Nº2 Diseño de la aplicación Android.

N.º	Ud.	Descripción	Medición
2.1	h	Diseño y estructuración de la interfaz gráfica	11
2.2	h	Establecimiento de las funciones básicas	6
		Total h…..:	**17**

Presupuesto Parcial Nº3 Implementación del sistema de envío de datos.

N.º	Ud.	Descripción	Medición
2.1	h	Estudio de Alternativas para el envío	10
2.2	h	Implementación métodos envío y recepción de datos de la Aplicación Android por Wifi	37
3.3	h	Implementación métodos envío y recepción de datos de la Aplicación Android por Bluetooth	31
3.4	h	Creación del Servidor	16
3.5	h	Implementación métodos envío y recepción de datos en Arduino por Wifi	28
3.6	h	Implementación métodos envío y recepción de datos en Arduino por Bluetooth	29
		Total h……:	**151**

Presupuesto Parcial Nº4 Implementación métodos y diseño electrónico.

N.º	Ud.	Descripción	Medición
4.1	h	Creación y diseño de los métodos para llevar a cabo las funciones	16
4.2	h	Diseño del circuito para realizar las funciones	8
4.3	h	Implementación del guardado de datos en la memoria interna (EEPROM)	19
		Total h……:	**43**

4. PRECIOS DESCOMPUESTOS.

Código	Ud.	Descripción		Total
1.1 Hor1	**h**	**Hora de trabajo dedicada investigar las diferentes tecnologías para realización de sistemas domóticos.**		
Maq1	1 H	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049
MO1	1 H	Graduado en GITI	35	35
MO2	0,2 H	Programador (JAVA Y PHP)	25	5
%	2 %	Costes directos complementarios	40,05	0,801
	2 %	Costes Indirectos	40,851	0,81702
			Precio Total por h.	**42,15702**
2.1 Hor2	**h**	**Hora de trabajo dedicada al Diseño y estructuración de la interfaz gráfica**		
Maq1	1 H	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049
MO1	0 H	Graduado en GITI	35	0
MO2	1 H	Programador (JAVA y PHP)	25	25
%	2 %	Costes directos complementarios	25	0,5
	2 %	Costes Indirectos	25,5	0,51
			Precio Total por h.	**26,059**

2.2 Hor3	**h**		**Hora de trabajo dedicada al establecimiento de las funciones básicas**		
Maq1	1	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049
MO1	0	h	Graduado en GITI	35	0
MO2	1	h	Programador (JAVA y PHP)	25	25
%	2	%	Costes directos complementarios	25	0,5
	2	%	Costes Indirectos	25,5	0,51
				Precio Total por h.	**26,059**

3.1 Hor4	**h**		**Hora de trabajo dedicada al estudio de Alternativas para el envío**		
Maq1	1	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049
MO1	0.5	h	Graduado en GITI	35	17,5
MO2	0,5	h	Programador (JAVA y PHP)	25	12,5
%	2	%	Costes directos complementarios	30	0,6
	2	%	Costes Indirectos	30,6	0,612
				Precio Total por h.	**31,261**

3.2 Hor5		**h**	**Hora de trabajo dedicada a la Implementación métodos envío y recepción de datos de la Aplicación Android por Wifi**			
Maq1	1	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049	
MO1	0.2	h	Graduado en GITI	35	7	
MO2	1	h	Programador (JAVA y PHP)	25	25	
%	2	%	Costes directos complementarios	32	0,64	
	2	%	Costes Indirectos	32,64	0,6528	
				Precio Total por h.	**33,3418**	

3.3 Hor6		**h**	**Hora de trabajo dedicada a la Implementación métodos envío y recepción de datos de la Aplicación Android por Bluetooth**			
Maq1	1	H	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049	
MO1	1	H	Graduado en GITI	35	35	
MO2	0,2	H	Programador(JAVA Y PHP)	25	5	
%	2	%	Costes directos complementarios	40,05	0,801	
	2	%	Costes Indirectos	40,851	0,81702	
				Precio Total por h.	**42,15702**	

3.4 Hor7		**h**	**Hora de trabajo dedicada a la creación del Servidor**			
Maq1	1	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049	
MO1	0.2	h	Graduado en GITI	35	7	
MO2	1	h	Programador (JAVA Y PHP)	25	25	
%	2	%	Costes directos complementarios	32	0,64	
	2	%	Costes Indirectos	32,64	0,6528	
				Precio Total por h.	**33,3418**	

3.5 Hor8		**h**	**Hora de trabajo dedicada a la Implementación métodos envío y recepción de datos en Arduino por Wifi**			
Maq1	1	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049	
MO1	0.5	h	Graduado en GITI	35	17,5	
MO2	0,5	h	Programador (JAVA y PHP)	25	12,5	
%	2	%	Costes directos complementarios	30	0,6	
	2	%	Costes Indirectos	30,6	0,612	
				Precio Total por h.	**31,261**	

3.6 Hor9	**h**		**Hora de trabajo dedicada a la Implementación métodos envío y recepción de datos en Arduino por Bluetooth**			
Maq1		1	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049
MO1		0.2	h	Graduado en GITI	35	7
MO2		1	h	Programador (JAVA y PHP)	25	25
%		2	%	Costes directos complementarios	32	0,64
		2	%	Costes Indirectos	32,64	0,6528
					Precio Total por h.	**33,3418**

4.1 Hor10	**h**		**Hora de trabajo dedicada a la creación y diseño de los métodos para llevar a cabo las funciones**			
Maq1		0	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049
MO1		1	h	Graduado en GITI	35	35
MO2		0	h	Programador (JAVA Y PHP)	25	0
%		2	%	Costes directos complementarios	35	0,7
		2	%	Costes Indirectos	35,7	0,714
					Precio Total por h.	**36,463**

4.2	**Hor11**	**h**		**Hora de trabajo dedicada al diseño del circuito para realizar las diferentes funciones**		
	Maq1	0	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049
	MO1	1	h	Graduado en GITI	35	35
	MO2	0	h	Programador (JAVA y PHP)	25	0
	%	2	%	Costes directos complementarios	35	0,7
		2	%	Costes Indirectos	35,7	0,714
					Precio Total por h.	**36,463**

4.3	**Hor12**	**h**		**Hora de trabajo dedicada a la Implementación del guardado de datos en la memoria interna (EEPROM)**		
	Maq1	0	h	Ordenador (CPU, pantalla, teclado, etc.)	0,049	0,049
	MO1	1	h	Graduado en GITI	35	35
	MO2	0	h	Programador (JAVA y PHP)	25	0
	%	2	%	Costes directos complementarios	35	0,7
		2	%	Costes Indirectos	35,7	0,714
					Precio Total por h.	**36,463**

5. JUSTIFICACIÓN DE LOS PRECIOS UNITARIOS

Para establecer el precio de los materiales y maquinaria que se requieren en el desarrollo del presente proyecto se ha decidido calcular el coste por hora en base a su precio y a su vida útil.

En el caso del ordenador se ha considerado que su precio ronda unos 500€ y que su vida útil es en torno a los 5 años. Por lo tanto, considerando que un año está compuesto por 52 semanas con 5 días laborables cada una y 8h de trabajo por día, el coste de utilizar una hora el ordenador resulta en:

$$Coste\ Ordenador = \frac{500}{5 \times 52 \times 1 \times 8} = 0{,}049\ €/h$$

Las licencias de los programas utilizados son gratuitas por los que su coste será nulo.

Vamos ahora ver el listado de precios de los componentes electrónicos:

- Placa Arduino NodeMCU ESP8266 : 9€ (Se requiere una por habitación)
- 40 Cables Macho-Macho : 3€
- 40 cables Macho-Hembra: 3€
- Modulo Bluetooth HC-06 para Arduino: 10€ (Se requiere uno por Arduino)
- Relés Arduino: 1,8 € (Requeridos 2 por persiana o puerta y 1 por luces)
- 20 resistencias para componentes Arduino: 1,5€

Coste total por habitación suponiendo una persiana, una puerta y una luz general: 35,5€

Determinando que una casa dispone de 4 habitaciones a automatizar el coste ascendería a 142€.

Estos costes son suponiendo que la vivienda consta ya de persianas eléctricas y puertas corredizas, sino habría que agregarle los costes de instalación de las mismas para poder ser controladas por el sistema domótico.