ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE CENTRAL TELEFÓNICA CON APLICACIÓN DOMÓTICA DE BAJO COSTO

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

JUAN CARLOS ESTÉVEZ AYALA

juan.estevez@epn.edu.ec

DIRECTOR: PhD. LUIS FELIPE URQUIZA AGUIAR

luis.urquiza@epn.edu.ec

Quito, marzo 2018

AVAL

Certifico que el presente trabajo fue desarrollado por Juan Carlos Estévez Ayala, bajo mi supervisión.

PhD. LUIS FELIPE URQUIZA AGUIAR
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo, Juan Carlos Estévez Ayala, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

JUAN CARLOS ESTÉVEZ AYALA

DEDICATORIA

A mi madre Berónica en el cielo, por darme la vida, quererme mucho, creer en mí, por sembrar en mí el gusto por los estudios y porque siempre me apoyaste.

A mi padre Carlos, por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica, como de la vida, por su incondicional apoyo perfectamente mantenido a través del tiempo.

A mi hermano Israel, para que puedas ver en mí un ejemplo a seguir y notes que cualquier meta se la consigue con dedicación y esfuerzo.

A mi novia Dolly, quien me apoyó y me alentó a continuar.

A mis compañeros de la Poli, por compartir los buenos y malos momentos durante la carrera universitaria.

AGRADECIMIENTO

En primer lugar, agradezco a mis padres, ya que gracias a ellos pude formarme en esta prestigiosa institución, a mis familiares y mi novia por apoyarme durante toda mi carrera.

Doy gracias a mis compañeros “Los Carguas” por hacer de este paso por la poli más placentero, gracias por haber reído y llorado a mi lado.

Agradezco también al Ing. José Estrada y al Dr. Luis Urquiza, por sus aportes al desarrollo de este proyecto de titulación con su conocimiento y sus consejos.

A mis amigos John, Diegazo y Fernando por su ayuda en el desarrollo de esta tesis.

A los miembros de mi tribunal por sus recomendaciones y consejos.

Finalmente, agradezco a mis compañeros y compañeras de trabajo del Centro de Educación Continua (CEC-EPN), por ayudarme en tiempos difíciles.

A todos ellos les quedo eternamente agradecido.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE CÓDIGOS

RESUMEN

La telefonía es un servicio de telecomunicaciones que se ha potenciado enormemente gracias a la evolución en el trasporte de la voz mediante IP y a la flexibilidad de Linux para la implementación de este servicio. Es así que, sin tecnologías propietarias que limiten la aplicación de los diferentes protocolos de comunicaciones, las posibilidades de integración con distintos sistemas (en este caso la domótica) son inmensas.

Por otro lado, aunque el acceso a Internet ha permitido que los usuarios interactúen remotamente con sistemas domóticos, el acceso a este servicio no siempre está disponible, especialmente en nuestro país, y muchos sectores no están muy familiarizados con su uso. Por esa razón, es conveniente apoyarse o complementar las capacidades de Internet con otro mecanismo de comunicación, más familiar, e igualmente potente (aunque no tan flexible), como es la telefonía.

La domótica, asimismo, orientada a extender el control que tiene el usuario sobre los dispositivos de su hogar para su comodidad y seguridad, ha causado mucho interés, y gracias a la evolución tecnológica, los mecanismos para su implementación se han vuelto menos costosos y más funcionales.

Se habla, por lo tanto, de tecnologías vigentes que mediante su integración es posible brindarle al usuario un mecanismo flexible para controlar y obtener información de los dispositivos de su hogar de manera remota y sencilla.

PALABRAS CLAVE: Telefonía IP, Domótica, Linux.

ABSTRACT

Telephony is a telecommunications service that has been greatly enhanced thanks to the evolution in voice transport through IP and the flexibility of Linux for the implementation of this service. Thus, without proprietary technologies that limit the application of different communication protocols, the possibilities of integration with different systems (in this case home automation) are immense.

On the other hand, although access to the Internet has allowed users to interact remotely with home automation systems, access to this service is not always available, especially in our country, and many sectors are not very familiar with its use. For this reason, it is convenient to support or complement the Internet's capabilities with another communication mechanism, more familiar, and equally powerful (although not as flexible), such as telephony.

Home automation, also, aimed at extending the control that the user has over the devices of their home for their comfort and safety, has caused much interest, and thanks to technological evolution, the mechanisms for their implementation have become less expensive and more functional.

There is therefore talk of current technologies that through its integration is possible to provide the user with a flexible mechanism to control and obtain information from their home devices remotely and easily.

KEYWORDS: IP Telephony, Domotics, Linux.

1. INTRODUCCIÓN

En el presente proyecto de titulación se presenta el desarrollo de un prototipo de central telefónica con aplicación domótica, la misma permitirá al usuario controlar dispositivos de su hogar de manera local y remota, el acceso remoto se lo realizará mediante una llamada telefónica al hogar (central telefónica) y el acceso local mediante una aplicación web o móvil (Android).

Además de controlar dispositivos del hogar, el prototipo será capaz de levantar alertas de sensores incorporados al sistema domótico y realizar una llamada al propietario del hogar para informarle acerca de la alerta que se ha provocado.

El prototipo cuenta también con la funcionalidad de poder comunicar mediante una llamada telefónica el portero eléctrico del hogar con el dueño de la casa, en caso de que el dueño del hogar no se encuentre en casa.

En el primer capítulo se describe los principales conceptos de los servicios utilizados en el prototipo desarrollado; se describen las características de las plataformas utilizadas: Raspberry Pi y Arduino; se hace una revisión del estado actual de la domótica y de sistemas de seguridad que se pueden implementar en el hogar; también se revisa conceptos de telefonía IP, así mismo; se describen las herramientas utilizadas para el desarrollo del prototipo.

En el segundo capítulo se detalla el proceso de diseño del prototipo el mismo que se fundamentará en los requisitos tanto del hardware como del software, así como también de protocolos y servicios necesarios para el buen funcionamiento del prototipo. Además, se describe el proceso de implementación de los servicios y componentes del prototipo, se presentan las pruebas realizadas para verificar y validar la implementación y por último se presenta el costo referencial del desarrollo del prototipo.

En el tercer capítulo se presentan las pruebas realizadas para verificar y validar la implementación y se presenta el costo referencial del desarrollo del prototipo.

En el cuarto capítulo se presentan el análisis de los resultados obtenidos en el trabajo realizado.

1.1 Objetivos

El objetivo general de este Proyecto Técnico es: Diseñar e implementar un prototipo de central telefónica con aplicación domótica y de bajo costo a través de la PSTN (Red de Telefonía Pública) y la red de datos local.

Los objetivos específicos de este Proyecto Técnico son:

- Describir los fundamentos teóricos de la telefonía IP que están relacionados con el prototipo propuesto y los elementos de una solución domótica.
- Diseñar e implementar un prototipo de central telefónica de bajo costo con una aplicación domótica.
- Implementar un servidor web, con alcance a nivel LAN, que facilite el monitoreo y control del sistema domótico.
- Verificar la funcionalidad del prototipo mediante la realización de pruebas.
- Presentar el costo del prototipo propuesto.

1.2 Alcance

En este proyecto se propone el diseño e implementación de un prototipo de central telefónica de bajo costo a través de la que se ofrezca una aplicación domótica utilizando la plataforma de placa reducida Raspberry Pi y el microcontrolador Arduino.

Para empezar, se describirá los conceptos básicos relacionados con la telefonía IP, su implementación mediante herramientas open source (Asterisk) y la interacción que le ofrece al usuario con las redes de datos y la PSTN. Del mismo modo, se hará una breve revisión de los elementos y objetivos de un sistema domótico orientado a la comodidad y seguridad en el hogar.

Posteriormente, se realizará el diseño del prototipo de central telefónica de bajo costo con aplicación domótica. Para ello, se determinará algunas de las necesidades más comunes de un usuario para la automatización de una vivienda de modo que éstas se puedan satisfacer a través de comandos (tonos DTMF) mediante la línea telefónica. El análisis y la implementación se concentrarán en permitir al menos las siguientes tareas:

- Control de dispositivos.
- Monitoreo de variables ambientales.
- Simulación de presencia.

Luego, se implementará el prototipo de central telefónica con aplicación domótica, de acuerdo a las necesidades determinadas en el proceso de diseño.

Para asegurar un costo reducido del prototipo se empleará un ordenador de placa reducida (Raspberry Pi) para la central telefónica, una plataforma de microcontrolador Arduino (hardware libre) para la recolección de datos de una serie de sensores y para el envío de comandos (actuadores), y software de código abierto (basado en Linux).

El prototipo permitirá que un usuario pueda comunicarse con ciertos dispositivos de su hogar, a través de una llamada telefónica, para controlarlos o recibir información de ellos. La comunicación entre el usuario y su hogar se concentrará en la central telefónica

propuesta. Ésta, a su vez, se conectará a los dispositivos del hogar para actuar sobre ellos (actuadores) o recibir información del entorno doméstico (mediante sensores). El control domótico telefónico será posible también desde los dispositivos (computadores o teléfonos) de la red de datos interna. El prototipo implementado estará compuesto al menos de los siguientes elementos:

- Central telefónica implementada en la plataforma Raspberry Pi.
- 2 terminales telefónicos (en *software* o *hardware,* en red interna y externa).
- *Gateway* de voz para la conexión con la PSTN.
- 1 Plataforma de sistema microprocesado (Arduino) para la interacción con sensores y actuadores.
- 1 Módulo Ethernet Shield de Arduino para la comunicación con la central telefónica.
- 5 Sensores (sensor de luz, sensor de temperatura, sensor de movimiento, sensor de gas y sensor de llama).
- 4 actuadores (Motor a pasos y 3 actuadores on/off).
- 1 *switch* o punto de acceso inalámbrico para la conexión interna de datos.

Las tareas del prototipo se implementarán de la siguiente manera:

- El control de dispositivos le permitirá al usuario encender una luz, por ejemplo, abrir una puerta (portero eléctrico), o que se genere una llamada remota que comunique al visitante con el dueño de la casa (si este último no se encuentra en su casa), a través de la central telefónica.
- El monitoreo de variables ambientales le permitirá al usuario, por ejemplo, llamar al hogar para determinar el valor exacto de una variable ambiental; o que luego de detectar una variación brusca de temperatura se dispare una llamada telefónica de alerta mediante la central.
- Dentro de los sensores para el monitoreo de variables ambientales se dispondrá de sensores de luz, temperatura, detector de movimiento, sensor de llama y sensor de gas, mientras que por parte de los actuadores se dispondrá de interruptores para el encendido y apagado de luces y motores a pasos para automatizar y controlar cortinas. La llamada de emergencia se realizará cuando se active el sensor de llama y/o el sensor de presencia.
- La simulación de presencia permitirá activar, mediante la central telefónica, un proceso automático de encendido y apagado de dispositivos, simulando mediante ello la presencia humana en un hogar, aparte se dispondrá de sensores de presencia que permitan advertir algún movimiento y notificarlo remotamente al usuario vía PSTN.

- Dado que la Raspberry Pi no cuenta con entradas y salidas analógicas, la interacción de los sensores y actuadores se lo realizará a través de la plataforma Arduino mediante el módulo Ethernet Shield de Arduino, es decir, la plataforma Arduino recolectará la información de los sensores y controlará a los actuadores. La plataforma Arduino se comunicará con la Ethernet Shield mediante comunicación SPI y el módulo Ethernet Shield se comunicará a su vez con la Raspberry pi mediante el stack de protocolos TCP/IP (sockets). Esto permitirá liberar de tareas a la Raspberry pi, realizando consultas de los estados de los sensores y actuadores al Arduino solamente cuando sea necesario. De igual manera el Arduino le informará a la Rapsberry Pi cuando salte una alarma o alguna variable ambiental tome un valor crítico, liberando nuevamente a la Raspberry Pi de trabajo. La ventaja de usar el stack de protocolos TCP/IP para comunicar la Raspberry Pi con el Arduino radica en poder utilizar la red de datos del hogar para ello, permitiendo así cubrir distancias relativamente largas y además se pudiendo contar con una gran escalabilidad pues se podrá incorporar varios Arduinos/EthernetShield y por ende más sensores y actuadores sin afectar el rendimiento del prototipo.
- Se levantará un servidor web, accesible sólo desde el interior del hogar, ya que el acceso remoto a este servidor se lo realizará normalmente vía PSTN (mediante tonos DTMF). Este servidor web podrá brindar una interfaz gráfica al usuario, quien interactuará directamente con la central telefónica. Mediante esta interfaz web también se permitirá el monitoreo de las variables ambientales sensadas y el control de los actuadores.
- La central telefónica tendrá también una función que controlará un portero eléctrico, de modo que quienes están dentro del hogar puedan abrir una puerta remotamente utilizando la central telefónica. Si la casa está vacía, el sistema telefónico llamará al dueño (a su celular, por ejemplo) para comunicarle con el visitante.
- Finalmente, se integrará el sistema con un servicio de correo electrónico local, de modo que el control domótico permita el envío de comandos mediante mensajes de correo electrónico, lo que le dará al usuario otra forma de interacción con el hogar, esto se lo realizará al igual que el servidor web de manera local, es decir, al interior del hogar, esto con el fin de demostrar su funcionamiento.

Se realizará las pruebas sobre el prototipo que permitan verificar la funcionalidad propuesta en este plan, y finalmente se presentará el costo referencial del mismo.

1.3 Marco Teórico

Domótica

El término domótica viene de la unión de la palabra *domus*, que en latín significa "casa" y de la palabra tica que viene de *automatica*, palabra en griego, cuyo significado es "funciona por sí sola" [1], [2]. El término domótica tiene varias acepciones, entre ellas la más común es "conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda", definición otorgada por la Real Academia de la Lengua [3].

Los dispositivos que forman parte del sistema domótico se comunican entre sí por medio de una red cableada o inalámbrica. Por lo general, existe un dispositivo central encargado de recopilar información de los dispositivos, analizar esta información, y enviar órdenes oportunas a los mismos.

Los sistemas domóticos pueden también recopilar información de variables físicas, como por ejemplo la temperatura ambiente del hogar. Asimismo, pueden realizar informes de los estados de estas variables, o a su vez, realizar alguna acción oportuna cuando estas variables tomen valores críticos.

La domótica posee varios niveles de complejidad, por ejemplo, el caso más simple puede ser un temporizador para encender y apagar una luz o aparato a una hora deseada. Mientras que, un sistema más complejo puede ser capaz de interactuar con cualquier elemento eléctrico del hogar dependiente de variables físicas que el sistema domótico se encuentre monitoreando. Por ejemplo, al incrementar la temperatura en el hogar, el sistema domótico puede interactuar con el sistema de aire acondicionado, para mantener una temperatura estable y así brindar confort en el hogar.

Como parte de las funciones de automatización del hogar, una central domótica debe cumplir con varios servicios que se pueden agrupar en los siguientes aspectos principales: confort, comunicaciones, seguridad y ahorro energético.

Confort

El hogar es el espacio cerrado donde las personas pasan más tiempo, es por eso que se busca hacer del hogar un espacio más cómodo y acogedor para el habitante. Las principales funciones de la domótica en el área de confort son: control de la intensidad en la iluminación del hogar, control y automatización del encendido/apagado de las luminarias, servicio de climatización y control remoto de los dispositivos en el hogar [1], [4].

Comunicaciones

Esta categoría abarca la infraestructura de comunicaciones que posee el hogar para la comunicación entre habitantes, entre habitantes y el sistema domótico y entre el sistema domótico y los dispositivos que controla [5].

Los medios de comunicación a nivel local pueden ser inalámbricos o cableados. Entre las tecnologías de comunicaciones inalámbricas que podría utilizarse están: wifi, bluetooth, zigbee e infrarrojo. Por otro lado, se puede utilizar medios guiados como: cable multipar, cable coaxial, el cableado eléctrico de la casa e inclusive fibra óptica [4].

El principal medio de comunicación remota es el Internet. Para esto, además de levantar un servidor web en el hogar, usualmente se debe realizar las configuraciones y convenios necesarios con el proveedor para que el servidor web sea alcanzable vía Internet.

Integrar porteros eléctricos al sistema domótico es otra característica en el área de comunicaciones. De modo que, se pueda capturar audio y/o video en el portero y transferirlo a un dispositivo por el cual un visitante puede comunicarse con un usuario dentro de la casa.

Seguridad

La seguridad se ha vuelto un aspecto cada vez más importante en los sistemas domóticos. La integración de cámaras, sensores y alarmas hacen del hogar un sitio más seguro y podría prevenir lamentables accidentes [4], [5]. Por ejemplo, se podría instalar sensores de movimiento y activarlos cuando los habitantes no se encuentren en el hogar, si la casa es invadida por extraños, los sensores activarían alarmas sonoras, o incluso no sonoras. Las alarmas también podrían informar, de alguna manera, al dueño del hogar de lo sucedido en la casa.

La seguridad no solo abarca robos a la propiedad, sino también prevención de accidentes. Por ejemplo, un sensor de gas puede alertar acerca de una fuga de gas doméstico en la casa, previniendo así un lamentable accidente.

Ahorro energético

Gracias a mecanismos de encendido y apagado automático de luminarias, los sistemas domóticos aportan un ahorro energético considerable. Un sistema domótico puede también desconectar equipos no prioritarios en función del consumo eléctrico y/o en función de ciertas condiciones [4], [5].

La central domótica puede ser configurada para apagar todas las luminarias a una determinada hora de la noche, previniendo así que permanezcan encendidas

innecesariamente toda la noche. Además, la central puede automatizar el movimiento de persianas para que permitan el paso de luz en el día y así lograr una mejor iluminación del hogar, esto permitiría apagar las luminarias y reemplazarlas por luz natural. Además, con sensores de movimiento sería posible detectar personas en zonas de paso y encender las luces solo cuando es necesario. Con estos ejemplos se puede apreciar la contribución de la domótica al ahorro energético.

Servicios de red

En una red de computadores, un servicio de red es una aplicación que se encuentra ejecutándose en un host, la cual está al pendiente de peticiones de otros hosts y es capaz de recibir y manipular la información recibida, realizar consultas a otros servicios, si fuera el caso, y responder adecuadamente a las peticiones recibidas [6].

Normalmente, los servicios son implementados utilizando la arquitectura cliente-servidor [7] basado en la capa aplicación del modelo OSI. La arquitectura cliente-servidor tiene dos tipos de participantes: clientes y servidores. Clientes son aquellos hosts que demandan el servicio. Mientras que, servidores son los equipos que brindan el servicio respondiendo las peticiones de los clientes.

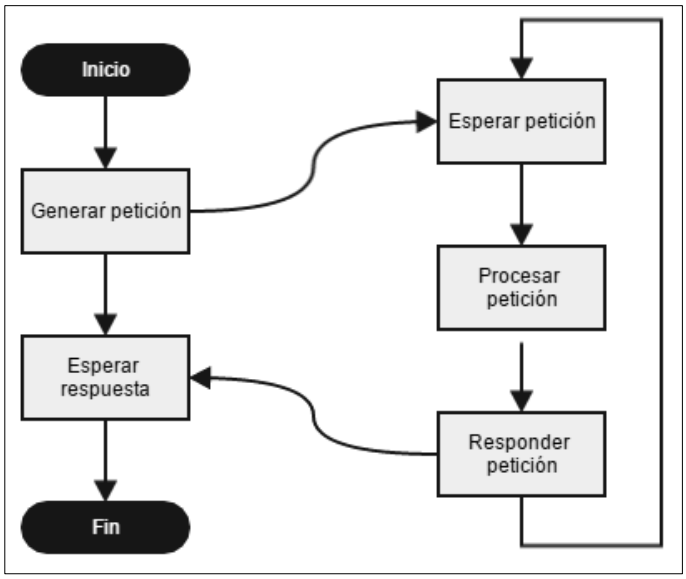

Figura 1.1. Arquitectura Cliente-Servidor

Telefonía

Es la tecnología que permite intercambiar sonidos, voz esencialmente, entre dos terminales a larga distancia [8]. Los medios de transmisión que se usa en telefonía son: par trenzado, ondas electromagnéticas (tecnología celular o satelital) y fibra óptica para comunicaciones internacionales.

Telefonía fija

El servicio de telefonía fija es uno de los servicios de telecomunicaciones más antiguo y consiste en el intercambio bilateral de voz en tiempo real entre dos terminales a través de una red de comunicaciones fija (que no ofrece mayor movilidad a sus usuarios).

PSTN (Public Switched Telephone Network)

La PSTN o red de telefonía pública conmutada, es una red de conmutación de circuitos optimizada para el intercambio bilateral de voz en tiempo real [9]. La PSTN garantiza una excelente calidad, en el intercambio de información, al dedicar un circuito a la llamada hasta que esta finaliza, independientemente de si los participantes estén o no hablando.

Telefonía IP

La telefonía IP es una tecnología que permite la trasmisión de voz mediante el protocolo TCP/IP. Es decir, permite enviar la información (voz) por medio de paquetes IP utilizando el protocolo de Internet.

Entre las ventajas que ofrece la telefonía IP se tiene:

- Ahorro en costos: Es posible utilizar una única plataforma técnica para voz y datos, lo que implica una menor inversión, mantenimiento y administración. Asimismo, se puede utilizar la misma red para voz y datos, con la posibilidad de compartir el mismo punto de red. Las extensiones pueden ser reubicadas simplemente cambiando la ubicación del terminal de un punto de red a otro. Las llamadas entre dos sedes de una empresa pueden ser gratuitas a través de Internet. Finalmente, es posible utilizar *softphones* en lugar de teléfonos (IP o analógicos).
- Movilidad: Si un usuario tiene que cambiarse de despacho, o de sucursal, llevar su extensión a su nueva posición requerirá un mínimo cambio en la configuración, y en la mayoría de los casos, lo podrá realizar el propio usuario.
- Tecnología escalable: La telefonía IP ofrece arquitecturas flexibles y escalables que facilitarán futuros crecimientos y expansiones a la empresa. Se simplifica los procedimientos de soporte y configuración. La expansión a nuevas sucursales y nuevos usuarios es sumamente fácil. Y finalmente permite una rápida instalación de nuevos servicios.

Terminales telefónicos

Un terminal telefónico es un dispositivo que permite al usuario conectarse a un sistema telefónico. Puede implementarse a nivel de hardware (teléfono IP) o de software (*softphone*).

Un teléfono IP es un dispositivo que permite al usuario comunicarse con un sistema telefónico mediante la red de datos, utilizando una interfaz Ethernet o WiFi. Y un softphone es una aplicación que implementa las funciones de un teléfono IP físico común y que puede instalarse en el sistema operativo de un computador o dispositivo móvil. Esta aplicación utiliza los recursos de estos dispositivos para permitirle al usuario la comunicación telefónica.

Gateway telefónico

Un Gateway es un dispositivo que sirve de interfaz para poder conectar una red con otra (posiblemente basada en una tecnología diferente). Por ejemplo, en el caso de la telefonía IP, el gateway es un dispositivo de red que sirve de interfaz entre la red IP y la PSTN, es decir, permite enviar las llamadas generadas en la red IP a la PSTN y a su vez permite a la red IP recibir llamadas provenientes de la PSTN. También existen gateways que interconectan con redes de telefonía móvil.

Central telefónica

Es un equipo de conmutación encargado de llevar a cabo la conexión entre terminales telefónicos para que puedan intercambiar información (voz). Una central telefónica privada o PBX (Private Branch Exchange) se encarga de realizar esta conmutación para conectar, en principio, dos o más terminales dentro de una red local. Asimismo, la PBX se encarga de conmutar y gestionar las comunicaciones entre la red interna y la PSTN.

Al tener el control sobre el ingreso o salida de llamadas, la PBX permite al administrador obtener información detallada del tráfico telefónico.

Tarjetas analógicas

Son interfaces que permiten la conexión de una PBX o un Gateway con redes de telefonía analógica. Pueden estar integradas por módulos de dos tipos:

- Foreign eXchange Station (FXS). Son módulos que generan tono de marcado, alimentación y señalización hacia un terminal telefónico analógico. Permiten conectar teléfonos analógicos y son generalmente de color verde.
- Foreign eXchange Office (FXO). Son capaces de recibir señalización, alimentación y tono de marcado. Permiten conectarse a la PSTN y generalmente son de color rojo.

Tarjetas digitales

Permiten la conexión de una PBX IP con la Red Digital de Servicios Integrados (RDSI). La RSDI es una red que facilita la conexión digital extremo a extremo. Se encuentran tarjetas

digitales del tipo BRI (Basic Rate Interface) de 192 kbps y PRI (Primary rate Interface) en velocidades de un E1 o un T1.

Un E1 se encuentra formado por 32 canales de 64 Kbps multiplexados (en total 2.048 Mbps), 30 de estos canales son de voz y los dos restantes son de control y sincronización. En cambio, un T1 se encuentra formado por 24 canales de voz de 64 kbps multiplexados, cada 192 bits (una muestra de 8 bits por 24 canales resulta 192 bits) se inserta un bit de separación, logrando así una velocidad total de 1.544 Mbps.

Señalización

Se conoce como señalización al conjunto de información intercambiada entre dispositivos de red y que permite el control y supervisión de los servicios de telecomunicaciones. En telefonía, la señalización es la responsable del establecimiento, mantenimiento y liberación de una llamada.

El Dual-Tone Multi-Frequency es un mecanismo de señalización "in-band"[1] para redes telefónicas analógicas. Su funcionamiento consiste en transmitir la combinación de dos tonos al pulsar una tecla en el terminal telefónico. Los tonos se encuentran distribuidos según la Tabla 1.1 mostrada a continuación.

Tabla 1.1. Distribución de frecuencias DTMF

	1209 Hz	**1336 Hz**	**1477 Hz**	**1633 Hz**
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	7	8	9	C
941 Hz	*	0	#	D

SIP

El protocolo SIP (Session Initiation Protocol) es un protocolo cliente-servidor especificado en el RFC 3261. La semántica y sintaxis de este protocolo se encuentran basadas en HTTP y SMTP. El protocolo SIP utiliza, en capa de transporte, tanto el protocolo TCP (Transmission Control Protocol) como UDP (User Datagram Protocol), ambos mediante el puerto 5060 por defecto.

[1] In-band es un término en inglés utilizado para indicar que la señalización se transmite junto con la información.

SIP es un protocolo de señalización cuyo funcionamiento se define en capa de aplicación y permite establecer, modificar y liberar sesiones multimedia en una red TCP/IP entre dos o más usuarios.

SIP basa su funcionamiento en el intercambio de mensajes, los mismos que pueden ser solicitudes o respuestas. Las solicitudes se listan a continuación:

- INVITE: El cliente envía este mensaje al servidor indicando que desea establecer una llamada.
- ACK: Mensaje mediante el cual el cliente confirma la recepción de una solicitud INVITE, este mensaje es enviado para establecer comunicación con el cliente que envío la solicitud INVITE.
- OPTIONS: Este mensaje es utilizado para intercambiar información de las capacidades y el estado de clientes y servidores.
- BYE: Mensaje que indica la finalización de una sesión.
- CANCEL: Mensaje que indica la cancelación de una solicitud pendiente.
- REGISTER: Este mensaje es usado para transmitir información del cliente al servidor, información como su dirección IP por ejemplo.

Cada solicitud SIP le debe corresponder una respuesta SIP, a excepción del ACK. Las respuestas deben ser codificadas para identificación de las mismas. Existen seis tipos de respuestas detalladas a continuación.

- Mensajes provisionales (1XX): Indican que existe una petición en progreso que aún no ha sido completada.
- Respuesta final de éxito (2XX): Esta respuesta indica que una solicitud ha sido completada.
- Respuesta de redirección (3XX): Contiene información nueva de la localización del cliente.
- Respuesta de fallo de solicitud (4XX): Indican la existencia de un error en procesar la solicitud de un cliente.
- Respuesta de fallo de servidor (5XX): Se utilizan cuando el servidor no ha sido capaz de procesar una solicitud.
- Respuesta de fallo global (6XX): Este mensaje indica que una solicitud no puede ser procesada por ningún servidor.

El proceso de establecimiento, mantenimiento y liberación de la llamada mediante el intercambio de mensajes SIP se muestra en la Figura 1.2.

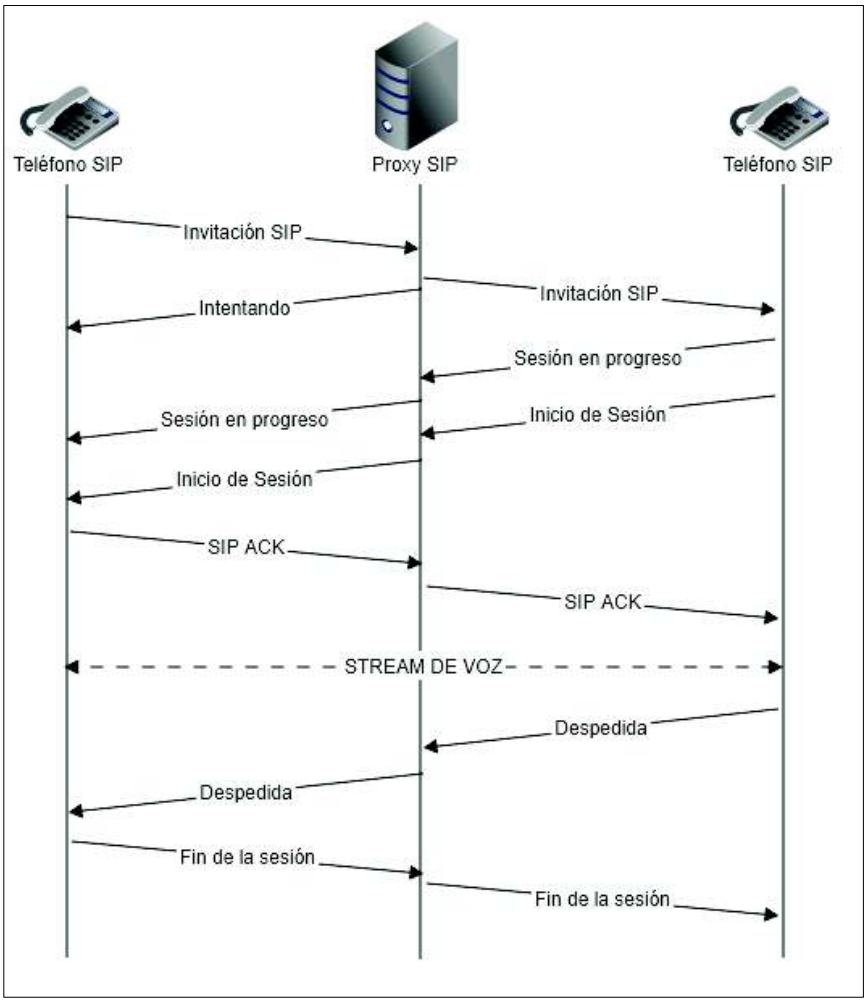

Figura 1.2. Proceso de establecimiento, mantenimiento y liberación de la llamada

Asterisk

Asterisk es una aplicación de software libre bajo la licencia GPL (General Public License) que proporciona a un computador las funcionalidades de una central telefónica. Asterisk puede ser instalado en varios sistemas operativos; sin embargo, GNU/Linux es el sistema operativo por excelencia para este software ya que cuenta con mayor documentación.

Asterisk contiene muchas funciones que anteriormente solo se las podía encontrar en costosos sistemas propietarios de PBX. Entre estas funciones de tiene: transferencias de llamadas, desvío de llamadas, conferencia múltiple, grupo de llamadas, correo de voz, operadora automática – IVR (Interactive Voice Response), música en espera, colas de llamadas, gestión de llamadas según horarios, llamadas automáticas en respuesta a llamadas perdidas, informes detallados, entre otros.

Zoiper

Zoiper es una aplicación de tipo cliente (softphone), propietaria, con muchas funcionalidades de un teléfono IP. Zoiper puede ser instalado en computadores o en dispositivos móviles.

CSipSimple

CSipSimple es un softphone de software libre bajo la licencia GPL instalable únicamente en dispositivos móviles Android. Al ser software libre, esta aplicación puede ser modificada por los usuarios o desarrolladores.

Festival

Festival es un software TTS (Text To Speech), es decir, es un programa de síntesis de voz y de propósito general que soporta múltiples lenguajes. Festival se distribuye como software libre permitiendo así su uso comercial y no comercial sin restricción.

Servicio web

El servicio web es un conjunto de protocolos y estándares con capacidad de intercambiar datos entre aplicaciones de software mediante redes de computadores, como Internet por ejemplo. Para que las aplicaciones puedan interactuar a través de la web es necesario una serie de estándares abiertos y equipos interconectados mediante redes de computadoras.

La principal razón para el uso de servicios web es el uso de los protocolos HTTP (HyperText Transfer Protocol) y TCP en el puerto 80. Es posible utilizar otros protocolos como UDP y otro puerto cualquiera. Sin embargo, TCP en el puerto 80 es el más utilizado, porque los navegadores web utilizan esta configuración por defecto.

Es el protocolo de comunicación que permite las transferencias de información en la World Wide Web. HTTP está definido en varios RFC siendo el más importante de ellos el RFC 2616, donde se define la sintaxis y la semántica de este protocolo en la versión 1.1. Además, el RFC 2616 define los elementos de software de la arquitectura web para lograr la comunicación, elementos como por ejemplo: clientes, servidores y proxies.

Por años, la herramienta más utilizada para brindar un servicio web ha sido el servidor Apache. Apache HTTP Server es un servidor web HTTP de código abierto que posee características altamente configurables, base de datos de autenticación y negociado de contenido. Apache tiene alta aceptación a nivel mundial y por lo tanto es el servidor web más utilizado des de 1996 y ha tenido un papel importante en el desarrollo de los servicios web.

Servicio de correo electrónico

El correo electrónico (e-mail en inglés) es un servicio de red que permite el enviar y recibir mensajes a través de una red de computadoras. El correo electrónico permite además anexar documentos digitales a los mensajes.

Para el correcto funcionamiento del servicio y para la interoperabilidad entre servidores de correo electrónico se ha definido tres protocolos sumamente importantes SMTP, IMAP y POP.

SMTP (Simple Mail Transfer Protocol) es un protocolo de red utilizado para el intercambio de mensajes de correo electrónico, el cual se encuentra definido en el RFC 2821. SMTP es un protocolo orientado a conexión basado en texto. El puerto más utilizado por SMTP es el TCP 25, aunque también se puede utilizar el puerto 587 o también el puerto 465 para SMTP seguro.

Para intercambiar parámetros e iniciar una sesión SMTP es necesario comandos originados por un cliente SMTP y las respuestas SMTP correspondientes del servidor. Una sesión puede incluir cero o más transacciones SMTP, una transacción consiste de tres secuencias comando/respuesta las cuales son:

- MAIL: Comando para establecer la dirección origen o dirección de retorno.
- RCPT: Comando para indicar el destino o los destinos del correo electrónico, se envía un comando por cada destino.
- DATA: Este comando contiene el contenido del mensaje, a este comando se debe responder dos veces, la primera para avisar que está listo para recibir un mensaje y la segunda después de haber recibido el mensaje, es decir para aceptar o rechazar todo el mensaje.

El protocolo SMTP es un protocolo que se encarga únicamente de la entrega de mensajes de correo, es decir, se asegura de la entrega al servidor de correo destino. Para que los usuarios puedan recuperar este correo se usa el protocolo IMAP o el protocolo POP.

POP (Post Office Protocol) es un protocolo utilizado para recuperar los mensajes de correo electrónico almacenados en un servidor remoto. Este protocolo fue diseñado para conexiones lentas o intermitentes, el protocolo se encarga de descargar el correo, permitiéndole al usuario revisarlo después, incluso sin conexión a Internet. La versión actual de este protocolo es la versión 3 (POP3) y se encuentra definido en el RFC 1939.

Por otro lado, IMAP (Internet Message Access Protocol) es un protocolo de capa aplicación que permite el acceso a mensajes almacenados en un servidor de correo. El puerto en el

que funciona por defecto es el TCP 143, aunque también puede utilizar los puertos TCP 220 (IMAP3) ó 993 (IMAP seguro). La versión actual de IMAP es la versión 4 revisión 1 y está definida en el RFC 3501.

IMAP tiene varias ventajas en comparación con POP, como por ejemplo: IMAP trabaja en modo permanente, por lo que el aviso de un correo nuevo es inmediato; la descarga del correo en IMAP se produce solamente cuando el usuario desea leer el mensaje; gestiona carpetas, plantillas y borradores en el servidor, permite la búsqueda de mensajes por palabras claves, etc.

Sockets

Un socket es un método para la comunicación entre un programa del cliente y un programa servidor, un socket está formado por un par de direcciones IP (local y remota), un protocolo de transporte (TCP o UDP) y un par de números de puerto (local y remoto). La dirección IP sirve para identificar a la máquina remota en el cual está corriendo el programa con el que se desea comunicar. El protocolo de transporte se encarga de llevar la información a esa máquina y, finalmente, el número de puerto identifica al programa en específico en la máquina remota.

Para que dos programas puedan comunicarse deben primero ser capaces de encontrarse el uno al otro por medio de la red y segundo ambos programas deben ser capaces de intercambiar información e interpretarla.

Los sockets permiten implementar la arquitectura cliente-servidor, la comunicación debe ser iniciada por un programa cliente realizando una petición con algún comando especial. Mientras que, el programa servidor procesa la petición y responde a la misma.

Raspberry PI

Es una computadora de placa reducida de bajo coste desarrollado por la fundación Raspberry Pi. Ordenador de placa reducida significa (en inglés Single Board Computer - SBC) que todo el computador y sus partes están implementados en un solo circuito o placa.

Raspberry Pi es un producto con propiedad registrada pero de uso libre. Por otro lado, el software es código abierto, siendo su sistema operativo por excelencia una distribución especial de Debian para Raspberry denominado Raspbian. Sin embargo, también es posible instalar otros sistemas operativos, como Windows 10 por ejemplo.

Figura 1.3. Raspberry PI 2 modelo B [10]

Las especificaciones técnicas de la Raspberry Pi 2 modelo B son las siguientes:

Tabla 1.2. Especificaciones de la Raspberry PI 2 modelo B [10]

Característica	Descripción
Sistema en chip (SoC)	Broadcom BCM2836
CPU	ARM11 ARMv7 ARM Cortex-A7 4 núcleos @ 900 MHz
Overclocking	arm_freq=1000, sdram_freq=500, core_freq=500
GPU	Broadcom VideoCore IV 250 MHz. OpenGL ES 2.0
RAM	1 GB LPDDR2 SDRAM 450 MHz
USB	2 puertos USB 2.0
Salida de video	HDMI 1.4 @ 1920x1200 píxeles
Almacenamiento	MicroSD
Ethernet	1 puerto de 10/100 Mbps
Tamaño	85,60 x 56,5 mm
Peso	45 gramos
Consumo	5v, 900mA, depende de la carga de trabajo de los 4 cores

Arduino

Arduino es una compañía desarrolladora de hardware libre, esta compañía se encarga de diseñar y manufacturar placas de desarrollo de hardware con su propio entorno de desarrollo. Ambos, hardware y software, son liberados bajo licencia de código abierto.

Arduino UNO es una placa de desarrollo de la compañía de hardware libre Arduino, es un proyecto basado en hardware libre y software libre, de precio asequible y fácil de utilizar.

La placa de desarrollo Arduino está basada en el microcontrolador ATmega328p y posee las siguientes características:

Tabla 1.3. Especificaciones de la placa de desarrollo Arduino UNO [11]

Característica	Descripción
Pines	14 pines Entrada/Salida digitales
Entradas analógicas	6 entradas analógicas
Cristal	Un cristal de cuarzo de 16 MHz
Comunicación con PC	Conexión USB
Voltaje de operación	5 V
Voltaje de entrada	Recomandado 7-12 V
Voltaje de salida	Límite 6-20 V
Memoria Flash	32 KB
SRAM	2 KB
EEPROM	1 KB
Tamaño	68.6 x 53.4 mm
Peso	25 g

Figura 1.4. Arduino UNO [11]

El módulo Arduino Ethernet Shield [12] permite a las placas de Arduino conectarse a la red de datos, mediante una interfaz RJ45, utilizando el stack de protocolos TCP/IP. Se monta fácilmente sobre Arduino Uno y se comunica con él por comunicación SPI (Serial Peripherial Interface). Sus principales características son:

Tabla 1.4. Especificaciones del módulo Arduino Ethernet Shield [12]

Característica	Descripción
Voltaje de operación	5V, alimentación desde la placa Arduino
Controlador Ethernet	Wiznet W5100 con memoria de 16KB
Conector	RJ45

Velocidad de transmisión	10-100 Mbps
Comunicación	SPI con placa Arduino
Memoria	MicroSD

Figura 1.5. Arduino Ethernet Shield [12]

Sensores

Son dispositivos que poseen la capacidad de captar y transformar variables físicas o químicas, llamadas variables de instrumentación, en variables eléctricas. Las variables de instrumentación pueden ser, por ejemplo: temperatura, luminosidad, presión, fuerza, humedad, etc. Por otro lado, una magnitud eléctrica puede ser un voltaje, una corriente eléctrica, una resistencia eléctrica, una capacitancia, etc. Por ejemplo, la magnitud física puede ser la temperatura y la propiedad alterada puede ser la resistencia eléctrica del sensor que varía proporcionalmente en relación con la variable medida.

A diferencia de un transductor, un sensor siempre está en contacto con la variable de instrumentación, convirtiéndola a un tipo de variable entendible por el sistema [13].

Los sensores tienen las siguientes características:

- Rango de medida: Dominio de la magnitud a medirse.
- Precisión: Error máximo esperado en la medición.
- Offset: Valor de la variable de salida, cuando la variable de entrada es nula.
- Linealidad: Es el grado de aproximación de la curva de calibración[2] a una recta.
- Sensibilidad: La sensibilidad es la pendiente de la curva de calibración. Un sistema será lineal si la sensibilidad es constante.
- Resolución: Mínima variación a la entrada que un sensor puede detectar.

[2] Curva de calibración: Es la relación entre la entrada del sensor y su correspondiente salida.

- Precisión: Grado de concordancia entre los resultados, a mayor precisión, menor es la dispersión de los valores medidos alrededor del valor medido.
- Exactitud: Capacidad del instrumento de producir errores pequeños.

A partir de estas características, se puede comparar los sensores y seleccionar el que más se adapte a los requerimientos y al presupuesto.

Actuadores

Los actuadores son elementos mecánicos que generan una acción o efecto sobre un proceso automatizado, los actuadores reciben las órdenes de un regulador o controlador. Entre los actuadores más comunes se tienen: motores, interruptores, válvulas, cilindros hidráulicos o neumáticos, resistencias calefactoras, ventiladores, etc.

2. METODOLOGÍA

El presente trabajo de estudio técnico se basa en investigación de tipo aplicado, a que su objeto de estudio se concentrará en el desarrollo de un prototipo en base a tecnologías existentes. Tanto software como hardware utilizados en el presente trabajo serán seleccionadas de manera que el costo de implementación sea mínimo, sin descuidar de ninguna manera los objetivos planteados en el capítulo anterior.

Para la realización de este proyecto técnico se utilizará información publicada en la documentación oficial del software y hardware. De esta forma, se asegurará que la información utilizada en el diseño e implementación del prototipo sea verídica y actualizada.

El desarrollo del proyecto se lo realizará en tres etapas principales: diseño del prototipo, implementación del prototipo y pruebas de funcionamiento. En la etapa de diseño se recopilarán los requerimientos del prototipo, y en base a estos, se describirá la estructura general del prototipo. Por otro lado, en la etapa de implementación se describirá la implementación de las funcionalidades del prototipo de acuerdo a lo expuesto en la etapa de diseño. Finalmente, se realizará pruebas de funcionamiento de cada uno de los elementos principales involucrados en el desarrollo del prototipo.

Se presentará, en el siguiente capítulo, el presupuesto referencial de la implementación del prototipo y se discutirá los resultados del presente trabajo de estudio técnico.

2.1 Fase de diseño

Requerimientos

El prototipo a desarrollar está enfocado en ofrecer funciones básicas que la domótica define en sus diferentes áreas, en especial, para las áreas de confort y seguridad.

Para el desarrollo del prototipo se ha considerado una casa básica conformada por dos dormitorios, un baño y una sala, suficiente para exponer la capacidad del prototipo. Razón por la cual, el prototipo contará con los siguientes dispositivos que el habitante será capaz de controlar:

- 4 luminarias (focos de 110V, uno en cada dormitorio, uno el baño y otro en la sala).
- 2 cortinas (motores a pasos, uno en cada dormitorio).
- Comunicación con el portero eléctrico.

El habitante será capaz de encender y apagar las luminarias del hogar, abrir y cerrar las cortinas, y comunicarse con el portero eléctrico, tanto dentro como fuera del hogar. Para

hacer posible la manipulación de estos componentes, desde dentro del hogar, se plantea implementar una interfaz gráfica intuitiva.

Se propone que esta interfaz gráfica sea una interfaz web, debido a que en la mayoría de dispositivos modernos, por ejemplo: smartphones, laptops, PCs o tablets, tienen incorporado por defecto un navegador web independientemente de cual sea su fabricante. Se propone, además, implementar una aplicación Android para smathphones y tablets que soporten este sistema operativo. Tanto la interfaz web como la aplicación Android serán destinadas a la manipulación de los dispositivos desde el interior del hogar.

La manipulación de los componentes del hogar (iluminarias, cortinas y comunicación con el portero eléctrico), desde cualquier sitio remoto, se lo realizará mediante una llamada al teléfono convencional del hogar. Esta llamada será recibida por la central telefónica mediante un audio de bienvenida. Después de esto, el habitante deberá ingresar una extensión especial, la misma que será la encargada de interactuar con la central domótica y ésta a su vez con los componentes a manipular. Adicionalmente, se deberá ingresar una contraseña de 10 dígitos para asegurar que solamente el habitante tenga acceso a la central domótica. Finalmente, la interacción entre el habitante y los componentes del hogar se lo realizará mediante audios, por parte de la central telefónica (menú interactivo - IVR), y por señalización DTFM por parte del habitante.

Además, la central domótica podrá monitorear, mediante sensores, las siguientes variables físicas:

- Nivel de luminosidad.
- Temperatura.
- Gas doméstico.
- Llama.

Adicionalmente, se incorporará un sensor de presencia (sensor infrarrojo) con el fin de monitorear si la casa es invadida cuando el habitante no se encuentre en el hogar.

Los sensores medirán constantemente las variables físicas de interés y se reportarán estos valores a la central domótica periódicamente para que ésta tome decisiones oportunas y genere alertas, si es necesario, cuando ciertos eventos ocurran. Contribuyendo de esta manera, con el área de seguridad. Los eventos y sus respectivas alertas se detallan en la siguiente tabla:

Tabla 2.1. Eventos y sus respectivas alertas

Evento	Alerta
El sensor llama detecta la presencia de fuego.	El prototipo genera una llamada al teléfono móvil del habitante, esté o no en el hogar, para reproducir un audio alertándole que se ha detectado la presencia de fuego en el hogar.
El sensor de gas detecta un valor alto de gas doméstico en el hogar.	El prototipo genera una llamada al teléfono móvil del habitante, esté o no en el hogar, para reproducir un audio alertándole que se ha detectado una fuga de gas doméstico en el hogar.
El sensor de presencia detecta intrusos en la casa.	El prototipo genera una llamada al teléfono móvil del habitante cuando éste no se encuentre en el hogar para reproducir un audio alertándole que se ha detectado la presencia de un intruso en la casa.

Se propone, además, implementar un sistema de simulación de presencia, el mismo que encenderá y apagará las luminarias del hogar en una secuencia definida para dar la sensación a personas ajenas al hogar de que hay personas dentro, cuando en realidad no hay nadie en el interior del hogar.

En la interfaz web, anteriormente mencionada, se deberá mostrar el estado de los componentes del hogar, y deberá contar, además, con botones para cambiar el estado de los mismos. Por otro lado, la interfaz web debe mostrar la información actual de luminosidad y temperatura en el hogar. Por último, se propone que la interfaz muestre dos botones adicionales para activar/desactivar las alertas del sensor de presencia y las alertas del simulador de presencia.

Por último, se propone integrar al sistema domótico un portero eléctrico que interactúe con la central telefónica para comunicar a un visitante con el dueño de casa, aunque este último no se encuentre en el domicilio.

Además de los mecanismos de interacción ya nombrados, se incorporará un mecanismo de interacción con el prototipo mediante mensajes de correo electrónico para el envío de

comandos al sistema domótico. Esto se lo realizará únicamente de manera local, como demostración de la funcionalidad.

Visión general del prototipo

Se ha considerado dividir las tareas del prototipo en varias secciones, denominadas módulos, que interactuarán entre sí por medio de un módulo principal, que se lo llamará módulo central. El módulo central será el encargado de tomar decisiones en base a la información recibida de los otros módulos que estarán conformados tanto por hardware como de software.

De esta manera, se logra un desarrollo modular del prototipo, facilitando así su implementación y brindando la posibilidad de añadir más funciones al proyecto desarrollando más módulos que intercambien información con el módulo central. Por otro lado, si algún módulo sufre algún imprevisto, no afectaría a los demás módulos.

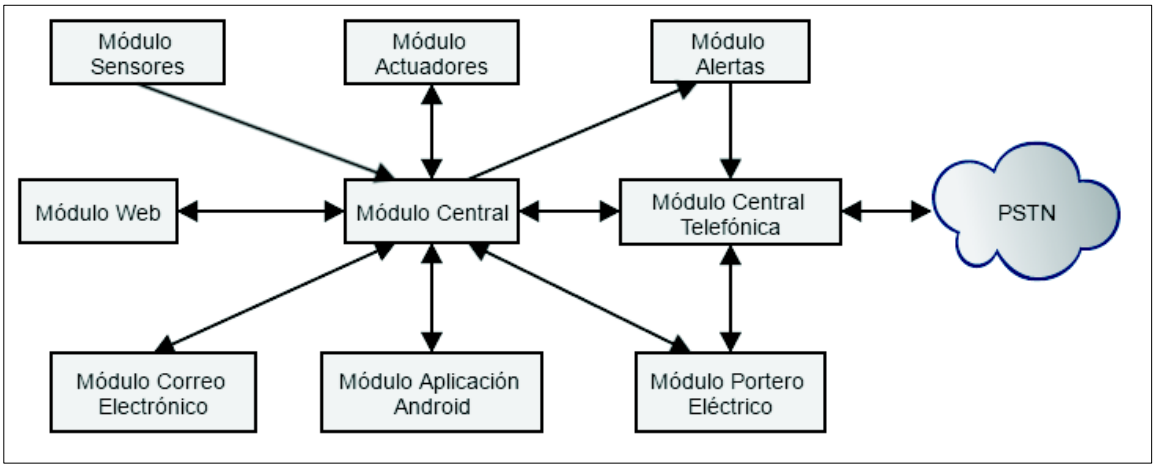

Figura 2.1. Diagrama de bloques del prototipo

En la Figura 2.1, se expone el diagrama de bloques del prototipo, en donde se puede apreciar cómo los diferentes módulos del prototipo interactúan entre sí. Se puede observar además, que todos los módulos interactúan con el módulo central.

Así por ejemplo, si se desea cambiar el estado de un actuador mediante la interfaz web, el módulo web se encargará de tomar la instrucción de la interfaz web, y no lo envía directamente al módulo actuadores, sino que lo envía al módulo central para que lo procese y sea el modulo central quien decida qué instrucción se envía al módulo actuadores según el estado de los actuadores reportado por el módulo actuadores.

Se propone utilizar una Raspberry PI para alojar los servicios necesarios de los que hará uso el prototipo, y así, asegurar un costo de implementación bajo. Los siguientes servicios podrán ser instalados y configurados en la Raspberry PI:

- Servidor de telefonía IP.
- Servidor web.
- Servidor de correo electrónico.
- Servidor sockets.

Debido a que la Raspberry PI no posee pines de conversión analógico digital, se propone el uso de placas de desarrollo Arduino. Los mismos que, con un Ethernet Shield, podrán hacer uso del stack de protocolos TCP/IP, y así, comunicarse con la Raspberry PI. La placa Arduino será encargada de manejar los sensores y los actuadores.

Adicionalmente, se utilizará un punto de acceso inalámbrico para que los habitantes del hogar puedan acceder a los servicios a los servicios del prototipo de manera local desde cualquier dispositivo con conexión WiFi.

Será necesario incluir un Gateway telefónico para que los usuarios puedan interactuarán remotamente con el prototipo usando tonos DTMF al hacer una llamada mediante la PSTN.

La topología de red del prototipo se muestra en la Figura 2.2. Es necesario un punto de acceso que cuente con al menos cuatro puertos Ethernet, los cuales estarán ocupados por la Raspberry PI, el Gateway telefónico y las dos placas Arduino.

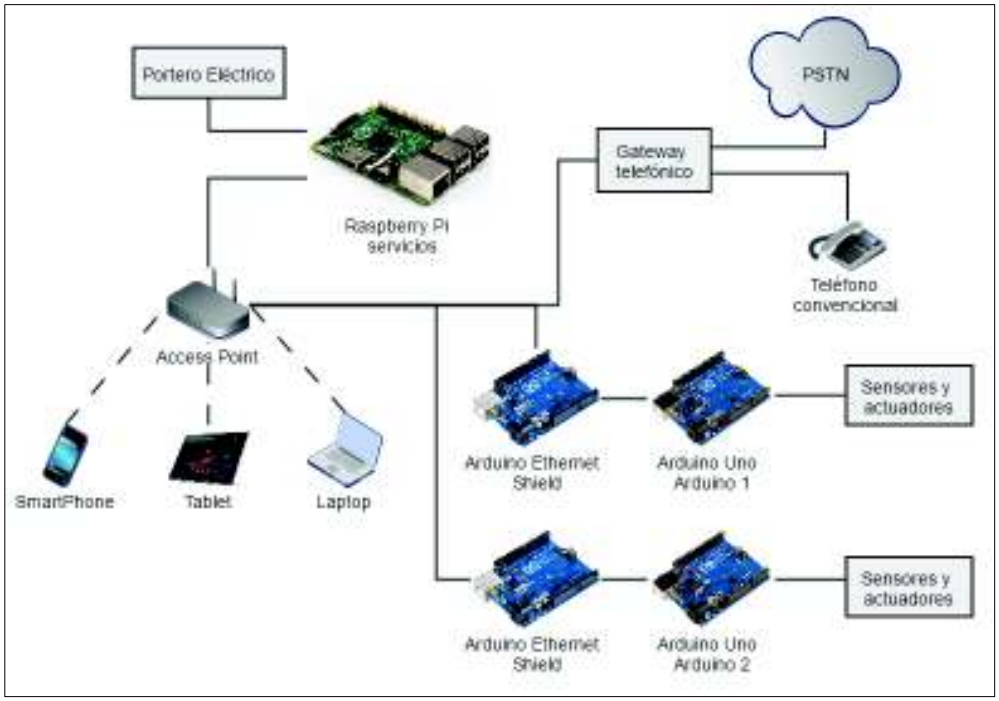

Figura 2.2. Topología del prototipo

Todos los dispositivos deben estar en la misma red. Por lo tanto, se define los siguientes parámetros de red:

Tabla 2.2. Parámetros de red

Parámetro	Valor
Dirección de red	192.168.100.0
Máscara de red	255.255.255.0
Default Gateway	192.168.100.1

Los dispositivos y su direccionamiento se resumen en la Tabla 2.3. Los dispositivos que no se encuentren en esta tabla, como por ejemplo celulares o tablets, tendrán direcciones adquiridas desde el servidor DHCP que brinda el Access Point.

Tabla 2.3. Direccionamiento IP de los dispositivos

Dispositivo	Dirección IP	Máscara de red	Default Gateway
Raspberry PI	192.168.100.200	255.255.255.0	192.168.100.1
Gateway telefónico	192.168.100.205	255.255.255.0	192.168.100.1
Arduino 1	192.168.100.201	255.255.255.0	192.168.100.1
Arduino 2	192.168.100.202	255.255.255.0	192.168.100.1

Módulo Central

El módulo central será el módulo principal, es decir, el cerebro del prototipo y será el encargado de interactuar con los demás módulos. Por lo tanto, será capaz de recibir información, procesarla y enviar comandos al módulo pertinente para que realice la acción respectiva.

Este módulo se desarrollará de tal forma que intercambie información implementando sockets. El módulo central actuará tanto como cliente, como servidor, logrando así, una comunicación bilateral con la mayoría de módulos, excepto con el módulo sensores y el módulo alertas, para los cuales se propone una comunicación unilateral, como se muestra en la Figura 2.1.

El módulo central dará instrucciones a los demás módulos mediante los comandos, los mismos que se detallan en la Tabla 2.4.

Tabla 2.4. Lista de comandos con los que el módulo central dará instrucciones a los demás módulos

Comando	Interpretación
F1-ON	Solicita al módulo actuadores que encienda la iluminaria 1
F1-OFF	Solicita al módulo actuadores que apague la iluminaria 1
F2-ON	Solicita al módulo actuadores que encienda la iluminaria 2
F2-OFF	Solicita al módulo actuadores que apague la iluminaria 2
F3-ON	Solicita al módulo actuadores que encienda la iluminaria 3
F3-OFF	Solicita al módulo actuadores que apague la iluminaria 3
F4-ON	Solicita al módulo actuadores que encienda la iluminaria 4
F4-OFF	Solicita al módulo actuadores que apague la iluminaria 4
V1-ON	Solicita al módulo actuadores que abra la cortina 1
V1-OFF	Solicita al módulo actuadores que cierre la cortina 1
V2-ON	Solicita al módulo actuadores que abra la cortina 2
V2-OFF	Solicita al módulo actuadores que cierre la cortina 2
F1?	Solicita al módulo actuadores información acerca del estado de la iluminaria 1
F2?	Solicita al módulo actuadores información acerca del estado de la iluminaria 2
F3?	Solicita al módulo actuadores información acerca del estado de la iluminaria 3
F4?	Solicita al módulo actuadores información acerca del estado de la iluminaria 4
LUZ?	Solicita al módulo actuadores información acerca del estado del sensor de luminosidad
TEMP?	Solicita al módulo actuadores información acerca del estado del sensor de temperatura
TIMBRE-ON	Solicita al módulo actuadores que se administre corriente eléctrica para activar el timbre
TIMBRE-OFF	Solicita al módulo actuadores que se deje de suministrar corriente eléctrica para desactivar el timbre
SPON	Se informa que el simulador de presencia se encuentra activado
SPOF	Se informa que el simulador de presencia se encuentra desactivado
A1ON	Se informa que el sensor de movimiento se encuentra activado
A1OF	Se informa que el sensor de movimiento se encuentra desactivado

SPOF	Se informa que el simulador de presencia se encuentra desactivado
S1XXX	Se informa el valor de la luminosidad, este que puede estar entre 000 hasta 100
S2XXX	Se informa el valor de la temperatura, este puede estar entre 000 hasta 999

Adicionalmente, el módulo central recibirá peticiones de los otros módulos, a las cuales el módulo central será capaz de responder. Se define, entonces, los comandos con los que el módulo central responderá a las peticiones en la Tabla 2.5.

Tabla 2.5. Comandos que el módulo central puede recibir

Comando	Interpretación
CONSULTAF	Otro módulo se encuentra consultando el estado de los actuadores. El módulo central responderá con el estado de las luminarias y las cortinas.
CONSULTAS	Otro módulo se encuentra consultando el estado de los sensores. La central domótica responderá con el estado del sensor de luminosidad, el estado del sensor de temperatura, si está activado o no la alarma del sensor de movimiento y si está activado o no el simulador de presencia.
FOCO1-ON	Otro módulo realiza la petición a la central domótica que encienda la luminaria 1
FOCO1-OFF	Otro módulo realiza la petición a la central domótica que apague la luminaria 1
FOCO2-ON	Otro módulo realiza la petición a la central domótica que encienda la luminaria 2
FOCO2-OFF	Otro módulo realiza la petición a la central domótica que apague la luminaria 2
FOCO3-ON	Otro módulo realiza la petición a la central domótica que encienda la luminaria 3
FOCO3-OFF	Otro módulo realiza la petición a la central domótica que apague la luminaria 3
FOCO4-ON	Otro módulo realiza la petición a la central domótica que encienda la luminaria 4

FOCO4-OFF	Otro módulo realiza la petición a la central domótica que apague la luminaria 4
F1-ON	La placa Arduino a cargo de la luminaria 1 informa que se ha encendido manualmente la luminaria.
F1-OFF	La placa Arduino a cargo de la luminaria 1 informa que se ha apagado manualmente la luminaria
F2-ON	La placa Arduino a cargo de la luminaria 2 informa que se ha encendido manualmente la luminaria.
F2-OFF	La placa Arduino a cargo de la luminaria 2 informa que se ha apagado manualmente la luminaria.
F3-ON	La placa Arduino a cargo de la luminaria 3 informa que se ha encendido manualmente la luminaria.
F3-OFF	La placa Arduino a cargo de la luminaria 3 informa que se ha apagado manualmente la luminaria.
F4-ON	La placa Arduino a cargo de la luminaria 4 informa que se ha encendido manualmente la luminaria.
F4-OFF	La placa Arduino a cargo de la luminaria 4 informa que se ha apagado manualmente la luminaria.
MOTOR1-ON	Otro módulo realiza la petición a la central domótica que abra la cortina 1
MOTOR1-OFF	Otro módulo realiza la petición a la central domótica que cierre la cortina 1
MOTOR2-ON	Otro módulo realiza la petición a la central domótica que abra la cortina 2
MOTOR2-OFF	Otro módulo realiza la petición a la central domótica que cierre la cortina 2
MOV1-ON	Otro módulo realiza la petición a la central domótica para activar la alarma del sensor de movimiento
MOV1-OFF	Otro módulo realiza la petición a la central domótica para desactivar la alarma del sensor de movimiento
SIMULACION-ON	Otro módulo realiza la petición a la central domótica para activar el simulador de presencia
SIMULACION-OFF	Otro módulo realiza la petición a la central domótica para desactivar el sensor de movimiento

ALERTA-MOV1	La placa Arduino a cargo del sensor de movimiento reporta que se ha activado el sensor
ALERTA-GAS	La placa Arduino a cargo del sensor de gas reporta que se ha activado el sensor
ALERTA-FUEGO	La placa Arduino a cargo del sensor de fuego reporta que se ha activado el sensor
TIMBRE	La placa Arduino a cargo del timbre reporta que se ha presionado el botón del timbre

Módulo Central Telefónica

Este módulo se encargará de recibir y realizar llamadas telefónicas tanto dentro como fuera del domicilio. Al interior del hogar, el módulo central telefónica utilizará la telefonía IP para aprovechar la red de datos del hogar y la facilidad que brinda la telefonía IP para instalar aplicativos clientes de telefonía IP en dispositivos modernos. Por otro lado, este módulo hará uso de un Gateway telefónico para conectarse a la PSTN, permitiendo así, la comunicación desde y hacia fuera del hogar.

Por último, el módulo central telefónica se encargará de hacer llegar las notificaciones que el módulo alertas genere, mediante una llamada telefónica al teléfono móvil del habitante para reproducir, apenas conteste, un audio pregrabado informando acerca del evento emergente.

Módulo Sensores

Este módulo se encargará de los sensores y reportará al módulo central los valores de temperatura y luminosidad cuando el módulo central lo requiera. Por otro lado, si se detecta una fuga de gas, la presencia de fuego o de algún intruso, el módulo sensores reportará el evento al módulo central para que ejecute la acción correspondiente.

Tabla 2.6. Comandos que puede recibir el módulo sensores

Comando	Interpretación
F1?	El módulo central quiere consultar el estado de la luminaria 1
F2?	El módulo central quiere consultar el estado de la luminaria 2
F3?	El módulo central quiere consultar el estado de la luminaria 3
F4?	El módulo central quiere consultar el estado de la luminaria 4
LUZ?	El módulo central quiere consultar el valor que está midiendo el sensor de luz

TEMP?	El módulo central quiere consultar el valor que está midiendo el sensor de temperatura

El módulo sensores responderá a los comandos que recibirá por parte del módulo central, los mismos que están detallados en la Tabla 2.6. De manera similar, el módulo sensores será capaz de reportar el valor de sus sensores mediante los comandos indicados en la Tabla 2.7.

Tabla 2.7. Comandos que usa el módulo sensores para reportar el estado de los sensores al módulo central

Comando	Interpretación
S1XXX	Se informa el valor de la luminosidad que puede ser desde 000 hasta 100
S2XXX	Se informa el valor de la temperatura que puede ser desde 000 hasta 999
ALARMA-MOV1	Se informa que se ha detectado un intruso
ALARMA-GAS	Se informa que se ha detectado una fuga de gas
ALARMA-FUEGO	Se informa que se ha detectado fuego

Los sensores estarán conectados a los pines de la placa Arduino, el mismo que se comunicará mediante la red de datos del hogar a la Raspberry PI, en donde estará alojado el software del módulo central.

Módulo Actuadores

El módulo actuadores se encarga de la interacción entre los actuadores y el módulo central, es decir, el módulo actuadores recibirá comandos del módulo central y según cual sea el comando realizará una acción en específico. Adicionalmente, el módulo deberá reportar el estado de los actuadores cuando el módulo central se lo consulte. Los posibles comandos que el módulo actuadores recibirá se muestran en la tabla a continuación:

Tabla 2.8. Comandos que puede recibir el módulo actuadores

Comando	Interpretación
F1-ON	El módulo central solicita encender la luminaria 1
F1-OFF	El módulo central solicita apagar la luminaria 1
F2-ON	El módulo central solicita encender la luminaria 2

F2-OFF	El módulo central solicita apagar la luminaria 2
F3-ON	El módulo central solicita encender la luminaria 3
F3-OFF	El módulo central solicita apagar la luminaria 3
F4-ON	El módulo central solicita encender la luminaria 4
F4-OFF	El módulo central solicita apagar la luminaria 4
V1-ON	El módulo central solicita abrir la cortina 1
V1-OFF	El módulo central solicita cerrar la cortina 1
V2-ON	El módulo central solicita abrir la cortina 2
V2-OFF	El módulo central solicita cerrar la cortina 2
TIMBRE-ON	El módulo central solicita administrar corriente eléctrica al portero eléctrico
TIMBRE-OFF	El módulo central solicita quitar la corriente eléctrica al portero eléctrico

La mayoría de porteros eléctricos se abren al aplicar corriente eléctrica a su cerradura por 2 o 3 segundos. En el prototipo propuesto, la cerradura será simbolizada por un diodo que se encenderá durante 2 segundos, simulando así, la apertura del portero eléctrico.

El módulo actuadores reportará el estado de los actuadores al módulo central mediante el sistema de comandos de la Tabla 2.9 mostrada a continuación.

Tabla 2.9. Comandos que usa el módulo actuadores para reportar el estado de los actuadores al módulo central

Comando	Interpretación
F1ON	Se informa que la luminaria 1 se encuentra encendida
F1OF	Se informa que la luminaria 1 se encuentra apagada
F2ON	Se informa que la luminaria 2 se encuentra encendida
F2OF	Se informa que la luminaria 2 se encuentra apagada
F3ON	Se informa que la luminaria 3 se encuentra encendida
F3OF	Se informa que la luminaria 3 se encuentra apagada
F4ON	Se informa que la luminaria 4 se encuentra encendida
F4OF	Se informa que la luminaria 4 se encuentra apagada
M1ON	Se informa que la cortina 1 se encuentra abierta
M1OF	Se informa que la cortina 1 se encuentra cerrada
M2ON	Se informa que la cortina 2 se encuentra abierta

M2OF	Se informa que la cortina 2 se encuentra cerrada

Módulo Alertas

Este módulo será capaz de generar alertas de acuerdo a la información recibida por el módulo central. Estas alertas dependerán de los eventos generados por los sensores según la Tabla 2.10. Además, el módulo alertas coordinará con la central telefónica para hacer llegar las notificaciones al habitante del hogar, reproduciendo audios pregrabados, mediante llamadas telefónicas.

Tabla 2.10. Lista de los posibles eventos y su respectiva alerta en el módulo alertas

Evento	Alerta
Se detecta la presencia de un intruso	Se genera una llamada al teléfono móvil del habitante y se reproduce un audio pregrabado indicando que se ha detectado la presencia de un intruso en el domicilio.
Se detecta la presencia de fuego	Se genera una llamada al teléfono móvil del habitante y se reproduce un audio pregrabado indicando que se ha detectado fuego dentro en el domicilio.
Se detecta la presencia de gas doméstico	Se genera una llamada al teléfono móvil del habitante y se reproduce un audio pregrabado indicando que se ha detectado una fuga de gas en el domicilio.

Módulo Web

El servicio web ofrecerá al habitante una interfaz gráfica amigable, sencilla e intuitiva donde se mostrará los valores de temperatura y luminosidad del hogar. Además, permitirá controlar los diferentes actuadores que el sistema domótico ofrece. Finalmente, el servicio web debe tener un alcance únicamente local.

Módulo Correo electrónico

Este módulo permitirá al usuario enviar un mensaje de correo electrónico a la central domótica, es decir, a una dirección especial de correo electrónico. En el asunto del correo electrónico, el habitante del hogar deberá ingresar uno de los comandos de la Tabla 2.11. El mismo que, el módulo central será capaz de interpreta para realizar la acción pertinente. Este servicio se implementará también de manera local.

Tabla 2.11. Lista de comandos que admite el módulo correo electrónico

Asunto	Interpretación
FOCO 1 ENCENDER	Se solicita encender la luminaria 1
FOCO 1 APAGAR	Se solicita apagar la luminaria 1
FOCO 2 ENCENDER	Se solicita encender la luminaria 2
FOCO 2 APAGAR	Se solicita apagar la luminaria 2
FOCO 3 ENCENDER	Se solicita encender la luminaria 3
FOCO 3 APAGAR	Se solicita apagar la luminaria 3
FOCO 4 ENCENDER	Se solicita encender la luminaria 4
FOCO 4 APAGAR	Se solicita apagar la luminaria 4
VENTANA 1 ABRIR	Se solicita abrir la cortina 1
VENTANA 1 CERRAR	Se solicita cerrar la cortina 1
VENTANA 2 ABRIR	Se solicita abrir la cortina 2
VENTANA 2 CERRAR	Se solicita cerrar la cortina 2
SENSOR DE MOVIMIENTO ENCENDER	Se solicita activar el mecanismo para la detección de intrusos
SENSOR DE MOVIMIENTO ENCENDER	Se solicita desactivar el mecanismo para la detección de intrusos
SIMULADOR DE PRESENCIA ENCENDER	Se solicita activar el simulador de presencia
SIMULADOR DE PRESENCIA ENCENDER	Se solicita desactivar el simulador de presencia

Adicionalmente, el módulo correo electrónico responderá, a la petición realizada por el habitante del hogar confirmando, mediante correo electrónico, si se logró o no realizar la solicitud recibida. Finalmente, se añadirá un informe detallado del estado de los sensores y actuadores al correo electrónico de respuesta.

Módulo Aplicación Android

Como alternativa a la interfaz web, se desarrollará una aplicación para Android donde se muestren los valores de temperatura e iluminación, Así como también, opciones para interactuar con los actuadores, como botones por ejemplo.

Módulo Portero eléctrico

Este módulo será el encargado de manipular el portero eléctrico, recibir órdenes del módulo central y notificar al módulo central si hay algún visitante fuera del domicilio.

Este módulo tendrá un mecanismo para que el visitante pueda hacer sonar el timbre del hogar, simbolizado en el prototipo por un led, hasta tres veces consecutivas. En la cuarta ocasión se generará automáticamente una llamada desde el prototipo, mediante el módulo central telefónica, al teléfono móvil del habitante del hogar, para inmediatamente comunicarlo con el visitante que se encuentra fuera del domicilio.

Se propone desarrollar una aplicación Android que permita realizar lo anteriormente planteado. Es decir, la aplicación tendrá un botón que, al presionarlo, enviará un comando al módulo principal para activar el timbre. Finalmente, el dispositivo donde se tendrá instalada la aplicación del portero deberá tener instalado un cliente de telefonía IP para que pueda comunicarse con el módulo central telefónica.

Raspberry PI

Se propone el uso del ordenador de placa reducida Raspberry PI [10] para la implementación del prototipo. Puesto que, al tener un sistema operativo basado en GNU/Linux y por lo tanto libre, puede soportar la instalación de programas basados, también, en software libre. Además, la Raspberry PI posee una interfaz de red, lo cual es necesario para conectarse a la red de datos del hogar. Finalmente, su uso asegura un costo mínimo en la implementación del prototipo.

La Raspberry PI soporta varios sistemas operativos diseñados para este ordenador de placa reducida. De entre todos ellos, se ha seleccionado Raspbian, por ser el sistema operativo más estable desarrollado para Raspberry PI. Raspbian es una versión Linux basada Debian Whezzy (Debian 7.0) [14].

Otro motivo por el cual se ha seleccionado la Raspberry PI es su bajo consumo de energía, haciéndolo ideal para la implementación en un hogar. Finalmente, al ser de placa reducida, sus dimensiones son considerablemente pequeñas al compararlo con un ordenador convencional y, por lo tanto, no ocupa mucho espacio.

Los siguientes módulos serán implementados en la Raspberry:

- Módulo central
- Módulo central telefónica
- Módulo web
- Módulo correo electrónico
- Módulo alertas

Arduino

Debido a que el ordenador de placa reducida Raspberry PI no posee pines de conversión analógico digital, se plantea el uso de placas de desarrollo Arduino [11] para realizar la lectura de los sensores y reportar esta información cuando el módulo central lo requiera. Así también, será el encargado de interactuar directamente con los actuadores cuando reciba órdenes del módulo central.

La placa de desarrollo Arduino será capaz de intercambiar información con el módulo central haciendo uso del stack de protocolos TCP/IP, gracias a su módulo Arduino Ethernet Shield [12]. Por lo tanto, se debe implementar sockets en las placas de desarrollo para que actúe tanto como cliente, como servidor.

Vale la pena destacar la asequibilidad de la placa de desarrollo Arduino y de su módulo Ethernet Shield. Así, con el uso de la placa Arduino se contribuye a un bajo costo de implementación del prototipo.

Por último, a la placa Arduino, en conjunto con su módulo Ethernet Shield, se le puede asignar una dirección IP, brindando de esta manera escalabilidad al proyecto. Puesto que, el número de placas Arduino que se pueden incorporar al prototipo dependerían del número de hosts libres en la red local.

De la variedad de modelos existentes de Arduino, se ha seleccionado la placa de desarrollo Arduino UNO. Por lo tanto, se propone utilizar dos de estas placas Arduino en la implementación del prototipo. Es decir, tanto el módulo sensores como el módulo actuadores serán implementados en las placas Arduino, con la distribución de pines detallada de la Tabla 2.12 para el primer Arduino, y en la Tabla 2.13 para el segundo Arduino.

Tabla 2.12. Distribución de pines, Arduino 1

No. de pin	Función del pin	Tipo de pin
A0	Sensar luminosidad	Entrada analógica
A1	Sensar temperatura	Entrada analógica
2	Sensar movimiento	Entrada digital
4	Motor a pasos, Dirección	Salida digital
5	Motor a pasos, Step	Salida digital
6	Final de carrera Izquierda	Entrada digital
7	Final de carrera derecha	Entrada digital
8	Foco 1 encender/apagar	Salida digital

9	Foco 1 verificar estado	Entrada digital
10	Reservado para Ethernet Shield	
11	Reservado para Ethernet Shield	
12	Reservado para Ethernet Shield	
13	Reservado para Ethernet Shield	
18	Foco 2 encender/apagar	Salida digital
19	Foco 2 verificar estado	Entrada digital
0	Timbre	Salida digital

Tabla 2.13. Distribución de pines, Arduino 2

No. de pin	Función del pin	Tipo de pin
A0	Sensor de gas	Entrada analógica
A1	Sensor de llama	Entrada analógica
4	Motor a pasos, Dirección	Salida digital
5	Motor a pasos, Step	Salida digital
6	Final de carrera Izquierda	Entrada digital
7	Final de carrera derecha	Entrada digital
8	Foco 1 encender/apagar	Salida digital
9	Foco 1 verificar estado	Entrada digital
10	Reservado para Ethernet Shield	
11	Reservado para Ethernet Shield	
12	Reservado para Ethernet Shield	
13	Reservado para Ethernet Shield	
18	Foco 2 encender/apagar	Salida digital
19	Foco 2 verificar estado	Entrada digital

Finalmente, se propone utilizar el puerto TCP 5000 en las dos placas Arduino para recibir peticiones y consultas desde el módulo central.

Sensor de luminosidad

El sensor de luminosidad será capaz de medir la cantidad de luz en el hogar. Este valor se representará en la escala del 0% al 100%, debido a que es una variable física que depende de varios factores, entre ellos: la geometría del hogar, la posición del sol, el número y disposición de las ventanas, etc.

Se utilizará un LDR (Ligth Dependent Resistor), es decir, una resistencia que varía en función de la cantidad de luz que reciba en su superficie. El circuito que se utilizará se presenta en la Figura 2.3.

En el diseño propuesto se tiene un potenciómetro en serie al LDR para fines de calibración. El LDR, al variar su resistencia con la cantidad de luz incidente, provocará un cambió de voltaje a la salida del circuito, el mismo que, se conectará a una entrada analógica de la placa Arduino, para posteriormente convertirlo a un formato digital.

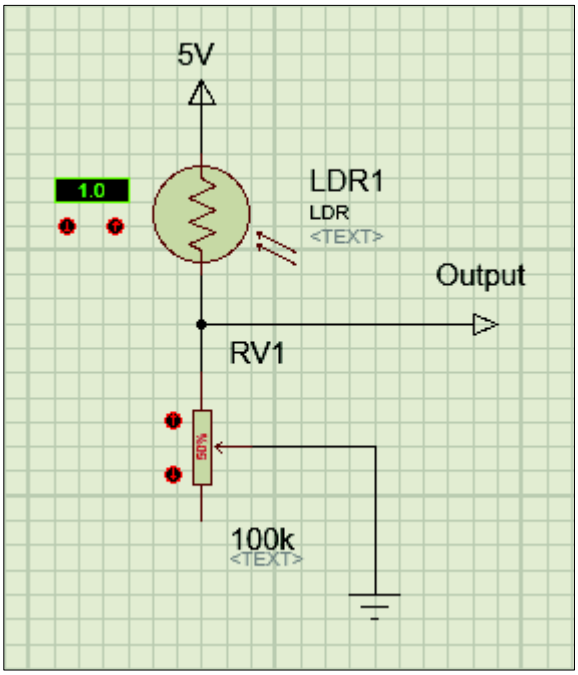

Figura 2.3. Circuito para el sensor de luminosidad

Sensor de temperatura

En el mercado existe una gran variedad de sensores de temperatura, y debido a que se requiere asegurar el menor costo posible para el prototipo, se ha decidido trabajar con el sensor LM35 [15].

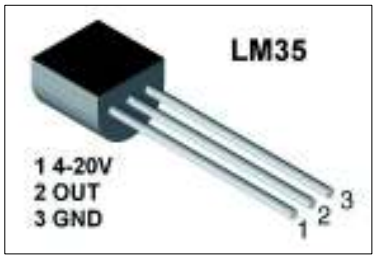

Figura 2.4. Sensor LM35 [16]

El sensor LM35 cuenta con tres pines, como se muestra en la Figura 2.4. El pin de alimentación puede soportar un voltaje máximo de 30 Voltios. El sensor entrega un voltaje variable proporcional a la temperatura, ver Figura 2.5. La salida de este pin se conecta a una entrada analógica en la placa de desarrollo Arduino.

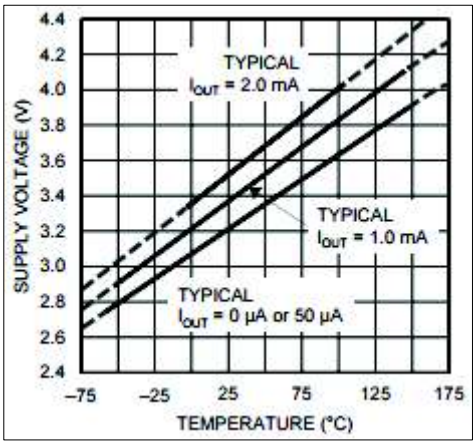

Figura 2.5. Curva Voltaje de salida vs. Temperatura del sensor LM35 [15]

Sensor de movimiento

El sensor de movimiento que se plantea utilizar en el prototipo es el sensor PIR[3] HC-SR501 [17] , puesto que no se requiere de mucha inversión económica en comparación a otros sensores de movimiento más complejos que se pueden encontrar en el mercado.

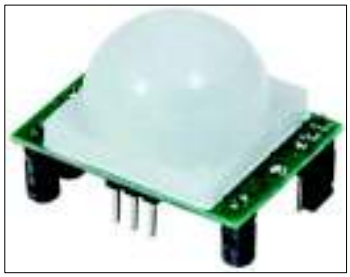

Figura 2.6. Sensor PIR HC-SR5011 [18]

El sensor HC-SR501, como se muestra en la Figura 2.6, posee únicamente tres pines, por lo que es muy sencillo de utilizar. El primer pin es utilizado alimentar el sensor. El voltaje de alimentación debe estar en el rango de 5 y 20 Voltios. El segundo pin es la salida del sensor, el sensor pondrá este pin en 5 Voltios cuando detecte movimiento y esperará un

[3] PIR viene del Inglés Passive Infrared, se le denomina pasivo puesto que el dispositivo no emite radiación alguna para la detección de movimiento.

intervalo de tiempo, que puede estar entre 30 segundos y 5 minutos, para regresar a su estado normal (0 Voltios). Por lo tanto, este pin se conectará a la entrada digital de la placa Arduino. El tercer y último pin se conecta a tierra.

Por último, en la parte inferior, el sensor cuenta con dos tornillos para calibrar el tiempo entre mediciones y la sensibilidad de la medición, como se muestra en la Figura 2.7.

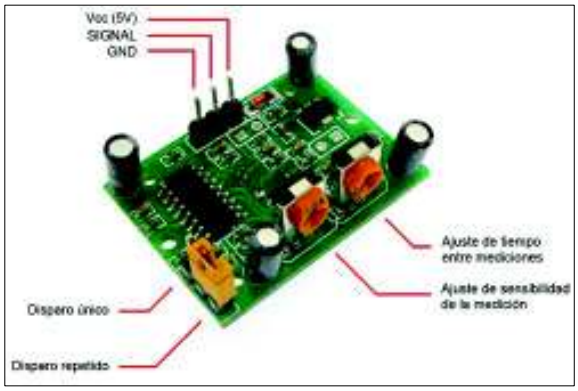

Figura 2.7. Sensor PIR HC-SR5011, vista inferior [19]

Sensor de gas

Para detectar la presencia de gas doméstico, se plantea el uso del sensor MQ-2 [20] de la Figura 2.8. Este sensor es adecuado para la detección de gas LP, i-butano, propano, metano, alcohol, hidrógeno. Y puesto que, el gas doméstico se encuentra conformado por metano en su mayoría, es decir entre el 80 y el 95%, el sensor MQ-2 se lo puede incorporar al prototipo. Por otro lado, este sensor tiene una alta sensibilidad que puede ser calibrada mediante un potenciómetro, y además, tiene un tiempo de respuesta rápido. Este pequeño sensor de gas puede detectar la presencia de gas combustible en concentraciones de 300 a 10.000 ppm.

Figura 2.8. Sensor de gas MQ-2 [21]

El sensor MQ-2 se presenta sobre una pequeña placa con cuatro pines, como se muestra en la Figura 2.9. El pin Vcc sirve para la alimentación del sensor y puede recibir entre 5 Voltios y 20 Voltios. El pin GND va conectado a tierra y el pin A0 es la salida de voltaje analógico, el mismo que, irá conectado a una entrada analógica de la placa Arduino, para finalmente ser convertido a formato digital.

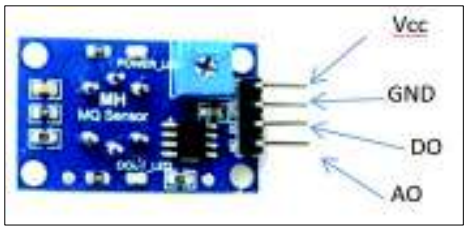

Figura 2.9. Sensor de gas MQ-2, vista inferior [22]

Sensor de llama

En el presente prototipo se planteó notificar alarmas cuando se detecte la existencia de fuego. El sensor de llama de la Figura 2.10 utiliza un led que responde a longitudes de onda entre los 760 y los 1100 nm (infrarrojo), longitud de onda que emite por lo general el fuego, por lo que se plantea el uso de este sensor en la implementación del prototipo.

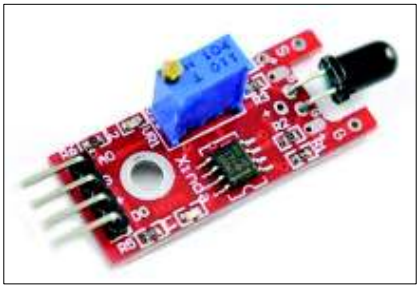

Figura 2.10. Sensor de llama [23]

Este sensor cuenta con cuatro pines de los cuales se utilizarán solo tres. Al pin + se conecta la alimentación del sensor. El pin G se conecta a tierra. Y el pin A0 es el pin de voltaje de salida que entrega el sensor de fuego, este pin debe ser conectado a una entrada analógica en la placa Arduino.

Luminarias

El presente prototipo debe ser capaz de encender o apagar las luminarias de forma manual y automática. Para que esto sea posible, es necesario utilizar un par de conmutadores, dispuestos como se muestra en la Figura 2.11.

El conmutador 1, de la Figura 2.11, brindará la posibilidad de encender/apagar manualmente la luminaria. Mientras que, el conmutador 2 será controlado por el prototipo mediante la placa Arduino. De esta manera, se podrá encender/apagar la luminaria manualmente o haciendo uso del prototipo.

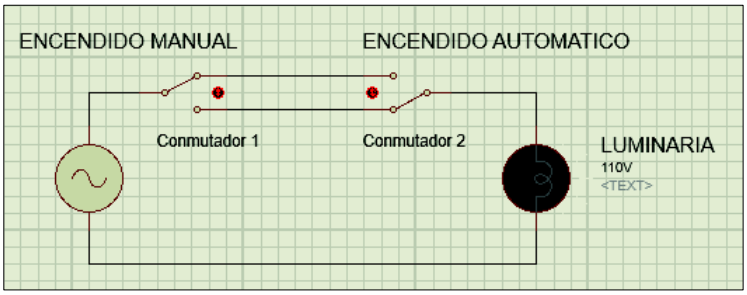

Figura 2.11. Circuito con conmutadores para las luminarias

El conmutador 1, para la manipulación manual de la luminaria, será un conmutador cualquiera que se encuentre en el mercado que soporte los 110 Voltios. Mientras que, para el conmutador 2 se propone utilizar un relé. A su vez, el prototipo debe monitorear el estado de las luminarias para presentarlo a los habitantes del hogar, por lo que es necesario un circuito que permita al prototipo conocer si en la luminaria se está aplicando, o no, el voltaje de alimentación (110 V).

Entre tantos circuitos integrados, se plantea el uso del circuito H11AA1 [24]. El mismo que, como se muestra en la Figura 2.12, cuenta con dos diodos rectificadores en antiparalelo, con lo que se obtiene en el pin 4 un cero lógico cuando el voltaje entre los pines 1 y 2 se encuentre entre -0.7 y +0.7 Voltios. Es decir, el circuito H11AA1 se comporta como un circuito “detector de cruce por cero”. La forma de onda en la salida (pin 4) del circuito, con respecto a su entrada (pin 1), se muestra en la Figura 2.13, en donde la entrada del circuito es una onda senoidal (110 V).

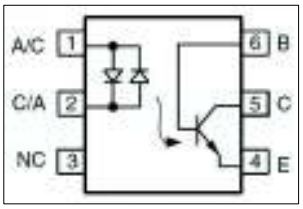

Figura 2.12. Circuito Integrado H11AA1 [25]

Según el datasheet del circuito H11AA1, la corriente de entrada máxima que soporta el integrado es de ±60 mA, por lo que será necesario incorporar una resistencia de potencia de 10 KOhms en el pin 1 del circuito H11AA1. Así la corriente en la entrada del H11AA1

variará entre -16 y +16 mA, valor que está por debajo del máximo soportado. Por último, para que no se caliente el circuito la resistencia debe ser capaz de disipar 10W o más.

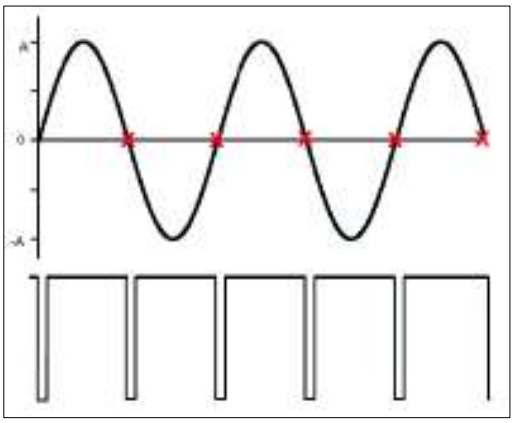

Figura 2.13. Salida del circuito H11AA1

Finalmente, para eliminar los flancos de bajada, cuando se produce un cruce por cero, se incorpora un capacitor de 2.2 uF en paralelo a la resistencia de 10 KOhms a la salida del circuito H11AA1, como se muestra en la Figura 2.14. De esta manera, se asegura que el circuito entregue 5 Voltios cuando exista 110 Voltios a la entrada del circuito y 0 Voltios cuando no exista voltaje en la entrada del circuito.

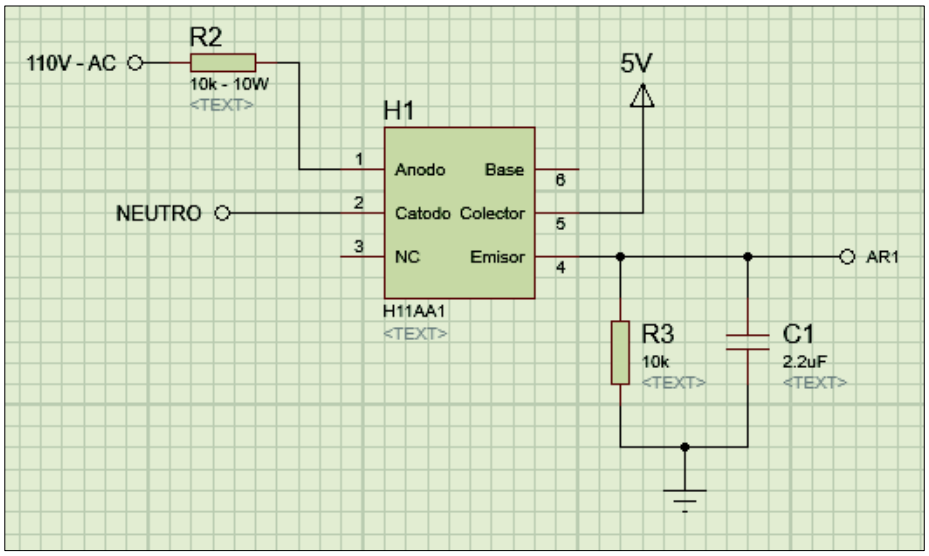

Figura 2.14. Circuito para monitorear el estado de las luminarias

El circuito propuesto se lo debe conectar en paralelo a la luminaria, como se muestra en la Figura 2.15. De esta manera, se asegura que el circuito entregue 5 Voltios cuando la luminaria esté encendida y 0 Voltios cuando esté apagada. En otras palabras, con los 5 ó 0 Voltios que entrega el circuito propuesto, conectados a una entrada digital de la placa Arduino, el prototipo será capaz de detectar si la luminaria está encendida o apagada.

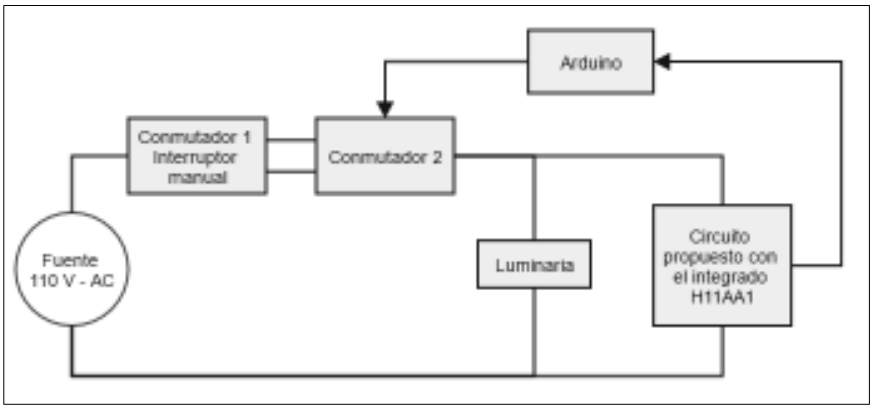

Figura 2.15. Esquema de conexión para las luminarias

Cortinas

Para abrir/cerrar las cortinas se plantea el uso de un motor a pasos. El motor a pasos permitirá abrir o cerrar la cortina girando su eje en sentido horario o antihorario, según sea el caso. Además, será necesario incorporar un driver que se conectará al motor. El driver deberá permitir alimentar al motor con una fuente independiente. Y a la vez, facilitar el control del motor. Finalmente se propone el uso de finales de carrera para detener al motor cuando la cortina llegue a estar completamente abierta o cerrada.

El driver para motor a pasos que se ha seleccionado es el A4988[4] [26]. Este driver puede manejar una fuente de alimentación para los motores de hasta 35 Voltios, con una corriente de salida de ±2 Amperios. El driver deberá conectarse como se muestra en la Figura 2.16.

Gracias al driver A4988, la placa Arduino se encargará únicamente los pines DIR y STEP. El pin DIR le indica al driver la dirección en la que debe girar el motor. Mientras que, con un flanco de subida en el pin STEP, se le indica al driver que el motor debe girar un paso en la dirección indicada en el pin DIR.

[4] Para más información del driver A4988 revise el datasheet alojado en la siguiente página web: https://www.pololu.com/file/download/a4988_DMOS_microstepping_driver_with_translator.pdf?file_id=0J450

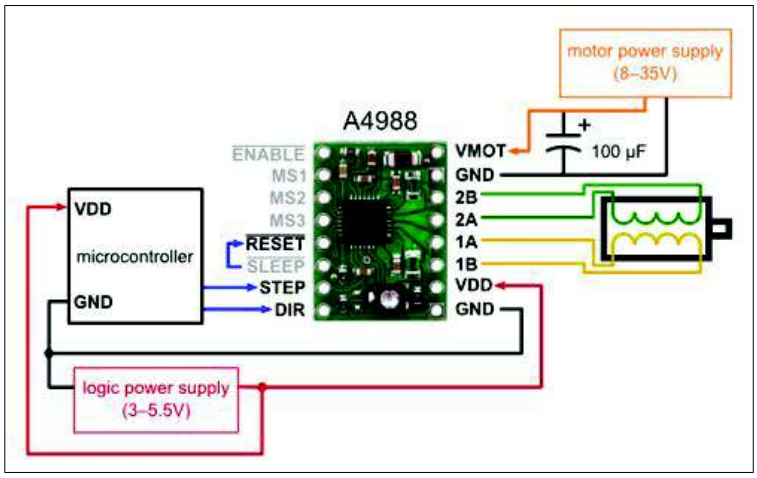

Figura 2.16. Esquema de conexión del driver A4988 [26]

Por último, para los finales de carrera, se propone el uso de pulsadores, los mismos que entregarán 5V cuando sean pulsados, es decir se tendrá 5V cuando la cortina llegue a estar abierta o cerrada completamente. Por lo tanto, será necesario incorporar dos finales de carrera en cada cortina. Si se conecta los finales de carrera a una entrada digital de la placa Arduino, el prototipo será capaz de determinar el estado de las cortinas.

Lenguaje de programación Ruby

Para el desarrollo del presente prototipo se propone el uso, de entre varios lenguajes de programación, de Ruby [27] debido a las siguientes razones:

- Es un lenguaje muy fácil de escribir, leer y entender el código.
- Se considera un lenguaje multiparadigma (procedimental, funcional u orientado a objetos).
- Soporta threads (hilos) de ejecución gestionados por su intérprete.
- Por último, el dominio del desarrollador, del presente proyecto, sobre este lenguaje de programación.

Ruby es un lenguaje de programación creado por el japonés Yukihiro “Matz” Matsumoto en 1995 [28]. Combina la sintaxis de Python y Perl con características de programación orientada a objetos y otras funcionalidades de lenguajes como Lua, Dylan, CLU, etc. Ruby es un lenguaje de programación interpretado y su implementación oficial es distribuida bajo una licencia de software libre.

Se dice que Ruby es completamente orientado a objetos, es decir, todos los tipos de datos utilizados en Ruby son objetos, incluidos los tipos de datos que en otros lenguajes se consideran primitivas.

Entre las características de Ruby se destacan las siguientes:

- Orientado a objetos.
- Cuatro niveles de ámbito de variables: global, clase, instancia y local.
- Manejo de excepciones.
- Expresiones regulares nativas.
- Posibilidad de redefinir operadores.
- Altamente portable.
- Hilos de ejecución simultáneos en todas las plataformas.

Además de las características mostradas anteriormente, Ruby es un lenguaje de programación bastante amigable y limpio.

Software módulo Central

Dado que este módulo está conformado por software únicamente, el módulo central será implementado en la Raspberry PI y desarrollado en lenguaje Ruby. Como se ha mencionado anteriormente, el módulo central necesita comportarse tanto como cliente como servidor. Por lo tanto, es necesario implementar Threads (hilos) en Ruby para que los procesos se ejecuten en paralelo.

El primer proceso, el proceso servidor, será el encargado de recibir peticiones de los demás módulos, y según el comando recibido el módulo central realizará la acción correspondiente. Por otro lado, el segundo proceso, el proceso cliente, será el encargado de monitorear el estado de los sensores y actuadores cada dos segundos, es decir, será el encargado de intercambiar información con los módulos sensores y actuadores periódicamente.

Implementar la comunicación de los módulos mediante sockets y códigos es muy útil, ya que, es muy fácil modificar y añadir más módulos al prototipo. De esta manera, será muy sencillo desarrollar e incorporar al prototipo una aplicación web, aplicación móvil, aplicación de escritorio, cualquier otro servicio que interactúe con el módulo central, o aumentar placas Arduino para añadir más actuadores y/o sensores.

Software módulo Central Telefónica

El módulo central telefónica se encargará, entre otras cosas, de las comunicaciones telefónicas. Para esto, se hace necesario el uso de la herramienta de software libre Asterisk [29]. Por el momento, Se ha elegido la versión 13.5 de Asterisk.

Una vez instalado correctamente Asterisk, se crearán las cuentas SIP para los habitantes del hogar. Para el prototipo se crearán cuatro cuentas SIP, las mismas que se muestran en la Tabla 2.14.

Tabla 2.14. Cuentas SIP para usuarios del prototipo

Dispositivo	Usuario	Contraseña	Servidor SIP
PC	0000XXXX0000	tesis12345	192.168.100.200
Laptop	0000XXXX0001	tesis12345	192.168.100.200
Celular	0000XXXX0002	tesis12345	192.168.100.200
Portero	0000XXXX0003	tesis12345	192.168.100.200

Es necesario crear una cuenta independiente para el portero eléctrico porque el módulo central telefónica hará uso de la misma para lograr comunicar el visitante con el habitante del hogar cuando el mismo no se encuentre en el domicilio. Además, se deberá realizar las configuraciones necesarias para la correcta interacción con el Gateway telefónico.

En el plan de marcado, es necesario crear las extensiones que se indican en la Tabla 2.15, entre estas se incluye la extensión especial para la central domótica.

Tabla 2.15. Extensiones para el plan de marcado

Extensión	Usuario / Aplicación
500	IVR (Interactive Voice Response), Extensión de la contestadora automática. Todas las llamadas que la central reciba desde la PSTN serán redirigidas a esta extensión.
1000	Teléfono fijo
1001	0000XXXX0001 (Laptop)
1002	0000XXXX0002 (Celular)
1003	0000XXXX0003 (PC)
2000	Portero eléctrico
9685	Central domótica

Para la extensión de central domótica (9685), será necesario incorporar algún mecanismo de autenticación. Por ello, quien desee interactuar con la central domótica deberá ingresar una clave. La clave de ingreso al menú de la central domótica será 2525363614. No obstante, por motivos de seguridad, esta clave no podrá guardarse en texto plano.

Para cifrar la clave, se utilizará el algoritmo MD5 (Message-Digest Algorithm 5) [30]. Se eligió este algoritmo ya que es un algoritmo de reducción criptográfica de 128 bits ampliamente utilizado.

La central domótica será capaz de recibir la contraseña de autenticación, aplicar el algoritmo MD5 y comparar el hash creado con el hash guardado para verificar si la contraseña ingresada es correcta.

La central domótica deberá tener un menú interactivo en donde el usuario pueda consultar el estado de los sensores y actuadores, y además, poder controlar los actuadores. Para esto, se ha diseñado un menú interactivo a base de audios según lo indicado en los diagramas de flujo de las Figura 2.17 a Figura 2.29.

Será necesario instalar un sistema de síntesis de voz para reproducir sonidos hablados a partir de un texto. Para el presente prototipo se ha decidido utilizar Festival TTS [31], el cual, es un sistema TTS (Text-to-Speech) de propósito general para múltiples lenguajes que es distribuido como software libre.

Se desarrollará un programa que consulte el estado de los actuadores a la central domótica y retorne un texto en función del estado de los actuadores. Asterisk debe ser capaz de tomar este texto y convertirlo en audio mediante Festival TTS.

Finalmente, el módulo central telefónica debe ser capaz de realizar llamadas cuando la central domótica detecte si se ha activado alguna alarma. Asterisk tiene la capacidad de generar llamadas automáticamente cuando se crean archivos con extensión .call en la carpeta de llamadas de salida /var/spool/asterisk/outgoing/.

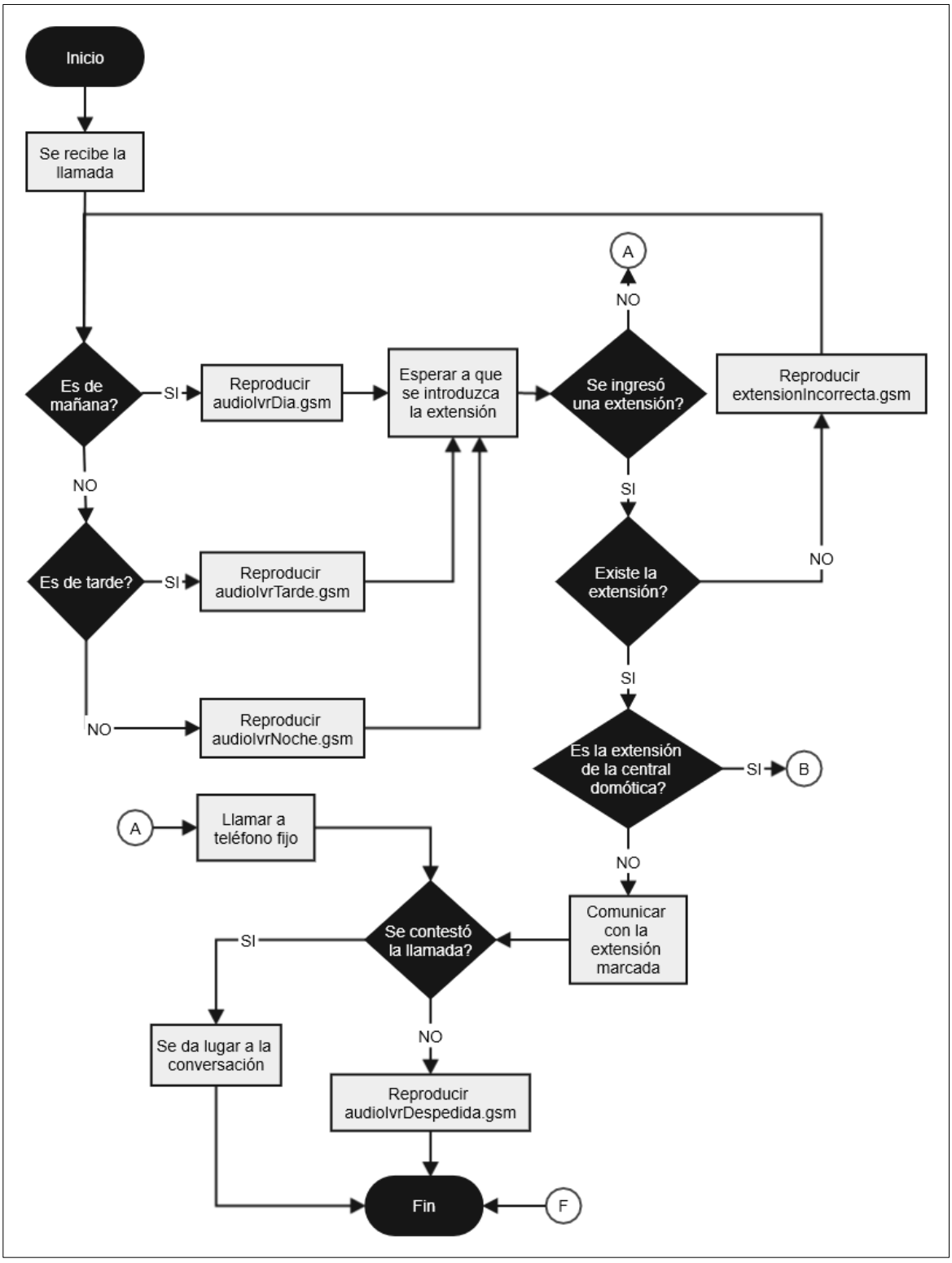

Figura 2.17. Diagrama de flujo, menú interactivo del módulo central telefónica, menú principal

En la Figura 2.17 se presenta el menú interactivo (IVR), el mismo que se aplicará a cualquier llamada recibida de la PSTN. Al inicio, se reproducirá uno de los tres audios pregrabados: audiolvrDia.gsm si es de día (entre las 06:00 y las 11:59), audiolvrTarde.gsm

si es de tarde (desde las 12:00 hasta las 17:59) y audioIvrNoche si es de noche o madrugada (entre las 18:00 y las 05:59). En cualquiera de estos casos, el audio solicitará a quien llame que ingrese la extensión con la que desea comunicarse. Si la extensión no es ingresada después de un tiempo (5 segundos), la llamada será dirigida al teléfono convencional. Mientras que, si es ingresada alguna extensión, primero se verificará si la extensión existe para transferir la llamada a la extensión marcada, sino, se reproducirá el audio extensionIncorrecta.gsm indicando que la extensión marcada es incorrecta y de inmediato se reproduce audioIvrDia.gsm, audioIvrTarde.gsm o audioIvrNoche.gsm, según sea el caso.

Si la extensión marcada existe y no es contestada en 30 segundos, se reproducirá el archivo audioIvrDespedida.gsm y finalmente se dará por terminada la comunicación.

Por último, si la extensión ingresada es la extensión de la central domótica, se referirá como central domótica a la parte del menú interactivo de Asterisk con las opciones que brinda el módulo central telefónica para interactuar con el prototipo y sus componentes (sensores y actuadores), se pasará la llamada al bloque B, es decir, al menú de la central domótica, que se detalla en la Figura 2.18.

Cuando la extensión de la central domótica, extensión 9685, es marcada, se reproducirá el audio domoticaIvr.gsm, en el cual, se da la bienvenida a la central domótica y se solicitará ingresar la contraseña de autenticación de 10 dígitos.

Si la contraseña no es ingresada correctamente se reproducirá el archivo claveIncorrecta.gsm, en el que se indica que la contraseña ingresa ha sido errónea y se reproducirá de nuevo el audio domoticaIvr.gsm. Se permitirá un máximo de cuatro intentos para ingresar la contraseña, después de los cuales se reproducirá el archivo domoticaDespedida.gsm indicando que se ha superado el número de intentos máximos permitidos y se finalizará la comunicación.

Por otro lado, si la contraseña ingresada es correcta, se ingresará al bloque D del diagrama de flujo de la Figura 2.18 y se reproducirá el archivo opcionesDomotica.gsm, en este audio se presenta al usuario tres opciones:

- Opción 1: Consultar el estado del domicilio.
- Opción 2: Cambiar el estado de los actuadores.
- Opción 3: Opciones del sistema de seguridad.

Cada opción tendrá un submenú como se indica en el diagrama de flujo de la Figura 2.19. Por otro lado, si se ingresa alguna opción inválida la llamada será dirigida al bloque D, es decir, se reproducirá de nuevo el audio opcionesDomótica.gsm.

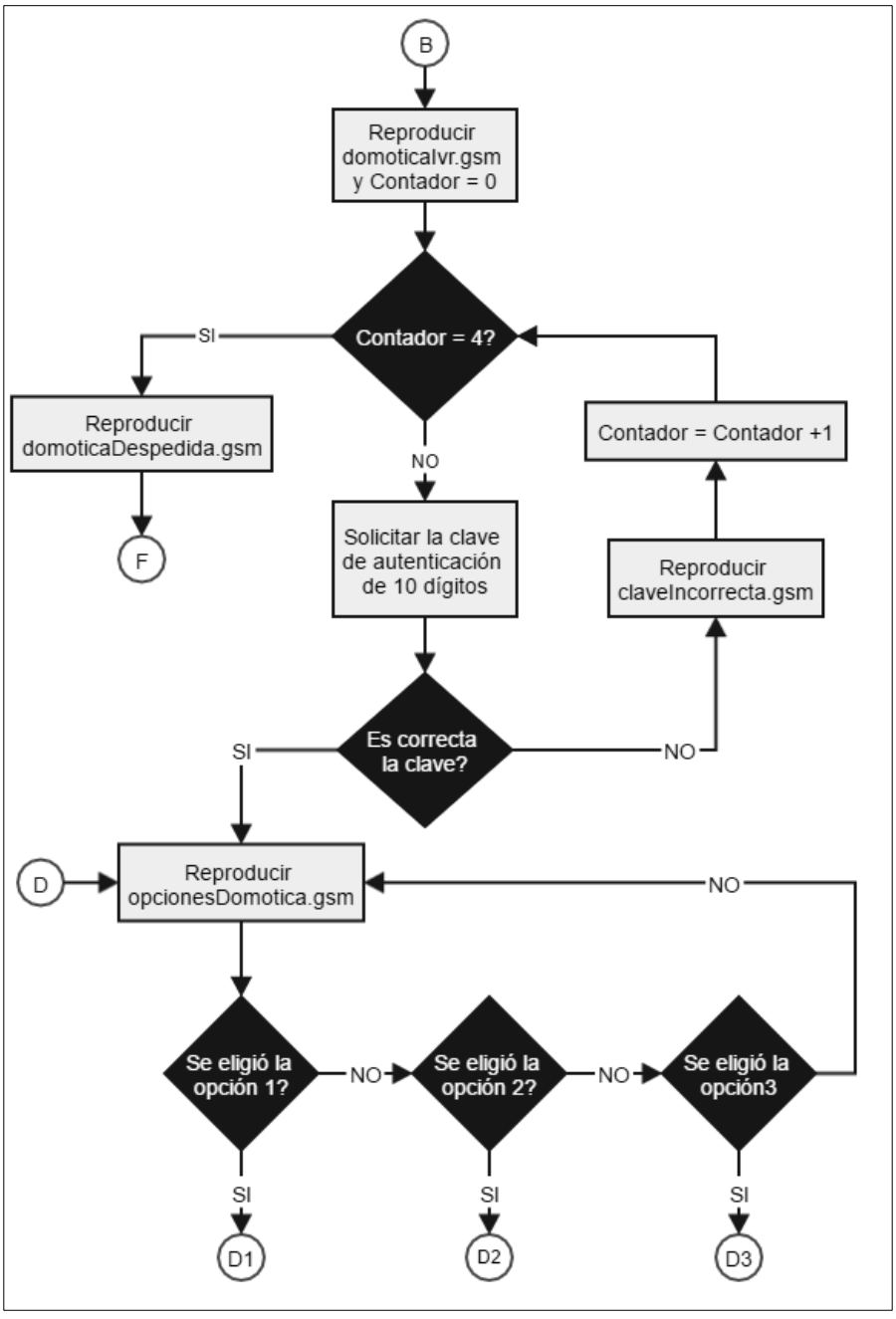

Figura 2.18. Diagrama de flujo, menú interactivo del módulo central telefónica, menú para la extensión 9685

El bloque D1, es decir, el submenú 1, es la sección dedicada para consultar el estado de del domicilio. En el submenú 1 se reproducirá el audio opcionesD1.gsm, en donde se presentará cuatro opciones, las cuales se enumeran a continuación:

- Opción 1: Consultar el estado completo de la casa.

- Opción 2: Consultar la temperatura de la casa.
- Opción 3: Consultar la luminosidad de la casa.
- Opción 0: Regresar al menú principal (bloque D).

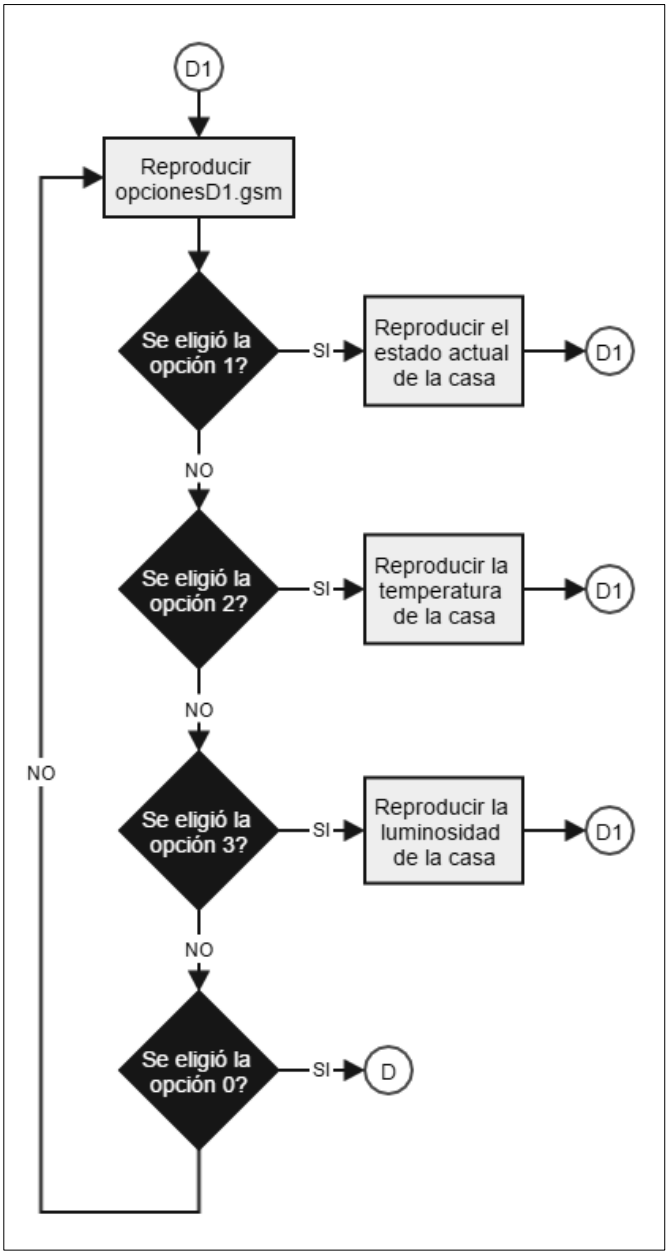

Figura 2.19. Diagrama de flujo, menú interactivo, bloque D1

En el caso de seleccionar la opción 1, el módulo central telefónica consultará al módulo central el estado de todos los sensores y actuadores del prototipo. Se plantea usar Festival TTS para convertir la información adquirida de texto a audio para reportar esta información al usuario. De la misma manera se procederá para la opción 2 y opción 3, con la diferencia

que el módulo central telefónica consultará el estado del sensor de temperatura y luminosidad respectivamente.

Si se ingresa la opción 0, el flujo de la llamada se retorna al bloque D, reproduciendo inmediatamente el audio opcionesDomotica.gsm. Por otro lado, si se ingresa una opción incorrecta, se debe reproducir el audio opcionesD1.gsm del bloque D1. Regresando al menú principal de la extensión central domótica, al seleccionar la opción 2 (Cambiar el estado de los actuadores), se reproducirá el audio opcionesD2.gsm, el mismo que presentará siete opciones, es decir, una opción para cada actuador y uno para regresar al nivel principal, como se indica en la Figura 2.20. Las opciones disponibles en el submenú 2 son:

- Opción 1: Opción para ingresar al menú de la luminaria 1 (bloque D21).
- Opción 2: Opción para ingresar al menú de la luminaria 2 (bloque D22).
- Opción 3: Opción para ingresar al menú de la luminaria 3 (bloque D23).
- Opción 4: Opción para ingresar al menú de la luminaria 4 (bloque D24).
- Opción 5: Opción para ingresar al menú de la cortina 1 (bloque D25).
- Opción 6: Opción para ingresar al menú de la cortina 2 (bloque D26).
- Opción 0: Opción para regresar al menú principal (bloque D).

Si se ingresa una opción no válida se volverá a reproducir el archivo opcionesD2.gsm correspondiente al bloque D2.

En esta sección del menú interactivo se plantea un menú por cada opción, es decir, un menú por cada actuador.

En la Figura 2.21 se muestra el bloque 21, es decir, el menú para el primer actuador (la luminaria 1). Iniciará reproduciendo el archivo opcionFoco1.gsm en el que se solicitará elegir entre las siguientes tres opciones:

- Opción 1: Consultar el estado de la luminaria 1.
- Opción 2: Cambiar el estado de la luminaria 1.
- Opción 0: Regresar al submenú 2 (bloque D2).

Para la opción 1, el módulo central telefónica realizará la consulta del estado de la luminaria 1 al módulo central, para después, presentarlo en forma de audio mediante Festival TTS. Mientras que, al seleccionar la opción 2, el módulo central dará la instrucción, según la taba 2.xx, al módulo central solicitando se cambie el estado de la luminaria 1, es decir, si la luminaria se encontraba encendida, se solicita apagar la luminaria, y visceversa.

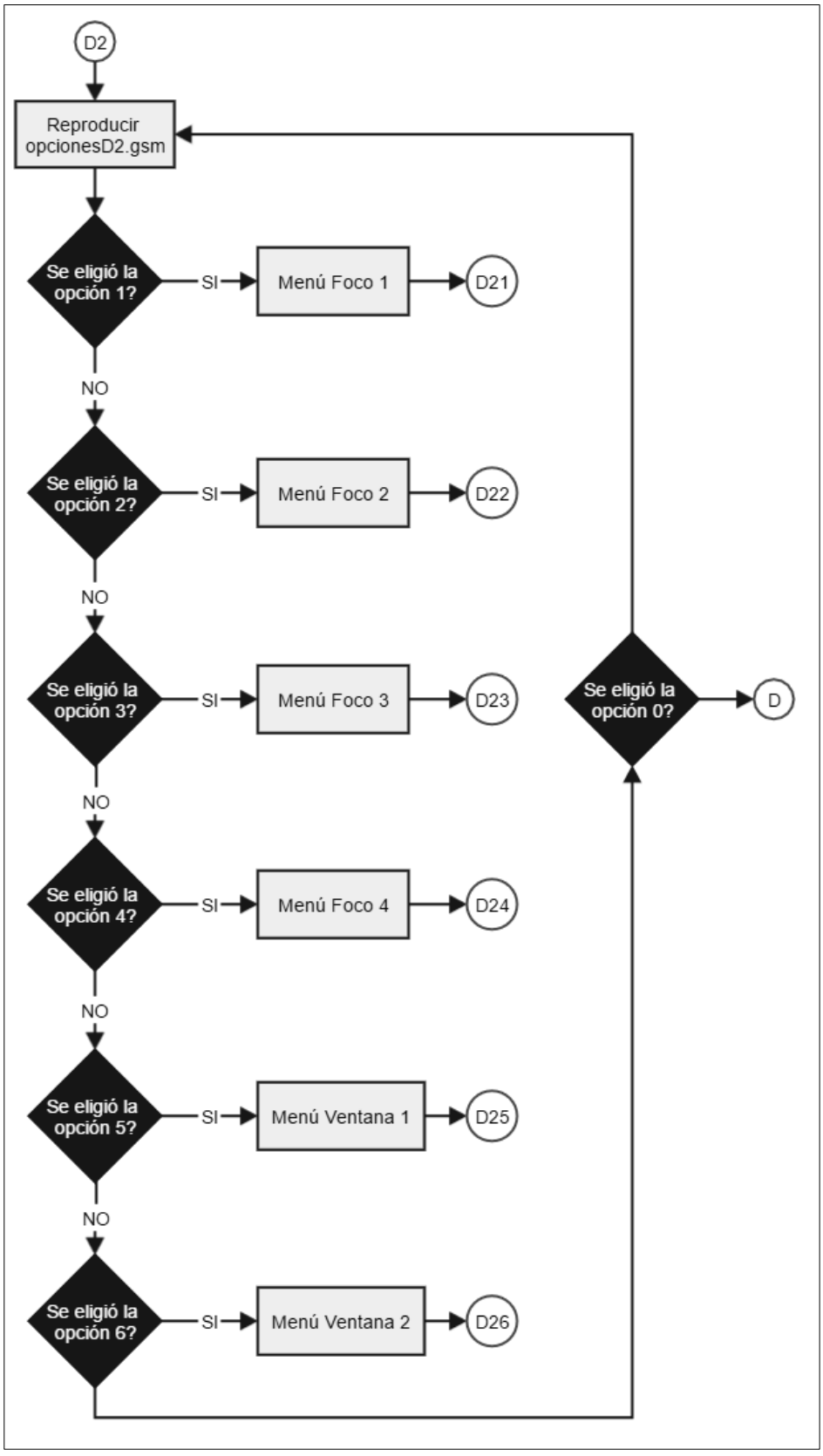

Figura 2.20. Diagrama de flujo, menú interactivo, bloque D2

Después, será necesario confirmar si la acción tuvo lugar, o si se tuvo algún inconveniente, mediante el uso de Festival TTS. Si la opción 0 fue seleccionada, el flujo de la llamada será redirigida al bloque D2, el submenú 2. Finalmente, si la opción seleccionada no es ninguna de las anteriores se reproducirá de nuevo el archivo opcionFoco1.gsm.

Lo mismo sucederá para la luminaria 2, la luminaria 3, la luminaria 4, la cortina 1 y la cortina 2, como se muestra en las Figura 2.22 a Figura 2.26 respectivamente.

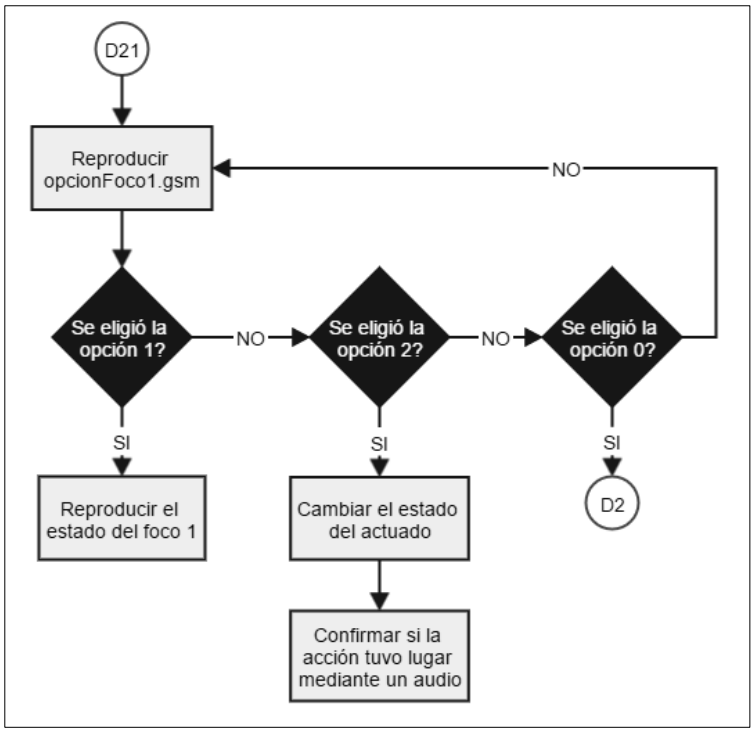

Figura 2.21. Diagrama de flujo, menú interactivo, luminaria 1

Regresando al bloque D, al menú principal, la tercera y última opción de este menú es la denomidada opciones de seguridad. Al seleccionar esta opción, se reproducirá el archivo opcionesD3.gsm, el cual presentará un menú con la siguientes opciones:

- Opción 1: Opción para ingresar al menú del sensor de movimiento.
- Opción 2: Opción para ingresar al menú del simulador de presencia.
- Opción 0: Opción para regresar al menú principal.

Si se ingresa una opción incorrecta se reproducirá de nuevo el archivo opcionesD3.gsm.

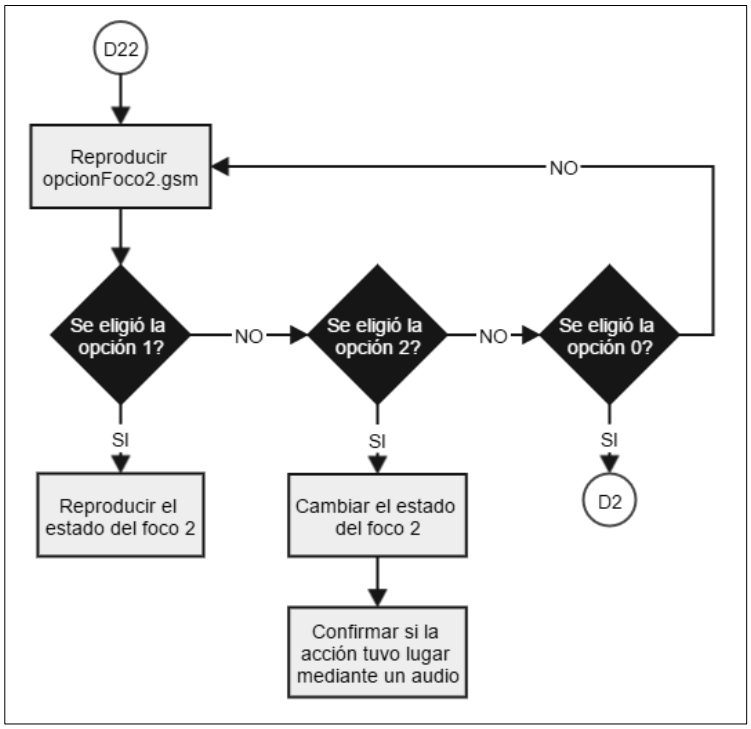

Figura 2.22. Diagrama de flujo, menú interactivo, luminaria 2

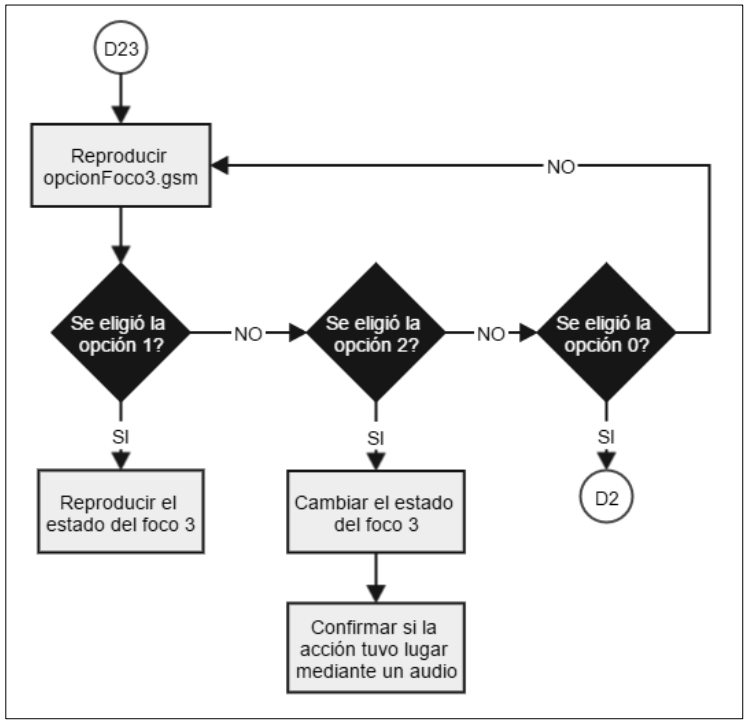

Figura 2.23. Diagrama de flujo, menú interactivo, luminaria 3

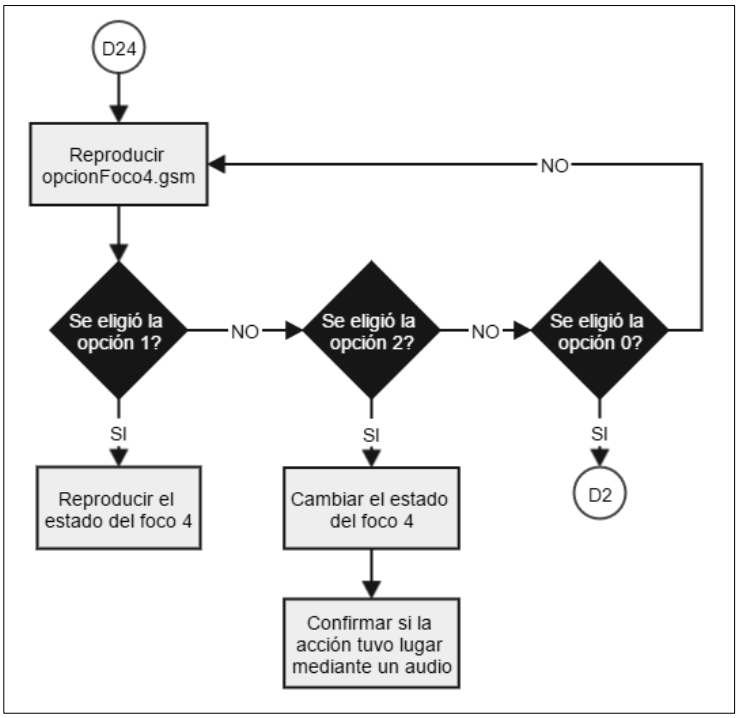

Figura 2.24. Diagrama de flujo, menú interactivo, luminaria 4

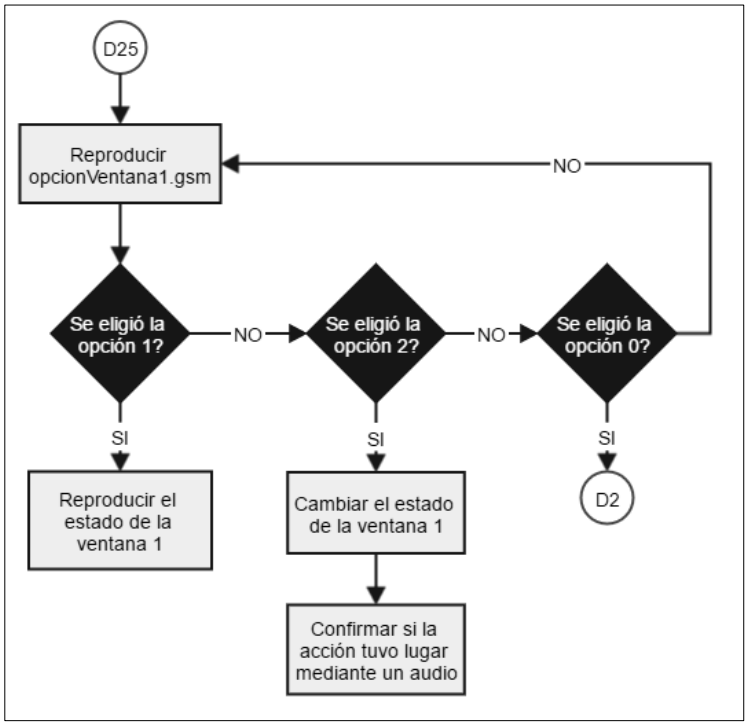

Figura 2.25. Diagrama de flujo, menú interactivo, cortina 1

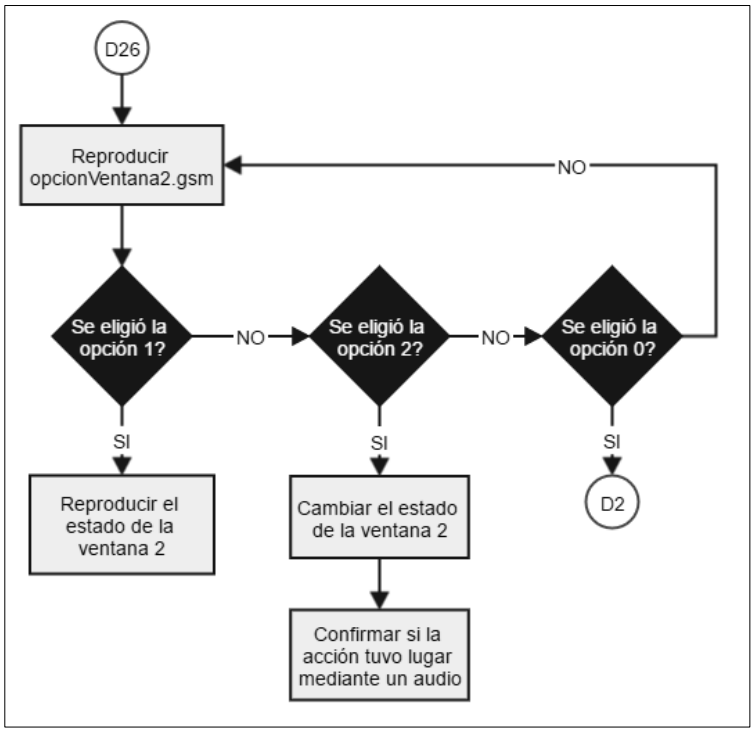

Figura 2.26. Diagrama de flujo, menú interactivo, cortina 2

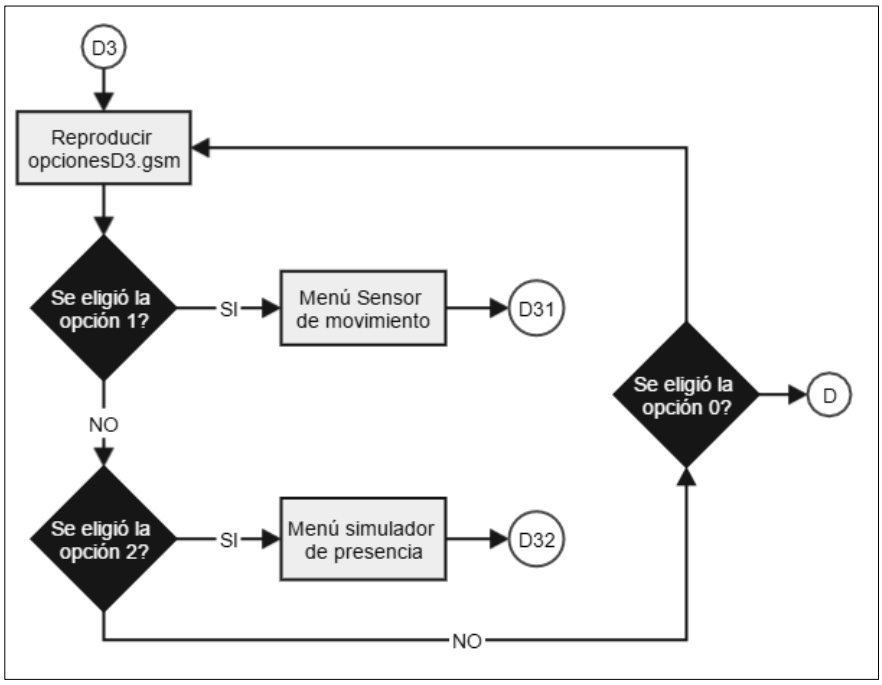

Figura 2.27. Diagrama de flujo, menú interactivo, bloque D3

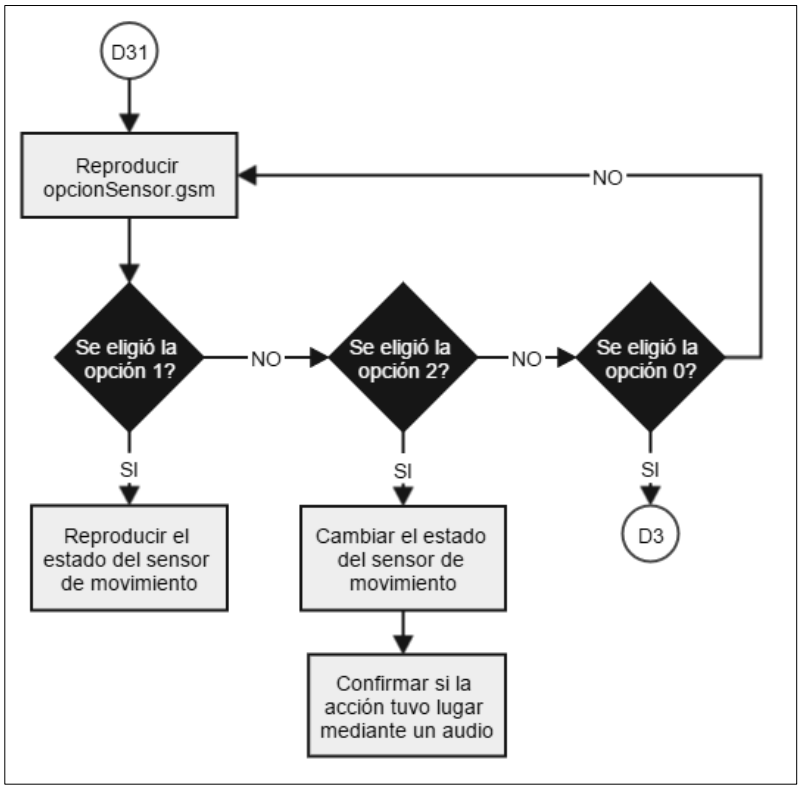

Figura 2.28. Diagrama de flujo, menú interactivo, bloque D31

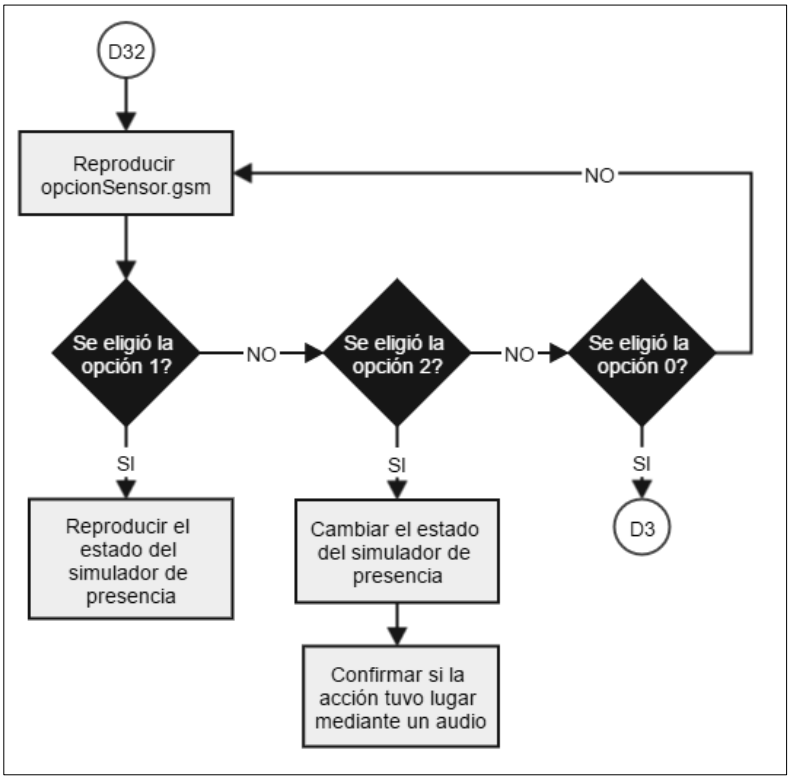

Figura 2.29. Diagrama de flujo, menú interactivo, bloque D32

Los boques D31 y D32, menú del sensor de movimiento y menú simulador de presencia respectivamente, tendrán la misma funcionalidad que los menús de los actuadores, como se puede apreciar en las Figura 2.28 a Figura 2.29.

Adicionalmente, el módulo central telefónica contará con un scipt desarrollado en ruby que será llamado desde el plan de marcado de Asterisk, según como se navegue por el menú intervactivo. Este script será el encargado de comunicarse con el módulo central y de generar el texto que posteriormente Asterisk lo convertirá, por medio de Festival TTS, a audio.Para que Asterisk haga uso del script, Asterisk de ejecutar el script desde el plan de marcado, y además, pasar como argumento el comando correspondiente a la acción que se debe realizar según la tabla mostrada a continuación:

Tabla 2.16. Lista de comandos que utilizará Asterisk para ejecutar el script del módulo central telefónica

COMANDO	INTERPRETACIÓN
ESTADOCASA	Se solicita un informe completo del estado de los actuadores y sensores del hogar
TEMPERATURA	Se solicita información sobre la temperatura del domicilio
LUMINOSIDAD	Se solicita información sobre la luminosidad del domicilio
ESTADOFOCO1	Se solicita información sobre el estado de la luminaria 1
CAMBIARFOCO1	Se solicita al módulo central que cambie el estado de la luminaria 1
ESTADOFOCO2	Se solicita información sobre el estado de la luminaria 2
CAMBIARFOCO2	Se solicita al módulo central que cambie el estado de la luminaria 2
ESTADOFOCO3	Se solicita información sobre el estado de la luminaria 3
CAMBIARFOCO3	Se solicita al módulo central que cambie el estado de la luminaria 3
ESTADOFOCO4	Se solicita información sobre el estado de la luminaria 4

CAMBIARFOCO4	Se solicita al módulo central que cambie el estado de la luminaria 4
ESTADOVENTANA1	Se solicita información sobre el estado de la cortina 1.
CAMBIARVENTANA1	Se solicita al módulo central que cambie el estado de la cortina 1.
ESTADOVENTANA2	Se solicita información sobre el estado de la cortina 2.
CAMBIARVENTANA2	Se solicita al módulo central que cambie el estado de la cortina 2.
ESTADOSENSOR	Se solicita información sobre el estado del sensor de movimiento, es decir, si está activado o no.
CAMBIARSENSOR	Se solicita al módulo central que cambie el estado del sensor de movimiento.
ESTADOPRESENCIA	Se solicita información sobre el estado del simulador de presencia, es decir, si está activado o no.
CAMBIARPRESENCIA	Se solicita al módulo central que cambie el estado del simulador de presencia.

Software módulo Web

Una interfaz gráfica servida por una aplicación web permitirá a un usuario enviar los comandos al módulo central según la Tabla 2.5. Además, para una mejor visualización, se desarrollará la interfaz en dos pestañas. La primera mostrará seis botones, una por cada actuador. La segunda pestaña mostrará la información de los sensores de luminosidad y temperatura, un botón para activar/desactivar la alarma del sensor de movimiento y un botón para activar/desactivar el simulador de presencia.

Se plantea desarrollar la interfaz de la aplicación web con HTML [32], CSS3 [33] y Javascript [34] para darle contenido, estilo y funcionalidad a la interfaz. Además, será necesario, en el lado del servidor, un pequeño script que sirva como puente entre el HTML y el módulo central, es decir, que reciba la acción solicitada en la interfaz web y se comunique con el módulo central mediante sockets.

Se propone el uso de la librería Sinatra [35] de Ruby, el cual nos permitirá realizar la interfaz web y comunicar los requerimientos del usuario al módulo central. Finalmente, el servicio web debe tener un alcance únicamente local.

Software módulo Correo electrónico

El software elegido para el servicio de correo electrónico es Postfix [36], debido a que este servidor de correo es de código abierto y muy fácil de utilizar. Este software también será instalado en la Raspberry Pi. El dominio con el que se lo configurará será tesis.epn y el nombre de usuario del servidor será central.tesis.epn.

Para demostrar el funcionamiento del módulo correo electrónico se creará cuatro cuentas de correo electrónico en el servidor Postfix. Las cuentas que se deberá crear se muestran en la Tabla 2.17.

Tabla 2.17. Usuarios para correo electrónico

Usuario	Correo electrónico	Carpeta
juan	juan@tesis.epn	/home/juan/Maildir
gabriela	gabriela@tesis.epn	/home/gabriela/Maildir
leonardo	leonardo@tesis.epn	/home/leonardo/Maildir
carla	carla@tesis.epn	/home/carla/Maildir
john	john@tesis.epn	/home/john/Maildir
central	central@tesis.epn	/home/pi/Maildir

Se define, además, una lista de comandos que los usuarios deberán colocar en el “Asunto” del correo electrónico para indicarle qué acción debe realizar el prototipo, según la Tabla 2.18.

Tabla 2.18. Comandos para enviar al módulo central vía correo electrónico

COMANDO	INTERPRETACIÓN
FOCO 1 ENCENDER	Se solicita a central domótica que encienda la luminaria 1
FOCO 1 APAGAR	Se solicita a central domótica que apague la luminaria 1
FOCO 2 ENCENDER	Se solicita a central domótica que encienda la luminaria 2
FOCO 2 APAGAR	Se solicita a central domótica que apague la luminaria 2
FOCO 3 ENCENDER	Se solicita a central domótica que encienda la luminaria 3

FOCO 3 APAGAR	Se solicita a central domótica que apague la luminaria 3
FOCO 4 ENCENDER	Se solicita a central domótica que encienda la luminaria 4
FOCO 4 APAGAR	Se solicita a central domótica que apague la luminaria 4
VENTANA 1 ABRIR	Se solicita a central domótica que abra la cortina de la ventana 1
VENTANA 1 CERRAR	Se solicita a central domótica que cierre la cortina de la ventana 1
VENTANA 2 ABRIR	Se solicita a central domótica que abra la cortina de la ventana 2
VENTANA 2 CERRAR	Se solicita a central domótica que cierre la cortina de la ventana 2
SENSOR DE MOVIMIENTO ENCENDER	Se solicita a central domótica que habilite la alarma del sensor de movimiento
SENSOR DE MOVIMIENTO APAGAR	Se solicita a central domótica que habilite la alarma del sensor de movimiento
SIMULADOR DE PRESENCIA ENCENDER	Se solicita a central domótica que habilite la simulación de presencia
SIMULADOR DE PRESENCIA APAGAR	Se solicita a central domótica que deshabilite la simulación de presencia

Una vez que el correo llegue al servidor de correo. El módulo correo electrónico debe ser capaz de tomar el Asunto del correo en la bandeja de entrada, procesarlo, e indicar al módulo central la acción que esta debe tomar, según los comandos mostrados en la Tabla 2.4.

Se desarrollará un cliente POP3, utilizando el lenguaje Ruby, en la misma Raspberry Pi, con la dirección de correo central@tesis.epn. A su vez, este cliente será el encargado de revisar periódicamente su bandeja de entrada, extraer el Asunto del correo recibido y comunicarse mediante sockets con el módulo central.

Software módulo Aplicación Android

Al igual que con la aplicación web, la aplicación Android servirá como una interfaz gráfica para poder consultar el estado de los sensores e interactuar con los actuadores.

La aplicación Android será muy similar a la página web, tendrá dos pestañas separando sensores y actuadores y será capaz de enviar los mismos comandos presentados en la Tabla 2.4 al módulo central por medio de sockets.

Simulador de presencia

El objetivo del simulador de presencia es evitar robos a la propiedad simulando que la casa se encuentra habitada, cuando en realidad, los habitantes de hogar no se encuentran en casa.

Para el simulador de presencia, se desarrollará un programa en Ruby que interactúe con la central domótica enviándole comandos cada cierto tiempo según la Tabla 2.4. De esta manera, el programa podrá encender/apagar las luminarias para dar la ilusión de que hay personas en el domicilio.

Para el presente proyecto se desarrollará el simulador siguiendo la siguiente secuencia:

- Se enciende la luminaria 1.
- Después de 30 segundos, se enciende la luminaria 2.
- Después de 20 segundos se apaga la luminaria 1.
- Después de 30 segundos se apaga la luminaria 2.
- Después de 5 segundos se enciende la luminaria 3.
- Después de 10 segundos se enciende la luminaria 4.
- Después de 20 segundos se apaga la luminaria 4.
- Después de 15 segundos se apaga la luminaria 3.

Después de diez segundos se repite la secuencia hasta que el usuario desactive el simulador de presencia.

2.2 Fase de implementación

En esta fase, se describe la implementación de las funcionalidades del prototipo, se documentan las pruebas realizadas para verificar y validar la implementación y se detalla el costo referencial de desarrollo del prototipo.

Se empezará explicando las configuraciones que se realizó a los equipos que forman parte de la topología propuesta en la Figura 2.2. Después, se detalla cómo fueron instalados y configurados los servicios de los que hará uso el prototipo. Luego, se mostrará la circuitería que se utilizó en el prototipo para integrar los sensores y actuadores, así como también, la maqueta que se construyó para montar los diferentes elementos del prototipo.

Una vez terminado el hardware, se procederá con la explicación del software desarrollado para cada módulo del prototipo. Finalmente, se presentará los resultados de las pruebas realizadas al prototipo y el costo referencial del proyecto.

Configuración de red

Se configuró el Access Point según los parámetros de red planteados en la fase de diseño. Se asignó, al Access Point, la dirección IP 192.168.100.1 con máscara de red 255.255.255.0, como se muestra en la Figura 2.30, y el nombre de la red (SSID) se asignó "TesisTelefonia" como se muestra en la Figura 2.31.

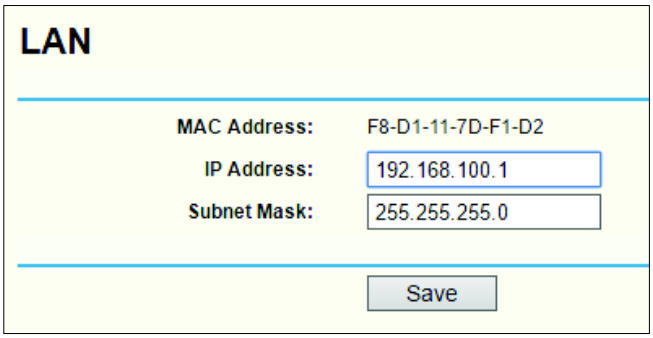

Figura 2.30. Configuración del Access Point, parte 1

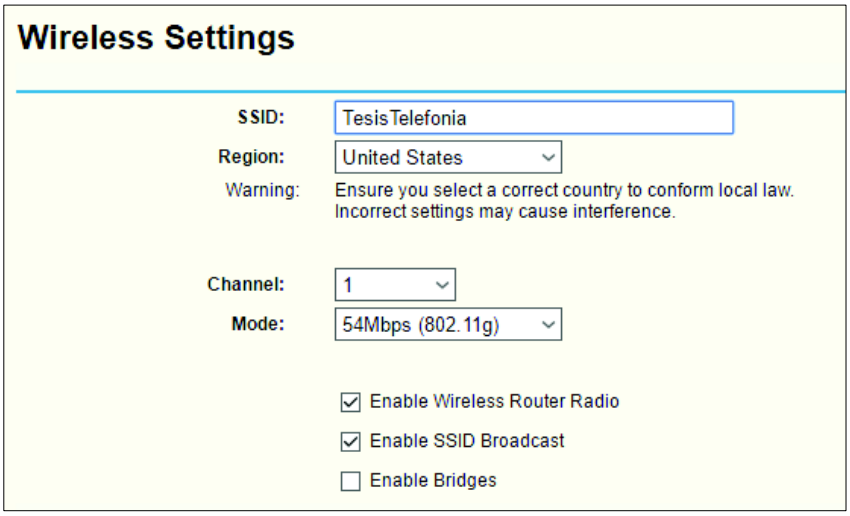

Figura 2.31. Configuración del Access Point, parte 2

Finalmente, se configuró el rango de direcciones IP que podrán ser utilizadas por los dispositivos que no tengan, desde la dirección IP 192.168.100.101 a la 192.168.100.120. Esta configuración se muestra en la Figura 2.32.

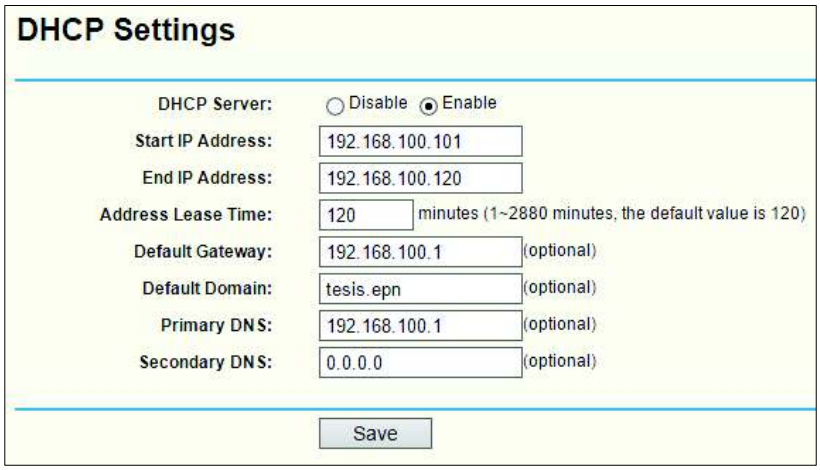

Figura 2.32. Configuración del Access Point, parte 3

Preparación de la Raspberry PI

Previo a la instalación del sistema operativo Raspbian en la Raspberry Pi, se debe estar seguro que la tarjeta microSD se encuentre vacía, caso contrario, se debe formatear la microSD. Una aplicación muy útil para este propósito, y que se utilizó en este proyecto, es el programa SDFormatter.

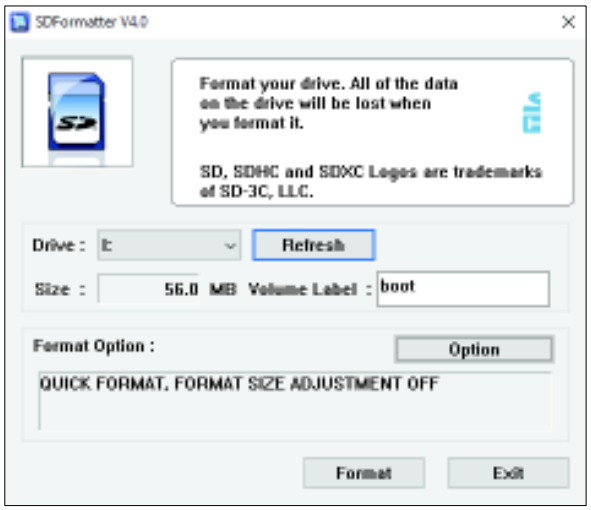

Figura 2.33. Programa SDFormatter para preparar la microSD

En la Figura 2.33 se muestra la interfaz gráfica de SDFormatter. Como se puede apreciar, la interfaz es muy sencilla e intuitiva para formatear unidades microSD.El resultado de este proceso se muestra en la Figura 2.34.

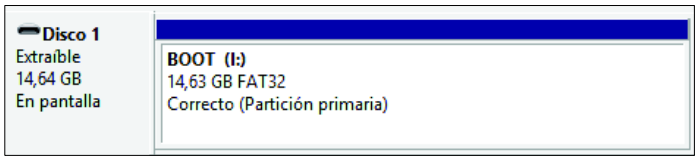

Figura 2.34. Unidad microSD formateada

Después, con el programa Win32 Disk Imager, se carga el sistema operativo Raspbian en la tarjeta microSD, ver Figura 2.35 yFigura 2.36.

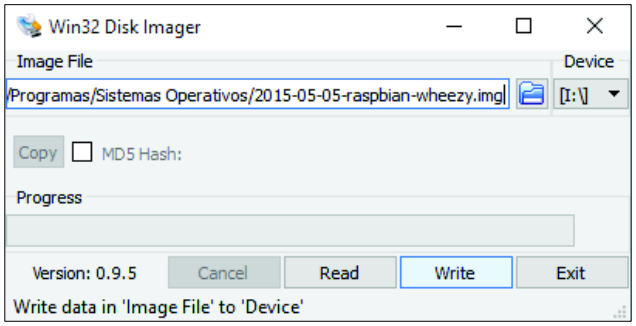

Figura 2.35. Uso del programa Win32DiskImager

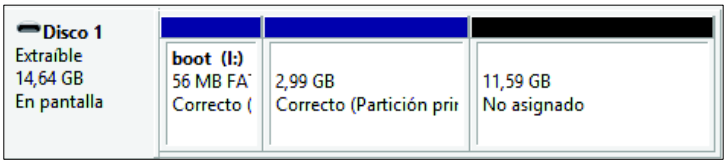

Figura 2.36. Tarjeta microSD con Raspbian

A continuación, se debe actualizar la Raspberry Pi y configurarla con una dirección IP estática para, en adelante, conectarse con la Raspberyy PI mediante SSH (protocolo de conexión remota - Secure Shell).

Con los comandos que se muestran a continuación se actualiza los repositorios, los paquetes informáticos, el firmware y el kernel de la Raspberry PI.

```
sudo apt-get update
sudo apt-get upgrade
sudo rpi-update
sudo reboot
```

Una vez actualizada la Raspberry PI, se edita el archivo /etc/network/interfaces con los parámetros de planteados en la sección 2.2, como se muestra en la Figura 2.37.

```
auto lo
iface lo inet loopback

auto eth0
allow-hotplug eth0
iface eth0 inet static
address 192.168.100.200
netmask 255.255.255.0
gateway 192.168.100.1

auto wlan0
allow-hotplug wlan0
iface wlan0 inet manual
wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
```

Figura 2.37. Parámetros de red en la Raspberry PI

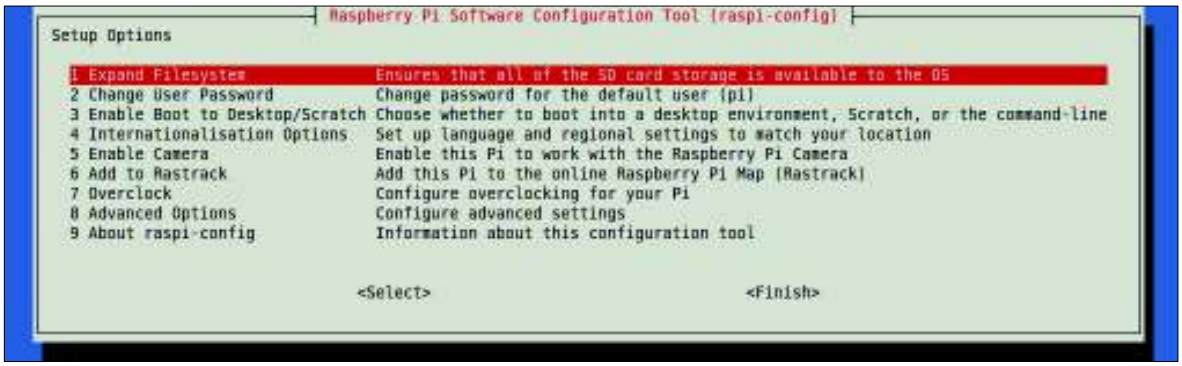

Figura 2.38. Menú del comando sudo raspi-config

Finalmente, con el comando sudo raspi-config, se despliega el menú de la Figura 2.38. En este menú se seleccionó la primera opción para poder utilizar todo el espacio disponible de la microSD.

Instalación de Asterisk

Antes de descargar e instalar Asterisk es necesario instalar las dependencias con el siguiente comando:

```
sudo apt-get install libncurses5-dev libsqlite3-dev libssl-dev
libiksemel-dev uuid-dev libjansson-dev libxml2-dev
```

Después, se procedió a descargar Asterisk en la versión 13 de su página oficial, mediante el siguiente comando:

```
wget http://downloads.asterisk.org/pub/telephony/asterisk/asterisk-13-
current.tar.gz
```

Luego, Para descomprimir, compilar e instalar Asterisk se utilizó los siguientes comandos:

```
tar -zxvf asterisk-13-current.tar.gz
```

```
cd asterisk-13.5.0
./configure
make
sudo make install
```

Una vez instalado Asterisk se debe, además, crear el usuario “asterisk” y cambiar de dueño a los archivos perteneciente al servidor Asterisk. Entonces, con el siguiente comando se creó el usuario Asterisk:

```
sudo useradd asterisk
```

En la Figura 2.39 se muestra como se cambió de dueño a los archivos que el servidor Asterisk utiliza.

```
pi@raspberrypi ~ $ sudo chown -R asterisk:asterisk /usr/lib/asterisk/
pi@raspberrypi ~ $ sudo chown -R asterisk:asterisk /var/lib/asterisk/
pi@raspberrypi ~ $ sudo chown -R asterisk:asterisk /var/spool/asterisk/
pi@raspberrypi ~ $ sudo chown -R asterisk:asterisk /var/log/asterisk/
pi@raspberrypi ~ $ sudo chown -R asterisk:asterisk /var/run/asterisk/
pi@raspberrypi ~ $ sudo chown -R asterisk:asterisk /usr/sbin/asterisk
```

Figura 2.39. Cambio de dueño a los archivos de Asterisk

Luego, en la carpeta de configuración, /etc/Asterisk/, se copió los archivos indications.conf, asterisk.conf y modules.conf de la carpeta configs/samples del archivo descargado para la instalación de Asterisk, puesto que, se utilizará las configuraciones por defecto de estos archivos. Los archivos fueron copiados usando los siguientes comandos:

```
sudo cd configs/samples/
sudo cp indications.conf.sample /etc/asterisk/indications.conf
sudo cp asterisk.conf.sample /etc/asterisk/asterisk.conf
sudo cp modules.conf.sample /etc/asterisk/modules.conf
```

Después, se cambió de dueño a la carpeta de configuración del servidor Asterisk al usuario asterisk con el siguiente comando:

```
sudo chown -R asterisk:asterisk /etc/asterisk/
```

En el archivo de configuración asterisk.conf, se modificó el usuario y grupo en las opciones “runuser” y “rungroup”, asignándoles a las dos el valor “asterisk”. Como se muestra en la Figura 2.40.

Para verificar si la instalación de Asterisk fue la correcta, se ingresa a la terminal de Asterisk, con el comando sudo asterisk -rvvvv. Se obtuvo así, el resultado de la Figura 2.41, lo que indica que Asterisk fue instalado con éxito.

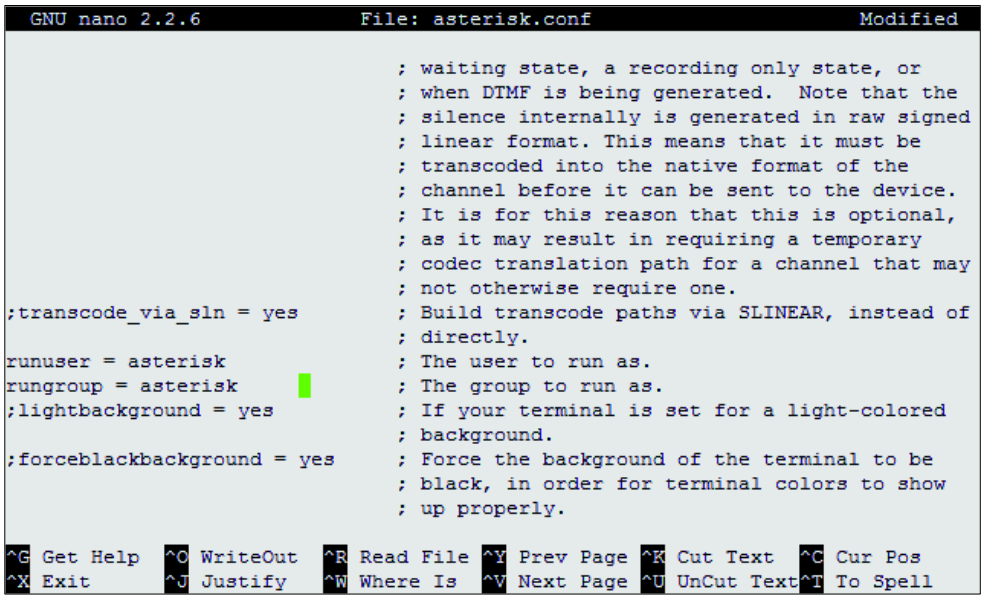

Figura 2.40. Cambios en el archivo /etc/asterisk.conf

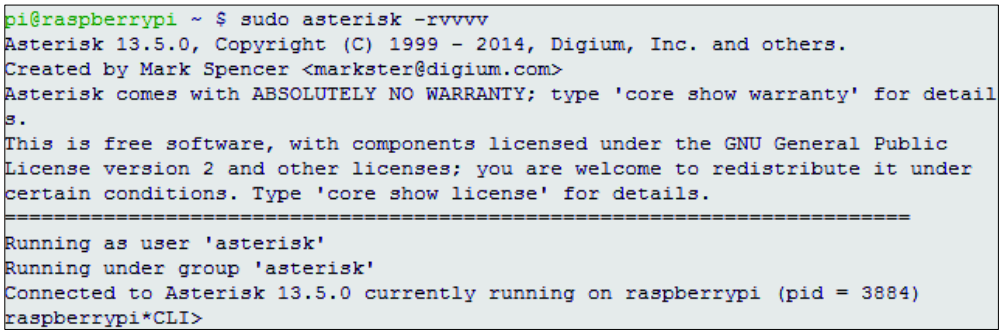

Figura 2.41. Terminal de Asterisk, instalación exitosa

Finalmente, es muy recomendable copiar el archivo rc.debian,asterisk de la carpeta contrib/init.d del archivo descargado para la instalación de Asterisk en la carpeta /etc/init.d/Asterisk, y aplicar el comando `update-rc.d asterisk defaults` para que la Raspberry PI añada a Asterisk a su colección de servicios.

Instalación de Postfix

Postfix se ha seleccionado en la fase de diseño como el servidor de correo electrónico en el que el módulo correo electrónico basará su funcionamiento. Sin embargo, antes de instalar el servidor Postfix en la Raspberry PI es necesario aplicar el comando “modprobe ipv6”, y añadir el texto “ipv6” al final del archivo /etc/modules.

Para instalar Postfix, y sus dependencias, se debe ingresar el siguiente comando:

```
sudo apt-get install dovecot-common dovecot-imapd dovecot-pop3d dovecot-
sieve dovecot-managesieved postfix
```

Una vez terminada la instalación se modificó en el archivo /etc/dovecot/conf.d/10-ssl.conf la línea correspondiente a la seguridad (ssl = required) para brindar un canal encriptado, es decir, seguro. Después, en el archivo /etc/dovecot/conf.d/10-mail.conf se editó la línea mail_location señalando el directorio donde se guardará la información de los correos electrónicos para cada ususario (mail_location = maildir:/home/%u/Maildir), en donde, %u hace referencia todos los usuarios registrados.

A continuación, se creó cada uno de los usuarios propuestos en la Tabla 2.17 con su respectiva carpeta Maildir. En las Figura 2.42 y Figura 2.43 se muestra cómo se creó el usuario juan y su carpeta Maildir.

```
pi@central ~ $ sudo useradd -d /home/juan -m juan
pi@central ~ $ sudo passwd juan
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

Figura 2.42. Se crea el usuario juan y su contraseña

```
pi@central ~ $ su juan
Password:
juan@central /home/pi $ cd
juan@central ~ $ mkdir Maildir
```

Figura 2.43. Se crea la carpeta Maildir para el usuario juan

Luego, en el archivo /etc/postfix/main.cf se verificó que el parámetro “virtual_alias_maps” tenga el valor “hash:/etc/postfix/virtual”. Después, se editó el archivo /etc/postfix/virtual con los usuarios de la Tabla 2.17, como se muestra en la Figura 2.44.

```
 GNU nano 2.2.6          File: /etc/postfix/virtual

juan@tesis.epn          juan
gabriela@tesis.epn      gabriela
leonardo@tesis.epn      leonardo
carla@tesis.epn         carla
jhon@tesis.epn          jhon
central@tesis.epn       pi
usuarios@tesis.epn      juan, gabriela, leonardo, carla, jhon
```

Figura 2.44. Contenido del archivo /etc/postfix/virtual

El archivo /etc/postfix/virtual relaciona la dirección de correo electrónico con el usuario o los usuarios registrados en el servidor. Así por ejemplo, si se envía un correo a la dirección

usuarios@tesis.epn, el correo electrónico será enviado a los usuarios juan, gabriela, leonardo, carla y jhon.

Luminarias

Se construyó los actuadores encargados de las luminarias en base al circuito de la Figura 2.11, donde se muestra el esquema conexión con dos conmutadores para encender/apagar las luminarias tanto manual como automáticamente, y la Figura 2.14, donde se define cual será el circuito para monitorear si la luminaria se encuentra o no encendida. Se utilizó el circuito de la Figura 2.45 como Conmutador 2 de la Figura 2.11, permitiendo así, la interacción del prototipo con la luminaria.

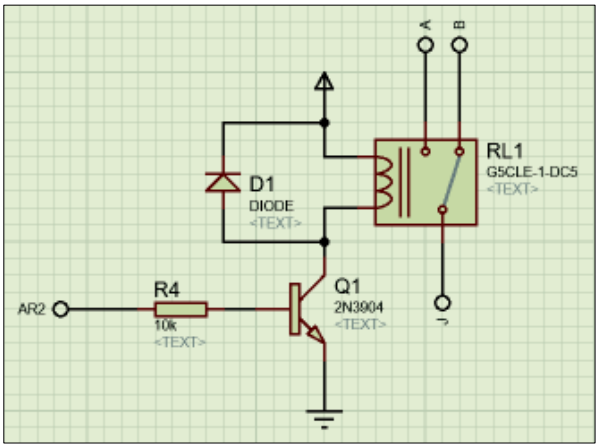

Figura 2.45. Circuito para el control del relé, luminarias

El pin AR2 de la Figura 2.45 se conectó a una salida digital de la placa Arduino, permitiéndole al Arduino tomar control del encendido/apagado de la luminaria, y por medio de este, al prototipo.

Así, se ha implementado en una sola placa (baquelita) el circuito para enceder/apagar la luminaria de la Figura 2.45 y el circuito para monitorear si la luminaria se encuentra encendida/apagada de la Figura 2.14. El resultado de la placa se muestra en la Figura 2.46.

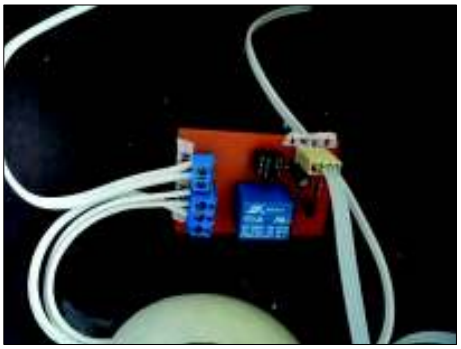

Figura 2.46. Baquelita para la luminaria

Cortinas

Para los actuadores de las cortinas, se diseñó una baquelita en donde se incorpora el driver del motor a pasos y conectores para la fuente del motor. En la Figura 2.47 se muestra la baquelita implementada según el diseño de la Figura 2.16.

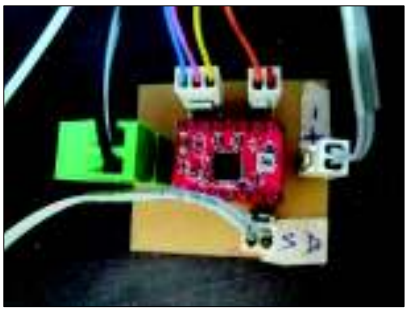

Figura 2.47. Diseño de la baquelita para el actuador de las cortinas

Sensor de luminosidad

De acuerdo a lo planteado en la fase de diseño, se realizó la baquelita del circuito de la Figura 2.3. El resultado se muestra en la Figura 2.48.

Figura 2.48. Baquelita para el sensor de luminosidad

Baquelitas para las placas Arduino

Se diseñó una baquelita para cada placa Arduino de acuerdo a la distribución de pines de las Tabla 2.12 y Tabla 2.13, estas placas facilitarán la conexión de los sensores y actuadores mediante conectores molex. Además, fueron diseñadas para ser colocadas encima del Ethernet Shield de las placas Arduino.

Las Figura 2.49 y Figura 2.50 corresponden a la baquelita implementada para el primer Arduino. Mientras que, las Figura 2.51 y Figura 2.52 corresponden a la baquelita para el segundo Arduino.

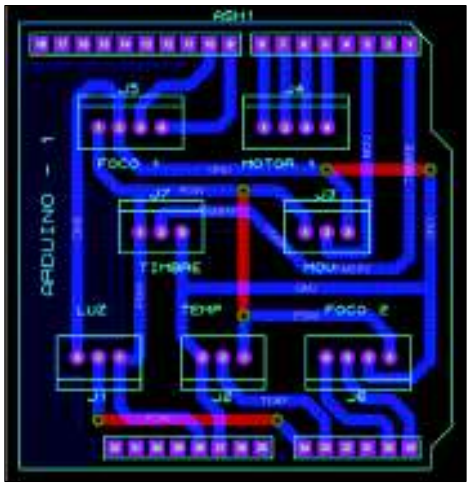

Figura 2.49. Diseño de la baquelita para la placa Arduino 1

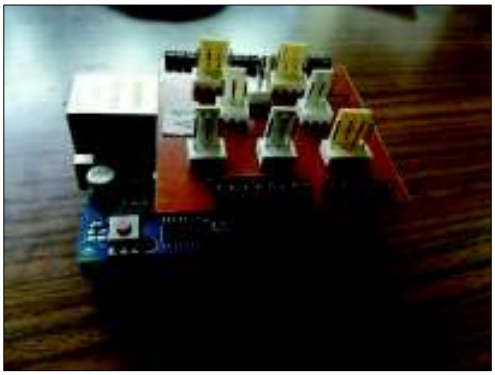

Figura 2.50. Baquelita implementada para la placa Arduino 1

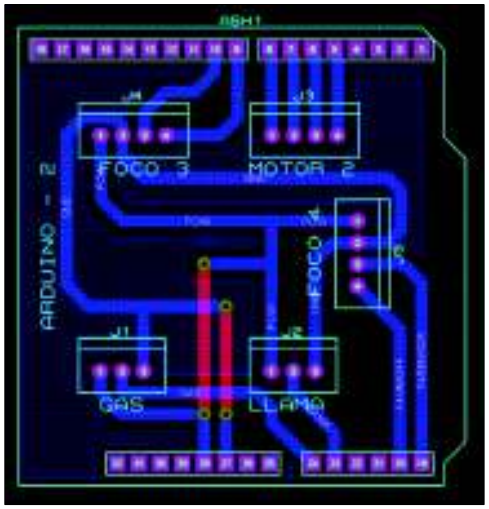

Figura 2.51. Diseño de la baquelita para la placa Arduino 2

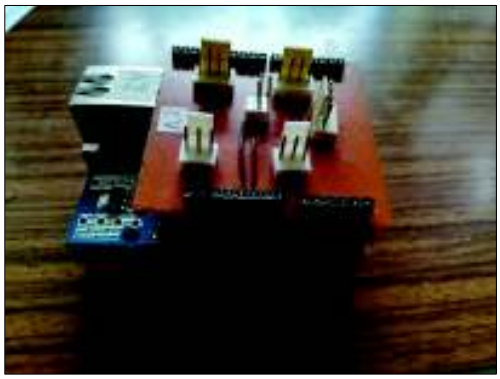

Figura 2.52. Baquelita implementada para la placa Arduino 2

Maqueta

Se construyó una maqueta con el fin de montar el prototipo y demostrar su funcionamiento. La maqueta fue construida con las dimensiones suficientes para apreciar el funcionamiento de cada componente del prototipo sin que estos se encuentren muy próximos como para interferir en la función de cada uno.

En la Figura 2.53 se muestra la maqueta con todos los elementos del prototipo.

Figura 2.53. Maqueta del prototipo

Instalación de Ruby

En la fase de diseño se planteó el uso de Ruby como lenguaje de programación para el desarrollo de los módulos que forman parte del prototipo. Raspbian, el sistema operativo de la Raspberry PI, no viene con este lenguaje de programación, por lo tanto, se instaló la versión más reciente, ingresando los siguientes comandos:

```
wget http://ftp.ruby-lang.org/pub/ruby/2.2/ruby-2.2.1.tar.gz
tar -zxvf ruby-2.2.1.tar.gz
cd ruby-2.2.1/
./configure
make
sudo make install
```

Se puede verificar la instalación exitosa de Ruby ejecutando el comando “ruby -v”. El resultado de este comando es el siguiente:

```
pi@central ~ $ ruby -v
ruby 2.2.1p85 (2015-02-26 revision 49769) [armv6l-linux-eabihf]
```

Figura 2.54. Lenguaje Ruby instalado

Software del Módulo Central

Según lo planteado en la fase de diseño, el módulo central actuará como cliente y como servidor al mismo tiempo. De manera que se utilizó la clase Thread que brinda Ruby para ejecutar dos secuencias de código en paralelo.

En el siguiente bloque de código se muestra las variables utilizadas en el módulo central. En donde se guarda la dirección IP y los puertos TCP utilizados en el prototipo, para la Raspberry, y para las placas Arduino 1 y Arduino 2, según lo planteado en la Tabla 2.2.

```
6   # IP y puerto del módulo central
7   $ownIP = '192.168.100.200'
8   $ownPuerto = 5000
9
10  # IP y puerto del "Arduino 1"
11  $arduino1IP = '192.168.100.201'
12  $arduino1Puerto = 5000
13
14  # IP y puerto del "Arduino 2"
15  $arduino2IP = '192.168.100.202'
```

```
16  $arduino2Puerto = 5000
```

Código 1. Variables que alojan las direcciones IP y los puertos TCP

Mientras que, en Código 2, se muestran los valores iniciales de las variables para los actuadores y sensores, así como también, para las variables que representan si el sensor de movimiento y el simulador de presencia están activados/desactivados.

```
18  # Valores iniciales para los actuadores
19  $foco1 = 'F1OF'
20  $foco2 = 'F2OF'
21  $foco3 = 'F3OF'
22  $foco4 = 'F4OF'
23  $motor1 = 'M1CE'
24  $motor2 = 'M2CE'
25
26  # Valores iniciales para los sensores
27  $sensor1 = 'S1050'  # Sensor de luminosidad
28  $sensor2 = 'S2027'  # Sensor de Temperatura
29
30  # Valor inicial del sensor de movimiento
31  $AlertasMov = 'A1OF'
32
33  # Valor inicial del simulador de presencia
34  $simulacion = 'SPOF'
```

Código 2. Variables con los valores iniciales de los actuadores y sensores

A continuación, en Código 3 se muestra el método domoticaServidor que cumplirá la función de recibir las peticiones y consultas de otros módulos, y que será ejecutado posteriormente mediante la clase Thread.

```
45  # Modulo central - SERVIDOR
46  def domoticaServidor
47    server = TCPServer.open($ownIP, $ownPuerto)
48    loop do
49      Thread.start(server.accept) do |client|
50        comando = client.gets.chomp
...         …
```

```
336         client.close
337       End
338     end
339   end
```

Código 3. Método domoticaServidor

En el método domoticaServidor se debe crear una instancia de la clase TCPServer y pasarle como parámetros la dirección IP de la Raspberry y el puerto donde recibirá las peticiones (5000/TCP). A continuación, en el lazo loop, el bloque de código que se repetirá continuamente, se crea una instancia de la clase Thread por cada petición que se realice guardando en la variable “comando” el contenido de la solicitud recibida para procesarlo posteriormente.

En Código 4 se muestra el método domoticaCliente. Este método también será una instancia de la clase Thread, por lo que se ejecutará en paralelo al método domoticaServidor. El método domoticaCliente consta de un lazo que se repetirá indefinidamente cada 5 segundos. Dentro de este lazo, se ejecutará las consultas a los módulos sensores y actuadores.

```
342   # Modulo central - CLIENTE
343   def domoticaCliente
344     while true
 …          …
383         sleep(5)
384     end
385   end
```

Código 4. Método domoticaCliente

Para que se ejecuten los métodos paralelamente, se utilizó el código mostrado en Código 5. En la primera y segunda línea se declaran las instancias de la clase Thread. Mientras que, en las dos últimas líneas se ejecutan los Threads.

```
398   th1 = Thread.new{domoticaServidor()}
399   th2 = Thread.new{domoticaCliente()}
400   th1.join
401   th2.join
```

Código 5. Ejecución en paralelo de los métodos

En el método domoticaServidor, el módulo central realizará la acción correspondiente según los comandos recibidos de la Tabla 2.5. En Código 6 se muestra lo que realiza el módulo central al recibir el comando FOCO1-ON. Primero, se comprueba si la luminaria se encuentra encendida o apagada mediante la sentencia if. Si la luminaria se encuentra apagada ($foco1 == 'F1OF'), se crea una instancia de la clase SocketCliente, ver Anexo I, pasándole como parámetro la dirección IP de la placa Arduino encargado de la luminaria 1 y el puerto en el que la placa Arduino recibe las peticiones. Después, se invoca el método enviarRecibir() de la clase SocketCliente pasándole como parámetro el comando correspondiente según la Tabla 2.8 (comandos que puede recibir el módulo actuadores). Si el método enviarRecibir devuelve 'OK' significa que el modulo actuadores realizó correctamente la acción solicitada y se cambia el valor a la variable $foco1. De la misma manera, el programa realizará acciones similares con el resto de comandos.

```
53  case comando
54    # ---------- SECCION ACTUADORES ---------- #
55    # Comando solicitando informacion de los actuadores
56    when "CONSULTAF"
57      respuesta =
    "#{$foco1}#{$foco2}#{$foco3}#{$foco4}#{$motor1}#{$motor2}"
58      client.puts respuesta
59      #puts respuesta
60
61    # -- COMANDOS PARA EL FOCO 1 -- #
62    # Comando para encender el Foco 1
63    when "FOCO1-ON"
64      if $foco1 == 'F1OF'
65        conexion = SocketCliente.new($arduino1IP,
    $arduino1Puerto)
66        aux = conexion.enviarRecibir("F1-ON")
67        if aux == 'OK'
68          $foco1 = 'F1ON'
69        end
70      end
71      respuesta =
    "#{$foco1}#{$foco2}#{$foco3}#{$foco4}#{$motor1}#{$motor2}"
72      client.puts respuesta
…   …
```

Código 6. Bloque de código que se ejecuta al recibir el comando FOCO1-ON

Cuando el módulo central recibe un comando correspondiente a una alerta, se crea una instancia de la clase Alertas, que forma parte del módulo alertas, pasando como parámetros el usuario a quien se le notificará la alerta, el nombre del audio que se va a reproducir y un nombre para la alerta. Luego, se ejecuta el método llamarAhora.

```
317  # Comando para disparar alerta del sensor de gas
318  when 'ALERTA-GAS'
319      alarma2 = Alertas.new('0000XXXX0001', 'alertaGas', 'Alerta
     Fuego')
320      alarma2.llamarAhora
```

Código 7. Bloque de código para generar una alerta cuando se activa el sensor de gas

Finalmente, cuando el módulo central reciba el comando TIMBRE desde el módulo portero eléctrico hará sonar el timbre. Para esto, se invoca al método timbrePortero mediante una instancia de la clase Thread, como se muestra a continuación.

```
328    # ---------- PORTERO ELECTRICO ---------- #
329    #Comando para hacer sonar el timbre
330    when 'TIMBRE'
331      timbreHilo = Thread.new{timbrePortero()}
332      timbreHilo.join
333
334  End
 ...  …
389  def timbrePortero
390    timbre = SocketCliente.new($arduino1IP, $arduino1Puerto)
391    timbre.enviar('TIMBRE-ON')
392    sleep(2)
393    timbre.enviar('TIMBRE-OFF')
394  end
```

Código 8. Bloque de código que se ejecuta para hacer sonar el timbre

Software del Módulo Central Telefónica

Siguiendo lo planteado en la fase de diseño se ha configurado el plan de marcado del servidor de telefonía IP modificando el archivo /etc/asterisk/extensions.conf. Primero, se declaró las siguientes variables para facilitar el código en el plan de marcado.

```
1   [globals]
2   ;SE DECLARAN LAS VARIABLES
3   PC = SIP/0000XXXX0000        ;PC
4   LAPTOP = SIP/0000XXXX0001    ;Laptop
5   CEL = SIP/0000XXXX0002       ;Cel
6   PORTERO = SIP/0000XXXXPORT   ;Portero
7   TELEFONO = SIP/1000          ;Telefono fijo (FXS)
8   TIEMPO = 30
```

Código 9. Variables globales para el archivo extensions.conf

Segundo, se configuran las extensiones según lo planteado en la Tabla 2.14, como se muestra en Código 10. En donde, además, se ha configurado un tiempo de espera ${TIEMPO}, es decir 30 segundos, después del cual se reproduce el archivo audioIvrDespedida, si la llamada no es atendida.

```
53  ;EXTENSION PARA TIMBRAR EL TELEFONO FIJO
54  exten => 0,1,Goto(1000,1)
55
56  ;EXTENSIONES PARA LLAMAR A LOS TERMINALES MOVILES
57  ;En la prioridad 1 se redirige la llamada hacia su destino
58  ;con una musica de fondo.
59  ;Si la llamada no es contestada despues de ${TIEMPO} segundos
60  ;se reproduce el sonido "audioIvrDespedida" y finalmente se
61  ;cuelga la llamada.
62
63  exten => 1000,1,Dial(${TELEFONO},${TIEMPO},m) ;TELEFONO FIJO
64    same => n,Playback(audioIvrDespedida)
65    same => n,Hangup()
66
67  exten => 1001,1,Dial(${LAPTOP},${TIEMPO},m) ;LAPTOP
68    same => n,Playback(audioIvrDespedida)
69    same => n,Hangup()
70
71  exten => 1002,1,Dial(${CEL},${TIEMPO},m)  ;CEL
72    same => n,Playback(audioIvrDespedida)
73    same => n,Hangup()
```

```
74
75  exten => 1003,1,Dial(${PC},${TIEMPO},m)    ;PC
76    same => n,Playback(audioIvrDespedida)
77    same => n,Hangup()
```

Código 10. Configuración de las extensiones de los equipos terminales

Posteriormente, se configuró las cuentas SIP en el archivo /etc/asterisk/sip.conf. Para esto, es necesario configurar varios parámetros. En el Código 11 se muestra los parámetros generales de las cuentas SIP. Mientras que, en el Código 12 se configura parámetros de seguridad y de codificación para la transferencia de información entre el servidor de telefonía IP y los equipos terminales.

```
1  [general]
2  context = unauthenticated   ;Contexto por defecto para las
   llamadas entrantes
3  allowguest = no          ;Evitar llamadas no autenticadas
4  srvlookup = yes          ;Habilitar búsqueda del registro DNS
   SRV en llamadas salientes
5  udpbindaddr = 0.0.0.0     ;Escuchar peticiones UDP en todas
   las interfaces
6  tcpenable = no           ;Deshabilitar soporte TCP
```

Código 11. Configuración del archivo sip.conf, parámetros generales

```
8  [proyecto-phone](!)        ;Crear un template para nuestros
   terminales
9  type = friend
10 context = Main           ;A dónde irán las llamadas de este
   dispositivo cuando entren al plan de marcado
11 host = dynamic           ;El dispositivo se registrará con
   Asterisk
12 nat = yes              ;Indica que el terminal está detrás de
   un NAT
13                 ;*** NAT stands for Network Address
   Translation, which allows
14                 ;multiple internal devices to share an
   external IP address.
15 secret = tesis12345       ;Contraseña segura para este
   terminal
```

```
16  dtmfmode = auto          ;Acepta tonos de marcado de otros
    dispositivos
17  disallow = all           ;Desabilita los codecs
18  allow = gsm             ;Admite solamente estos codec
19  allow = alaw
20  allow = ulaw
```

Código 12. Configuración del archivo sip.conf, varios parámetros

Finalmente, se define identificadores para los equipos terminales, como se muestra en Código 13. De esta manera, el equipo terminal deberá utilizar uno de estos indicadores con la contraseña “tesis12345” definida en la línea 15 de Código 12.

```
22  ;Se define un nombre de terminal y usar el template proyecto-
    phone
23  [0000XXXX0000](proyecto-phone)   ;PC
24  [0000XXXX0001](proyecto-phone)   ;LAPTOP
25  [0000XXXX0002](proyecto-phone)   ;CEL
26  [0000XXXXPORT](proyecto-phone)   ;PORTERO
```

Código 13. Configuración del archivo sip.conf, varios parámetros

Regresando al archivo de configuración /etc/asterisk/extensions.conf, se configuró el servicio de IVR para las llamadas recibidas desde la PSTN. Como se muestra en Código 14, el servicio de IVR se ha sido asignado a la extensión 500. Por lo tanto, cualquier llamada recibida por el Gateway telefónico será dirigida a la extensión 500.

El servicio de IVR, extensión 500, se configuró como se planteó en el diagrama de flujo de la Figura 2.17. Se dispone de tres audios de bienvenida: audioIvrDia, audioIvrTarde y audioIvrNoche, para la mañana, tarde y noche respectivamente. El comando WaitExten(5) fue utilizado para esperar cinco segundos, si la extensión no es ingresada en cinco segundos, el módulo central telefónica redirigirá esta llamada al teléfono fijo.

```
12  ;A la siguiente extension entran las llamadas de la PSTN
13  ;SERVICIO DE IVR
14  exten => 500,1,Answer()
15    Same => n(contestadora),GotoifTime(06:00-
    11:59,*,*,*?ivrdia)
16    same => n,GotoifTime(12:00-17:59,*,*,*?ivrtarde)
17    same => n,BackGround(audioIvrNoche)
```

```
18      same => n,WaitExten(5)
19      same => n,Goto(telefonofijo)
20      same => n(ivrdia),BackGround(audioIvrDia)
21      same => n,WaitExten(5)
22      same => n,Goto(telefonofijo)
23      same => n(ivrtarde),BackGround(audioIvrTarde)
24      same => n,WaitExten(5)
25      same => n(telefonoFijo),Dial(${TELEFONO},${TIEMPO},m)
26      same => n,Playback(audioIvrDespedida)
27      same => n,Hangup()
```

Código 14. Configuración del servicio de IVR

Para las llamadas provenientes de la PSTN, se debe configurar el Gateway Telefónico para pasar el control de la llamada a la extensión 500. El Gateway Telefónico seleccionado para el prototipo es el Grandstream modelo HT503 de la Figura 2.55. El Gateway HT503 cuenta con un puerto FXS, un puerto FXO, un puerto LAN y un puerto WAN.

Figura 2.55. Gateway telefónico Grandstream HT503

La configuración del Gateway telefónico se realizó mediante una interfaz web que provee el equipo HT503. En la información básica del equipo se debe configurar una dirección IP y la información del servidor de telefonía IP.

Al Gateway telefónico se le ha asignado la dirección IP 192.168.100.205, como se muestra en la Figura 2.56. Y se ha añadido a información del servidor, y de paso el número de extensión de la IVR, extensión 500. Ver Figura 2.57. Adicionalmente, en el archivo /etc/asterisk/sip.conf se debe crear una plantilla para el Gateway Telefónico con los parámetros que se muestran en Código 15.

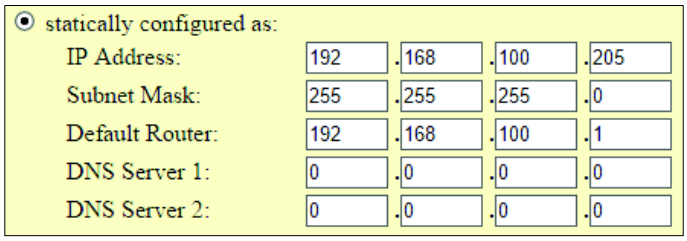

Figura 2.56. Configuración del gateway telefónico, direccionamiento IP

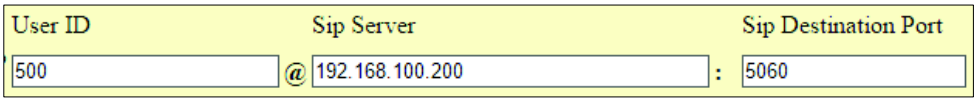

Figura 2.57. Configuración del gateway telefónico, información del servidor SIP

```
[ht503fxo]
  ;[pstn]
type=friend
  ;username=ht503fxo
  ;secret=telefonia
canreinvite=no
host=192.168.100.205
  ;host direccion del gateway telefonico
nat=no
port=5062
disallow=all
allow=alaw
allow=ulaw
dtmf=rfc2833
qualify=yes
context=default
```

Código 15. Configuración del gateway telefónico en el archivo sip.conf

Para finalizar con la configuración del Gateway Telefónico, se configuró el puerto FXS del equipo HT503 como en la Figura 2.58 y el puerto FXO, como se muestra en la Figura 2.59.

Se ha definido que el teléfono fijo tendrá asignada la extensión 1000. Para esto, se añadió en el archivo /etc/asterisk/sip.conf una plantilla para el puerto FXS, como se muestra a continuación en el Código 16.

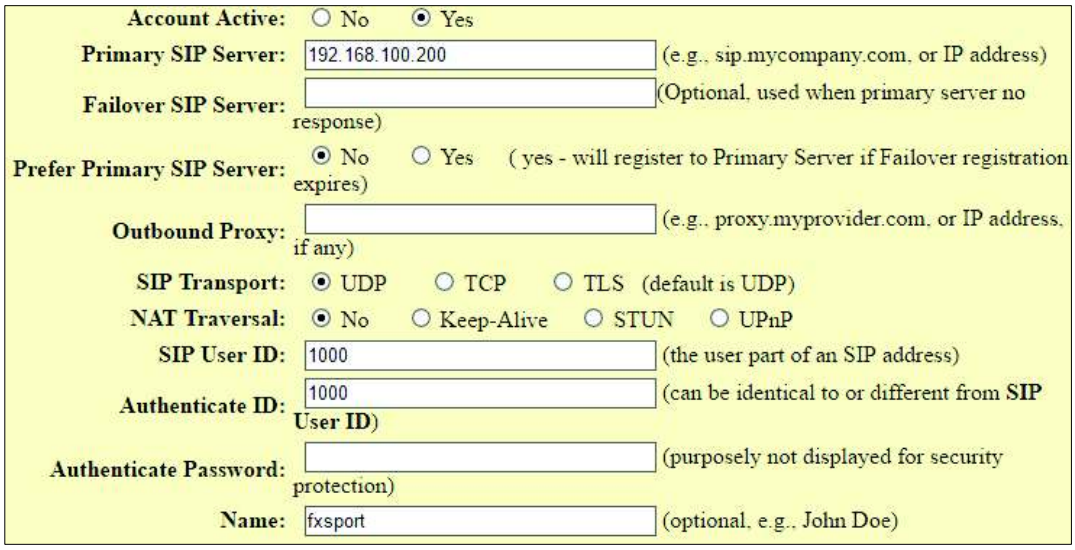

Figura 2.58. Configuración del gateway telefónico, puerto FXS

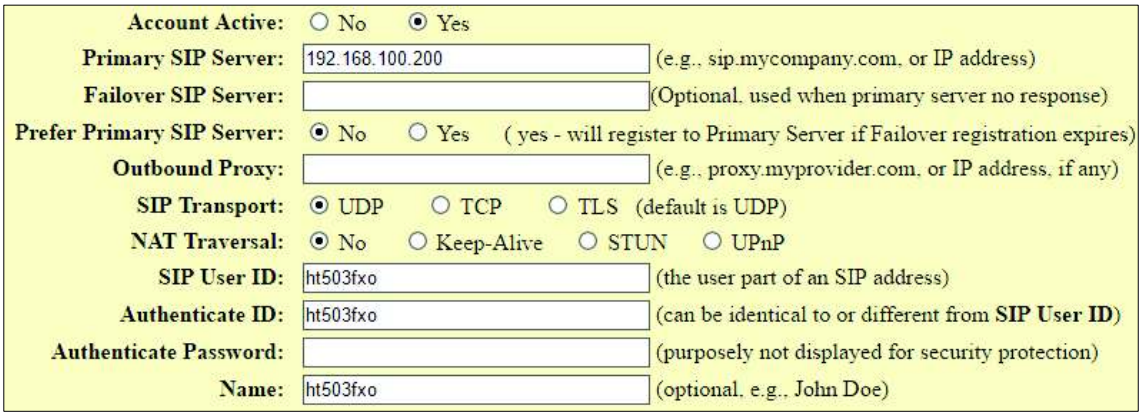

Figura 2.59. Configuración del gateway telefónico, puerto FXO

```
50  [1000]
51  type = friend
52  context = Main
53  host = dynamic  ; El dispositivo se registrará con Asterisk
54  nat = yes   ; indica que el terminal está detrás de un NAT
55  dtmfmode = auto ; acepta tonos de marcado de otros dispositivos
```

Código 16. Configuración del archivo sip.conf para el puerto FXS

Para las llamadas salientes, ya sea a celular, convencional o interprovincial, se añadió al archivo /etc/asterisk/extensions.conf la información del bloque de Código 17. En cada una de estas extensiones, se redirige las llamadas hacia el puerto FXO del Gateway Telefónico pasándole, además, el número con el que se desea comunicar mediante la variable ${EXTEN}.

```
37  ;LLAMADAS A CELULAR
38  exten => _09XXXXXXXX,1,Dial(SIP/ht503fxo/${EXTEN})
39    same => n,Hangup()
40
41  ;LLAMADAS DENTRO DE QUITO
42  exten => _XXXXXXX,1,Dial(SIP/ht503fxo/${EXTEN})
43    same => n,Hangup()
44
45  ;LLAMADAS INTERPROVINCIALES
46  exten => _XXXXXXXX,1,Dial(SIP/ht503fxo/${EXTEN})
47    same => n,Hangup()
```

Código 17. Configuración del archivo extensions.conf para llamadas salientes

Es necesario incluir la extensión del Portero Eléctrico en el archivo /etc/asterisk/extensions.conf. Como se muestra a continuación, se ha asigando la extensión 2000 y se ha seteado el CallerID como “PORTERO” para identificar a la llamada proveniente del portero eléctrico. En la línea 82 del archivo extensions.conf se redirecciona la llamada al número celular del usuario (${CEL}).

```
80  ; EXTENSION DEL PORTERO ELECTRICO
81  exten => 2000,1,Set(CALLERID(name)=PORTERO)
82    same => n,Dial(${CEL},${TIEMPO},m)
83    same => n,hangup()
```

Código 18. Configuración del archivo extensions.conf para la extensión del portero eléctrico

Para la implementación del diagrama de flujo de la Figura 2.18, se ha desarrollado en Ruby un script mostrado en Código 19. El script será ejecutado desde el archivo /etc/asterisk/extensions.conf pasándole como parámetro la contraseña introducida por el usuario.

El script guarda el parámetro que recibe en una variable. Después, le aplica el algoritmo MD5 mediante la clase Digest y la compara con la variable passMD5, que contiene la contraseña 2525363614 cifrada. Si las contraseñas coinciden, el script devuelve 'OK', caso contrario, devuelve el string 'ERROR'.

```
1  #!/usr/bin/ruby
2
3  require 'digest/md5'
4  require_relative 'logs'
5
6  # Se recibe el comando desde Asterisk
7  pass = ARGV[0]
8
9  # Se calcula el hash usando MD5
10 # (Message-Digest Algorithm 5 - RFC 1321)
11 passHash = Digest::MD5.hexdigest(pass)
12
13 # Contraseña 2525363614 despues de aplicar MD5
14 passMD5 = 'aa573d35b12f9671f67818a774213a5a'
15
16 if passHash == passMD5
17   # Si la contraseña es correcta devuelve OK
18   respuesta = 'OK'
19 else
20   # Si es incorrecta guarda el mensaje de error
21   # en un archivo de log y devuelve ERROR
22   error = "Error en la autenticacion"
23   insertarLog(error)
24   respuesta = 'ERROR'
25 end
26
27 print respuesta
```

Código 19. Script para la autenticación

Si la extensión 9685 es marcada, el módulo central telefónica ejecutará las líneas de Código 20. Primero, el módulo central telefónica reproduce el audio domoticaIvr y espera 10 segundos para que el usuario ingrese la contraseña mediante el comando Read, y la

almacena en la variable PASSW. Para verificar si la contraseña es correcta, se ejecuta el script de Código 19 pasándole como argumento la variable PASSW y el resultado se almacena en la variable AUTH, esto se lo realiza con el comando Set. Si la variable AUTH contiene el string "OK" se pasa el control al bloque D. Caso contrario, se reproduce el audio claveIncorrecta y se aumenta en uno a la variable CONTADOR. Por último, si el usuario supera el número máximo de intentos establecido, se reproduce el audio domoticaDespedida y finalmente se cuelga la llamada.

```
86  ; EXTENSION PARA LA CENTRAL DOMOTICA
87  exten => 9685,1,Answer()
88    same => n,Set(CONTADOR=1)
89    same =>
    n(contadorDomotica),Gotoif($[${CONTADOR}=4]?colgarDomotica)
90    ; CONTADOR < 4
91    same => n,Read(PASSW,domoticaIvr,10)
92    same => n,Set(AUTH=${SHELL(ruby
    /home/central/rubyScripts/autenticacion.rb ${PASSW})})
93    same => n,GotoIf($["${AUTH}"="OK"]?bloqueD)
94    ; FALLA LA AUTENTICACION
95    same => n,Playback(claveIncorrecta)
96    same => n,Set(CONTADOR=$[${CONTADOR}+1])
97    same => n,Goto(contadorDomotica)
98    ; CONTADOR = 4
99    same => n(colgarDomotica),Playback(domoticaDespedida)
100   same => n,Wait(1)
101   same => n,Goto(colgar)
```

Código 20. Configuración del archivo extensions.conf para la extensión 9685

Si la contraseña ingresada es la correcta, es decir 2525363614, Se pasará el control a la línea 104 del archivo /etc/asterisk/extensions.conf (ver Código 21). En el bloque D, se reproducirá el audio opcionesDomotica, y guardará la opción seleccionada por el usuario en la variable OPCION. Después, si la opción ingresada es igual a 1, 2 ó 3, se pasará el control de la llamada al bloque D1, D2 ó D3 respectivamente, según se estableció en el diagrama de flujo de la Figura 2.18.

```
103 ; AUTENTICACION CORRECTA
104   same => n(bloqueD),Wait(1)                    ; D
```

```
105    same => n,Read(OPCION,opcionesDomotica,1)
106    same => n,Gotoif($[${OPCION}=1]?bloqueD1)
107    same => n,Gotoif($[${OPCION}=2]?bloqueD2)
108    same => n,Gotoif($[${OPCION}=3]?bloqueD3)
109    ; NO SE ELIGIO OPCION O LA OPCION NO ES VALIDA
110    same => n,Goto(bloqueD)
```

Código 21. Implementación del Bloque D

La implementación del bloque D1 del diagrama de flujo de la Figura 2.19, fue muy similar a la del bloque D, ver Código 22. De la misma manera, se implementó los bloques D2 y D3.

```
112    ; OPCION = 1
113    same => n(bloqueD1),Wait(1)                    ; D1
114    same => n,Read(OPCION1,opcionesD1,1)
115    same => n,Set(TEXTO='')
116    same => n,Gotoif($[${OPCION1}=1]?bloqueD11)
117    same => n,Gotoif($[${OPCION1}=2]?bloqueD12)
118    same => n,Gotoif($[${OPCION1}=3]?bloqueD13)
119    same => n,Gotoif($[${OPCION1}=0]?bloqueD10)
120    same => n,Goto(bloqueD1)
```

Código 22. Implementación del Bloque D1

Las líneas de código de Código 23 son las encargadas de reproducir el estado actual del hogar (sensores y actuadores). Para esto, el módulo central telefónica se ayuda de un script, también desarrollado en Ruby llamado moduloCestralTelefonica.rb, al cual se le pasa como parámetro un comando de la Tabla 2.5 y el resultado lo almacena en la variable TEXTO. Posteriormente, el módulo central telefónica hace uso del software Festival TTS para reproducir el contenido de la variable TEXTO. Finalmente, se pasa el control de la llamada al bloque D1.

De manera similar se ha implementado los diagramas de flujo de las Figura 2.20 a Figura 2.29.

```
122    ;;; D11
123    same => n(bloqueD11),Wait(1)                    ; D11
124    ;;; Reproducir el estado actual de la casa
```

```
125     same => n,Set(TEXTO=${SHELL(ruby
    /home/central/rubyScripts/moduloCentralTelefonica.rb
    ESTADOCASA)})
126     same => n,Festival(${TEXTO})
127     same => n,Wait(1)
128     same => n,Goto(bloqueD1)
```

Código 23. Implementación del Bloque D11

Finalmente, el script moduloCentralTelefonica.rb será el encargado de recibir comandos del plan de marcado (/etc/asterisk/extensions.conf) e interactuar con el módulo central, utilizando sockets y los comandos planteados en la Tabla 2.5.

```
1   #!/usr/bin/ruby
2
3   require_relative 'socketCliente'
4   require_relative 'logs'
5
6   # Se recibe el comando desde asterisk
7   comando = ARGV[0]
…   …
91  ### PROGRAMA PRINCIPAL ###
92  case comando
93  # Submenu 1 (Estado de la casa)
94  #
95  when 'ESTADOCASA'
96     # Retorna el estado actual de la casa
97     begin
98      consulta = SocketCliente.new($ipServidor,
    $puertoServidor)
99      respuesta = ''
100     aux = consulta.enviarRecibir('CONSULTAF')
101     respuesta += actuadoresTexto(aux)
102     aux = consulta.enviarRecibir('CONSULTAS')
103     respuesta += '. '
104     respuesta += sensoresTexto(aux)
105    rescue ExceptionName => error
```

```
106       insertarLog(error)
107       respuesta = 'Se ha producido un error'
108     end
...   …
538   end
539
540   print respuesta
```

Código 24. Parte del script moduloCentralTelefonica.rb

Software del Módulo Sensores y del Módulo Actuadores

Como se planteó en fase de diseño, los módulos sensores y actuadores serán implementados en las placas Arduino. Para que esto sea posible, las dos placas Arduino deberán actuar como servidor y como cliente. Además, deberán trabajar con los comandos establecidos en las Tabla 2.6 a Tabla 2.9, es decir la lista de comandos que pueden enviar y recibir los módulos sensores y actuadores.

Primero, se debe configurar los parámetros de red de las placas Arduino. En Código 25, se configura los parámetros de red para la primera placa Arduino. Además, se importa las librerías Ethernet y SPI que utiliza la placa para comunicarse con la placa Arduino Ethernet Shield.

```
30  #include <Ethernet.h>
31  #include <SPI.h>
32
33  byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
34
35  //Parametros de red para Arduino
36  IPAddress ip(192,168,100,201);
37  IPAddress gateway(192,168,100,1);
38  IPAddress subnet(255,255,255,0);
39  EthernetServer server(5000);
40
41  //Direccion del servidor
42  byte servidor[] = {192, 168, 100, 200};
43  int puertoServidor = 5000;
```

Código 25. Programa Arduino, parámetros de red

A continuación, se muestra en Código 26 como se definen los pines de la placa Arduino 1, según lo establecido en la Tabla 2.12.

```
45  // Pines
46  int foco1 = 8;                  // Pin de salida Foco1
47  int foco1Ent = 9;               // Pin de entrada Foco1
48  int foco2 = 18;                 // Pin de salida Foco2
49  int foco2Ent = 19;              // Pin de entrada Foco1
50  int timbre = 0;                 // Pin de salida para el timbre
    (portero)
51
52  int  sensorMov  =  2;                  // Pin de entrada sensor
    movimiento
53
54  int motorManual = 3;             // Cierre manual de la ventana
55  int motorDireccion = 4;         // PinA salida motor a pasos
56  int motorStep = 5;              // PinB salida motor a pasos
57  int motorFinR = 6;              // Entrada para Fin de carrera
58  int motorFinL = 7;              // Entrada para fin de carrera
```

Código 26. Programa Arduino, distribución de pines

```
61  // Variables auxiliares
62  int auxMov = 0;                // Variable auxiliar para sensor
    de movimiento
63  int auxMotor = 0;               // Vaiable auxiliar para motor
    a pasos
64  boolean auxBtnM = 0;
65  boolean aux;
66
67  // Variables valores
68  boolean estadoFoco1 = 0;         // Valor estado foco1
69  boolean estadoFoco2 = 0;         // Valor estado foco2
70  boolean estadoVentana = 0;         // Valor estado motor
71  int luzValor = 0;
72  int temperatura = 0;
```

```
73
74  String cad = String(20);
```

Código 27. Programa Arduino, declaración de variables

También, se define las variables con las que trabajará el Arduino 1 (ver Código 27). A continuación, se inicializa el servidor del Arduino 1 con los parámetros de red definidos anteriormente y se configuran los pines de la placa Arduino, como entrada o como salida.

```
76  void setup() {
77    Ethernet.begin(mac, ip, gateway, subnet);
78    server.begin();
79    pinMode(timbre, OUTPUT);
80    pinMode(foco1, OUTPUT);
81    pinMode(foco1Ent, INPUT);
82    pinMode(foco2, OUTPUT);
83    pinMode(foco2Ent, INPUT);
84    pinMode(sensorMov, INPUT);
85    pinMode(motorManual, INPUT);
86    pinMode(motorDireccion, OUTPUT);
87    pinMode(motorStep, OUTPUT);
88    pinMode(motorFinL, INPUT);
89    pinMode(motorFinR, INPUT);
90  }
```

Código 28. Programa Arduino, configuración de pines

Luego, en Código 29, se espera por una petición y se almacena el comando recibido en la variable cad. Dependiendo del contenido de la variable cad, la placa Arduino realizará una acción correspondiente a la Tabla 2.6, o a la Tabla 2.7. Por ejemplo, cuando la placa Arduino 1 recibe el comando F1-ON, cambiará el estado del pin 8 a uno lógico, encendiendo la luminaria.

```
156 // Servidor
157   cad = "";                  // Se guarda el comando recibido
158   EthernetClient cliente = server.available();
159   if(cliente){
160     while(cliente.connected()){
161       if(cliente.available()){
```

```
162         char c = cliente.read();
163         cad = cad + c;
164
165         if(c == '\n'){
166
167           if(cad.lastIndexOf("F1-ON") > -1){
168             tooglePin(foco1);
169             cliente.print("OK");
170           }
```

Código 29. Programa Arduino, servidor socket

Por otro lado, la placa Arduino debe ser capaz de conectarse al módulo central mediante socket y enviar el comando respectivo según la Tabla 2.5 cuando se produzca un evento que necesite ser notificado al usuario. Por ejemplo, en Código 30 se muestra el código usado en el Arduino 1 para el sensor de movimiento. Primero lee el estado del sensor de movimiento. Después, si el pin que está conectado al sensor de movimiento percibe un uno lógico, enviará el comando ALARMA-MOV1 al módulo central, invocando al método enviarComando. Por último, el método enviarComando creará la conexión con el módulo central, enviará el comando y cerrará la conexión.

```
94    // Sensor de movimiento
95    aux = digitalRead(sensorMov);
96    if(aux == LOW){
97      auxMov ++;
98    }
99    if(aux == HIGH){
100     auxMov = 0;
101   }
102   if(auxMov == 1){
103     enviarComando("ALARMA-MOV1");
104   }
105   if(auxMov == 3){
106     auxMov --;
107   }
```

Código 30. Programa Arduino, sensor de movimiento

Software del Módulo Web

Para el módulo web se ha implementado una interfaz web según los requerimientos planteados en la fase de diseño. La interfaz se muestra en la Figura a continuación.

Figura 2.60. Interfaz web

Al presionar un botón, la aplicación web envía un comando, de acuerdo a la Tabla 2.5, al módulo central, para que realice la acción correspondiente. Y después de haber confirmado que la acción se ha realizado con éxito, los botones cambiarán de color.

Software Módulo Aplicación Android

El módulo aplicación Android cuenta con una interfaz gráfica dividida en dos pestañas con botones para enviar comandos según la Tabla 2.5.

En la Figura 2.61 se muestra la pestaña actuadores. Por otro lado, la pestaña sensores se muestra en la Figura 2.62.

Figura 2.61. Aplicación Android, pestaña actuadores

Figura 2.62. Aplicación Android, pestaña sensores

Módulo Correo Electrónico

De manera similar al módulo central, el módulo correo electrónico implementado es capaz de recibir los comandos planteados en la Tabla 2.11, e interactuar con el módulo central mediante sockets, haciendo uso de los comandos establecidos en la fase de diseño.

Primero, se desarrolló un cliente de correo electrónico POP3, para poder consultar periódicamente el último correo recibido en la bandeja de entrada para el usuario “central”.

En Código 31 se muestra la configuración para el cliente de correo electrónico POP3 desarrollado en Ruby.

```
106 # Al iniciar el programa se guarda el ID del ultimo correo
107 begin
108   Mail.defaults do
109     retriever_method :pop3, {
110       :address      => "192.168.100.200",
111       :port         => 995,
112       :user_name    => 'central',
113       :password     => 'central12345',
114       :enable_ssl   => true }
115   end
116   correo = Mail.last
117   ultimoID = correo.message_id.to_s
118
119 rescue Exception => e
120   puts 'Error de conexion con servidor de correo'
121   puts e
122 end
```

Código 31. Fragmento de código del módulo correo electrónico

Por otro lado, en el fragmento de código de Código 32, se muestra cómo el módulo correo electrónico consulta el último correo. El módulo verifica si el ID del correo es diferente al último ID guardado. Si son diferentes, se entiende que el correo consultado es un correo nuevo y se procede a extraer el subject del correo electrónico para finalmente enviar el comando correspondiente al módulo central.

Finalmente, el módulo correo electrónico responderá a cualquier petición presentando un informe detallado del estado del hogar, mediante otro correo.

```
125 # Toma el ultimo correo y toma las decisiones en base al
    subject
126 while true
127   begin
128     correo = Mail.last    # Se lee el ultimo correo
129     #
130     if correo.message_id.to_s != ultimoID
```

```
131         case correo.subject.to_s
132
133         # FOCO 1
134         when 'FOCO 1 ENCENDER'
135           # Se envia el comando al actuador correspondiente
136           comando = SocketCliente.new($ipServer, $ipPuerto)
137           auxenvio = comando.enviarRecibir('FOCO1-ON')
138           unless auxenvio == false
139             # Se genera el subject para el correo de respuesta
140             asunto = 'Peticion a Central telefonica'
141             # Se genera el body para el correo de respuesta
142             texto = 'Se ha encendido el Foco 1' + "\n\n"
143             texto += estadoActual()
144             # Se genera el destino para el correo de respuesta
145             origenCorreo = correo.from.to_s
146             origenCorreo = origenCorreo[2..-3]       # Se
    elimina corchetes y parentesis
147             # Se envia el correo de respuesta
148             CorreoEnviar.new.enviar(origenCorreo, asunto,
    texto)
149           else
150             # Se genera el subject para el correo de respuesta
151             asunto = 'Peticion a Central telefonica'
152             # Se genera el body para el correo  de respuesta
153             texto = 'No se ha podido encender el Foco 1' +
    "\n\n"
154             texto += estadoActual()
155             # Se genera el destino para el correo de respuesta
156             origenCorreo = correo.from.to_s
157             origenCorreo = origenCorreo[2..-3]       # Se
    elimina corchetes y parentesis
158             # Se envia el correo de respuesta
159             CorreoEnviar.new.enviar(origenCorreo, asunto,
    texto)
160           end
```

Código 32. Módulo correo electrónico, recibir correo

Software del Módulo Alertas

Según lo establecido en el capítulo de diseño, el módulo alertas será el encargado de notificar al habitante del hogar cuando se presente una emergencia o un evento de interés, según lo planteado en la Tabla 2.10.

```
14  def llamarAhora
15    @archivoID = Time.new.hour.to_s + Time.new.min.to_s +
    Time.new.sec.to_s + @user
16
17    File.open("#{@directorio}#{@archivoID}.call", 'w+') {
    |file|
18      file.puts "Channel: SIP/#{@user}"
19      file.puts "CallerID: #{@callerID}"
20      file.puts "MaxRetries: #{@repeticiones}"
21      file.puts "RetryTime: #{@tRepeticion}"
22      file.puts "WaitTime: #{@tEspera}"
23      file.puts "Application: Playback"
24      file.puts "Data: #{@audio}"
25    }
26
27    system("mv #{@directorio}#{@archivoID}.call
    /var/spool/asterisk/outgoing/")
28  end
```

Código 33. Método llamarAhora del módulo alertas

El módulo alertas contará con el método llamarAhora, el mismo que creará un archivo con los parámetros mostrados en Código 33. Después, el módulo alertas colocará el archivo creado en el directorio /var/spool/asterisk/outgoing/. Asterisk realizará la llamada al usuario indicado en la línea Channel y reproducirá el archivo que se menciona en la línea Data, apenas el archivo sea colocado en el directorio outgoing, antes mencionado.

Software Módulo Portero eléctrico

De acuerdo a lo establecido en la fase de diseño, el módulo portero eléctrico debe ser capaz de hacer sonar el timbre por tres veces consecutivas, y a la cuarta ocasión comunicar al visitante vía telefónica con el móvil del habitante del hogar. El timbre se encuentra ubicado en el pin 0 de la placa Arduino, según la Tabla 2.12.

Se desarrolló una aplicación en Android con una interfaz sencilla, tal como se muestra en la Figura 2.63, con un solo botón, que al presionarlo, envía un comando al módulo central para se encargue de hacer sonar el timbre. Adicionalmente, la aplicación lanza la llamada telefónica mediante el código utilizado en Código 34. En el mismo, se crea una instancia de la clase Intent invocando el método ACTION_CALL. Después, se ejecuta el método setData pasándole como parámetro la extensión con la que se va a comunicar. Por último, se ejecuta la acción con el comando startActivity.

Figura 2.63. Interfaz del módulo portero eléctrico

```
88  if (timbres == 4){
89      callIntent = new Intent(Intent.ACTION_CALL);
90      callIntent.setData(Uri.parse("csip:1001"));
91      try{
92          startActivity(callIntent);
93          timbres = 0;
94      }
95      catch (android.content.ActivityNotFoundException ex){
96          Toast.makeText(getApplicationContext(), "Error:
    SipCall", Toast.LENGTH_SHORT).show();
97      }
98
99  }
```

Código 34. Fragmento de código para producir una llamada telefónica

3. RESULTADOS Y DISCUSIÓN

En el presente capítulo se presenta los resultados de las pruebas realizadas a los componentes del prototipo desarrollado en el presente proyecto de titulación. También, se describirá la metodología empleada en las pruebas aplicadas al prototipo. De igual manera, se comentará los principales inconvenientes que se fueron dando en la implementación del prototipo, y también, cómo fueron superados.

Al final del capítulo, se presentará el costo referencial del prototipo y se discutirá el por qué el prototipo presentado podría ser una buena opción para ser desarrollado comercialmente.

Pruebas aplicadas al Hardware

Para empezar, se decidió evaluar los componentes de hardware del prototipo, iniciando con las luminarias, continuando con las cortinas, y finalizando con los sensores.

Primero, se evaluó el encendido y apagado manual de las luminarias, esta evaluación es fundamental porque si existe un fallo en el prototipo, el habitante del hogar podrá encender y apagar las luminarias hasta resolver el inconveniente sin afectar las actividades diarias que se realizan en el domicilio. Después, se realizaron pruebas en el encendido y apagado mediante el prototipo, es decir, mediante una llamada telefónica, la interfaz web, la aplicación Android y mediante un correo electrónico. Finalmente, se evaluó el Circuito Integrado H11AA1 de la Figura 2.12, encendiendo y apagando la luminaria y verificando si se refleja el cambio en la interfaz web. En la siguiente tabla se muestra en resumen las pruebas realizadas las luminarias, cabe mencionar que todas las pruebas aplicadas al prototipo.

Tabla 3.1. Pruebas aplicadas a las luminarias

Prueba	Observaciones
Encendido manual	Ninguna
Apagado manual	Ninguna
Encendido desde una llamada telefónica	La navegación por el menú interactivo puede resultar un poco engorroso
Apagado desde una llamada telefónica	La navegación por el menú interactivo puede resultar un poco engorroso
Encendido desde la interfaz web	Ninguna
Apagado desde la interfaz web	Ninguna
Encendido desde la aplicación Android	Ninguna
Apagado desde la aplicación Android	Ninguna

Encendido vía correo electrónico	El tiempo que tarda es considerable
Apagado vía correo electrónico	El tiempo que tarda es considerable
El circuito Integrado H11AA1 detecta cuando se enciende la luminaria	Ninguna
El circuito Integrado H11AA1 detecta cuando se apaga la luminaria	Ninguna

Luego, se realizaron pruebas a los motores (cortinas). Primero, se evaluó si las cortinas abren y cierran, es decir, si el prototipo es capaz de girar los motores en sentido horario y antihorario. Para esto, se envió el comando respectivo de la Tabla 2.8 (comandos que puede recibir el módulo actuadores) a la dirección IP y puerto de los Arduinos mediante telnet. Después, se realizó las pruebas para abrir y cerrar las cortinas mediante una llamada telefónica, la interfaz web, la aplicación Android y mediante un correo electrónico. Los resultados de las evaluaciones de los motores se presentan a continuación.

Tabla 3.2. Pruebas aplicadas a las cortinas

Prueba	Observaciones
Abrir las cortinas mediante telnet	Ninguna
Cerrar las cortinas mediante telnet	Ninguna
Abrir las cortinas mediante una llamada telefónica	La navegación por el menú interactivo puede resultar un poco engorroso
Cerrar las cortinas mediante una llamada telefónica	La navegación por el menú interactivo puede resultar un poco engorroso
Abrir las cortinas desde la interfaz web	Ninguna
Cerrar las cortinas desde la interfaz web	Ninguna
Abrir las cortinas desde la aplicación Android	Ninguna
Cerrar las cortinas desde la aplicación Android	Ninguna
Abrir las cortinas vía correo electrónico	El tiempo que tarda es considerable
Cerrar las cortinas vía correo electrónico	El tiempo que tarda es considerable

De las pruebas realizadas se ha observado que la navegación por el menú interactivo que brinda el servidor de telefonía IP no es muy amigable, y además demoroso, por lo que se recomienda utilizarlo únicamente cuando se requiere interactuar con el hogar remotamente. Por otro lado, interactuar con el prototipo vía correo electrónico conlleva un retardo considerable para una aplicación domótica tal como encender luminarias o abrir cortinas.

Para los sensores de temperatura y luminosidad, se varío la magnitud que el sensor monitorea y se verificó cómo este cambio se refleja en la interfaz web. Mientras que, para los sensores de gas, fuego y movimiento se sometió a la presencia de gas doméstico, fuego y movimiento respectivamente y se verificó que la llamada de alerta se realice desde la central telefónica y se reproduzca el audio correspondiente. A continuación se presenta los resultados de las pruebas aplicadas a los sensores.

Tabla 3.3. Pruebas sobre el sensor de temperatura

Prueba	Observaciones
El sensor de temperatura refleja el aumento de temperatura	Ninguna
El sensor de temperatura refleja la disminución de temperatura	Ninguna
El sensor de luminosidad refleja el aumento de luz	Ninguna
El sensor de luminosidad refleja disminución de luz	Ninguna
Se produce la llamada cuando existe presencia de gas doméstico	Ninguna
Se reproduce el audio de alerta por presencia de gas doméstico	El audio se reproduce inmediatamente y no se escucha las primeras palabras
Se produce la llamada cuando existe presencia de fuego	A veces el sensor de fuego es activado por la luz de las luminarias
Se reproduce el audio de alerta por presencia de fuego	El audio se reproduce inmediatamente y no se escucha las primeras palabras
Se produce la llamada cuando se detecta movimiento	Ninguna
Se reproduce el audio de alerta por detección de movimiento	El audio se reproduce inmediatamente y no se escucha las primeras palabras

Es recomendable que los audios de las alertas tengan un silencio de al menos 2 segundos al inicio del audio para escuchar correctamente el audio, y además, que el mensaje de alerta se reproduzca al menos 2 veces.

El sensor de fuego se debe colocar a una distancia de al menos medio metro de las luminarias para que no sea activado por la luz que emite las luminarias. Se debe recordar

que el sensor de fuego es un dispositivo que capta las longitudes de onda del infrarrojo y que las luminarias emiten, aunque no en gran cantidad, estas longitudes de onda.

Pruebas aplicadas al software

Después de haber realizado pruebas a los componentes de hardware, se realizó pruebas a los distintos módulos que conforman el software del prototipo propuesto. Se inició probando el funcionamiento del servidor de telefonía IP (Asterisk) con llamadas entre equipos terminales, navegación por el menú interactivo (extensión 500) y verificando el correcto funcionamiento del sub menú interactivo de la central domótica (extensión 9685).

Tabla 3.4. Pruebas del funcionamiento de Asterisk

Prueba	Observaciones
Llamada desde la laptop (extensión 1001) al celular (extensión 1002)	Ninguna
Llamada al IVR (extensión 500)	Ninguna
Llamada a la extensión central domótica	Ninguna

Luego, se evaluó el funcionamiento del módulo más importante del prototipo, el módulo central. Para esto, se desarrolló un pequeño programa que consiste de un cliente socket que envía los comandos definidos en la Tabla 2.5. Comandos que el módulo central puede recibir) al módulo central, el programa espera que el módulo central realiza la acción correspondiente y despliega en pantalla la confirmación de que el requerimiento fue atendido con éxito o no.

Tabla 3.5. Pruebas de funcionamiento del módulo central

Prueba	Observaciones
Enviar al módulo central el comando para encender la luminaria 1	Ninguna
Enviar al módulo central el comando para apagar la luminaria 1	Ninguna
Enviar al módulo central el comando para encender la luminaria 2	Ninguna
Enviar al módulo central el comando para apagar la luminaria 2	Ninguna
Enviar al módulo central el comando para encender la luminaria 3	Ninguna

Enviar al módulo central el comando para apagar la luminaria 3	Ninguna
Enviar al módulo central el comando para encender la luminaria 4	Ninguna
Enviar al módulo central el comando para apagar la luminaria 4	Ninguna
Enviar al módulo central el comando para abrir la cortina 1	Ninguna
Enviar al módulo central el comando para cerrar la cortina 1	Ninguna
Enviar al módulo central el comando para abrir la cortina 2	Ninguna
Enviar al módulo central el comando para cerrar la cortina 2	Ninguna
Enviar al módulo central el comando para activar el simulador de presencia	Ninguna
Enviar al módulo central el comando para activar el simulador de presencia	Ninguna
Enviar al módulo central el comando para activar el detector de movimiento	Ninguna
Enviar al módulo central el comando para desactivar el detector de movimiento	Ninguna

Todos los comandos fueron recibidos, atendidos y contestados por el módulo central. Una vez asegurado el funcionamiento del módulo central se realizaron pruebas a los diferentes módulos.

Las pruebas de funcionamiento del módulo central telefónica consistieron en llamadas telefónicas. Se navegó por el menú interactivo para controlar y consultar los diferentes componentes que forman parte del prototipo. En la siguiente tabla se muestra los resultados de las pruebas realizadas.

Tabla 3.6. Pruebas del funcionamiento del módulo central telefónica

Prueba	Observaciones
Seleccionar la opción en el IVR para encender la luminaria 1	Ninguna
Seleccionar la opción en el IVR para apagar la luminaria 1	Ninguna
Seleccionar la opción en el IVR para encender la luminaria 2	Ninguna
Seleccionar la opción en el IVR para apagar la luminaria 2	Ninguna
Seleccionar la opción en el IVR para encender la luminaria 3	Ninguna

Seleccionar la opción en el IVR para apagar la luminaria 3	Ninguna
Seleccionar la opción en el IVR para encender la luminaria 4	Ninguna
Seleccionar la opción en el IVR para apagar la luminaria 4	Ninguna
Seleccionar la opción en el IVR para abrir la cortina 1	Ninguna
Seleccionar la opción en el IVR para cerrar la cortina 1	Ninguna
Seleccionar la opción en el IVR para abrir la cortina 2	Ninguna
Seleccionar la opción en el IVR para cerrar la cortina 2	Ninguna
Seleccionar la opción en el IVR para activar el simulador de presencia	Ninguna
Seleccionar la opción en el IVR para desactivar el simulador de presencia	Ninguna
Seleccionar la opción en el IVR para activar el detector de movimiento	Ninguna
Seleccionar la opción en el IVR para desactivar el detector de movimiento	Ninguna

Luego, se realizaron pruebas para el módulo Web, el módulo Correo Electrónico y el módulo Aplicación Android.

Para evaluar el módulo Web se ingresó la URL de la interfaz web (http://192.168.100.200:3000) en un navegador web y se interactuó con el prototipo. En la tabla a continuación se muestra las pruebas realizadas.

Tabla 3.7. Pruebas de funcionamiento del módulo web

Prueba	Observaciones
Encender la luminaria 1 mediante la interfaz web	Ninguna
Apagar la luminaria 1 mediante la interfaz web	Ninguna
Encender la luminaria 2 mediante la interfaz web	Ninguna
Apagar la luminaria 2 mediante la interfaz web	Ninguna
Encender la luminaria 3 mediante la interfaz web	Ninguna
Apagar la luminaria 3 mediante la interfaz web	Ninguna
Encender la luminaria 4 mediante la interfaz web	Ninguna
Apagar la luminaria 4 mediante la interfaz web	Ninguna
Abrir la cortina 1 mediante la interfaz web	Ninguna
Cerrar la cortina 1 mediante la interfaz web	Ninguna
Abrir la cortina 2 mediante la interfaz web	Ninguna

Cerrar la cortina 2 mediante la interfaz web	Ninguna
Activar el simulador de presencia mediante la interfaz web	Ninguna
Desactivar el simulador de presencia mediante la interfaz web	Ninguna
Activar el detector de movimiento mediante la interfaz web	Ninguna
Desactivar el detector de movimiento mediante la interfaz web	Ninguna

Después, para el módulo Correo Electrónico, se redactó correos ingresando en el asunto cada uno de los comandos de la Tabla 2.11. Lista de comandos que admite el módulo correo electrónico) y se verificó que el prototipo cumpla con el requerimiento del usuario.

Tabla 3.8. Pruebas de funcionamiento del módulo correo electrónico

Prueba	Observaciones
Encender la luminaria 1 mediante correo electrónico	Ninguna
Apagar la luminaria 1 mediante correo electrónico	Ninguna
Encender la luminaria 2 mediante correo electrónico	Ninguna
Apagar la luminaria 2 mediante correo electrónico	Ninguna
Encender la luminaria 3 mediante correo electrónico	Ninguna
Apagar la luminaria 3 mediante correo electrónico	Ninguna
Encender la luminaria 4 mediante correo electrónico	Ninguna
Apagar la luminaria 4 mediante correo electrónico	Ninguna
Abrir la cortina 1 mediante correo electrónico	Ninguna
Cerrar la cortina 1 mediante correo electrónico	Ninguna
Abrir la cortina 2 mediante correo electrónico	Ninguna
Cerrar la cortina 2 mediante correo electrónico	Ninguna
Activar el simulador de presencia mediante correo electrónico	Ninguna
Desactivar el simulador de presencia mediante correo electrónico	Ninguna
Activar el detector de movimiento mediante correo electrónico	Ninguna
Desactivar el detector de movimiento mediante correo electrónico	Ninguna

Las pruebas realizadas para el módulo Aplicación Android fueron muy similares a las del módulo Web. Se ejecutó la aplicación Android desde un teléfono móvil y se interactuó con el prototipo. Los resultados de las pruebas se muestran en la siguiente tabla.

Tabla 3.9. Pruebas de funcionamiento del módulo Aplicación Android

Prueba	Observaciones
Encender la luminaria 1 mediante la aplicación Android	Ninguna
Apagar la luminaria 1 mediante la aplicación Android	Ninguna
Encender la luminaria 2 mediante la aplicación Android	Ninguna
Apagar la luminaria 2 mediante la aplicación Android	Ninguna
Encender la luminaria 3 mediante la aplicación Android	Ninguna
Apagar la luminaria 3 mediante la aplicación Android	Ninguna
Encender la luminaria 4 mediante la aplicación Android	Ninguna
Apagar la luminaria 4 mediante la aplicación Android	Ninguna
Abrir la cortina 1 mediante la aplicación Android	Ninguna
Cerrar la cortina 1 mediante la aplicación Android	Ninguna
Abrir la cortina 2 mediante la aplicación Android	Ninguna
Cerrar la cortina 2 mediante la aplicación Android	Ninguna
Activar el simulador de presencia mediante la aplicación Android	Ninguna
Desactivar el simulador de presencia mediante la aplicación Android	Ninguna
Activar el detector de movimiento mediante la aplicación Android	Ninguna
Desactivar el detector de movimiento mediante la aplicación Android	Ninguna

Por último, se realizaron pruebas del funcionamiento del módulo portero eléctrico. Las pruebas consistieron en presionar el botón de timbre para hacer sonar el timbre del prototipo, y después de haber presionado cuatro veces, se verificó si se produce la llamada al teléfono móvil del usuario.

Tabla 3.10. Pruebas del funcionamiento del portero eléctrico

Prueba	Observaciones
Presionar una vez para activar el timbre	Ninguna
Presionar cuatro veces para activar la llamada telefónica	Ninguna

Costo referencial del prototipo

Para finalizar este capítulo, se detalla el costo referencial del prototipo. Primero, se presenta el precio de los principales equipos y elementos utilizados en el presente proyecto. Cabe mencionar que los equipos mencionados a continuación fueron adquiridos en Quito, Ecuador.

Tabla 3.11. Costo referencial de los equipos utilizados

Descripción	Valor Unitario (USD)	Cantidad	Valor Total (USD)
Raspberry PI	$85.00	1	$85.00
Tarjeta miniSD de 16 GB	$12.00	1	$12.00
Arduino UNO	$23.00	2	$46.00
Arduino Ethernet Shield	$24.00	1	$24.00
Cargadores mini USB 9V	$6.00	3	$18.00
Gateway telefónico Grandstream	$100.00	1	$100.00
Access Point WiFi	$25.00	1	$25.00
Patch core de 1 metro	$2.00	4	$8.00
TOTAL			**$318.00**

De la Tabla 3.11 se puede observar que el equipo más caro del proyecto es el Gateway telefónico Grandstream, seguido por el computador de placa reducida Raspberry PI.

En la siguiente tabla se muestra el costo referencial para la construcción de una baquelita para una luminaria. La baquelita contiene borneras para conectar la luminaria, borneras para conectar la alimentación de 110 Voltios, borneras para conectar la baquelita al Arduino y el circuito intergado H11AA1 para verificar si la luminaria se encuentra encendida o no.

Tabla 3.12. Costo referencial de la baquelita para una luminaria

Descripción	Valor Unitario (USD)	Cantidad	Valor Total (USD)
Circuito integrado H11AA1	$0.45	1	$0.45
Resistencia 10 KOhms a 10 W	$0.20	1	$0.20
Resistencia 10 KOhms 0.5 W	$0.02	2	$0.04
Capacitor 2.2uF	$0.05	1	$0.05
TBJ 2N3904	$0.15	1	$0.15
Relé 5V - 10A	$0.50	1	$0.50
Bornera 2 terminales	$0.10	1	$0.10
Bornera 3 terminales	$0.10	1	$0.10
Baquelita	$0.50	1	$0.50
Ácido para baquelitas	$0.50	1	$0.50
TOTAL			**$2.59**

A continuación se detallan los elementos utilizados en la implementación de la baquelita que controla el motor a pasos de una cortina. En el costo referencial incluye el motor a pasos, conectores tanto para el motor como para el Arduino y el driver que facilita el control del motor.

Tabla 3.13. Costo referencial de la baquelita para una cortina

Descripción	Valor Unitario (USD)	Cantidad	Valor Total (USD)
Driver A4988	$7.00	1	$7.00
Conector Mólex 2 pines	$0.20	1	$0.20
Conector Mólex 3 pines	$0.20	2	$0.40
Bornera 2 terminales	$0.30	1	$0.30
Motor a pasos	$3.00	1	$3.00
Baquelita	$0.50	1	$0.50
Ácido para baquelitas	$0.50	1	$0.50
TOTAL			**$11.90**

Continuando con el detalle de gastos, se presenta el costo referencial de los materiales utilizados en la construcción de las baquelitas para las placas Arduino.

Tabla 3.14. Costo referencial de la baquelita para el Arduino 1

Descripción	Valor Unitario (USD)	Cantidad	Valor Total (USD)
Hilera de espadines macho-macho	$0.40	1	$0.40
Conectores Molex 2 pines	$0.15	1	$0.15
Conectores Molex 3 pines	$0.20	4	$0.80
Conectores Molex 4 pines	$0.20	3	$0.60
Baquelita	$0.50	1	$0.50
Ácido para baquelitas	$0.50	1	$0.50
TOTAL			**$2.95**

En la Tabla 3.14 se detalla los elementos utilizados para el Arduino1, mientras que, en la Tabla 3.15 los elementos utilizados para la baquelita del Arduino 2. Como se puede observar, el costo de los elementos para las baquelitas es muy reducido, pero la fabricación de los mismos es muy demoroso.

Tabla 3.15. Costo referencial de la baquelita para el Arduino 2

Descripción	Valor Unitario (USD)	Cantidad	Valor Total (USD)
Hilera de espadines macho-macho	$0.40	1	$0.40
Conectores Molex 2 pines	$0.15	1	$0.15
Conectores Molex 3 pines	$0.20	2	$0.40
Conectores Molex 4 pines	$0.20	3	$0.60
Baquelita	$0.50	1	$0.50
Ácido para baquelitas	$0.50	1	$0.50
TOTAL			**$2.55**

El costo de los sensores utilizados en el prototipo se presenta en la Tabla 3.16. Como se puede observar los sensores son económicos debido a que, para un uso domótico, no es necesario sensores de alta gama.

Tabla 3.16. Costo referencial de los sensores

Descripción	Valor Unitario (USD)	Cantidad	Valor Total (USD)
Resistencia LDR	$0.23	1	$0.23
Potenciómetro 1 KOhms	$0.10	1	$0.10
Sensor de Temperatura LM35	$0.70	1	$0.70
Sensor de Fuego infrarojos	$6.30	1	$6.30
Sensor de Gas MQ-2	$7.00	1	$7.00
Sensor PIR HC-SR5011	$4.50	1	$4.50
Resistencia 330 Ohms a 0.5 W	$0.02	1	$0.02
TOTAL			**$18.85**

Adicional a los elementos detallados anteriormente, se presenta, en la tabla a continuación, el costo de varios elementos empleados en la implementación del prototipo.

Tabla 3.17. Costos varios

Descripción	Valor Unitario (USD)	Cantidad	Valor Total (USD)
Cable gemelo (1 metro)	$0.60	5	$3.00
Boquilla	$0.30	4	$1.20
Foco	$0.50	4	$2.00
Bus de datos (1 metro)	$0.35	1	$0.35
Estaño (1 metro)	$0.40	2	$0.80
TOTAL			**$7.35**

Finalmente, y tomando en cuenta todos los elementos que forman parte del prototipo, se presenta el costo referencial del proyecto en la tabla a continuación.

Tabla 3.18. Costo referencial del prototipo

Descripción	Valor Unitario (USD)	Cantidad	Valor Total (USD)
Equipos	$318.00	1	$318.00
Baquelita Luminaria	$2.59	4	$10.36
Baquelita Ventana	$11.29	2	$22.58
Baquelita Arduino 1	$2.95	1	$2.95

Baquelita Arduino 2	$2.55	1	$2.55
Sensores	$18.85	1	$18.85
Varios	$7.35	1	$7.35
SUBTOTAL			**$382.64**
Hora de desarrollo de software	$10.00	150	$1,500.00
TOTAL			**$1,882.64**

Cabe recalcar que el costo referencial presentado en la Tabla 3.18 no incluye el valor de los equipos terminales ni el costo de la maqueta. También, el costo de la construcción de las baquelitas no ha sido incluido, porque varía mucho dependiendo del técnico y las herramientas utilizadas.

El costo de construcción se ha incluido en “costo de ingeniería”. Debido a que en la implementación del presente proyecto, se tuvo que repetir cuatro veces las baquelitas debido a la falta de experiencia en la fabricación de las mismas.

De la Tabla 3.18 se puede observar que la inversión en equipos y elementos (subtotal) no supera los 400 dólares, logrando así, cumplir uno de los principales objetivos de este proyecto de titulación, que la implementación del prototipo sea de bajo costo.

Después de la presentación de las pruebas de funcionamiento y del costo referencial del prototipo, se puede concluir que se ha diseñado e implementado un prototipo potencialmente comercial, debido a que la inversión es baja y se obtiene un beneficio considerable de seguridad y confort.

4. CONCLUSIONES

- Se diseñó e implementó un prototipo de central telefónica con aplicación domótica utilizando software libre y hardware libre. Esto implica, un costo referencial bastante reducido, en comparación con tecnologías existentes en el mercado, y que podría impulsar el desarrollo de la domótica en el país.
- En el presente proyecto se demuestra la capacidad que tiene el software libre para incorporar alternativas, no tan comunes, pero al mismo tiempo eficaces, para interactuar con los dispositivos del hogar, como por ejemplo la interacción vía telefónica mediante la PSTN y la interacción mediante correo electrónico.
- Se ha logrado implementar un sistema bastante práctico de alertas, como son las llamadas telefónicas al teléfono móvil del habitante del hogar, cuando en el domicilio se presenta un evento potencialmente perjudicial, como una fuga de gas, la presencia de fuego, o la detección de intrusos. Este sistema resulta ser bastante conveniente y aporta muchísimo a la seguridad del hogar, porque de hecho podría salvar vidas.
- Se logró implementar varios servicios en un solo computador de placa reducida Raspberry PI, como el servicio de telefonía IP, servicio web, y servicio de correo electrónico, que funcionan sin ningún problema en un ambiente doméstico, puesto que, no se realizan muchas peticiones simultáneas.
- La utilización de sockets permitió la comunicación entre los módulos del prototipo. El desarrollo modular favorece al mantenimiento y depuración del software, disminuye la complejidad y tamaño del código, y facilita la incorporación de nuevos módulos al prototipo.
- El uso de la placa Arduino facilitó agregar al prototipo sensores y actuadores, debido a que la placa Arduino posee conversores análogo digital y un número significativo de pines. Por otro lado, el módulo Arduino Ethernet Shield permitió la comunicación entre la Raspberry PI y las placas Arduino. Por último, se debe mencionar que el número de placas Arduino que se puede incorporar está limitado por el número de direcciones IP libres, haciendo del prototipo desarrollado, un proyecto con mucha escalabilidad.
- El lenguaje de programación Ruby, al tener una sintaxis muy amigable y al ser orientada a objetos, facilitó el desarrollo, edición y mantenimiento del software que forma parte del prototipo.
- La telefonía IP representa una gran cantidad de posibilidades en lo que a aplicaciones corresponde, interacciones con otros servicios y automatización de tareas, son algunas de ellas. Además, la telefonía IP permite que dispositivos, hoy en día muy comunes,

como celulares, tablets o computadores, puedan comportarse como terminales telefónicos.

Recomendaciones

- El servidor de telefonía IP Asterisk no tiene un sistema sofisticado de sentencias de control y definición de funciones, por lo que se dificulta de sobremanera la implementación de tareas automáticas complejas. Por lo tanto, es altamente recomendable apoyarse en scripts que puedan llevar de mejor manera la toma de decisiones y el flujo de la información.
- Para evitar inconvenientes con servicios o programas complejos, se recomienda instalarlos y configurarlos en una máquina virtual con cualquier sistema operativo basado en Debian, como Ubuntu por ejemplo, antes de instalar y configurar en el ordenador de placa reducida Raspberry PI.
- Sería conveniente, para trabajos futuros, determinar el número máximo de llamadas telefónicas simultáneas que soporta el ordenador de placa reducida Raspberry PI sin comprometer la calidad de la llamada. Si existe un número significativo de llamadas máximas soportadas, este proyecto de titulación podría extenderse a una pequeña o incluso a una mediana empresa.
- Incluir en el software mecanismos para evitar posibles errores, mediante excepciones, y que guarden información de la causa de estos errores en archivos de log cuando el hardware u otro componente del sistema falle. Esto ayudará al diagnóstico y posterior mantenimiento del proyecto.

Es muy importante tener conocimientos sobre el manejo y administración del sistema operativo Linux. Puesto que, Raspbian, el sistema operativo por excelencia de Raspberry PI, es una distribución de Linux.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] D. S. Arias, “Domotica: diseno de una casa inteligente basado en la tecnologia Jini”, 18-may-2004. [En línea]. Disponible en: http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/sanchez_a_d/. [Consultado: 04-mar-2018].

[2] “Domótica”, *Wikipedia, la enciclopedia libre*. 26-feb-2018.

[3] R.- ASALE, “domótico, ca”, *Diccionario de la lengua española*. [En línea]. Disponible en: http://dle.rae.es/?id=E7W0v9b. [Consultado: 04-mar-2018].

[4] G. Morales, “La domótica como herramienta para un mejor confort, seguridad y ahorro energético”, *Cienc. E Ing.*, vol. 0, núm. 0, pp. 39–42, 2011.

[5] A. Recuero, “Estado actual y perspectivas de la domótica”, *Inf. Construccion*, vol. 50, feb. 1999.

[6] “Network service”, *Wikipedia*. 26-dic-2017.

[7] “Arquitectura Cliente Servidor - EcuRed”. [En línea]. Disponible en: http://www.ecured.cu/Arquitectura_Cliente_Servidor. [Consultado: 04-mar-2018].

[8] “telefonía”, *TheFreeDictionary.com*. [En línea]. Disponible en: https://es.thefreedictionary.com/telefon%c3%ada. [Consultado: 04-mar-2018].

[9] “Teléfono fijo”, *Wikipedia, la enciclopedia libre*. 26-ene-2018.

[10] “Raspberry Pi 2 Model B”, *Raspberry Pi*. .

[11] “Arduino Uno Rev3”. [En línea]. Disponible en: https://store.arduino.cc/arduino-uno-rev3. [Consultado: 04-mar-2018].

[12] “Arduino Ethernet Shield 2”. [En línea]. Disponible en: https://store.arduino.cc/arduino-ethernet-shield-2. [Consultado: 04-mar-2018].

[13] “Diferencia entre sensor, transductor y captador”, *Canal Gestión Integrada*, 23-ene-2015. .

[14] “Debian -- Información sobre la versión de Debian ‘wheezy’”. [En línea]. Disponible en: https://www.debian.org/releases/wheezy/. [Consultado: 09-abr-2018].

[15] “LM35 Datasheet(PDF) - Texas Instruments”. [En línea]. Disponible en: http://www.alldatasheet.com/datasheet-pdf/pdf/517588/TI1/LM35.html. [Consultado: 09-abr-2018].

[16] “Sensor LM35: medir temperaturas con Arduino - Arduino Tutoriales”, *Aprendiendo Arduino desde Cero*, 21-ene-2016. .

[17] “HC-SR501 PIR Sensor Working, Pinout & Datasheet”. [En línea]. Disponible en: https://components101.com/hc-sr501-pir-sensor. [Consultado: 09-abr-2018].

[18] "PIR Sensor HC-SR501| Toko Arduino dan Raspberry Pi di Denpasar", *Bali Electro | Toko Arduino dan Raspberry Pi di Denpasar*. [En línea]. Disponible en: https://toko.bali-electro.com/gel-elektromagnetik/44-hc-sr501-pir-sensor.html. [Consultado: 09-abr-2018].

[19] "Sensor PIR HC-SR501 - Tienda Oficial MCU-Tronics". [En línea]. Disponible en: https://mcutronics.com.mx/sensor-pir-hc-sr501.html. [Consultado: 09-abr-2018].

[20] "MQ-2 CH4 Gas Sensor Datasheet | Parallax Inc". [En línea]. Disponible en: https://www.parallax.com/downloads/mq-2-ch4-gas-sensor-datasheet. [Consultado: 09-abr-2018].

[21] "MQ-2 Módulo Sensor de Gas Inflamable", *Geekbot Electronics*. .

[22] "Sensor de Gas MQ2 con Arduino UNO R3 HETPRO TUTORIALES", *HETPRO/TUTORIALES*, 06-ago-2014. .

[23] designthemes, "Detector de llama | Tienda y Tutoriales Arduino". .

[24] "H11AA1 Datasheet(PDF) - Fairchild Semiconductor". [En línea]. Disponible en: http://www.alldatasheet.es/datasheet-pdf/pdf/52722/FAIRCHILD/H11AA1.html. [Consultado: 10-abr-2018].

[25] "Optoacoplador H11AA1". [En línea]. Disponible en: http://www.electronicoscaldas.com/optoacopladores/430-optoacoplador-de-entrada-ac-h11aa1.html. [Consultado: 10-abr-2018].

[26] "Pololu - A4988 Stepper Motor Driver Carrier". [En línea]. Disponible en: https://www.pololu.com/product/1182. [Consultado: 10-abr-2018].

[27] "Lenguaje de Programación Ruby". [En línea]. Disponible en: https://www.ruby-lang.org/es/. [Consultado: 10-abr-2018].

[28] "Acerca de Ruby". [En línea]. Disponible en: https://www.ruby-lang.org/es/about/. [Consultado: 10-abr-2018].

[29] "Homepage", *Asterisk.org*. [En línea]. Disponible en: https://www.asterisk.org/home. [Consultado: 10-abr-2018].

[30] R. Rivest, "The MD5 Message-Digest Algorithm". [En línea]. Disponible en: https://tools.ietf.org/html/rfc1321. [Consultado: 10-abr-2018].

[31] "Festvox: Festival". [En línea]. Disponible en: http://festvox.org/festival/. [Consultado: 10-abr-2018].

[32] "HTML", *Documentación web de MDN*. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/HTML. [Consultado: 10-abr-2018].

[33] "CSS3", *Documentación web de MDN*. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/CSS/CSS3. [Consultado: 10-abr-2018].

[34]“JavaScript”, *Documentación web de MDN*. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/JavaScript. [Consultado: 10-abr-2018].
[35]“Sinatra”. [En línea]. Disponible en: http://sinatrarb.com/. [Consultado: 10-abr-2018].
[36]“The Postfix Home Page”. [En línea]. Disponible en: http://www.postfix.org/. [Consultado: 04-mar-2018].