Universidad de Cuenca

Facultad de Ingeniería

Carrera de Electrónica y Telecomunicaciones

Diseño e implementación de un protocolo de comunicación domótico para interacción entre sensores y actuadores

Trabajo de titulación previo a la obtención del título de Ingeniero en Electrónica y Telecomunicaciones.

Autores :

Raúl Benito Suquinagua Otavalo C.I. 0105441257

Edgar Osvaldo Muñoz Abad C.I. 0302450051

Director :

Ing. Darwin Fabián Astudillo Salinas, PhD C.I. 0103907036

Co-Director :

Ing. Luis Ismael Minchala Ávila, PhD C.I. 0301453486

Cuenca - Ecuador

2018

Resumen

En la actualidad, los escenarios donde los seres humanos habitan, son cada vez más inteligentes, las personas desean mejorar su confort incorporando elementos que le faciliten o hagan más agradable su estancia en el hogar. Por este motivo, el usuario requiere que estos elementos sean: fáciles, rápidos y seguros de integrar al entorno.

En este trabajo, se implementó un protocolo sobre la capa de aplicación del modelo TCP/IP. El protocolo permite la interacción entre los sensores y actuadores (nodos cliente) que se ejecutan en una red domótica. En cada capa se usa un protocolo específico: 1.) en la capa física se usa el estándar IEEE 802.11 y define aspectos básicos como la potencia del transmisor y la sensibilidad del receptor; 2.) en la segunda capa se usa CSMA/CA como método de acceso al medio, permitiendo que múltiples estaciones utilicen un mismo medio de transmisión; 3.) en la capa de red, la interacción entre los nodos está configurada en modo malla, en donde cada nodo usa un identificador de 32 *bits* basado en su dirección MAC, propio del protocolo; 4.) la capa de transporte es configurada utilizando TCP, para garantizar que los datos se entregarán de manera confiable; por último, 5.) en la capa de aplicación se utiliza el protocolo MQTT.

Para probar el rendimiento del protocolo desarrollado, se fabricaron prototipos de los nodos: actuador y sensor. También, se desarrolló una aplicación para dispositivos móviles basados en *Android*, de manera que el usuario pueda controlar y gestionar la red desde su dispositivo. Además, se plantea que el envío de mensajes del protocolo propuesto sea seguro y soporte ataques de carácter malicioso, para lo cual, se usa el algoritmo de encriptación AES-256.

Las ventajas principales del protocolo planteado con respecto a los protocolos destinados al área de la domótica son: 1.) la manera simple en que se realiza la transferencia de información utilizando el protocolo MQTT, minimizando la sobrecarga de la red; 2.) la fácil incorporación de los nodos a la red, utilizando la aplicación del dispositivo móvil; y, por último, 3.) la comunicación a largo alcance y escalabilidad en la red, gracias a que la topología usada permite que los nodos realicen multi-salto.

Palabras clave : AES, Asociación, MQTT, Domótica, Protocolo, Red de malla.

Abstract

Nowadays, the scenarios where human beings live are getting smarter. People want to improve their comfort by incorporating elements that facilitate or do more pleasant their stay. For this reason, the user requires that these elements be: easy, fast and safe to integrate into the environment.

In this work, a protocol was implemented on the application layer of the TCP / IP model. The protocol allows the interaction between sensors and actuators (client nodes) that run in a home automation network. A specific protocol is used in each layer: 1.) the IEEE 802.11 standard is used in the physical layer and defines basic aspects such as transmitter power and receiver sensitivity; 2.) in the second layer, CSMA / CA was used as medium access method, allowing multiple stations to use the same transmission medium; 3.) in the network layer, the interaction between the nodes is configured in mesh mode, where each node uses a 32-bit identifier based on its MAC address, typical of the protocol; 4.) the transport layer is configured using TCP, to ensure that the data is delivered reliably; finally, 5.) in the application layer the MQTT protocol is used.

To test the performance of the developed protocol, prototypes of the nodes were manufactured: actuator and sensor. An application was also developed for mobile devices based on Android, so that the user can control and manage the network from his device. In addition, it is proposed that the sending messages of the proposed protocol be secure and support attacks of a malicious nature, for which the AES-256 encryption algorithm is used.

The main advantages of the proposed protocol with respect to other protocols in the home automation area are: 1.) the simple way in which the information transfer is carried out through the MQTT protocol, minimizing the network overload; 2.) The easy incorporation of the nodes to the network using the mobile device application; and, finally, 3.) long-range communication and scalability in the network because the topology used allows the nodes to perform multi-hop

Keywords : AES, Association, Domotic, Protocol, Mesh network

Índice general

Índice de figuras

Índice de tablas

Cláusula de Propiedad Intelectual

Edgar Osvaldo Muñoz Abad, autor del trabajo de titulación "Diseño e implementación de un protocolo de comunicación domótico para interacción entre sensores y actuadores", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor.

Cuenca, 24 de octubre de 2018

Edgar Osvaldo Muñoz Abad

C.I: 030245005-1

Cláusula de Propiedad Intelectual

Raúl Benito Suquinagua Otavalo, autor del trabajo de titulación "Diseño e implementación de un protocolo de comunicación domótico para interacción entre sensores y actuadores", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor.

Cuenca, 24 de octubre de 2018

Raúl Benito Suquinagua Otavalo

C.I: 010544125-7

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Edgar Osvaldo Muñoz Abad en calidad de autor y titular de los derechos morales y patrimoniales del trabajo de titulación "Diseño e implementación de un protocolo de comunicación domótico para interacción entre sensores y actuadores" de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 24 de octubre de 2018

Edgar Osvaldo Muñoz Abad

C.I: 030245005-1

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Raúl Benito Suquinagua Otavalo en calidad de autor y titular de los derechos morales y patrimoniales del trabajo de titulación "Diseño e implementación de un protocolo de comunicación domótico para interacción entre sensores y actuadores" de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 24 de octubre de 2018

Raúl Benito Suquinagua Otavalo

C.I: 010544125-7

Dedicatoria

Dedicado con mucho cariño a mi mamá, Josefina, por todo su esfuerzo para ayudarme a alcanzar esta meta y su apoyo en toda mi vida, a pesar de las dificultades. A mi hermana Mónica por su cariño y alentarme siempre a continuar.

Raúl Suquinagua

Dedico este trabajo, y sobre todo el titulo que me permitirá conseguir, a dos mujeres que llevo siempre en mi mente y en mi corazón, las cuales son la razón de mi esfuerzo y dedicación, que ahora no están presentes; mi madre Lorgia que me dio la vida y mi abuelita Lola que me enseño a como vivirla.

A mi papá Oswaldo, a mi segunda mamá Alexandra, por sus esfuerzos, sus palabras y consejos que me han brindado para poder alcanzar esta meta. A mis hermanos; Kenal, Sandy, Genaro y David por su apoyo incondicional y motivación que me han brindado.

También quiero dedicar a mi abuelito Bernardo, por sus consejos y enseñanzas, a mi tía Rosario por su cariño y ayuda. A mis sobrinos Adrian, Asher y Megan que son mi motivación para seguir adelante. Finalmente, a mi novia, por siempre apoyarme para seguir adelante y luchar por mis sueños.

Edgar Muñoz

Agradecimientos

Concluir este propósito académico requiere mucha perseverancia y dedicación, no solo de parte nuestra como autores, sino también de personas que día a día contribuyeron en este largo camino, es por esto que deseamos expresarles nuestro sincero agradecimiento.

A Dios por habernos brindado la vida, la sabiduría y la inteligencia para poder culminar esta etapa.

A nuestra familia, amigos y compañeros por su apoyo incondicional en todas las etapas de esta carrera universitaria.

A todos los profesores que marcaron cada etapa de nuestra vida universitaria, en especial a los Ingenieros Fabián Astudillo e Ismael Minchala, por haber aceptado dirigir este trabajo, por su tiempo, ayuda y motivación; no solo para el desarrollo de este trabajo, sino también, por los conocimientos brindados en las aulas de clase.

Al Club de Robótica de la Universidad de Cuenca (CRUC) por permitirnos el uso de sus instrumentos y su espacio físico..

LOS AUTORES

Abreviaciones y Acrónimos

AES Advanced Encryption Standard. 5, 14, 15, 34
AP Access Point. 67, 72
API Application Programming Interface. 69

CA Collision Avoidance. 18, 19, 22
CSMA Channel Sense Multiple Access. 18, 19, 22

DoS Denial of Service. 14

GPIO General Purpose Input/Output. 24
GSM Global System Mobile. 19, 20

HTTP Hypertext Transfer Protocol. 22, 72

IoT Internet of Things. 12, 13, 19, 59
IP Internet Protocol. 5, 12, 13, 19, 22, 26, 71

JSON JavaScript Object Notation. 26

LAN Local Area Network. 4, 85

MAC Media Access Control. 71
MQTT Message Queue Telemetry Transport. 7, 12, 22, 25, 26, 31, 39, 44, 55, 59, 60, 68–70
MS Master Slave. 4

NIST National Institute of Standards and Technology. 14

OSI Open System Interconnection. 4, 17–19

PTP Point To Point. 4
pubsub Publish–subscribe. 11, 12

QoS Quality of Service. 45, 70

RADIUS Remote Authentication Dial-In User Service. 56
RGB Red Green Blue. 36, 37
RSSI Received Signal Strength Indicator. 27, 72
RTC Real Time Clock. 39, 65, 66

SSID Service Set Identifier. 27, 72
SSL Secure Sockets Layer. 13

TCP Transmission Control Protocol. 5, 12, 19, 22, 26, 50
TLS Transport Layer Security. 13
TP Transmission Power. 4

UDP User Datagram Protocol. 19

VoIP Voice Over Internet Protocol. 13

WiFi Wireless Fidelity. 24, 25, 27, 36, 39, 44, 47, 56, 60, 69, 72, 87
WSN Wireless Sensor Network. 7, 8, 10–13

Capítulo 1

Introducción

En este capítulo se identifica el problema y se justifica el motivo porque se ha elegido diseñar el protocolo sobre la capa de aplicación. Dentro del alcance que tendrá el trabajo, se especifica el escenario y las características del protocolo propuesto. Por ultimo, se plantea los objetivos tanto generales como específicos.

1.1. Identificación del problema

En la actualidad los escenarios donde los seres humanos habitan son cada vez más inteligentes, las personas desean mejorar su confort incorporando elementos que le faciliten o hagan más agradable su estancia. Por este motivo, el usuario requiere que estos elementos sean: fáciles, rápidos y seguros de integrar al entorno [7]. La tecnología permite potenciar el desarrollo de aplicaciones domóticas de calidad, que brindan comodidad a los usuarios. El continuo desarrollo de estas tecnologías provee soluciones fáciles, útiles y económicas; con las finalidades claras de garantizar el bienestar y la seguridad [8]. Existen varias empresas que proveen tecnología domótica, como lo es +SPACIO [9], ésta se especializa en el desarrollo de soluciones de control en viviendas y edificios; actualmente son un referente en el ámbito de la tecnología aplicada a los espacios arquitectónicos en España. Mientras que en Ecuador existen empresas como SODEL

[10] y SmartHomeQuito [11].

En este contexto, existen varios trabajos relacionados a sistemas domóticos, que hacen uso de un protocolo de comunicación en diferentes capas del modelo OSI. Los protocolos son entes que permiten normalizar la interacción entre todos los dispositivos y programas asociados; estos gestionan los intercambios de información. Los protocolos de comunicación diseñados y empleados en la actualidad en una red domótica de tipo bus son; X-10, LonWorks, ModBUS, EHS, BatiBus [12], de los cuales, los 2 primeros pertenecen a buses propietarios, mientras que los otros son abiertos. El protocolo Lonworks ha conseguido establecerse en el mercado domótico como una de las soluciones con más potencial para la automatización de viviendas, está basado en paquetes en el que no existe el concepto de cliente-servidor, además implementa las 7 capas basada en el modelo OSI [13]. Las tecnologías de comunicación que utilizan la mayoría de protocolos inalambricos son: Zigbee [14] y Z-wave [15]; los cuales tratan de simplificar la arquitectura para consolidar una red simple y de bajo consumo. Por ejemplo ZigBee se basa en el modelo OSI, no utiliza las 7 capas del estándar de red, sino solamente 4.

Los protocolos propietarios necesitan un conjunto de chips patentados y componentes electrónicos especiales para su implementación. Además, la instalación requiere mano de obra especializada [13]. El protocolo EIB [16] distribuye la inteligencia entre los sensores y actuadores haciendo que éstos sean complejos y costosos. KNX [17] es la unión de lo mejor de los protocolos EIB, EHS y del BatiBUS lo cual aumenta considerablemente la oferta del mercado residencial al ser un protocolo que trabaja con dispositivos de diferentes fabricantes. Por otro lado, CEBUS [18] es un protocolo no propietario muy robusto y se implementa sobre las 5 capas del modelo OSI haciendo que los nodos sean complejos. También existe BACnet[19] que se diseñó para admitir varias capas físicas y de enlace diferentes, que incluyen: comunicaciones inalámbricas PTP y alámbicas de tipo LAN cableada de bajo costo MS/TP.

Si bien existen varios protocolos destinados al ámbito de la domótica, éstos a menudo utilizan la mayoría de las capas de modelo OSI, por lo que aumenta su complejidad. Existen muy pocos trabajos relacionados a protocolos sobre la capa de aplicación, la ventaja de trabajar sobre esta capa es que la red ya está formada y no hay que preocuparse por la gestión y administración de la misma; haciendo que el protocolo implementado en la última capa sea simple, pero al mismo tiempo debe ser lo suficientemente robusta en cuanto a calidad de servicio, funcionalidad y requerimientos de seguridad.

1.2. Justificación

El origen de la domótica se remonta a los años 70 cuando aparecieron los dispositivos de automatización de edificios. Desde entonces, se ha manifestado un creciente interés por parte de investigadores y la industria por la búsqueda de la casa ideal. Dentro de las principales servicios que ofrece la domótica se tiene: programación y ahorro energético, *confort*, comunicaciones, accesibilidad. Las arquitecturas típicas son: centralizada, distribuida y mixta. La arquitectura mixta más común está basada en *ZigBee* y son totalmente inalámbricos. La gran mayoría de protocolos usados en demótica son propietarios.

Hasta el momento se han realizado implementaciones sobre protocolos propietarios, con arquitecturas centralizadas donde el nodo central administra toda la red y todos los esclavos deben llegar a éste de manera directa, arquitecturas distribuidas donde los nodos son muy complejos por ende tienen un alto costo. Actualmente existen muy pocos estudios sobre la implementación de un protocolo en la capa de aplicación para una red de malla de sensores y actuadores en un entorno domótico de muy fácil integración y con un muy bajo costo.

1.3. Alcance

La contribución de este trabajo se centra, en el desarrollo e implementación de un protocolo sobre la capa de aplicación del modelo TCP/IP, que permita la interacción entre los sensores y actuadores (nodos cliente) que se ejecutan en una red domótica. Para lograr esto, tanto el *Gateway* (nodo central) como los nodos cliente, están conectados a una red en malla, dado que este tipo de redes tienen la característica de hacer multi-salto y por este motivo es una red ideal debido a su largo alcance y escalabilidad.

Se propone un escenario en que los nodos sensores y los dispositivos encargados de abstraer la interacción hombre máquina (teléfonos inteligentes) comandan cargas residenciales que básicamente son de control *ON/OFF*, estas cargas pueden ser: elementos de iluminación, motores eléctricos (para el control de puertas, ventanas, cortinas, etc), electrodomésticos, dispositivos de entretenimiento (televisión, radio, etc) y más.

Se plantea que el protocolo propuesto sea lo suficientemente seguro, para soportar ataques de carácter malicioso. Por lo cual, el protocolo envía todos los mensajes utilizando el tipo de encriptación denominada estándar de encriptación avanzada (AES, por sus siglas en inglés), de esta manera el sistema domótico tiene resistencia frente a ataques cripto-analíticos, en particular, al criptoanálisis lineal y diferencial [20].

Por otro lado, se plantea una interfaz humano-máquina amigable, de manera que, el usuario esté informado del estado de su vivienda y pueda gestionar los recursos con facilidad. Para llevar a cabo dicho objetivo, se diseña una aplicación para teléfonos inteligentes, esta aplicación se encargará de abstraer la información extraída de la base de datos del *Gateway* y mostrarla al usuario de una manera visualmente agradable y fácil de utilizar.

1.4. Objetivos

1.4.1. Objetivo general

Diseñar e implementar un protocolo que permita facilitar la integración de sensores y actuadores en un entorno domótico.

1.4.2. Objetivos específicos

- Diseñar un prototipo de sensor para control *ON/OFF*.
- Diseñar un prototipo de actuador para comando de motores, cargas eléctricas e iluminación
- Diseñar e implementar un protocolo a nivel aplicativo para interacción entre actuadores y sensores.
- Diseñar e implementar un método para la asociación de nodos sensores y actuadores a la red domótica tomando en cuenta criterios de seguridad.
- Diseñar e implementar una interfaz humano-máquina de entorno amigable para el usuario.
- Diseñar e implementar simulación de presencia.

Capítulo 2

Marco Teórico

En este capítulo se presenta una visión general de los sistemas domóticos y sus aplicaciones. Se describe la tipología de estos sistemas (centralizados, descentralizados y distribuidos); y el funcionamiento de una WSN, ya que es el tipo de red que se utiliza para este trabajo, debido a su bajo coste computacional. También se trata el funcionamiento del protocolo de comunicación MQTT, la seguridad en los dispositivos inalámbricos y diferentes métodos de encriptación. Por último se da una breve descripción de lo que es la simulación de presencia.

2.1. Introducción

Las tecnologías de la información y la comunicación se extienden a lo largo de nuestra vida para facilitar las tareas cotidianas y aumentar la calidad de nuestra vida en todos los ámbitos [21]. Al igual que en otros dominios, en los entornos domésticos muchos dispositivos se están equipando con diferentes protocolos de comunicación conectados a redes domésticas. La domótica representa un conjunto de aplicaciones domésticas de tecnologías modernas de los campos de la arquitectura, la construcción, la electrónica, la automatización, las telecomunicaciones y la informática. Estas aplicaciones tienen como objetivo la gestión de instalaciones de edificios, asegurando una reducción y optimización del consumo energético y también algunos servicios

como: seguridad, comodidad e intercambio de información, estas propiedades se reducen a una sola que es el común denominador en la sociedad: mejorar la calidad de vida [22].

Hoy en día, cuando se habla de domótica, se asocia inmediatamente con el término control remoto, utilizado para cualquier tipo de proceso. En el caso de una vivienda inteligente basada en protocolos de comunicación, sus ocupantes pueden controlar desde una computadora, un celular, relojes inteligentes; elementos como los sistemas de iluminación, climatización, así como los distintos electrodomésticos. La flexibilidad de este tipo de control y operación de los dispositivos de la red domótica depende de la arquitectura empleada, ya que para una aplicación determinada se encontrarán ciertas características que cumplan a cabalidad con una arquitectura específica [22]. Por otra parte, independientemente del tipo de arquitectura utilizada, los dispositivos integrados en los elementos del hogar presentan restricciones computacionales y de escalabilidad, en este sentido, en la literatura existen varias soluciones utilizando redes de sensores (WSN).

En [23], las WSN permiten el desarrollo de redes con gran número de dispositivos conectados y es posible adaptar parámetros como la tasa de transmisión, potencia, modulación, etc; de tal manera que los dispositivos finales, llamados nodos tengan un rendimiento y conectividad aceptable utilizando un mínimo de recursos. Pero al ahorrar recursos, no quiere decir que la información transmitida será insegura, por esto, surgen varios tipos de encriptación de datos, como se verá en el transcurso de este capítulo.

2.2. Tipología de los sistemas

Los sistemas domóticos pueden ser clasificados según su tipología dependiendo de la ubicación, espacio, características y distribución de los distintos dispositivos que serán parte del sistema [13].

2.2.1. Sistemas centralizados

En este tipo de sistemas se tiene una topología tipo estrella (ver Figura 2.1), es decir, que tiene un único elemento para controlar a los diferentes dispositivos que están conectados, por lo tanto, si este elemento centralizado falla o interrumpe su funcionamiento, todo el sistema deja de funcionar en su totalidad [22]. En aplicaciones críticas o susceptibles al colapso del elemento central se recomienda redundancia.

Las principales características que tiene este sistema son: costo reducido en la implementación,

fácil uso y formación e instalación sencilla. Uno de los principales inconvenientes es su escalabilidad, es decir, que al llegar a un cierto punto, el sistema no podría soportar más dispositivos conectados.

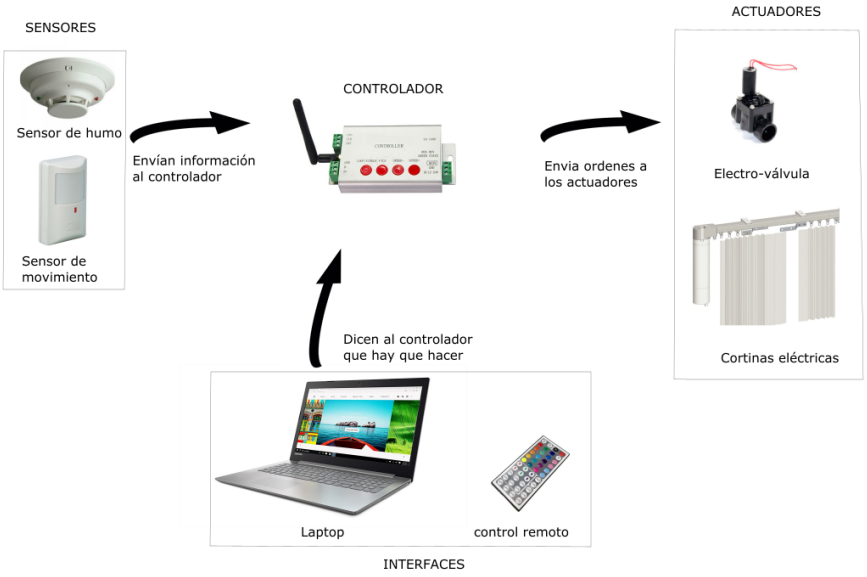

Figura 2.1: Arquitectura centralizada

2.2.2. Sistemas descentralizados

En este tipo de sistemas a diferencia del anterior, no existe un módulo central de control, por lo que todos los componentes del sistema tienen independencia en las funciones de control y mando. Entre las ventajas de este sistema esta su topología robusta, seguridad de funcionamiento, posibilidad de rediseño de la red y facilidad de ampliar la red. Pero presenta inconvenientes como por ejemplo: los elementos de la red no son universales, el costo de implementación es elevado y tiene una alta complejidad de programación[13].

2.2.3. Sistemas distribuidos

En este tipo de sistemas las funciones se reparten por áreas de trabajo diferentes que realizan tareas de forma coordinada para lograr a cabo su objetivo final. Es por eso que la arquitectura distribuida tiene los elementos de control generalmente cerca al dispositivo que se requiere comandar [22], la ventaja es que al colapsar un elemento de control, puede seguir funcionando el sistema sin disminuir su funcionalidad, dotándolo de gran flexibilidad.

La desventaja principal del sistema, es que es más compleja en cuanto a la programación. Sin embargo, el sistema presenta beneficios como: la posibilidad de rediseño de la red, facilidad de ampliación, sensores y actuadores universales.

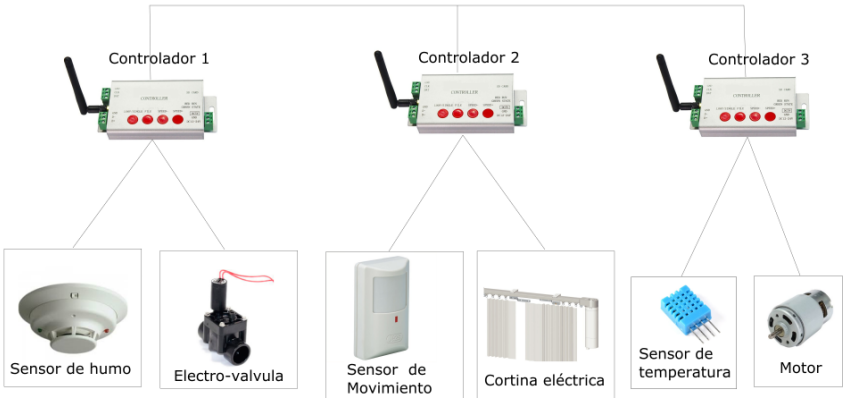

Figura 2.2: Arquitectura distribuida

2.3. Redes de sensores inalámbricos (WSN)

Las tecnologías inalámbricas permiten comunicar lugares sin la necesidad de una conexión por cable. En la implementación de estas nuevas tecnologías, el uso de WSN se ha vuelto un punto clave [24]. Las WSN se definen como sensores distribuidos que recolectan datos de los sensores y envían esta información hasta el servidor o un nodo central de manera inalámbrica. Los recientes avances en microelectrónica y sistemas inalámbricos han permitido el desarrollo de nodos de bajo costo, tamaño reducido y bajo consumo [25]. Se pueden identificar una serie de características principales de las WSN [25]:

- **Gran escalabilidad.** La cantidad de nodos que se despliega en una red puede llegar hasta los miles, pudiendo crecer su número.
- **Topología variable.** La posición en que se coloca cada nodo puede ser arbitraria y normalmente es desconocida por los otros nodos, lo que va a permitir un despliegue aleatorio.
- **Cooperación de nodos sensores:** Realizan operaciones simples antes de transmitir los datos, lo que se denomina un procesamiento parcial o local.
- **Comunicación:** Los nodos sensores usan comunicación por difusión y debido a que están densamente desplegados, los vecinos están muy cerca unos de otros y la comunicación *multi-hop* (salto múltiple de uno a otro) consigue un menor consumo de potencia.

En la Figura 2.3 se muestra los elementos básicos que componen de forma general una WSN [25], los cuales son:

- **Sensores**: toman la información del medio y las convierten en señales eléctricas.
- **Nodos sensores**: son los procesadores de radio, que toman los datos de los sensores a través de sus puertos y envían la información a la estación base.
- **Gateway:** nodo que actúa de interfaz de conexión entre la WSN y otra red.
- **Estaciones base**: Recolector de datos basado en un ordenador común o sistema integrado.

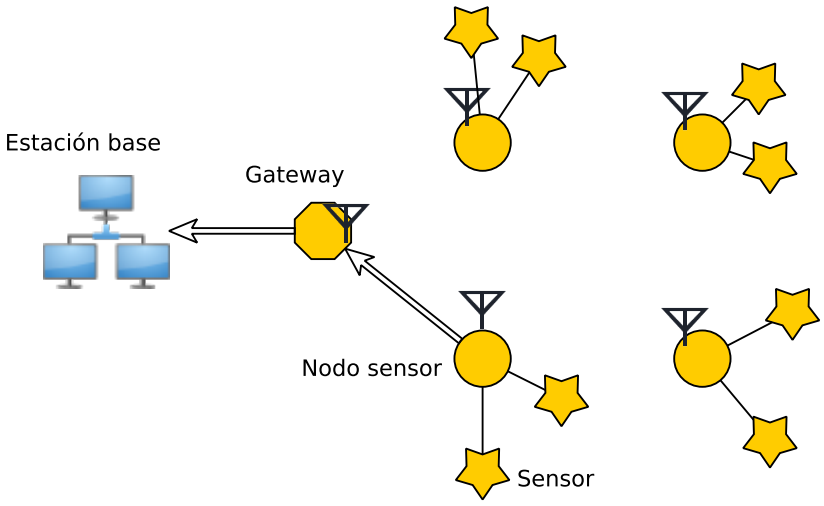

Figura 2.3: Elementos de una WSN básica.

2.4. Protocolo de comunicación MQTT

Generalmente las redes de sensores inalámbricas tienen capacidades de comunicación y de energía limitadas, lo que hace que no sea factible que los nodos entreguen datos directamente a un gran número de clientes externos. Estos factores indican que los nodos necesitan realizar la transferencia de datos lo más simple posible para minimizar la sobrecarga. Una solución a esto es la arquitectura *publish-subscribe* (pubsub). Este tipo de arquitectura usa un servidor intermediario llamado *broker* para manejar las solicitudes del cliente. La Figura 2.4 muestra el diseño general de un sistema pubsub. Los datos están organizados a través de tópicos o temas en el *broker*. Los tópicos se estructuran en una jerarquía similar a las carpetas y archivos en un sistema de archivos utilizando la barra diagonal (/) como delimitador. Usando este sistema

puede crear estructuras de nombres fáciles de usar y auto-descriptivos de su propia elección. Cuando un cliente publica contenido nuevo sobre un tema, los clientes suscritos al tema reciben una notificación con el nuevo contenido [1]. La Figura 2.5 presenta el patrón de publicación/-suscripción para MQTT. La ventaja del uso de un intermediario (*broker*) es que permite que los nodos consoliden el tráfico de red en la WSN enviando únicamente los datos a un punto final, simplificando las comunicaciones. Los sistemas pubsub que utilizan el protocolo heredado MQTT están ampliamente implementados en las aplicaciones IoT.

MQTT [26] es un protocolo de publicación y suscripción desarrollado por IBM [1]. Está diseñado para entornos restringidos, aunque su dependencia del protocolo orientado a la conexión (TCP) y la naturaleza compleja puede limitar su viabilidad para su uso en WSN. MQTT conecta las redes y dispositivos con *middleware* y aplicaciones. Es posible realizar conexiones tanto: máquina a servidor (M2S), como servidor a servidor (S2S), máquina a máquina; y varios tipos de enrutamiento (uno a muchos, uno-a-uno y muchos-a-muchos).

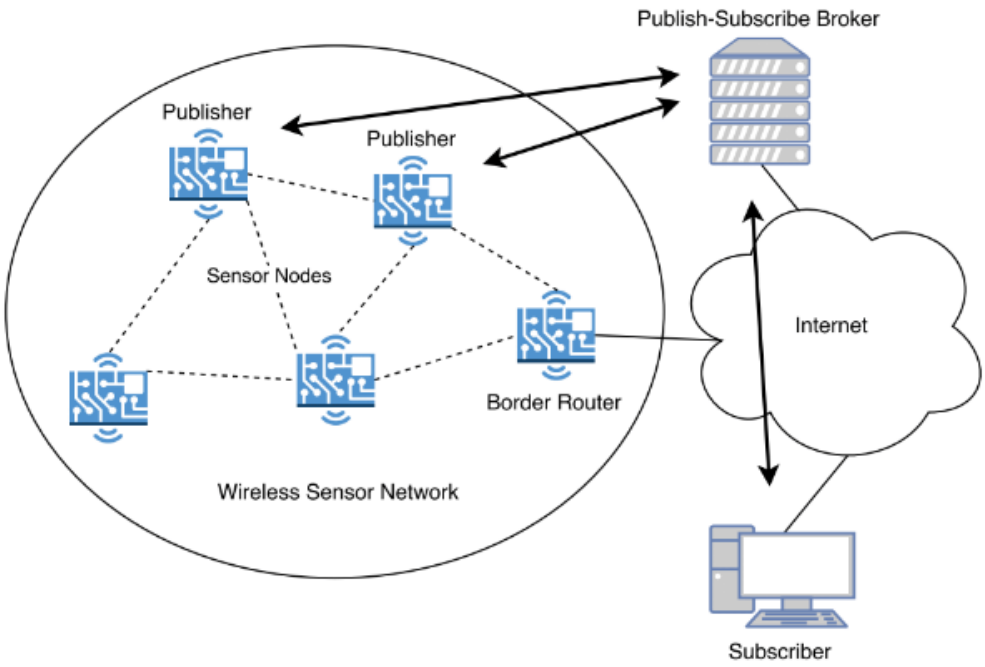

Figura 2.4: Arquitectura usada para publicar desde una red de sensores inalámbricos [1].

El puerto MQTT predeterminado para TCP/IP es el 1883. MQTT tiene diferentes tipos de servidores como: mosquito, hivemq, mosca y paho MQTT. También soporta *Transport Layer*

[1] https://www.ibm.com

Security (TLS) / *Secure Sockets Layer* (SSL), que son protocolos de seguridad en las comunicaciones que se utilizan en diferentes aplicaciones como correo electrónico, navegación web, fax por Internet, voz sobre IP (VoIP) y mensajería instantánea [27].

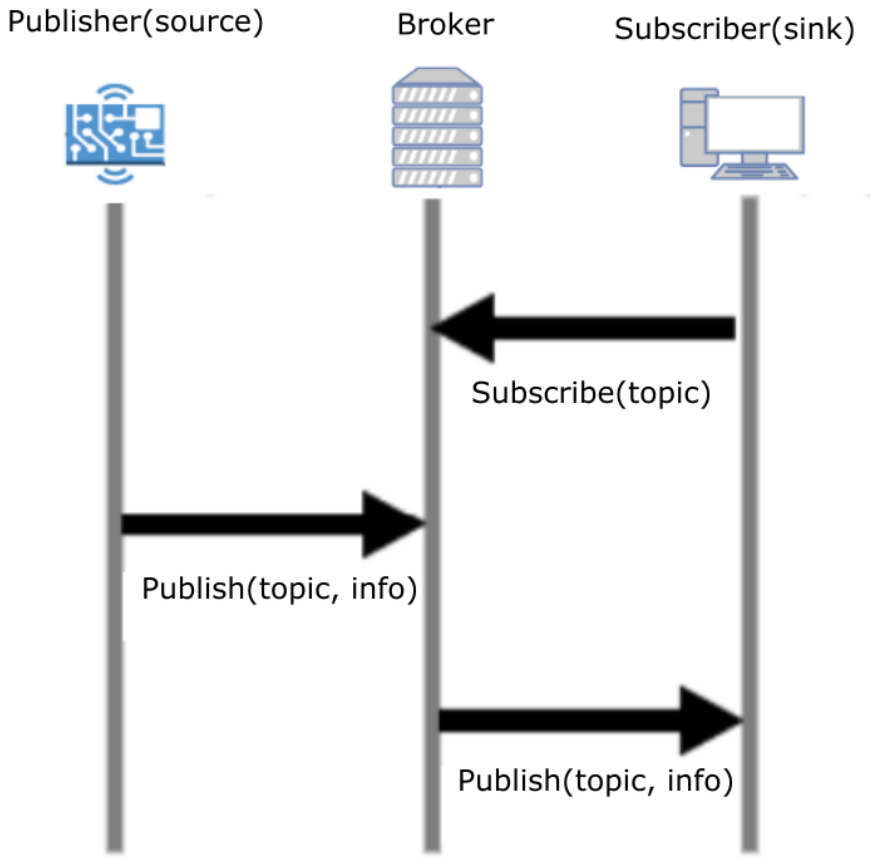

Figura 2.5: Proceso de publicar y subscribir en MQTT.

2.5. Seguridad

Con el rápido desarrollo de las comunicaciones por cable e inalámbricas, incluidas *Internet of Things* (IoT) y la red de sensores inalámbricos (WSN), la seguridad de la información se ha convertido en un problema cada vez más importante [28]. La seguridad en dispositivos inalámbricos e IoT tiene varios problemas, que incluyen confidencialidad de datos, comunicación segura con integridad y autenticidad.

Las principales amenazas que se deben de considerar al desarrollar aplicaciones IoT son:

- Acceso remoto: esta amenaza intenta acceder al dispositivo sin permiso.
- *Man-in-the middle*: ocurre cuando un atacante obtiene acceso al canal de comunicación del dispositivo. El atacante puede realizar actividades no autorizadas, como interceptar datos o modificar comunicaciones para cambiar la misión del dispositivo.
- *spoofing*: ocurre cuando un atacante crea paquetes con una dirección de origen falso para ocultar su identidad.
- *Spyware*: todo tipo de software que monitorea y roba información de dispositivos.
- Inyección de servicio: el atacante apunta a un servicio del dispositivo e inyecta código malicioso para corromper el servicio.
- *Malware*: se refiere a todo tipo de códigos de software, como son los virus, gusanos y troyanos; que están hechos para hacer operaciones maliciosas.
- *Ransomware*: cualquier tipo de software que bloquea un dispositivo y exige algún tipo de pago para desbloquearlo.
- Denegación de servicio (DoS): un atacante ocupa los recursos de la red inundándolo con solicitudes consecutivas, y por lo tanto, colapsando los dispositivos.

Una solución para enfrentar los problemas expuestos es el cifrado de datos, esto significa alterarlos, generalmente mediante el uso de una clave, de modo que no sean legibles para quienes no posean dicha clave. Luego, a través del proceso de descifrado, aquellos que sí poseen la clave podrán utilizarla para obtener la información original. Ésta es una técnica muy utilizada para evitar los problemas de seguridad. Existen muchos métodos de encriptación, como por ejemplo: AES, CAPE, ARIA, PRESENT, LEA y más [29–31]

AES: es un algoritmo de encriptación desarrollado por el Instituto Nacional de Estándares y Tecnología (NIST). Utiliza el algoritmo Rinjndael y el 26 de noviembre de 2001, la Publicación 197 de las Normas federales de procesamiento de la información lo nombró como el nuevo estándar para el cifrado. Este estándar se llamó AES *Advanced Encryption Standard* y actualmente sigue siendo el estándar para el cifrado [32].

AES es un cifrado de bloque simétrico, es decir, que usa la misma clave para cifrado y descifrado. El estándar establece que el algoritmo sólo puede aceptar un tamaño de bloque mínimo de 128 bits y las claves de cifrado podrían ser: AES-128, AES-192 o AES-256. Además no se utiliza la estructura *feistel* [33], ya que en *feistel* la mitad del bloque de datos se usa para modificar la otra mitad del bloque de datos y luego se intercambian las mitades. En el caso de AES, el bloque completo de datos se procesa en paralelo durante cada ronda usando sustituciones y permutaciones.

CAPE: implementa clave privada y cifrado de flujo simétrico basado en xor[2] desarrollado para ofrecer cifrado en microcontroladores con capacidades limitadas. Es un proyecto experimental, destinado para fines educativos y de investigación; no se lo recomienda

[2]Operación booleana

para la producción [34].

A pesar de utilizar pocos recursos de procesamiento, es un tipo de encriptación débil, ya que no ha sido diseñado teniendo en cuenta el principio de Kerckhoff (uno debe diseñar sistemas bajo la suposición de que el enemigo se familiarizará plenamente con ellos) [35], ni se ha considerado la vulnerabilidad conocida de texto sin formato (tiempo de fuerza bruta sobre el texto sin formato).

ARIA: es un tipo de cifrado de bloques diseñado en 2003 por un grupo de investigadores Coreanos. Es un algoritmo estándar seleccionado por la Agencia Coreana de Tecnología y Estándares. ARIA utiliza una estructura de red de sustitución-permutación basada en AES. La interfaz tiene un tamaño de bloque de 128 bits con un tamaño de clave de 128, 192 o 256 bits [36].

2.6. Simulación de presencia

La simulación de presencia surge como un mecanismo de seguridad para persuadir a los ladrones. La estrategia consiste en que la actitud de un intruso que desea ingresar en la casa puede verse afectada al ver que hay actividad y movimiento dentro de la vivienda, este hecho puede hacerle abandonar el intento de intrusión [37]. Existen varios mecanismos que se han desarrollado para resolver la problemática, algunos se muestran en el apartado 3.4.

Capítulo 3

Estado del Arte

Este capítulo presenta un resumen de los protocolos tipo bus e inalámbricos existentes aplicados a la domótica y se analiza las diferentes implementaciones desarrolladas hasta el momento.

3.1. Introducción

Existen varios trabajos relacionados a redes domóticas, que hacen uso de un protocolo de comunicación en diferentes capas del modelo OSI, algunos propietarios y otros de libre acceso. En general existe una continua evolución de la tecnología, la Figura 3.1 muestra los protocolos existentes y su evolución.

A continuación, se muestran algunos de los enfoques y características de los protocolos existentes para entornos domóticos.

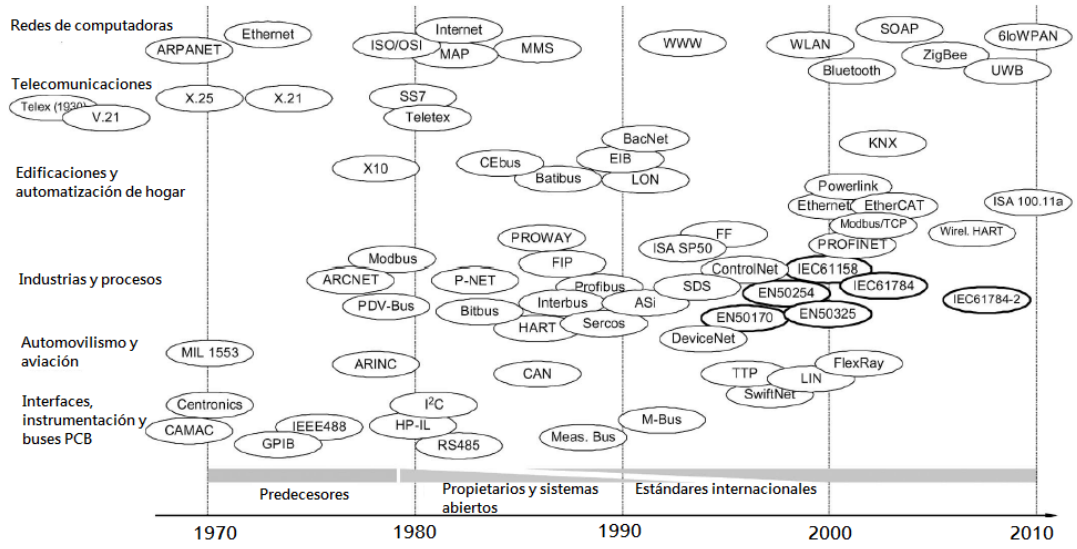

Figura 3.1: Evolución de los principales protocolos y tecnologías habilitantes [2].

3.2. Protocolos tipo bus

Los protocolos de comunicación diseñados y empleados en la actualidad en una red domótica de tipo bus son; X-10 [38], LonWorks, ModBus [39], EHS y BatiBus [40]; los 2 primeros pertenecen a buses propietarios, mientras que los otros son abiertos. El protocolo Lonworks se ha establecido en el mercado domótico como una de las soluciones con más potencial para la automatización de viviendas, presenta un control descentralizado permitiendo distribuir la inteligencia entre sensores y actuadores. La comunicación es de tipo bus, habitualmente el medio de que se utiliza par trenzado aunque existen otros; además es un sistema que permite la interconexión entre aparatos de diferentes fabricantes con poco cableado, flexible, fácilmente ampliable, mediante un protocolo estándar e implementa las 7 capas basadas en el modelo OSI [41]. Por otro lado, existen protocolos de libre acceso, como por ejemplo *BatiBus*, la característica de este protocolo es que soporta varias topologías como: estrella, anillo y árbol; todas las cuales exhiben los beneficios de una configuración simple y barata; además, controla el flujo de datos y evita las colisiones con CSMA/CA (Channel Sense Multiple Access / Carrier Avoid) [40].

3.3. Protocolos para tecnologías inalámbricas

Los protocolos más utilizados para entornos domóticos de tipo inalámbrico son: Zigbee [42], Enocean [43] y Z-wave [44]; los cuales tratan de simplificar la arquitectura para consolidar una

red simple y de bajo consumo; resultando una buena solución para la introducción de la domótica en la vivienda construida, evitando problemas de obras, ductos y cableados. Por ejemplo; ZigBee se basa en el modelo OSI, no utiliza las 7 capas del estándar de red, sino solamente 4. Enocean trabaja bajo una frecuencia de 868MHz por lo que no existen problemas con la saturada banda de los 2.4GHz, la velocidad de transmisión es de 25 kbit/s, ideal para aplicaciones de baja tasa de datos; una ventaja es que permite que los sensores puedan trabajar sin activar la recepción radio constantemente, incrementando la eficiencia energética, la desventaja es que no cuenta con un estándar internacional. Zwave utiliza solamente 3 capas del modelo OSI (capa de aplicación, red y física); además, es un sistema fiable, permite una topología mallada que admite hasta 232 dispositivos con una distancia de 30 metros entre dispositivos al aire libre o 20 metros en espacios cerrados [3]. A continuación se muestra una comparativa de los protocolos mencionados.

	Fiabilidad	Seguridad	Emisión Baja frec	Uso simple	Precio	Protección Inversión	Interop
ZigBee	✓	✓	✓	–	No aún	–	✗
enocean	✗	✗	✓	✓		✓	✓
Z-Wave	✓	✓	✓	✓	No aún	✓	✓

Figura 3.2: Comparación entre Zigbee, enocean y Zwave [3].

3.4. Implementaciones desarrolladas

Los protocolos de comunicación dependiendo de las necesidades de operación pueden ser desarrollados sobre cualquier capa del modelo OSI. Por ejemplo [45] presenta una aplicación de IoT, en la que se diseña e implementa una red domótica para control y monitoreo, utilizando el protocolo TCP/IP para la transmisión de datos a través de Internet, mientras que para la seguridad se usa GSM (Global System Mobile) a través de mensajes cortos. En [46] se propone un protocolo que es capaz de resolver la mayoría de los problemas sin depender de los sitios de Internet o proveedores. Emplea mensajes sin procesar para ahorrar los recursos del CPU y utiliza UDP para un uso eficiente del ancho de banda . Otro trabajo es [47], en donde se analiza el estándar IEEE 802.15.4 (Zigbee) y su fiabilidad en entornos domóticos; se usa un protocolo simple y ligero basado en el método de acceso al medio CSMA/CA. Una de las ventajas es que los paquetes individuales son reconocidos y se proporciona garantías de entrega a nivel de enlace; sin embargo, no hay garantías de calidad de servicio o soporte para niveles de prioridad de tráfico de red. Finalmente, [48] crea una red domótica de control, que conecta inalámbricamente

todos los nodos que formen parte del sistema, estableciéndose los parámetros de comunicación y de configuración; el sistema consta de un nodo de control de red y de otro nodo periférico, de tal manera que este último se pueda conectar y controlar elementos habituales en una vivienda; al mismo tiempo, la instalación es controlada por el usuario a través de una aplicación web, visualmente atractiva y de fácil manejo, alojada en la unidad central.

En cuanto a simulación de presencia, en [49] se propone un mecanismo para cambiar de forma aleatoria el estado de actuadores tal como, iluminarias, persianas y dispositivos del hogar en general, de manera aleatoria. Otro enfoque es [37], utiliza sensores de movimiento para realizar cambios en la actividad de los aparatos en el hogar y al mismo tiempo envía una alerta a los propietarios de la casa haciéndolos saber de la intrusión. En la industria existen varios dispositivos que ofrecen simular la presencia de un hogar, [50] aporta seguridad al simular presencia con pequeñas variaciones horarias cada día, para esto cuenta con sensores de luz y opciones de estaciones del año; además el equipo desarrollado se entrega pre-programado, pero en caso de ser necesario puede ser programado a pedido del cliente.

3.5. Conclusiones

La seguridad es muy importante debido a problemas de vulnerabilidad, como se evidencia en la sección 2.5. En [45] utiliza un mecanismo de seguridad utilizando GSM a través de mensajes cortos, pero únicamente para advertir a los usuarios sobre la actividad en el hogar. Sin embargo, los trabajos realizados hasta el momento, no toman demasiada importancia a la seguridad de los datos.

Además se ha visto que existen problemas de inter-operabilidad en varios trabajos y esto hace que no se pueda intercambiar información, ni procesarla. Como por ejemplo: Zigbee soporta redes malladas, pero no posee la capacidad de inter-operabilidad con otros sistemas pertenecientes a la red domótica.

Por último, Zwave soporta simulación de presencia como un mecanismo de seguridad, la activación de dispositivos generalmente se realiza de manera aleatoria o con temporizadores, y evidentemente no simula el comportamiento real de la red.

Capítulo 4

Diseño e implementación

El presente capítulo abarca la metodología empleada para enfrentar la problemática expuesta, mostrando las características de los dispositivos utilizados, diseño de los prototipos de los diferentes tipos de nodos de la red, esquema del protocolo propuesto, diseño de un mecanismo de simulación de presencia mediante redes neuronales y el diseño de la interfaz humano máquina.

4.1. Introducción

Como hemos visto en los Capítulos 2 y 3, existen varios enfoques y un sin número de trabajos relacionados. Las principales necesidades en el ámbito de la domótica con el fin de garantizar la automatización de tareas son: la seguridad, gestión energética y confort del usuario. Con el fin de suplir esa necesidades existe un desarrollo continuo de nuevas tecnologías y mecanismos; por consiguiente, el presente documento desarrolla un protocolo que suple las necesidades del entorno domótico, permitiendo asociar actuadores y sensores en una red tipo *mesh*.

4.2. Especificación de requerimientos del protocolo

En la Figura 4.1 se muestra de manera general la configuración del sistema, y los requerimientos implícitos de cada función del protocolo, los mismos que se describen a continuación:

- **Proporcionar comunicación entre los diferentes dispositivos**.- Esta funcionalidad es implementada sobre las capas del modelo TCP/IP. La primera es la capa de acceso a la red, utiliza CSMA / CA permitiendo que múltiples estaciones utilicen un mismo medio de transmisión y utiliza el estándar IEEE 802.11. La capa de enrutamiento es configurada en modo malla entre la red de nodos, en donde cada nodo tiene una identificación de acuerdo a su *chipId* de 32 *bits* propio del protocolo, el cual esta basado en su dirección MAC. La siguiente capa es la de transporte, es configurada utilizando TCP, para asegurar que los datos se entreguen de manera fiable. Por último, en la capa de aplicación se utiliza el protocolo MQTT, para lograr una eficiencia en la comunicación, debido a las restricciones de las redes de sensores.
- **Agregar nodos a la red**.- Se realiza mediante una configuración de cliente servidor, en donde inicialmente el nodo está en modo *access point* y el dispositivo móvil es el cliente; una vez realizada la configuración, el nodo deja de ser *access point* y se une a la red de sensores. Todo esto se realiza utilizando HTTP. Mientras para la eliminación del nodo se realiza automáticamente desde el servidor, es decir, se valida si algún nodo está en la base de datos pero no está en la red de malla, entonces se elimina.
- **Configurar emparejamiento de los nodos**.- Las configuraciones de emparejamiento incluyen; agregar un nuevo emparejamiento entre sensores y actuadores, cambiar un emparejamiento existente por uno nuevo y eliminar un emparejamiento existente. Todo esto se realiza mediante el protocolo MQTT.
- **Almacenar configuraciones y estados de los actuadores en una base de datos**.- Esta funcionalidad se realiza mediante MySQL[1], que está alojado en el servidor y almacena tanto las ubicaciones de los nodos, estados de los actuadores, emparejamiento y configuración de nodos.
- **Simular presencia**.- Para satisfacer este requerimiento se utiliza una red neuronal, que aprende el patrón de actividad del hogar mediante un entrenamiento con la información de la base de datos, luego al activar la opción simular presencia se replica el patrón aprendido.
- **Controlar el hogar de manera remota**.- Esta función se realiza mediante el dispositivo móvil, el cual realiza un descubrimiento de la dirección IP del servidor para poder acceder a la base de datos y mostrar el estado de los actuadores y realizar el control.

[1] https://www.mysql.com/

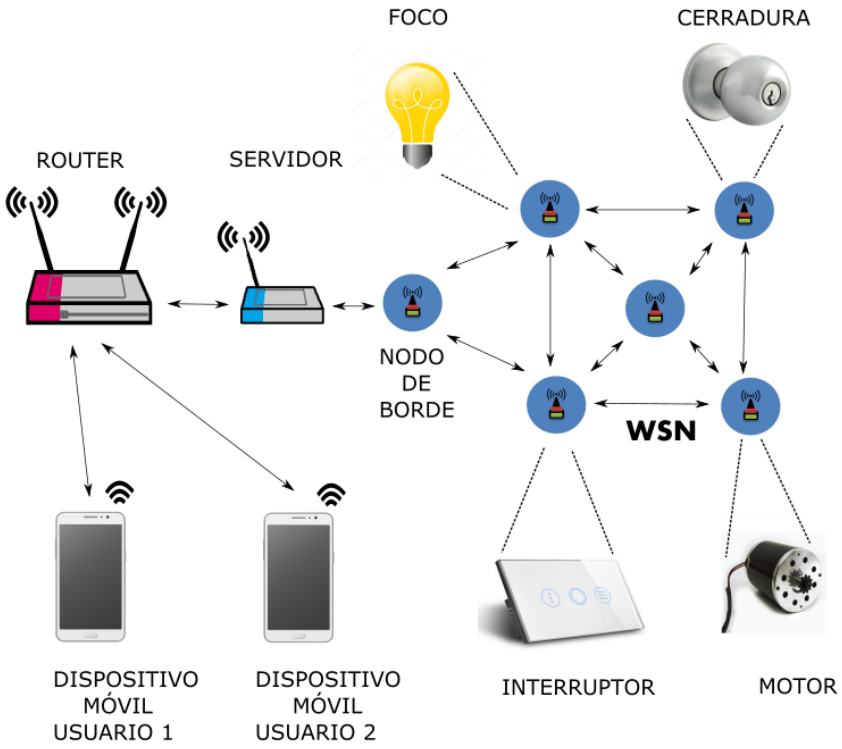

Figura 4.1: Esquema genera del prototipo propuesto.

4.3. Hardware del Prototipo

Para desplegar el sistema domótico descrito, es necesario *hardware* de diferentes capacidades. El que demanda mayor capacidad es el servidor, debido a que al menos debe permitir implementar una red neuronal. Mientras que para los nodos de la red de sensores, es necesario que se permita configurar una red de malla. Por lo tanto, a continuación se describe las características del *hardware* utilizado.

4.3.1. Raspberry Pi

La *Raspberry Pi* es una computadora del tamaño de una tarjeta de crédito, diseñada originalmente para educación y creada por la *Raspberry Pi Foundation* en Reino Unido. Gracias a su pequeño tamaño y precio accesible, fue rápidamente adoptado por fabricantes y entusiastas de la electrónica.

La Raspberry Pi es más lenta que una computadora portátil o de escritorio moderna, pero sigue

siendo una computadora Linux completa y puede proporcionar todas las capacidades esperadas que implica, con un bajo nivel de consumo de energía. En sus diferentes modelos está formado por un procesador ARM [2] modelo 28xx desarrollado por Broadcom[3]. En el Apéndice F se detallan las características del dispositivo.

La Raspberry Pi fue diseñada para trabajar con diferentes distribuciones Linux como por ejemplo: Aros, Android, Ubuntu Mate, Lubuntu y Fedora. Sin embargo, el sistema operativo recomendado es Raspbian [51], consistente en una distribución Linux basada en Debian y optimizada para la plataforma.

Figura 4.2: Tarjeta Raspberry Pi 3 modelo B+ [4]

4.3.2. Módulos Wifi

Los nodos de la red de sensores se implementan con módulos WiFi, específicamente utilizando módulos ESP8266 ESP-12. Mientras que para el *gateway* (nodo de borde de la red de sensores) se utiliza el módulo ESP8266 ESP-07, ya que esta cuenta con antena externa, necesario para tener una mayor área de cobertura. Estos módulos son soportados por el entorno Arduino, por tanto, permite usar las funciones y librerías conocidas, y ejecutarlas directamente en ESP8266 sin necesidad de un microcontrolador externo [52].

Además posee pines de entrada y salida de propósito general, 11 para el ESP8266 ESP-12 y 9 para el ESP-07. Esto permite realizar tareas de nivel medio / alto debido a que no existe restricciones de pines GPIO. En el Apéndice F están listadas algunas de las características de los módulos ESP-12 y ESP-07.

[2] Procesador con arquitectura diseñada para utilizar menor número de transistores, con direccionamiento de 32 y 64 bits.

[3] https://www.broadcom.com/

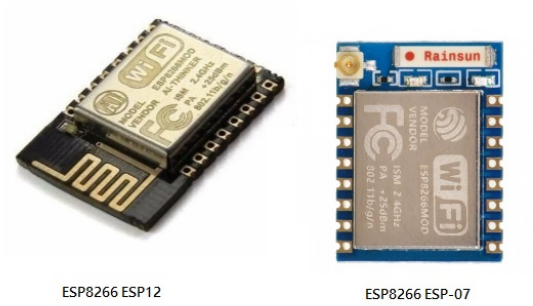

Figura 4.3: Módulos WiFi utilizados.

4.3.3. Sensores táctiles

Se utiliza específicamente el módulo TTP223B, es basado en un sensor capacitivo capaz de detectar cuando una parte del cuerpo toca el circuito. Básicamente el voltaje de alimentación es 5 Voltios y cada vez que toquemos el círculo blanco (ver Figura 4.4), pasará a nivel alto el pin SIG (señal). Pero al realizar unas pruebas de los sensores, se verificó que también puede trabajar con 3.3 voltios (mismo voltaje de alimentación de los módulos WiFi mencionados anteriormente). Estos sensores se usan para adquirir las señales de control para cambiar los estados de los actuadores del sistema domótico.

Figura 4.4: Sensor táctil.

4.4. Implementación del protocolo de comunicación

En esta Sección se presenta el desarrollo del protocolo propuesto y está organizada de la siguiente manera: se presenta una descripción general de la red en malla utilizada, la disposición de los dispositivos y como se unificó con el protocolo MQTT. Además, se explican los requerimientos

implícitos 4.2 de cada una de las funcionalidades del protocolo, como es el proceso para agregar nodos a la red domótica, configuración de emparejamientos y cuales son las tramas de mensajes utilizadas para el intercambio de información. Finalmente se explica los métodos utilizados para proveer seguridad al sistema.

4.4.1. Red en malla

La red en malla utilizada en el presente trabajo es basada en la librería [53]. Esta librería posibilita la creación de una verdadera red *ad-hoc*, lo que significa que no se requiere planificación ni un controlador central o enrutador. Cualquier sistema de uno o más nodos se auto-organizará en una malla completamente funcional.

Usa objetos JSON para todos sus mensajes debido a que el código y los mensajes son legibles, y posibilita la integración con los *front-ends* de javascript y las aplicaciones web.

No crea una red de nodos TCP / IP. En su lugar, cada uno de los nodos está identificado de forma única por su *chipID* de 32 *bits* [53]. Existen algunas limitaciones, como se lista a continuación:

- Restricciones en cuanto a la cantidad de mensajes enviados en multidifusión. Esto es para evitar que el hardware se sobrecargue.
- Capacidad de potencia y memoria del Esp8266 hace que sea fácil sobrecargar la malla y desestabilizarla.
- Perdida de mensajes debido al alto tráfico, no hay certeza que se entreguen todos los mensajes.

La última limitación no afecta al presente proyecto, ya que no existe una sobrecarga de tráfico, debido a que solamente se utiliza para control *ON/OFF* de los nodos actuadores.

4.4.2. Integración de la red de malla con MQTT

La configuración del *gateway* MQTT se basó en el trabajo desarrollado en [5], su función consiste en escuchar los mensajes del puerto serie, los retransmite a MQTT y viceversa. Se puede ejecutar en cualquier máquina Windows o Linux que admita *Node.js* [54]. En este caso, tanto el *broker* cómo el servidor *Node.js* fue implementado en la Raspberry pi (ver Figura 4.5). La instalación y configuración se puede ver en el Apéndice A

El proyecto [5] se adaptó para que la información de un tópico específico sea enviaba a todos

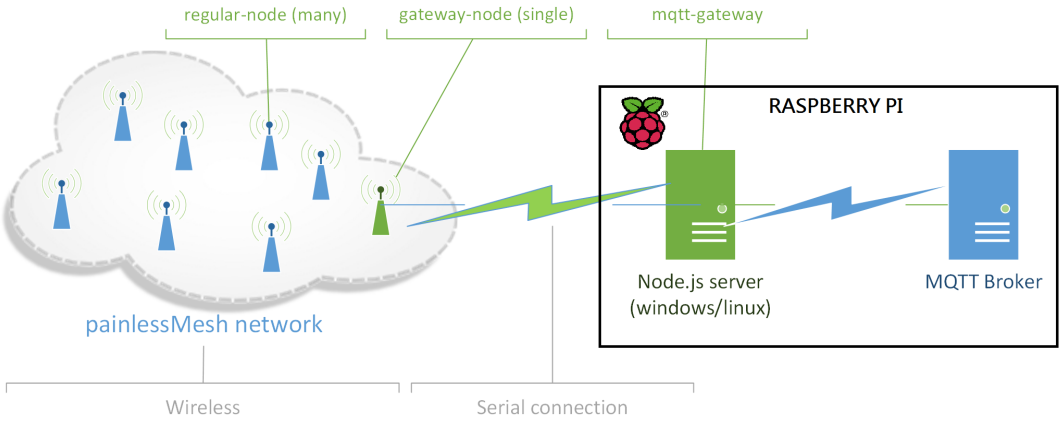

Figura 4.5: Estructura de la red en malla y comunicación MQTT [5].

los nodos, estén o no subscritos. Entonces se tuvo que validar ese inconveniente, se modificó las líneas de código en el servidor *Node.js*.

El tópico de los nodos de la *mesh* es la ubicación a la que pertenece cada uno. Cuando se agrega un nodo a la red, se configura con la ubicación, luego se almacena esta información en la base de datos presente en el servidor. Entonces cuando un nodo publica, se valida que se entregue la información solamente a los nodos pertenecientes a esa ubicación.

4.4.3. Incorporación de nodos a la red

Al abrir la aplicación del dispositivo móvil, se transfiere información desde el servidor (mediante al tópico "movil") como: nombre, identificación, ubicación de los nodos que pertenecen a la red y la lista de ubicaciones por defecto. Es necesario estos datos para validar la agregación de nodos (ver Figura 4.6).

Para agregar un nodo a la red en malla, el procedimiento es el siguiente: el nodo se establece como servidor en modo *access point* (1), con una SSID específica para cada uno. El dispositivo móvil realiza un escaneo de todas las redes WiFi disponibles, que tengan un SSID que indique que es un nodo (2). Los nodos son ordenados por nivel de RSSI(mayor a menor) y se escoje el primero (3). El dispositivo móvil envía un mensaje de inicialización al nodo escogido para encender un led indicador (4) y pregunta si el nodo que se escogió es el deseado (6); si es el nodo, entonces se realiza la transferencia de datos de configuración (nombre y ubicación del nodo; SSID, contraseña y puerto de la *mesh*; si es sensor, se solicita el número de *switch*) y si no, se escoge el siguiente nodo y se pregunta nuevamente si es el nodo deseado. Al finalizar la

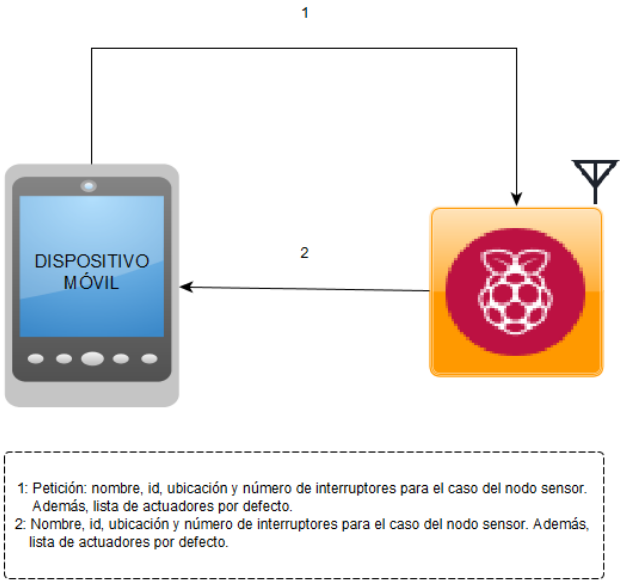

Figura 4.6: Transferencia de datos de inicialización servidor-dispositivo móvil.

configuración, el nodo deja de operar en modo *access point* y se une a la red en malla (9). Para más detalles se puede observar la Figura 4.7 y el Apéndice D.6.

Una vez que el nodo pertenece a la red en malla, existen dos casos:

1 **Nodo actuador:** Transfiere los datos de configuración a la base de datos, estos datos son: nombre, identificación y ubicación.
2 **Nodo sensor:** Transfiere los datos de configuración a la base de datos, estos datos son: nombre, identificación, ubicación y número de interruptores. Luego se pregunta si desea emparejar con algún actuador presente en la red en malla, y si es el caso, se solicita que actuador se desea controlar y se asigna un número de interruptor perteneciente al sensor para su emparejamiento. Finalmente se envía la configuración de emparejamiento a la base de datos y al nodo actuador de la red para validar los datos recibidos.

En la Figura 4.8 se puede explicar mediante un diagrama el proceso de incorporación y emparejamiento inicial.

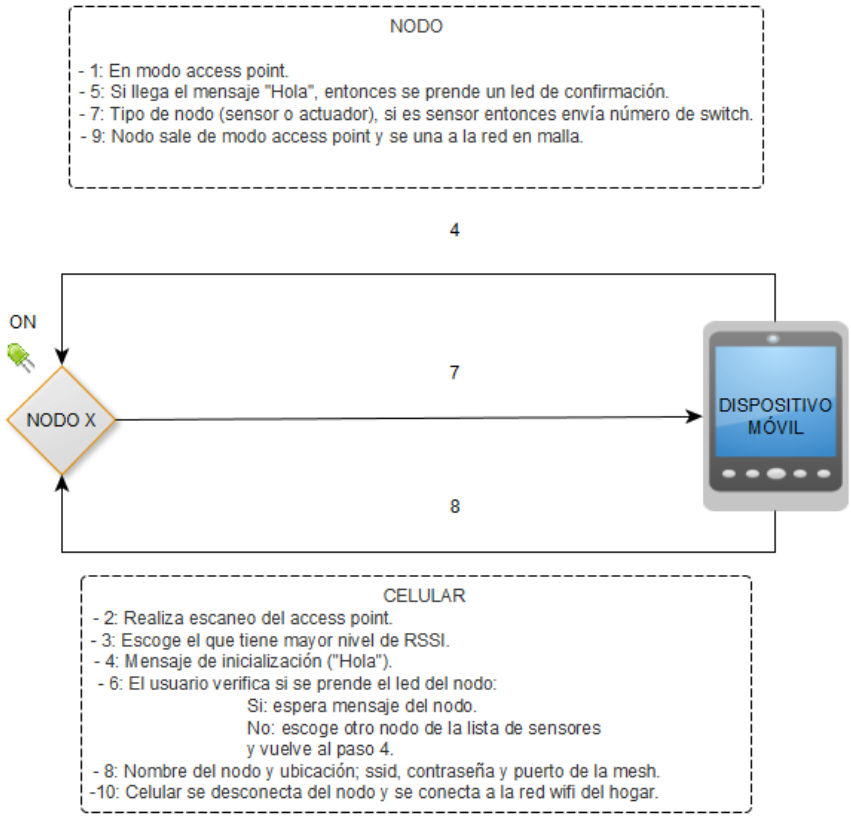

Figura 4.7: Procedimiento para agregar nodos a la red.

4.4.4. Configuración de emparejamientos

Inicialmente al ejecutar esta opción, se intercambian los siguientes mensajes entre el servidor y el dispositivo móvil para validar y configurar la interacción en la aplicación:

- Petición de las ubicaciones y código del hogar.
- Petición del nombre, identificación, ubicación y número de interruptores de los nodos sensores.
- Petición del nombre, identificación y ubicación de los nodos actuadores.
- Petición de los emparejamientos existentes.

Existen dos tipos de configuraciones de emparejamiento implementados: modificar y agregar.

Modificar

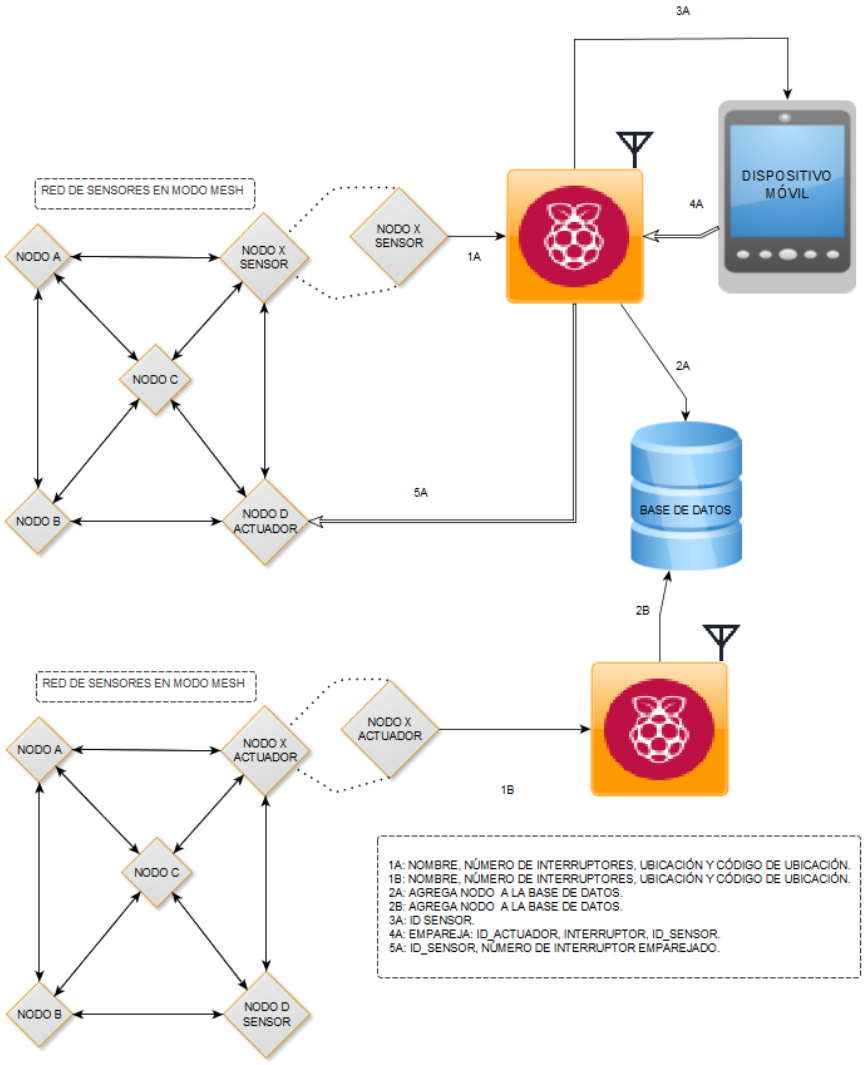

Figura 4.8: Agregación y emparejamiento inicial de los nodos.

La utilidad de esta función se da cuando se requiere modificar un emparejamiento que ya se ha establecido previamente. Por ejemplo, si la iluminación de una ubicación en específico se quiere controlar desde un punto diferente al que está configurado.

Después de realizar las validaciones automáticas y la configuración, se envía un mensaje de emparejamiento al servidor con el sensor y número de interruptor nuevo. Esta información se almacena en la base de datos y se envía en *unicast* al nodo actuador, haciéndolo saber que sensor actualmente lo controla.

Agregar

Mediante esta función el usuario puede agregar emparejamientos entre sensores y actua-

dores, un actuador puede tener n[4] sensores emparejados a él. Para agregar un enlace, el usuario escoge que sensor e interruptor desea que accione un actuador específico. El proceso de modificar y agregar emparejamiento se ilustra en la Figura 4.9.

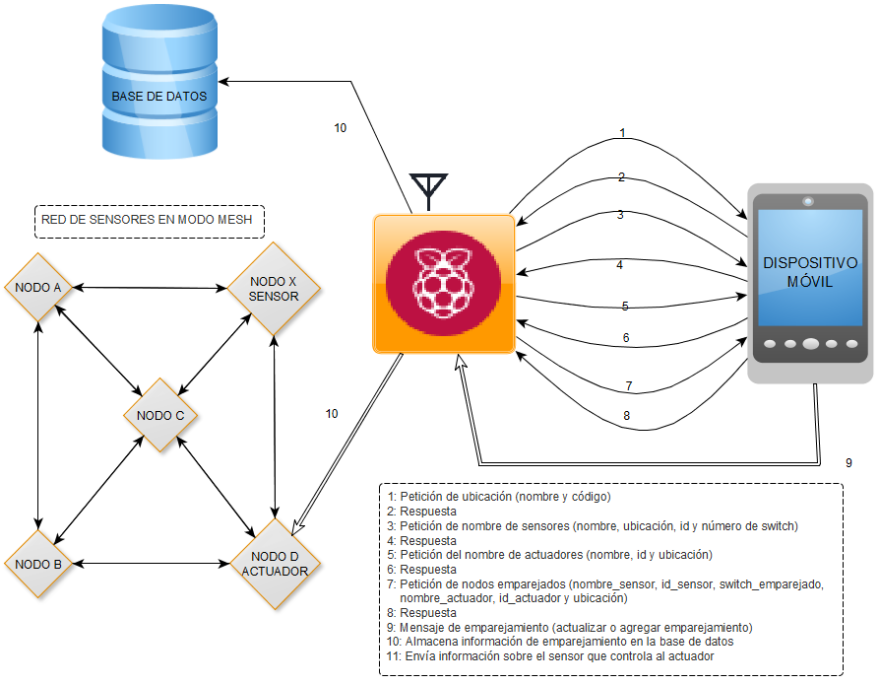

Figura 4.9: Proceso para modificar y agregar emparejamiento.

4.4.5. Trama de mensajes

Los mensajes utilizados para el intercambio de información entre el dispositivo móvil y el servidor se realizan utilizando el protocolo MQTT. Las tramas se envían mediante el tópico "movil" usando el mecanismo de petición respuesta.

El primer conjunto de mensajes son para almacenar y configurar información en la base de datos proveniente desde el dispositivo móvil, como: agregar nodos a la red, emparejar y actualizar emparejamiento. A continuación se muestra la disposición de los mensajes utilizados:

Actualización del emparejamiento de nodos.

1	2	id_actuador	id_sensor_pasado	id_sensor_nuevo	switch_nuevo

[4]n: número de sensores en una ubicación.

Actualización del tópico del nodo.

1	3	ubicación_nueva	código_nuevo	id_nodo

Insertar nodo actuador a la red.

1	4	nombre	ubicación	código	id

Insertar nodo sensor a la red.

1	5	nombre	número_de_switch	ubicación	código	id

Insertar emparejamiento de nodos.

1	6	id_sensor	switch	id_actuador

El segundo conjunto de mensajes son para consulta de información existente en la base de datos. Esta información es básicamente para validar y visualizar formularios de la aplicación en el dispositivo móvil. Se realiza una petición de la información mediante la cabecera de la trama (son los dos primeros números de la trama), mientras que la respuesta es la cabecera más la información solicitada. A continuación se muestra la disposición de los mensajes de respuesta utilizados:

Petición de lista de actuadores de la tabla estado.

2	0	nombre	id

Petición de ubicaciones.

2	1	ubicación	código

Petición de nombre del sensor.

2	2	nombre	id	número_de_switch

Petición de información de emparejamientos existentes en la red.

2	3	nombre_sensor	id_sensor	num_de_switch	nombre_actuador	id_actuador	ubicación

Petición de estado de actuadores.

2	5	id	estado

Petición de información de sensores.

2	7	nombre	ubicación	id	número_de_switch

Petición de información de actuadores.

2	8	nombre	id	ubicación

Finalmente, el tercer grupo de mensajes son para el control de los nodos. Estos mensajes se usan tanto para cambiar los estados de los nodos actuadores, como para activar y desactivar el simulador de presencia. A continuación se muestra la disposición de los mensajes utilizados:

Control de actuadores desde los sensores de la mesh.

0	estado	id_actuador

Control desde el dispositivo móvil.

2	6	id_actuador

Inicio de simulador de presencia.

2	9

Fin de simulador de presencia.

2	10

El primer mensaje es de control desde los nodos sensor hacia el servidor, para actualizar estados en la base de datos y luego se envía la actualización de estado hacia los nodos actuador. El segundo mensaje es para el control de los actuadores mediante el dispositivo móvil. Los dos últimos mensajes son para la activación y desactivación del simulador de presencia.

En el Apéndice G se realiza la captura de paquetes de la red, permitiendo visualizar las tramas de los mensajes mediante *Wireshark*[5].

4.4.6. Seguridad

***Autenticación*:** Una forma para que la red en malla sea segura, es hacerla privada. Esto evita que usuarios no autorizados puedan conectarse a la red de forma inalámbrica, robar información e incluso acceder al resto de nodos y al servidor. Por lo tanto, para acceder

[5] https://www.wireshark.org/

a la red en malla de nuestro proyecto se solicita una contraseña de acceso.

***Encriptación*:** Se utiliza la encriptación AES-256. Toda la información transmitida por los nodos de la *mesh* es encriptada y enviada hacia el nodo *gateway*. El nodo *gateway* desencripta la información y envía de forma serial al servidor. La información del servidor es encriptada y enviada a la *mesh*, cada nodo se encarga de desencriptar la información recibida. El procedimiento se puede apreciar en la Figura 4.10.

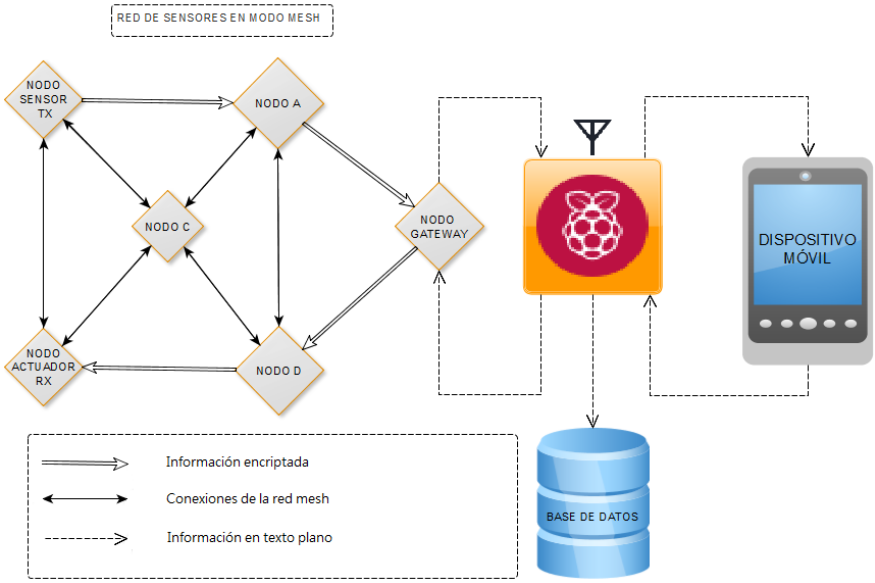

Figura 4.10: Proceso de encriptación en la *mesh*.

***Transferencia de información Servidor-Dispositivo móvil*:** Para prevenir acciones maliciosas sobre la captura e interpretación de la información cursada entre el servidor y los dispositivos móviles (interfaz humano máquina), se envían tramas de datos que contienen únicamente cabeceras e identificadores, los puntos finales son capaces de interpretarlos.

4.5. Base de datos

La base de datos es un elemento muy importante del prototipo. Es la encargada del almacenamiento y manejo de toda la información y configuraciones existentes en la red domótica. Está diseñada mediante ***MySQL*** [6] y la instalación se puede observar en el Apéndice B; se escogió este entorno porque además de su fácil uso, cuenta con un entorno gráfico para visualizar todo lo que sucede y así realizar el control y pruebas de las tareas. Específicamente cuenta con las siguientes tablas:

[6] https://www.mysql.com/

- **Actuadores:** información de los nodos actuadores.
- **Sensores:** información de los nodos sensores.
- **Estados:** información de los estados de los nodos actuadores.
- **Empareja:** información de los emparejamientos existentes.
- **Ubicación:** ubicaciones (tópicos) configurados en la red.

Mediante el diagrama de entidad-relación que se encuentra en la Figura 4.11, se representan las entidades relevantes de nuestro sistema de información, así como sus interrelaciones y propiedades.

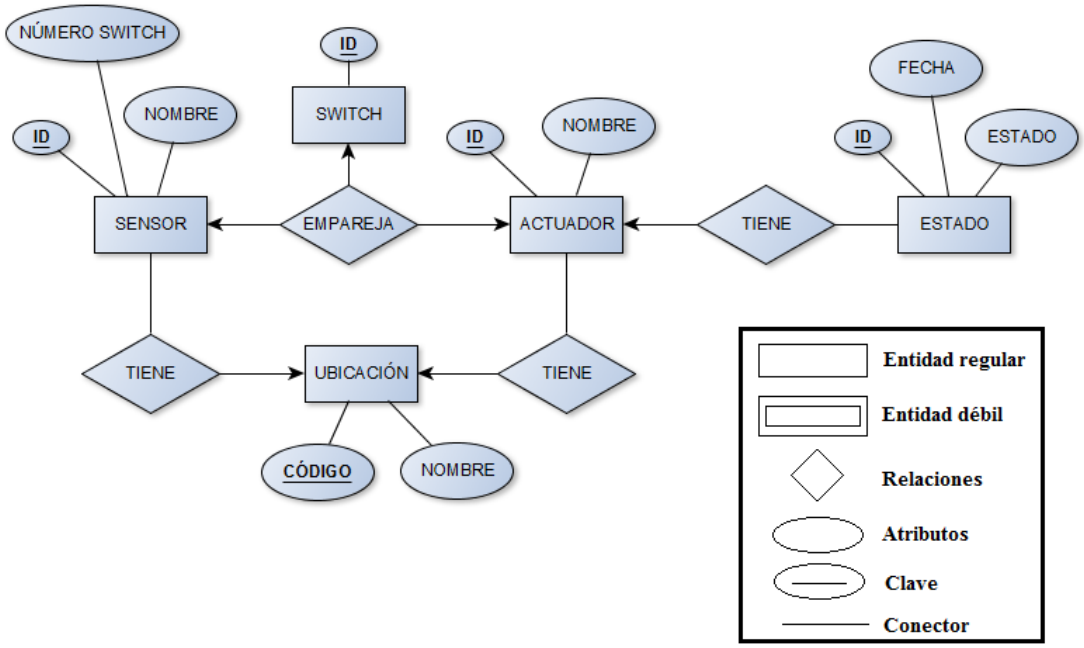

Figura 4.11: Diagrama entidad-relación de la base de datos.

4.6. Diseño de los prototipos de hardware

En esta Sección se detalla el diseño de los prototipos de hardware como son los nodos sensores, nodos actuadores y servidor. Existen 3 tres prototipos de nodos: nodo sensor, nodo actuador con relé de estado sólido y nodo actuador con relé mecánico. El primero es el encargado de percibir si el usuario desea cambiar el estado del actuador al cual se encuentra emparejado. El segundo puede manejar cargas eléctricas de poca potencia, mientras que el último puede

manejar cargas de mayor potencia.

4.6.1. Diseño del nodo sensor

El diseño de la placa del sensor es basado en el módulo WiFi ESP8266 ESP-12, pero el módulo necesita de un regulador de voltaje a 3.3 voltios. Se incorpora un led RGB que tiene varias funciones como: indicador de agregación a la red e indicador de estados en general (cada estado es representado por un color). Además tiene dos entradas para sensores táctiles. El esquemático del nodo sensor se muestra en la Figura 4.12.

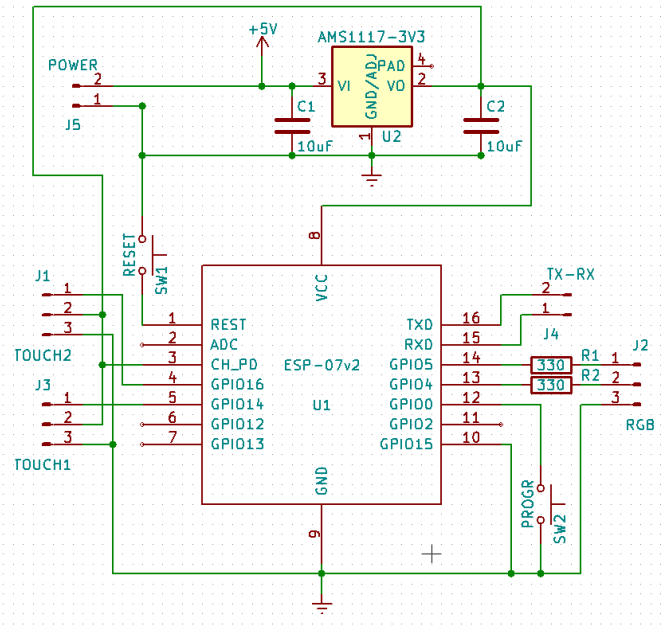

Figura 4.12: Esquemático del nodo sensor.

Para el diseño se utiliza componentes electrónicos de montaje superficial, de modo que la placa sea lo más compacta posible, ya que se va a remplazar por los interruptores tradicionales del hogar. La carcasa del nodo sensor está diseñada para que se incorporen: la placa del nodo, la fuente de alimentación y los sensores táctiles. El diseño final del prototipo se puede ver en la Figura 4.13.

4.6.2. Diseño del nodo actuador con relé de estado sólido

Este diseño está pensado para manejar cargas eléctricas de baja potencia, soporta hasta 550 mA, ideal para el control de iluminación led como: focos, plafones, lámparas, etc. Las ventajas de

Figura 4.13: Prototipo del nodo sensor.

utilizar relés de estado sólido es que ocupan muy poco espacio, aíslan la parte de potencia (carga) del control (microprocesador) y al no ser mecánicos tienen menos desgaste en sus componentes, por lo que tienen una mayor vida útil de los elementos. El esquemático de este nodo actuador se muestra en la Figura 4.14.

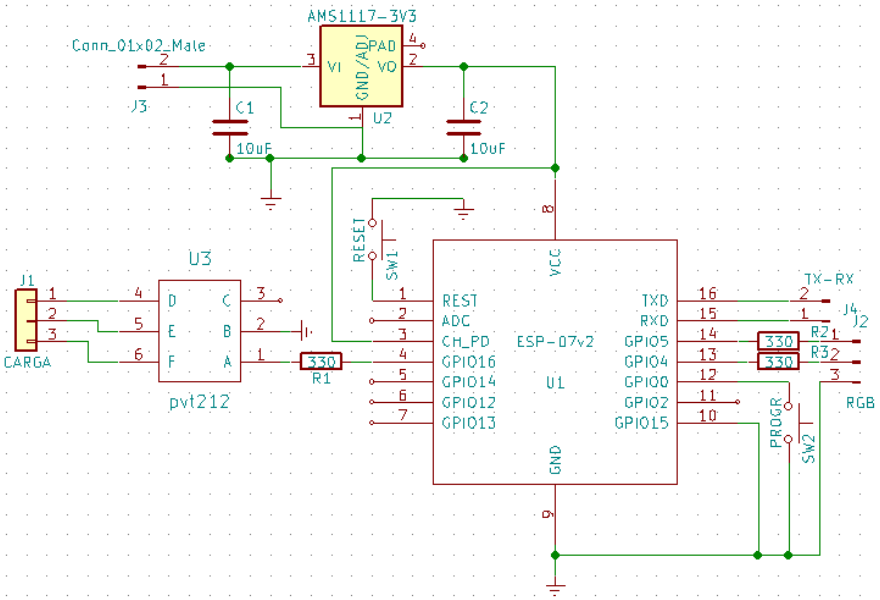

Figura 4.14: Esquemático del nodo actuador con relé de estado sólido.

La carcasa del nodo está diseñada para que sea compacto, de tal manera que se pueda incorporar dentro de los accesorios de iluminación. El prototipo consiste en la fuente de alimentación, la placa de control e indicadores de estados (mediante un led RGB), como se ve en la Figura 4.15.

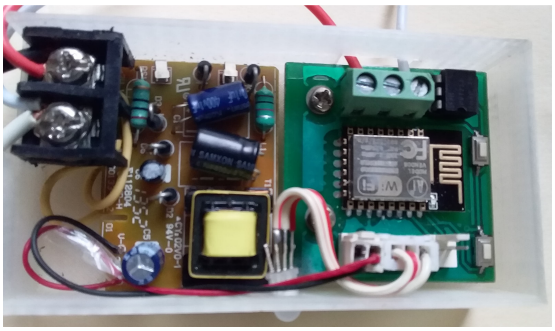

Figura 4.15: Prototipo del nodo actuador con relé de estado sólido.

4.6.3. Diseño del nodo actuador con relé mecánico

Este tipo de nodo está diseñado para manejar cargas eléctricas de mayor potencia, este puede soportar cargas que consuman hasta 10 A. El inconveniente de utilizar relés mecánicos, es que al ser un mecanismo electro-mecánico, los picos provenientes del bobinado puede regresar hacia los elementos de control (microprocesador) y eventualmente dañar el dispositivo. Este problema se soluciona al colocar un diodo anti-paralelo en el relé; para tener un margen más amplio de seguridad, se coloca un ópto-acoplador para aislar las partes de carga y control. El esquemático de este nodo actuador se muestra en la Figura 4.16.

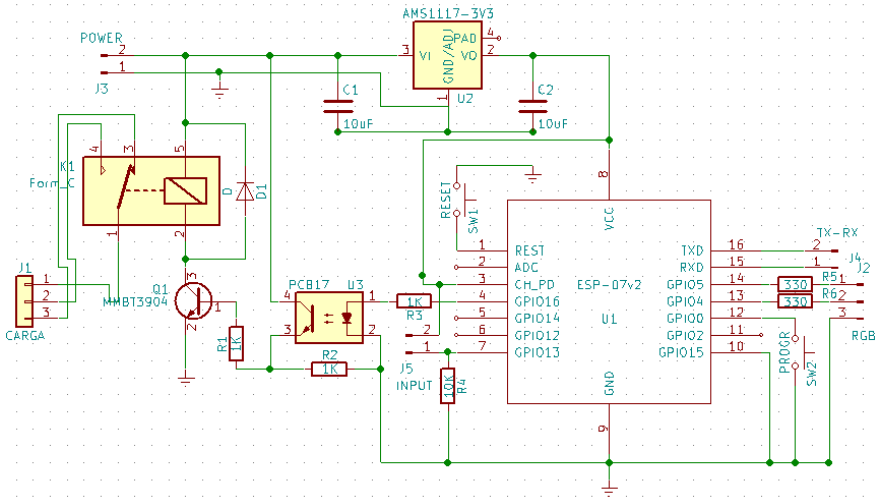

Figura 4.16: Esquemático del nodo actuador con relé mecánico.

En este caso, la carcasa contiene los mismos elementos que el nodo actuador anterior, la única diferencia que existe es el tamaño (es más grande). En la Figura 4.17 se muestra el diseño final de este prototipo.

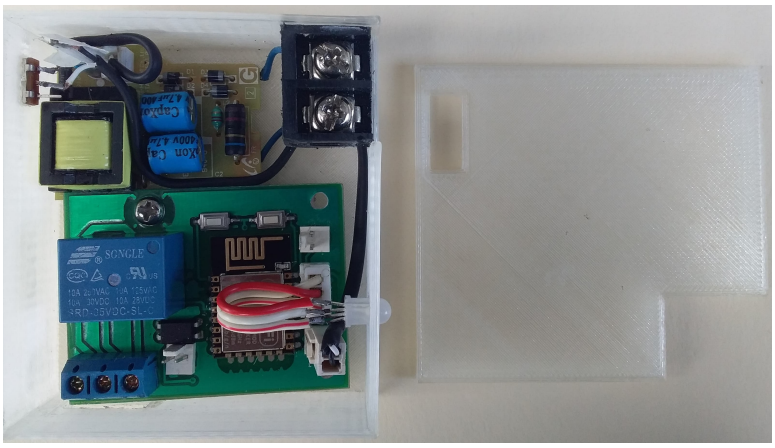

Figura 4.17: Prototipo del nodo actuador con relé mecánico.

4.6.4. Diseño del servidor

El prototipo para el servidor utiliza tres elementos:

- Raspberry pi 3.
- *Real Time Clock* (RTC).
- Módulo WiFi (nodo *Gateway* de la *mesh*.)

Estos tres elementos juntos forman el servidor. El primer elemento es el cerebro, encargado de todo el procesamiento; aquí se encuentra la base de datos, la red neuronal y el servidor MQTT. El segundo sirve para mantener la fecha y hora actualizada porque es muy importante registrar los cambios de estados de los actuadores en la fecha correcta para el entrenamiento de la red neuronal (la configuración del RTC se puede ver en el Apéndice C). Finalmente, el tercer elemento sirve como nodo *gateway* de la red *mesh*, posibilitando la comunicación del servidor con la red de nodos.

En la Figura 4.18 se aprecia el diseño final del servidor, con sus elementos incorporados.

Figura 4.18: Prototipo del servidor.

4.7. Simulación de presencia

En una vivienda común, existen rutinas; como por ejemplo: la hora de despertarse, horarios del trabajo, horarios alimenticios, etc. Pero esto generalmente son de lunes a viernes porque son días laborales. Teniendo esto en cuenta, la función de simulación de presencia de este proyecto, se basa en la idea de que una red neuronal podría aprender el patrón de interacción en el hogar, una vez aprendido se puede cada cierto periodo testear el entrenamiento para predecir que nodos deben estar activados o desactivados.

Se implementa mediante una red neuronal multi-capa, ya que es capaz de actuar como un aproximador universal de funciones entre un grupo de variables de entrada y salida. Por esto, las redes perceptrón multi-capa son herramientas de propósito general, flexibles y no lineales [55]. Se hace uso de esta no linealidad para el aprendizaje del patrón de interacción, caso contrario no sería más que una regresión lineal.

4.7.1. Configuración

Las herramientas utilizadas para la configuración y entrenamiento de la red neuronal es *Theano* y *Keras*. *Theano* es una biblioteca de *Python* que le permite definir, optimizar y evaluar expresiones matemáticas que involucran matrices multidimensionales de manera eficiente. Ésta es una herramienta de *deep-learning* que realiza todo el trabajo de aprendizaje detrás de *Keras*[7]. Se puede ver con mayor detalle la instlación y configuración en el Apéndice E.

[7] *Keras* es una API de redes neuronales de alto nivel, escrita en *Python* y capaz de ejecutarse sobre *TensorFlow* o *Theano*

La red neuronal multi-capa consta una capa de entrada, una capa de salida y una o más capas ocultas. El número de neuronas de la primera capa depende de las entradas de nuestro modelo (entradas de entrenamiento de la red neuronal), el número de neuronas de la segunda capa está relacionado con las variables de salida que se van a aproximar, y las capas ocultas y las neuronas dentro de ellas sirven como un mecanismo de variación del nivel de aprendizaje (se escogen de manera empírica el número de capas y neuronas ocultas). Para el caso de nuestra red neuronal se escogen:

- **Capa de entrada:** 4 neuronas (variables de entrada).
- **5 capas ocultas:** 15 neuronas, 8 neuronas, 17 neuronas, 25 neuronas y la última capa con 12 neuronas.
- **Capa de salida:** 1 neurona (variables de salida).

En la Figura 4.19 se puede apreciar la configuración de la red neuronal multi-capa. Además de las capas y número de neuronas, se configuran algunos parámetros adicionales como se muestra en la Tabla 4.1:

Tabla 4.1: Parámetros de configuración de la red neuronal multi-capa propuesta.

Parámetro	**Asignación**
Función de activación de la capa de entrada	Relu
Función de activación de las capas ocultas	Relu
Función de activación de la capa de salida	Sigmoid
Modelo de pérdida	Binary_crossentropy
Modelo de optimización	Adam
Métrica de validación	Accuracy (exactitud)

La función de activación de la capa de salida se escoge la “ sigmoid ” porque mediante esta función nos aseguramos que la salida varié entre 0 y 1. Esto, ya que se desea saber si el nodo actuador está activado (1) o desactivado (0) en una hora y fecha específica.

4.7.2. Datos de entrenamiento

La información para el entrenamiento de la red neuronal está contenida dentro de la base de datos del servidor. Se tiene cuatro parámetros que se han escogido, éstas son:

- **Día de la semana en que cambió de estado el nodo actuador:** es un número entero que varía entre 0 y 6 (7 días de la semana).
- **Hora del día en que cambió de estado:** un número entero que varía entre 0 y 23.

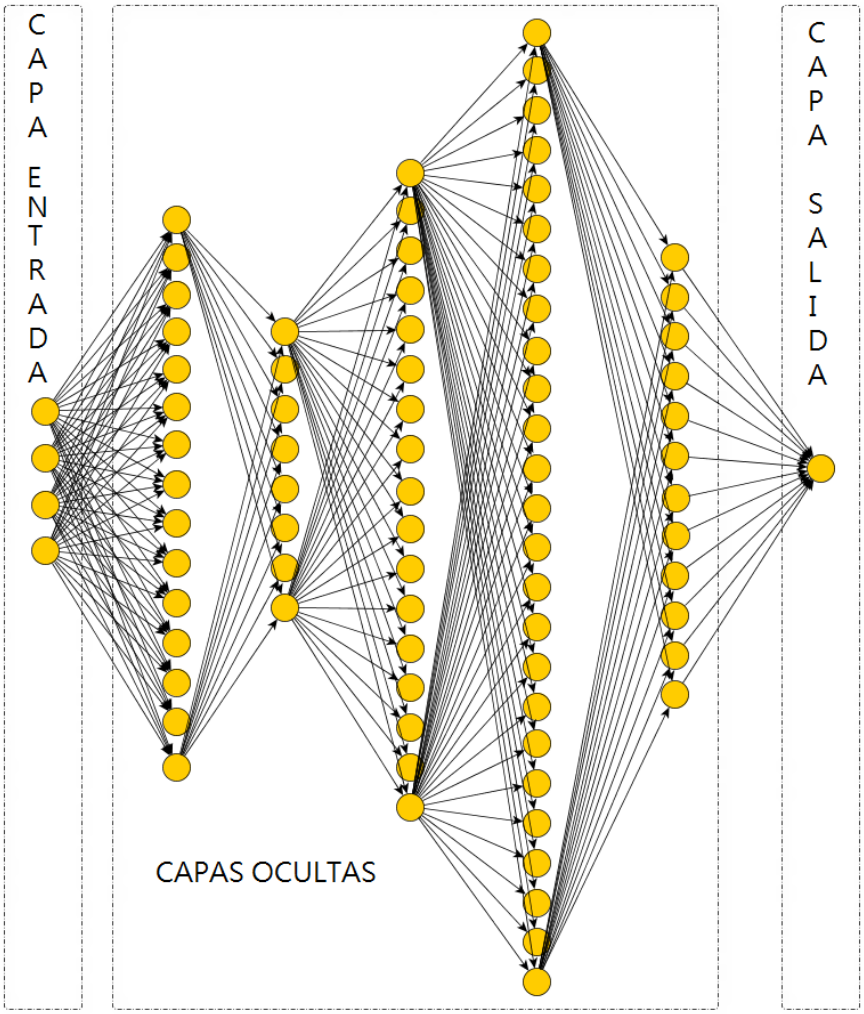

Figura 4.19: Configuración de la red neuronal multi-capa.

- **Minuto en que cambió de estado:** es un número entero que varía entre 0 y 59.
- **Número de actuador en la base de datos que cambió de estado:** es un número entero y es el número de actuador que está en la base de datos. Por ejemplo si la red tiene 10 nodos, este parámetro varía entre 0 y 9.

4.7.3. Proceso

Con la información contenida en la base de datos se entrena la red neuronal, se testea en determinados periodos de tiempo (por ejemplo cada 5 minutos) para predecir los estados de los actuadores en la fecha actual y finalmente se envía los mensajes de control hacia la *mesh* para

cambiar los estados de los actuadores. El proceso se explica en mayor detalle en el diagrama de bloques de la Figura 4.20.

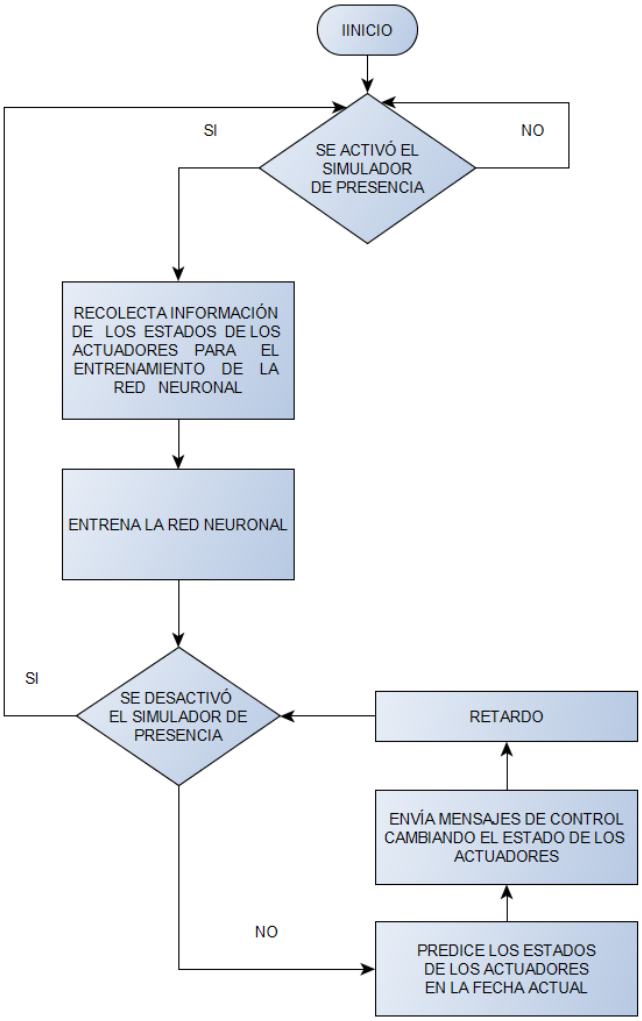

Figura 4.20: Diagrama de bloques del proceso de la simulación de presencia.

4.8. Conclusiones

Raspberry pi tiene la ventaja de poseer una arquitectura que posibilita alojar sistemas operativos basados en LINUX, por lo que se vuelve muy flexible en cuanto a las aplicaciones que se pueden desarrollar en este dispositivo. Otra ventaja es que hay la posibilidad de realizar conexiones físicas directamente con el procesador, mediante los pines de entrada/salida de propósito general, haciendo que los procesos sean ejecutados con mayor velocidad. Por otro lado,

los módulos WiFi escogidos son compactos y cuentan con un micro-procesador incluido, lo cual posibilita la programación de manera directa en el módulo y eventualmente ahorrar recursos de *hardware* y *software*.

Se diseña varios prototipos del nodo actuador, permitiendo incorporar una amplia variedad de cargas eléctricas a la red domótica. Pero hay que tomar en cuenta el tamaño de los prototipos porque en ciertos casos hay que colocarlos dentro de los dispositivos a controlar; por esto, se emplearon componentes de montaje superficial y así reducir al máximo el tamaño de los dispositivos diseñados.

El uso del protocolo MQTT posibilita ahorrar recursos en cuanto al trafico de información cursada por la red. La información es enviada solamente a los nodos que están subscritos a un tópico específico. Además facilita la agregación de los usuarios al sistema, ya que únicamente se debe subscribir al tópico "movil" y toda la información necesaria para el intercambio de mensajes está configurada previamente.

Capítulo 5

Resultados y discusión

En este capítulo se analizan las mediciones de parámetros de desempeño que se obtuvieron como resultado de la puesta en marcha del prototipo de sistema domótico. Se realizan mediciones del tiempo de procesamiento de las funciones del servidor, los tiempos de retardo al alejar los dispositivos, al hacer multi-salto y congestionar el canal de operación de la red *mesh*; y la eficiencia de la red neuronal.

5.1. Introducción

En el proyecto de domótica planteado, el tiempo es una medida crítica que refleja una buena calidad de servicio (QoS) y un buen desempeño del sistema, por este motivo se realizan mediciones de los tiempos de ejecución de funciones y tiempos que toma enviar la información de punto a punto. Las mediciones se efectúan en varios entornos y condiciones; Además, se tomaron varias mediciones en las mismas condiciones para obtener el intervalo de confianza del 95 % y los valores promedios.

Para el caso de la simulación de presencia, se realizan pruebas de eficiencia de la red neuronal. Estas pruebas consisten en el entrenamiento y validación con varios conjuntos de entrenamiento,

estos conjuntos son obtenidos mediante un patrón de interacción inicial y luego mediante una función de distribución normal se realizan modificaciones, utilizando la media y varianza de la distribución propuesta sobre el patrón inicial.

5.2. Medición 1: ejecución de las funciones del servidor

Es muy importante que los tiempos de ejecución de las funciones utilizadas en el sistema domótico sean reducidos, debido a que de esto depende directamente la calidad de servicio ofrecida al usuario y debido a que el umbral percepción del ojo humano es de 300 mili-segundos antes de que detecte cambios en el entorno [56]. Este tiempo de ejecución es crítico en cuanto a: tiempo de sincronización de la información para la interacción con la aplicación móvil y tiempo en que se puede cambiar de estado los actuadores. La mayoría de las funciones realizan búsquedas, inserciones y actualizaciones en la base de datos; esto hace que las consultas sean dependientes en gran manera de la cantidad de información almacenada. A continuación se muestran los resultados de ejecución de las funciones empleadas en el sistema y su variación con respecto al número de nodos almacenados en la base de datos.

Tabla 5.1: Tiempos de ejecución de las funciones realizadas en el servidor.

Función **Función**	**Sin nodos (segundos)**	**10 nodos (segundos)**	**20 nodos (segundos)**
Actualizar estado del actuador	0,02211	0,02754	0,03808
Consulta de nodos pertenecientes al tópico	0,01200	0,02149	0,01976
Actualización emparejamiento	0,04125*	0,02792*	0,03161*
Insertar actuador	0,04620*	0,06369*	0,04182*
Insertar sensor	0,05917	0,03638	0,02973
Insertar emparejamiento	0,03108*	0,03718*	0,02437*
Consulta información de estados de actuadores	0,03136*	0,01368*	0,03357*
Consulta ubicaciones	0,01657*	0,01617*	0,01574*
Consulta información del sensor	0,04135*	0,02286*	0,03151*
Consulta información del actuador	0,01693	0,02081	0,02589
Consulta estados de actuadores	0,02993*	0,02113*	0,02985*
Control desde el dispositivo móvil	0,00141*	0,00124*	0,00129*
Consulta de cantidad de sensores	0,00884	0,01827	0,02033
Consulta de cantidad de actuadores	0,02066*	0,01813*	0,02164*

***** son medidas que no varían al aumentar el número de nodos.

En la Tabla 5.1 se puede apreciar que el tiempo de ejecución de los procesos en el servidor son reducidos y están en el orden de las decenas de mili-segundos (por ejemplo el máximo valor es de 63 mili-segundos), siendo el intercambio de información de manera inmediata. Existen algunas funciones que no dependen de la cantidad de información alojada en la base de datos; pero hay otras que sí, sin embargo, el tiempo no aumenta de manera significativa y se podría

decir que es casi despreciable para el prototipo diseñado.

5.3. Medición 2: agregación de nodos a la red domótica

El primer paso para agregar los nodos a la red, es la transferencia de datos de configuración entre el nodo y el dispositivo móvil. El tiempo promedio para el nodo sensor es $5{,}31 \pm 0{,}3651$ segundos, mientras que para el nodo actuador es $4{,}83 \pm 0{,}2405$ segundos. Esto exhibe que los tiempo no varían demasiado según el tipo de nodo, además de tener intervalos de confianza reducidos. El siguiente paso es la agregación del nodo a la red de malla, este proceso es cuando el nodo se desconecta del dispositivo móvil y se une a la *mesh*. El tiempo promedio para el sensor de $5{,}86 \pm 0{,}4351$ segundos y para el actuador de $4{,}92 \pm 0{,}3185$ segundos. El último paso es la agregación a la red domótica, este proceso es cuando el nodo de la *mesh* envía la petición al servidor para formar parte de la red. El tiempo promedio para el sensor de $11{,}56 \pm 0{,}4281$ segundos y para el actuador de $22{,}44 \pm 0{,}3518$ segundos. Existe una mayor diferencia entre el promedio del nodo actuador y del nodo sensor, específicamente de 10,88 segundos; siendo el nodo actuador el que más se tarda, esto es porque el nodo actuador realiza más procesos antes de unirse a la red, siendo el tiempo mínimo.

En general, los tiempos totales de los nodos son grandes (el tiempo del actuador es $22{,}73 \pm 1{,}2283$ segundos y el tiempo del sensor es $32{,}19 \pm 0{,}9108$) porque el dispositivo móvil tiene que desconectarse del nodo que está en modo *access point* para unirse a la red WiFi del hogar.

5.4. Medición 3: retardos al variar la distancia entre los dispositivos

Existen dos casos cuando se alejan los nodos del servidor: el primer caso es cuando el nodo sensor permanece junto al servidor, mientras que el nodo actuador se aleja; y el segundo caso es cuando tanto el nodo sensor como el actuador se alejan del servidor (ver Figura 5.1).

El tiempo de retardo es mayor al alejarse los dos nodos del servidor, que uno solo. Esto es debido a que toda la información antes de ser recibida por el nodo actuador atraviesa el servidor. Evidentemente, lleva más tiempo en llegar el mensaje de control al servidor porque existe una mayor distancia que lo separa del nodo sensor (ver Figura 5.2).

Por otro lado, la funcionalidad que depende de la variación de la distancia entre el servidor y el *router* del hogar, es el control de los actuadores mediante el dispositivo móvil. Debido a que

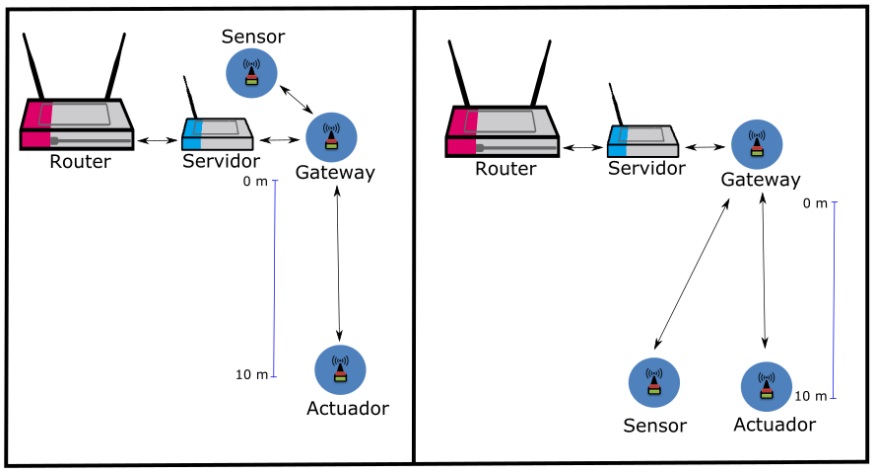

Figura 5.1: Método de medición para variar la distancia entre los nodos.

se envía la orden de control hacia el *router*, se retransmite hacia el servidor y de ahí a la red en malla en donde están los nodos actuadores; una vez que el estado del actuador cambia, se envía un mensaje de confirmación hacia el dispositivo móvil. Es decir el tiempo medido es de ida y vuelta de la información.

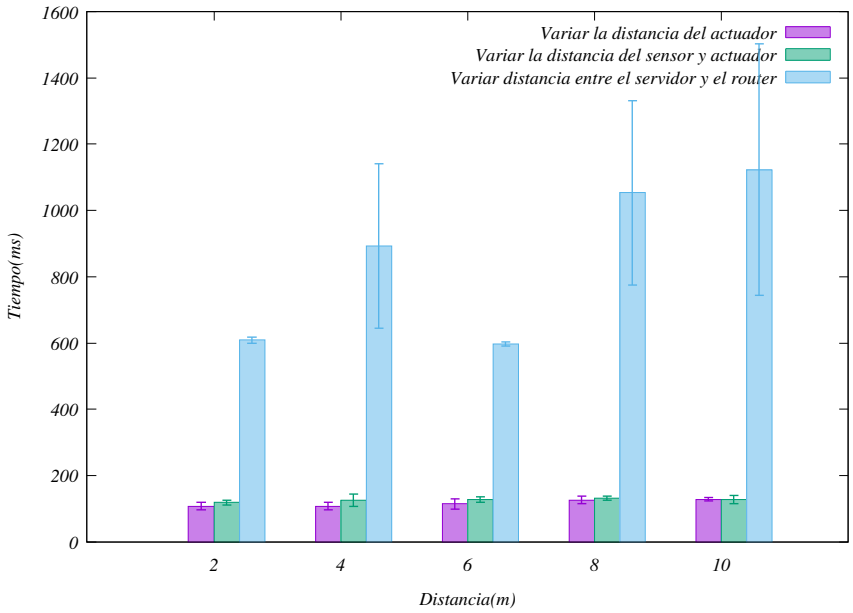

Figura 5.2: Retardos producidos al variar la distancia entre los dispositivos.

5.5. Medición 4: retardo según el número de saltos

Estos tiempos de retardo fueron obtenidos en un mismo escenario. Para las mediciones de un salto, el nodo sensor y actuador son alcanzables directamente desde el servidor. Para dos saltos, se colocó un nodo intermedio al borde del área de cobertura del servidor, luego se alejaron un nodo actuador y un nodo sensor hasta llegar al límite de cobertura del nodo intermedio. Mientras que para tres saltos, se realiza el mismo proceso que para dos saltos, pero se coloca un nodo intermedio adicional. En la Figura 5.3 se puede ver el mecanismo usado:

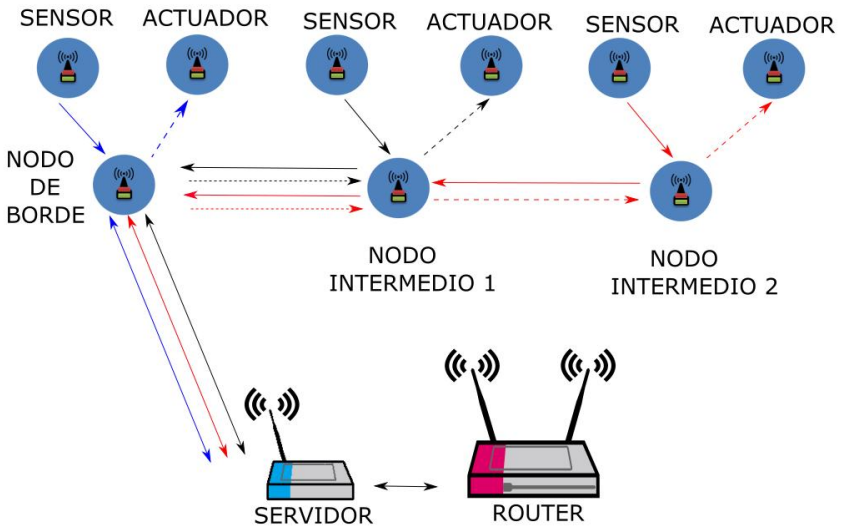

Figura 5.3: Mecanismo de toma de mediciones utilizando multi-salto.

Los resultados logrados (ver Figura 5.4) reflejan que no existe mucha diferencia de retardo al hacer un salto o dos saltos; pero, al hacer tres salos aumenta significantemente. El tiempo de retardo de tres saltos es un 90 % mayor que un salto y 77 % mayor que dos saltos. Estos resultados indican que la información transportada por la red tiene un tiempo de retardo sensible al número de saltos. Sin embargo, estos tiempos son aceptables de acuerdo a [56].

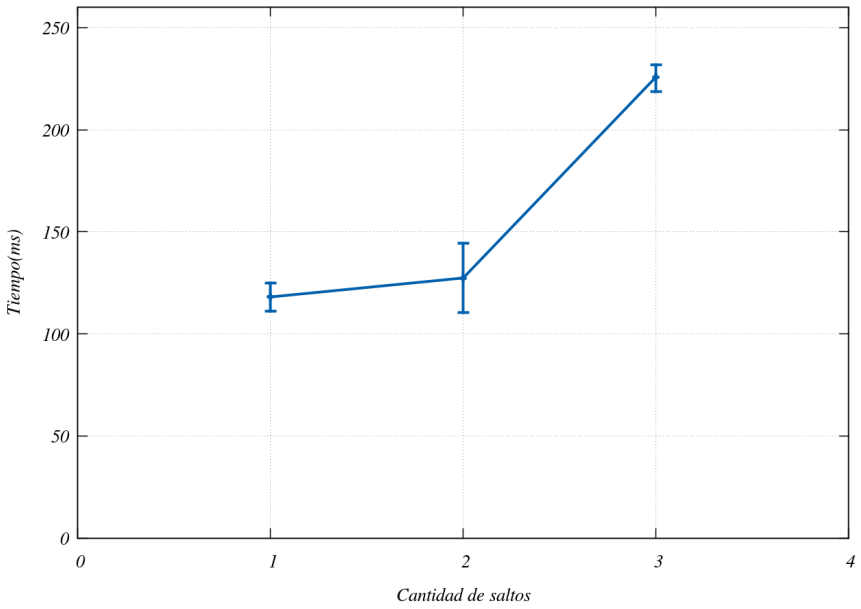

Figura 5.4: Resultados obtenidos utilizando multi-salto.

5.6. Mediciones 5: congestionamiento de la banda de frecuencia de la red de malla

La red en malla empleada trabaja en el canal 6 de la banda de frecuencia de los 2,4Ghz. Entonces, se generó tráfico variando el ancho de banda ocupado en el canal, específicamente se establecieron cinco sesiones paralelas en *iperf*[1] y se aumentó el ancho de banda con intervalos de 2,5$Mbps$ hasta llegar a 27,5$Mbps$ porque para 30$Mbps$ se saturó completamente el canal, causando la pérdida total de paquetes; debido a que se utiliza TCP.

En la Figura 5.5 se puede ver el tiempo de retardo aumenta conforme se aumenta el ancho de banda en el canal de operación. Hasta los 7.5 Mbps se tiene un tiempo reducido, pero después de esto, aumenta el retardo significativamente hasta llegar al máximo de 3.1 segundos.

[1] https://iperf.fr/

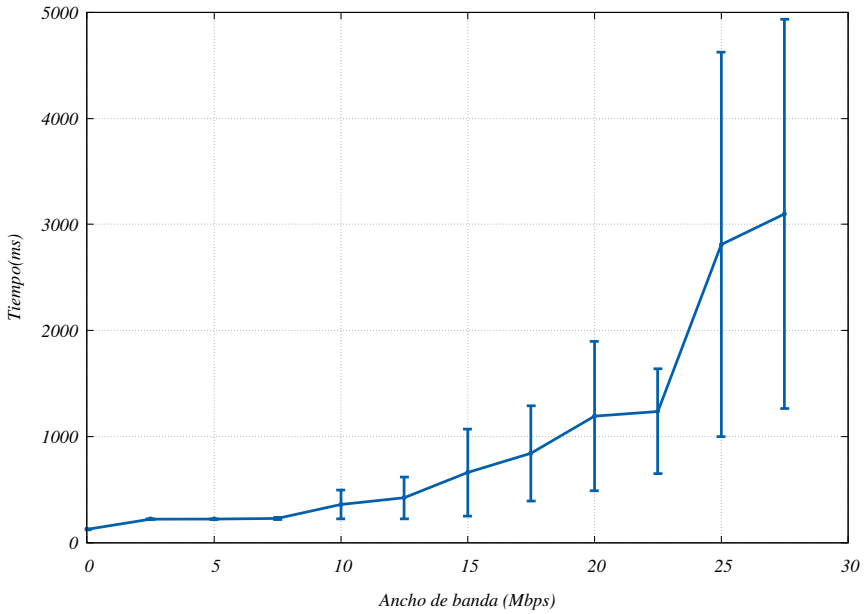

Figura 5.5: Resultados obtenidos al saturar el canal de operación de la *mesh*.

5.7. Simulación de presencia

En la Tabla 5.2 se planteó un patrón de interacción en el hogar, con horarios típicos de una familia. De lunes a viernes es el mismo horario, pero los fines de semana el horario de la mañana se desfasa dos horas más tarde. Además la simbología $X(Y)$ en donde X es el nodo actuador y Y es una variable booleana de estado 1(encendido) o 0(apagado).

A partir del patrón de interacción planteado se crean 10 conjuntos de pruebas y entrenamiento; modificando las horas con una distribución normal, es decir, que las horas de la Tabla 5.2 son la media y se generan variaciones de ella mediante una varianza, específicamente se tiene una varianza de 2 (los horarios pueden variar entre $\pm$ 2 minutos). En la Figura 5.6 se puede ver resultados al realizar el entrenamiento (línea azul) y validación (línea roja). En donde se obtiene que la eficiencia promedio de la red neuronal para el entrenamiento es del $96{,}48 \pm 1{,}05\,\%$ y para la validación es del $92{,}65 \pm 2{,}91\,\%$.

Tabla 5.2: Patrón de interacción de un hogar típico.

Hora	Lunes-Viernes	Sábado-Domingo
06:00	1(1)	-
06:05	2(1)	-
06:10	3(1)	-
06:20	4(1), 3(0)	-
06:30	4(0), 2(0), 1(0)	-
08:00	-	1(1)
08:05	-	2(1)
08:10	-	3(1)
08:20	-	4(1), 3(0)
08:30	-	4(0), 2(0), 1(0)
18:00	5(1)	5(1)
18:05	4(1)	4(1)
20:10	5(0)	5(0)
20:20	4(0)	4(0)
20:30	1(1)	1(1)
20:35	2(1)	2(1)
20:50	2(0)	2(0)
21:00	1(0)	1(0)

Nota: 1(dormitorio padres), 2(dormitorio hijo), 3(baño), 4(cocina), 5(sala)

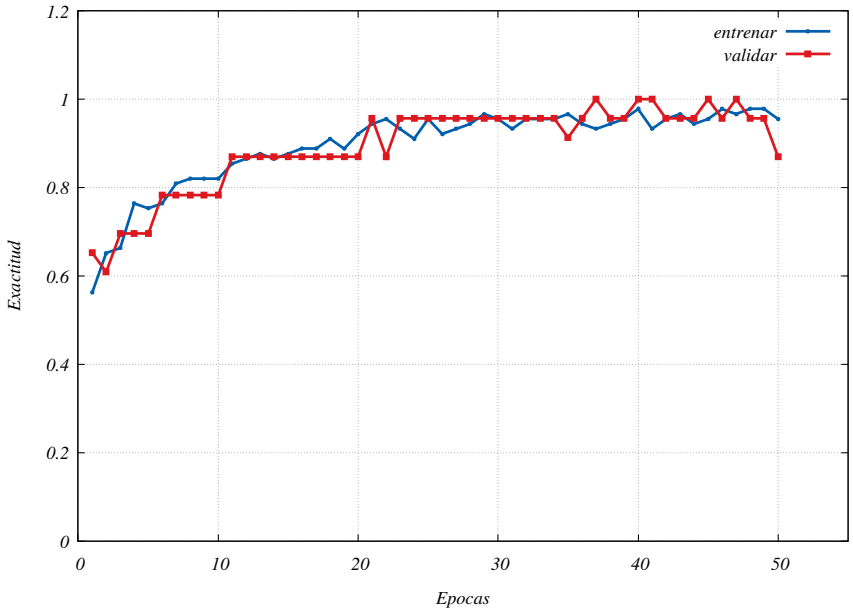

Figura 5.6: Resultados del entrenamiento (linea azul) y validación (linea naranja) con el primer conjunto de entrenamiento, se muestra la exactitud (izquierda) y la pérdida (derecha).

5.8. Conclusiones

Los tiempos de ejecución de las funciones dentro del servidor son menores a 63 mili-segundos en condiciones de poca carga computacional (no está en funcionamiento el simulador de presencia), incluso al aumentar la cantidad de información. Resultando ser un tiempo adecuado para la aplicación, ya que el tiempo de sensibilidad del ojo humano es de 300 mili-segundos [56].

Los resultados obtenidos con el aprendizaje de la red neuronal para la simulación de presencia son buenos, ya que al realizar pruebas con diferentes conjuntos de validación se logró una eficiencia de predicción del 92.65 %. Hay que recalcar resultados de eficiencia cercanos al 100 % no es recomendable porque se estaría sobre-entrenando a la red y el rendimiento para otros conjuntos de prueba es deficiente.

Capítulo 6

Conclusiones y Recomendaciones

En este capítulo se presentan las conclusiones finales del trabajo realizado así como la interpretación de los resultados, las limitaciones que se encontraron en el proceso, y finalmente, se proponen maneras de ampliar la investigación a futuro.

6.1. Conclusiones

En este estudio se ha demostrado que la red en malla utilizada tiene buen desempeño en cuanto al retardo, incluso al hacer multi-salto. Sin embargo, al congestionar el canal de operación se produce un retardo crítico en cuanto al envío de mensajes de control. Además relacionando los tiempos de ejecución de los procesos en el servidor y el tiempo transcurrido de transmisión de información punto a punto (entre un nodo actuador y un nodo sensor) resulta que el tiempo que viaja la información en la red en malla es mucho mayor que los tiempos de procesamiento en el servidor, teniendo sentido que los retardos aumenten cuando se satura el canal de transmisión de la red.

MQTT permitió reducir la carga en la red de malla. Pero también facilitó la comunicación entre el dispositivo móvil y el servidor, debido a que los dispositivos móviles que quieren controlar el

hogar únicamente deben subscribirse al tópico “móvil” e inmediatamente adquiere acceso a la red domótica.

6.2. Recomendaciones

Basándose en los resultados obtenidos, se recomienda colocar el servidor lo más cerca del *router* del hogar y de ser posible conectarlo con el cable de red. Esto para disminuir al máximo los retardos producidos. Además configurar el *router* en un canal diferente al seis, para evitar saturar esa frecuencia de operación, debido a que estos retardos producidos son los más críticos.

6.3. Trabajos futuros

El trabajo desarrollado funciona con la red WiFi del hogar, de modo que se puede controlar los actuadores de manera local. Pero el prototipo puede ser ampliado, haciendo que el servidor se pueda acceder desde Internet, posibilitando un sin número de aplicaciones.

Las cargas eléctricas a controlar por los nodos actuadores son de tipo ON/OFF, pero en un futuro se podría desarrollar para que soporte dispositivos con varios niveles de potencia, para aplicaciones que demanden eficiencia energética, disminución del consumo, entre otras. Las aplicaciones por ejemplo pueden ser: iluminación dimerizable o controlar la velocidad de motores (para el control de cortinas y puertas).

Con respecto a la problemática que se dio al saturar el canal de operación de la *mesh*, una solución puede ser que automáticamente los nodos se cambien de canal cuando se detecta que la red está congestionada, enviándose un mensaje de configuración desde el servidor.

En cuanto a la autenticación de los usuarios de los dispositivos móviles, no se realiza en el prototipo, pero mediante en servidor de autenticación como lo es RADIUS se lo lograría fácilmente.

Anexos

Apéndice A

Instalación y configuración del servidor MQTT

En este punto se detalla el procedimiento que se utilizó para poner en marcha el servidor MQTT. El procedimiento es basado en varios trabajos.

A.1. Instalación

Existe una plataforma que admite todos los intermediarios *brokers* de publicación/suscripción y es desarrollado por [57]. En el Listado A.1 en la línea 1, se muestra el comando de instalación. El servidor MQTT trabaja con *Node.js* y la comunicación con la base de datos se la realiza con *python*[1]. Para poder comunicarse a través de los dos tipos de lenguaje de programación, tanto *Node.js* como *python* son clientes MQTT, para el intercambio de mensajes. En el Listado A.1 en la línea 2, se muestra el comando de instalación. *MongoDB*[2] permite almacenar toda la información que cursa por el *broker*, esta información puede ser accedida desde el exterior (internet) para las aplicaciones de IoT. En el Listado A.1 en la línea 1, se muestra el comando de instalación, y en la línea 2 se muestra el comando para inicializar el servicio.

Listado A.1: Instalación de complementos para MQTT

```
1    sudo npm install mosca --save
2    pip3 install paho-mqtt
3    sudo apt-get install mongodb-server
```

[1] https://www.python.org/
[2] https://www.mongodb.com/

4 sudo /etc/init.d/mongodb start

A.2. Configuración del broker

Para que entre en funcionamiento el servidor MQTT es necesario indicar la dirección del broker, en este caso, está alojado en el mismo servidor y escuchando el puerto 1883. Mientras que el nodo *gateway* de la *mesh* realiza el intercambio de información mediante el puerto serial del *raspberry pi* a una velocidad de 115200 baudios, ya que es la maxima velocidad estable para los módulos WiFi. Para más detalle, ver el Listado A.2.

Listado A.2: Configuración del Broker

```
config.serial = {};
config.serial.port = process.env.MESH_SERIAL_PORT
|| "/dev/ttyS0";
config.serial.baud = process.env.MESH_SERIAL_BAUD
|| 115200;

config.mqtt = {};
config.mqtt.server = process.env.MESH_MQTT_SERVER_URL
|| "mqtt://localhost:1883";

config.topic = {};
config.topic.PREFIX_IN = "mesh-in/";
config.topic.PREFIX_OUT = "mesh-out/";
config.topic.GATEWAY_ID = "gateway";
config.topic.ONLINE = "online";
config.topic.STATUS = "status";

config.payload = {};
config.payload.ONLINE = "online";
config.payload.OFFLINE = "offline";

exports.config = config;
```

Apéndice B

Instalación de la base de datos

En este punto se detalla el procedimiento que se utilizó para instalar la base de datos (MySQL[1])y los problemas que se obtuvieron y como se solucionaron.

B.1. Instalación

Antes de instalar la base de datos se debe tener en cuenta algunos pre-requisitos como es: *Apache* y *PHP*. Con la línea 1 del listado B.1 se instala los pre-requisitos, mientras que la base de datos se instala con la línea 2. También es necesaria la instalación de *PHPAdmin*, para poder acceder desde el navegador a la base de datos, permitiendo ver todos los procesos visualmente. Se debe ejecutar el comando de la línea 3 del Listado B.1.

Listado B.1: Instalación de la base de datos

```
1 sudo apt-get install apache2 php5 libapache2-mod-php5
2 sudo apt-get install mysql-server
3 sudo apt-get install phpmyadmin
```

Eventualmente en el proceso de instalación, se desplegará un cuadro de diálogo pidiendo que ingrese una contraseña para el usuario *root*. Esto se lo hace debido a que luego necesitaremos esta información para acceder al servidor MYSQL y conectarlo a PHPMyAdmin, ver Figura B.1.

En el proceso de instalación se necesitará configurar *PHPAdmin* para conectarse al servidor de

[1] https://www.mysql.com/

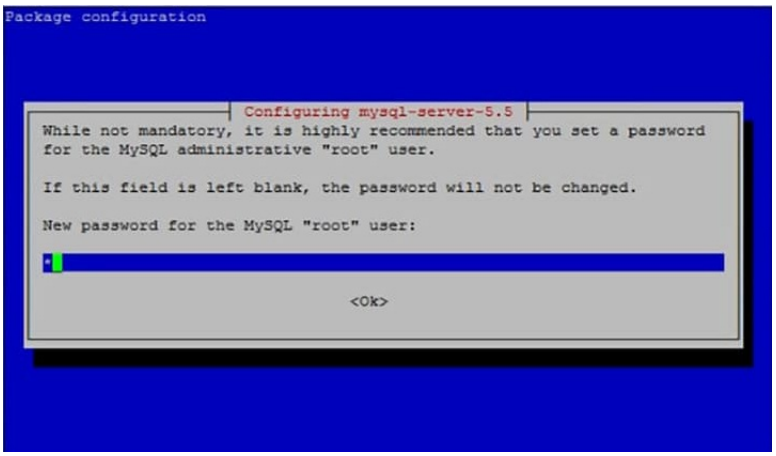

Figura B.1: Creación de la contraseña de autenticación MySQL.

la base de datos *SQL*. Para hacer esto, se da el visto bueno al mensaje de la Figura B.2:

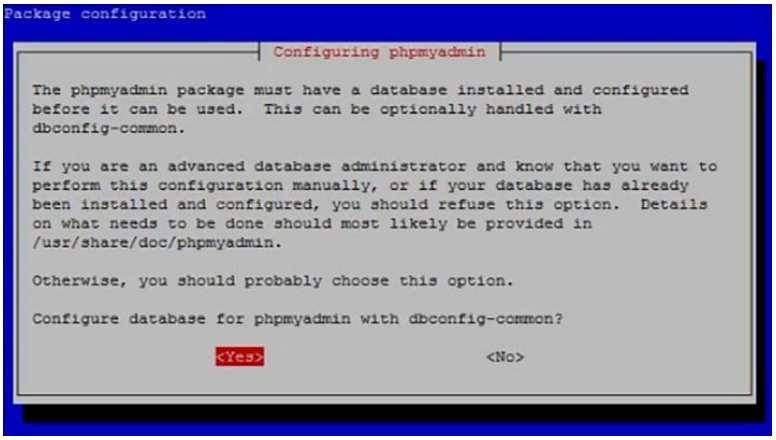

Figura B.2: Vinculación de *PHPAdmin* con *SQL*.

Una vez realizados todos los pasos anteriores y si todo salió bien, se mostrará la siguiente pantalla al ingresar desde el navegador a: `localhost/phpmyadmin`, como se muestra a continuación:

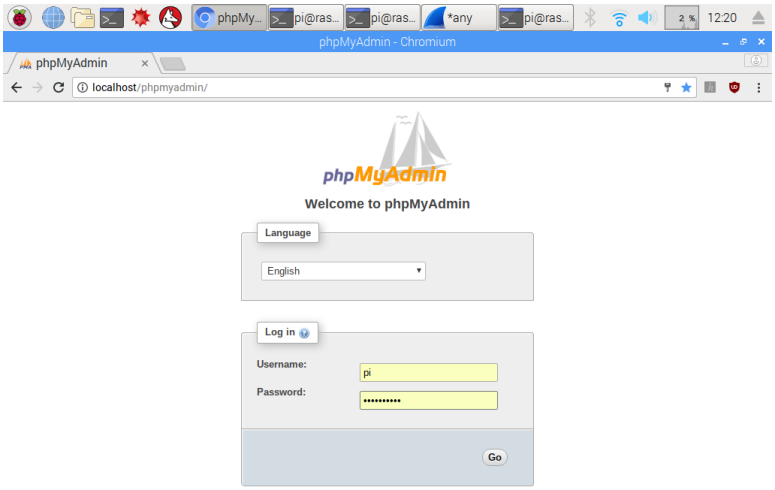

Figura B.3: Pantalla principal de la base de datos.

B.2. Conflictos y soluciones

Después de la instalación se obtuvo un problema. Este consistía en que en ciertos momentos no se permitía acceder a la base de datos mediante *phpadmin*, básicamente salia un error dando a saber que no sabía que hacer con la petición que se realizó.

Este error es un problema de permisos NGINX. Para solucionarlo se tiene que modificar el archivo `nginx.conf` ubicado la siguiente dirección `/etc/nginx`, escribiendo desde la línea 3 hasta la 12 del listado B.1:

Listado B.2: Solución al error de accesos a phpadmin

```
1  user www-data;
2  worker_processes auto;
3  pid /run/nginx.pid;
4  include /etc/nginx/modules-enabled/*.conf;
5
6  server{
7          listen 80;
8          server_name myapp.com;
9          location / {
10                 root /phpmyadmin;
11         }
```

```
12 }
13
14 events {
15         worker_connections 768;
16         #multi_accept on;
17 }
```

Para que los cambios efectuados en el archivo de configuración se establezcan, se debe ejecutar la siguiente línea de comando: `sudo nginx -s reload`

Apéndice C

Configuración del RTC

En este punto se detalla el procedimiento que se utilizó para configurar el módulo RTC, encargado de sincronizar y de mantener igual la hora en el servidor.

Para que el *raspberry pi* soporte el RTC, se debe modificar el archivo ***config.txt*** ubicado en la dirección `/boot`, agregando el modelo del módulo que se va a utilizar, como se en la línea 1 del Listado C.1. También se tiene que desactivar el " `fake hwclock` " que interfiere con el reloj en tiempo real. Para esto, se ejecutan las líneas 2 y 3. Finalmente, en el archivo `hwclock-set` de la ruta `/lib/udev/}` se debe comentar las líneas 4, 5 y 6 del Listado C.1.

Listado C.1: Configuración RTC

```
1 dtoverlay=i2c-rtc,ds1307
2 sudo apt-get -y remove fake-hwclock
3 sudo update-rc.d -f fake-hwclock remove
4 if [ -e /run/systemd/system ] ; then
5  exit 0
6 fi
```

Se reinicia el *raspberry pi* y se verifica que la comunicación entre el módulo y el servidor se efectúa satisfactoriamente (ver Listado C.2), mediante el comando `sudo i2cdetect -y 1`.

Listado C.2: Verificación del funcionamiento del RTC

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                 -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: 50 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- UU -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Inicialmente el módulo RTC, tendrá un tiempo incorrecto. Hay que sincronizarlo con el *raspberry pi* con `sudo hwclock -D -r` (debe haber una conexión con Internet para la sincronización). Luego se ejecuta `sudo hwclock -w` para escribir la hora y otro `sudo hwclock -r` para leer la hora.

Apéndice D

Desarrollo de la aplicación móvil

La aplicación se la implementó para *smartphones* basados en *Android*. La herramienta para el desarrollo de la aplicación es la plataforma oficial *Android Studio*. El repositorio de la aplicación se encuentra en la plataforma *GitHub* [58].

D.1. Descripción de la aplicación

La aplicación que se desarrolló tiene como función principal la interacción entre los usuarios y el sistema implementado de manera sencilla y práctica. La aplicación cumple dos funciones principales:

- Configurar nodos para agregar a la red (Figura 4.7).
- Configuración (Figura 4.9) y control de los nodos de la red.

La interfaz de usuario cuenta con cinco opciones para el control y configuración de los nodos como se muestra en la Figura D.1 .

1. **Control *ON/OF* de cargas**: esta opción permite al usuario accionar las cargas que están en la red. En una lista se muestra el nombre de la carga y el estado (*ON/OFF*), el usuario puede elegir cualquier carga de la lista para accionarla.
2. **Configurar nodos para agregar a la red**: esta opción provee la capacidad de conectarse a los nodos cuando estén trabajando en modo AP, es decir, cuando los nodos aún no están incorporados en la red. Mediante esta opción se envía la información de configuración a

Figura D.1: Interfaz de Usuario.

los nodos.

3. **Configuración de emparejamiento**: mediante este botón se puede cambiar, agregar y eliminar emparejamientos entre los nodos actuadores y nodos sensores.
4. **Configurar servidor**: con este botón se busca o se define la dirección del servidor central, si no existe el servidor central no será posible utilizar la aplicación.
5. **Activar/Desactivar simulador de presencia**: este botón sirve para activar o desactivar el simulador de presencia.

Todo el control y la configuración de emparejamiento se lo realiza mediante el protocolo MQTT. Las tramas para cada función fueron descritas en el Capítulo 4.

D.2. Requisitos de Hardware

La aplicación fue desarrollada para que funcione en cualquier *smarthphone Android*, siempre que cumpla con las siguientes características básicas:

- Sistema operativo *Android* 4.0.3 API 15 *(Ice Cream Sandwich)* o superior.
- Conexión WiFi
- Memoria RAM de 512 MB o superior.

Si el dispositivo no cuenta con alguna de estas características básicas la aplicación puede verse parcial o completamente inutilízale.

D.3. Implementación de un cliente MQTT

Para que la aplicación pueda comunicarse con la red implementada se necesita crear un cliente MQTT que esté suscrito a un tópico (para la aplicación el tópico específico es "*movil*"), una vez suscrito a un tópico podrá publicar.

Para la implementación del cliente MQTT se utilizó la librería *Eclipse Paho Java Client*[59]. *Paho Java Client* es una librería de cliente MQTT escrita en Java para desarrollar aplicaciones en plataformas compatibles con Java, como *Android*. *Paho Java Client* proporciona dos APIs:

- *MqttAsyncClient* proporciona una API completamente asíncrona en la que la finalización de las actividades se notifica a través de *registered callbacks*.
- *MqttClient* es un contenedor síncrono alrededor de *MqttAsyncClient* donde las funciones aparecen sincronizadas con la aplicación.

D.3.1. Funciones de un cliente MQTT

La función cliente provee al usuario la conexión con el *broker* cada vez que quiera realizar alguna acción. El primer paso es agregar las dependencias *Paho* y el repositorio *Maven* [59].

Una vez agregada las dependencias, se procede a crear un cliente MQTT mediante el objeto *MqttAndroidClient*; los parámetros básicos para este objeto son el contexto, el url del *broker* y el id del cliente.

La función publicar genera un mensaje *PUBLISH* con el tópico adecuado, que será usado por el *broker* para reenviar el mensaje a los clientes interesados. El protocolo MQTT no especifica como tiene que ser el formato de los datos (payload), por lo que, para este proyecto se utiliza una cadena de *strings*. Los parámetros principales de esta función son:

- ***Id del Cliente***: identifica al usuario. Cadena de tipo *String*.

- ***Mensaje***: es el contenido util. Cadena de tipo *String*.
- ***QoS***: este campo indica el nivel de calidad de servicio(1, 2 o 3). Cuanto más alto es el nivel, más estrategias pone en marcha el protocolo para asegurarse de que los mensajes llegan a los clientes, y si no llegan, al menos que estos sepan que el mensaje no va a llegar para tomar medidas. Es un número de tipo *int*.
- ***Tópico***: trata de una cadena UTF-8 utilizada por el *broker* para filtrar mensajes y repartirlos a los clientes interesados en ese tópico. En esta aplicación el tópico general es "*movil*" y es de un solo nivel.

La función suscribir permite unirse al tópico "*movil*", para poder recibir los mensajes que son publicados en este tópico y poder procesarlos. Los parámetros necesarios para esta función son: Id Cliente, tópico, QoS. A continuación muestra el código utilizado para lograr suscribirse a un tópico.

D.3.2. Servicio para escuchar mensajes MQTT

Una vez que se suscribe a un tópico, se necesita de un servicio para escuchar los mensajes que se publican en ese tópico. Para crear un servicio se siguen los siguientes pasos [60]:

1. Crear una Clase con extensión *Service* y crear todos los métodos.
2. En el método *Oncreate*, se crea un objeto de cliente MQTT. En el objeto creado se llama a la función *MqttCallbackExtends* que crea un método *messageArrived*, mediante este método se podrá acceder a los mensajes que son publicados en el tópico "*movil*", los mensajes se guardan en un *string* que luego será procesado.

D.4. Registro de SSID del hogar

Con el fin de que la aplicación se más robusta y no genere errores, se valida para que el usuario se registre en la red del hogar. Cuando el usuario instala por primera vez la aplicación, el algoritmo obtiene la información de la red a la que está conectado y el usuario determina si es o no la red de su hogar. Si la red es la del hogar, la información de la red es guardada en un archivo "$redinfo.txt$", caso contrario se abre la ventana de *Ajustes/WiFi* para que el usuario se conecte a la red de su hogar. Este proceso se muestra en la Figura D.2. Mientras no se registre la red del hogar el usuario no podrá realizar ninguna acción.

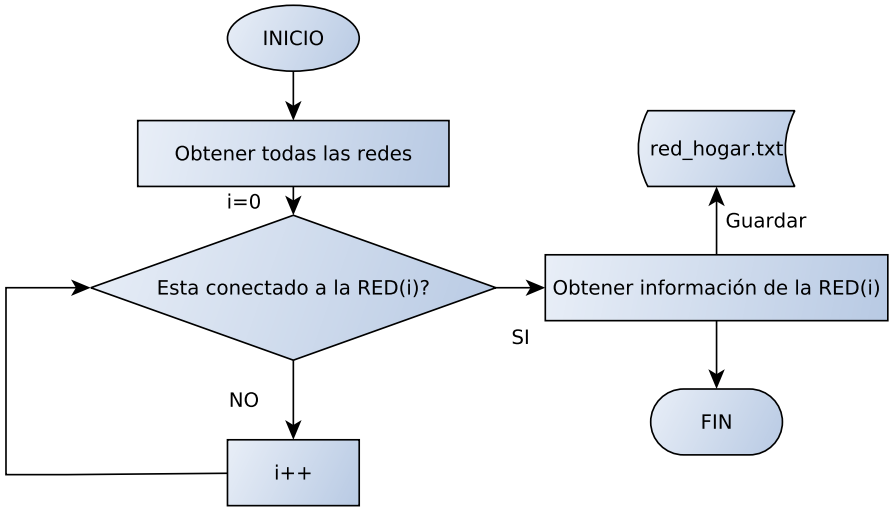

Figura D.2: Proceso para registrar la red del hogar.

D.5. Descubrimiento de dirección IP del servidor

Para que la aplicación realice cualquier acción en la red implementada, debe conocer la dirección IP del servidor. Para implementar el descubrimiento de la red, se ha tomado como base el proyecto *android-network-discovery*[61].

Luego de determinar si la red a la que se encuentra conectado es la del hogar, se procede a buscar la dirección IP del servidor. Primero busca si el archivo "*ip.txt*" está vacío, si es así, se procede a realizar la búsqueda. Se realiza una secuencia de búsqueda partiendo de la dirección IP del dispositivo, en cada dispositivo encontrado se evalúa si la dirección MAC del dispositivo coincide con dirección MAC del servidor, si es así, se guarda la dirección IP en el archivo "*ip.txt*" y termina el proceso; caso contrario se sigue la búsqueda. Todo este proceso se muestra en la Figura D.3.

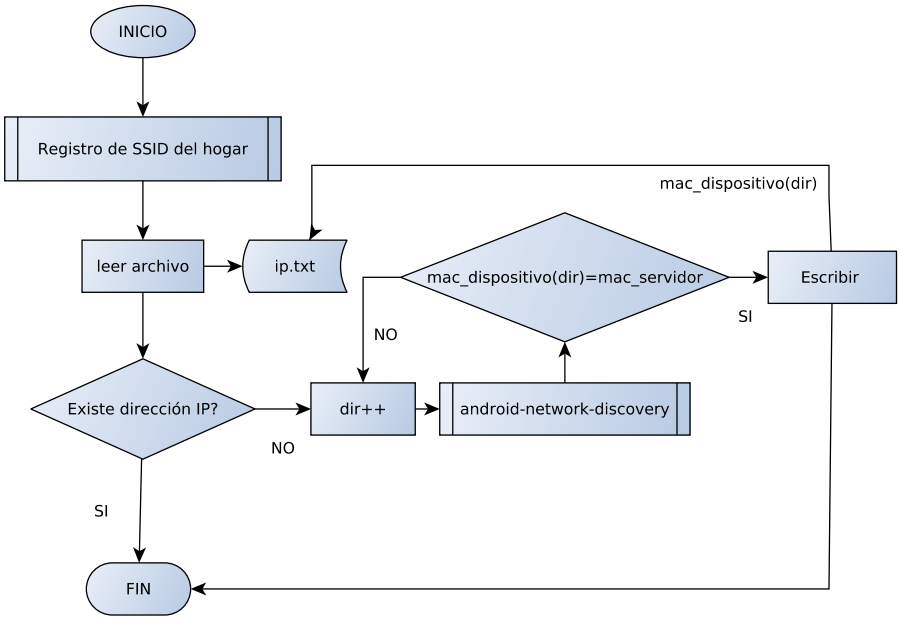

Figura D.3: Proceso para descubrir la dirección IP del servidor.

D.6. Configuración de nodos para agregar a la red

Un nodo para que pueda ser parte de la red, ésta debe ser configurada mediante la aplicación. La aplicación envía los siguientes datos de configuración al nodo:

- SSID de la *mesh*.
- Contraseña de *mesh*.
- Puerto de la *mesh*.
- Nombre asignado.
- Ubicación del nodo (tópico).

El primer paso es solicitar la lista de sensores y actuadores al servidor mediante la función *publicar*, esto se hace con el fin de que no exista nodos con el mismo nombre dentro de una ubicación. Una vez que el servidor responda la solicitud, se procede a conectarse al nodo en modo AP (cuando el nodo no tiene configuración trabaja como servidor en modo AP). La SSID del nodo tiene la serie ***MUSUXXXXX***, mientras que la contraseña es conocida por la aplicación ya que es única para todos los nodos. El dispositivo mediante la aplicación, escanea todas las redes WiFi disponibles y se conecta al nodo más cercano midiendo el RSSI, una vez conectado este envía un “ hola ” mediante una petición HTTP y el nodo inmediatamente enciende una

alerta. El usuario debe determinar si el nodo que dio la alerta es al que desea configurar, si es así, el nodo responde enviando: el tipo de nodo (sensor o actuador) y el número de interruptores en el caso de ser sensor. La aplicación almacena la información que respondió el nodo y arma la trama de configuración y envía al nodo. El último paso es esperar la confirmación del servidor de la nueva agregación. En el caso de que el nodo que dio la alerta, no es el que el usuario desea configurar, se procede a seguir buscando otros nodos cercanos. Todo el proceso se muestra en la Figura D.4

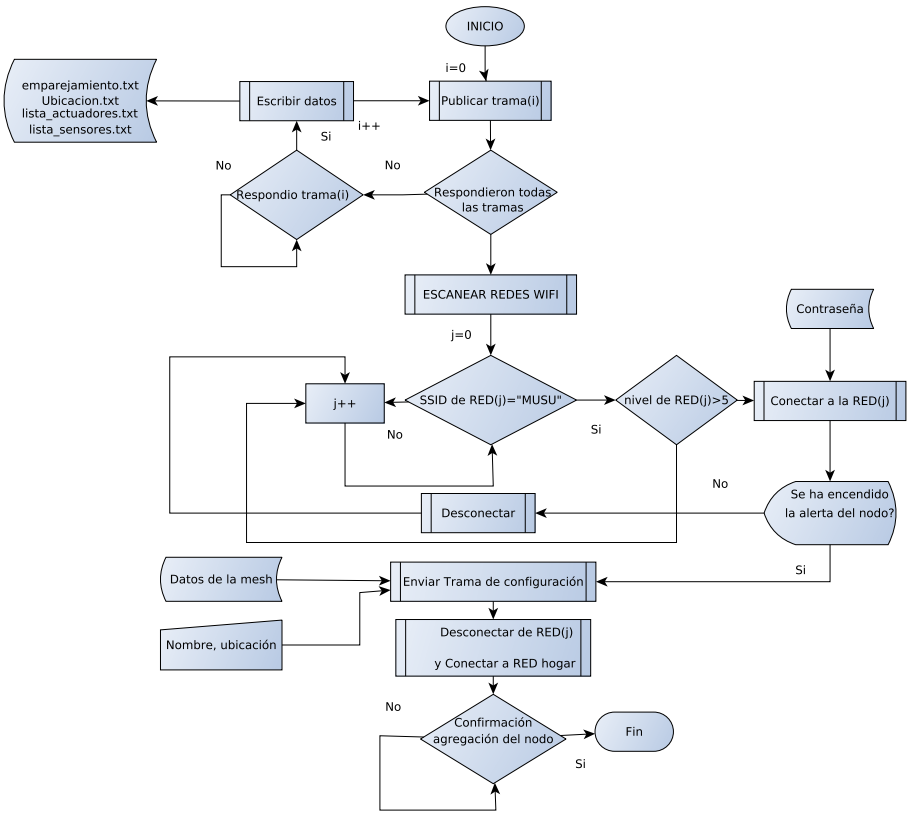

Figura D.4: Proceso de configuración de los nodos para que se agreguen a la red.

D.7. Configuración del emparejamiento de nodos en la red

El emparejamiento entre los nodos actuadores y sensores puede ser realizado en el momento que se agregan los sensores a la red, pero también se puede realizar las siguientes configuraciones una vez que los nodos estén en la red:

- **Agregar**: agrega un emparejamiento nuevo, sin importar que si el nodo ya tiene un

emparejamiento.

- **Cambiar**: elimina el emparejamiento anterior para realizar el nuevo emparejamiento.
- **Eliminar:** elimina el emparejamiento.

Estas funciones proveen al usuario la capacidad de gestionar el emparejamiento de los nodos en la red. Cualquiera de las configuraciones se puede realizar siempre y cuando los nodos estén dentro de la misma ubicación (tópico). El emparejamiento se lo realiza entre nodos sensores y actuadores únicamente.

D.7.1. Agregar

Esta función provee al usuario la opción de agregar n emparejamientos a un mismo nodo. Donde n es la cantidad de nodos disponibles para emparejamiento.

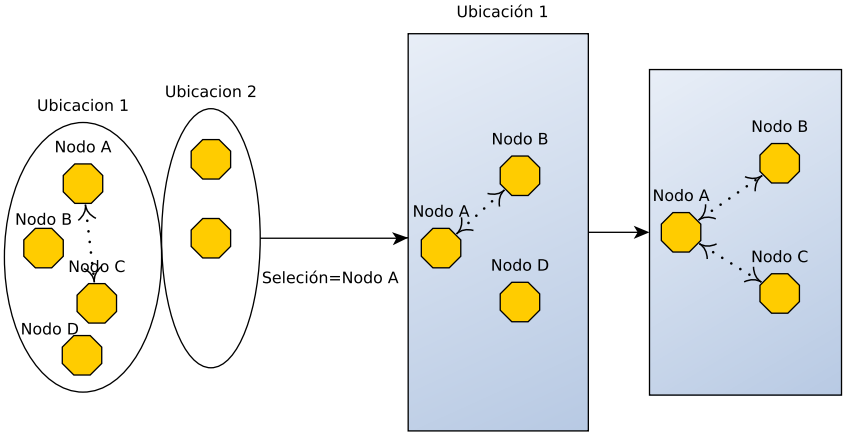

Figura D.5: Metodología para agregar un emparejamiento.

La Figura G.1 muestra la metodología para agregar un emparejamiento. Se solicita al servidor las lista de los actuadores, sensores y emparejamientos mediante la función *publicar*, se publica las tramas; Petición de lista de actuadores de la tabla estado, petición de ubicaciones, petición de información de sensores, petición de información de actuadores y petición de información de emparejamientos existentes en la red. Se publica las tramas de manera secuencial, esto se lo realiza con el fin de poder mostrar al usuario los nodos que pueden ser emparejados. Una vez que el servidor responde todas las tramas, el usuario debe seleccionar el tipo de nodo (actuadores o sensores) al que realizara el emparejamiento. El usuario selecciona un nodo A y se listan todos los nodos disponibles dentro de la ubicación del nodo A. Se selecciona un nodo B de la lista y se especifica las características y se arma la trama para el emparejamiento. El proceso se muestra en la Figura D.6.

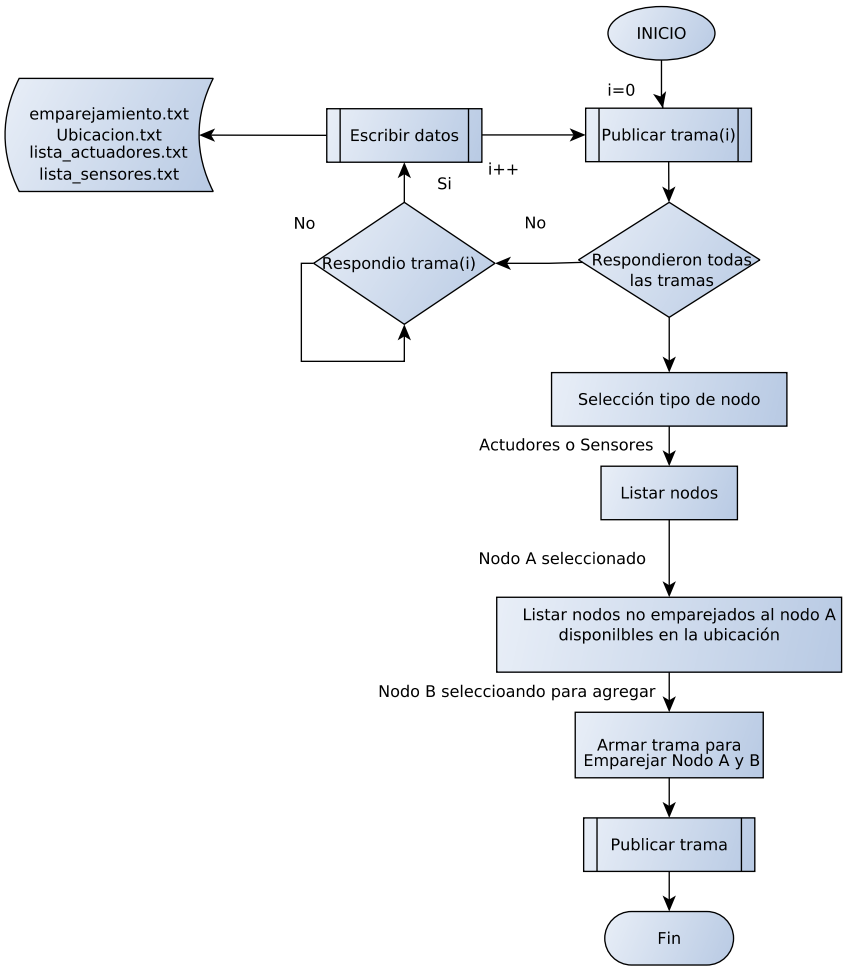

Figura D.6: Proceso para agregar un emparejamiento.

D.7.2. Cambiar

Esta función provee al usuario la opción de cambiar un emparejamiento ya establecido por uno nuevo, eliminado el anterior.

La Figura D.7 muestra la metodología para cambiar el emparejamiento entre nodos. Se solicita al servidor las lista de los actuadores y sensores. El usuario selecciona el tipo de nodo (Actuadores y sensores), posteriormente selecciona un nodo A y se muestran todos los sensores conectados a él. Se selecciona un Nodo B el cual esta emparejado para eliminar el emparejamiento, posteriormente se selecciona el nuevo nodo para el emparejamiento. Por ultimo se arma la trama y se publica. Todo el proceso se muestra en la Figura G.2.

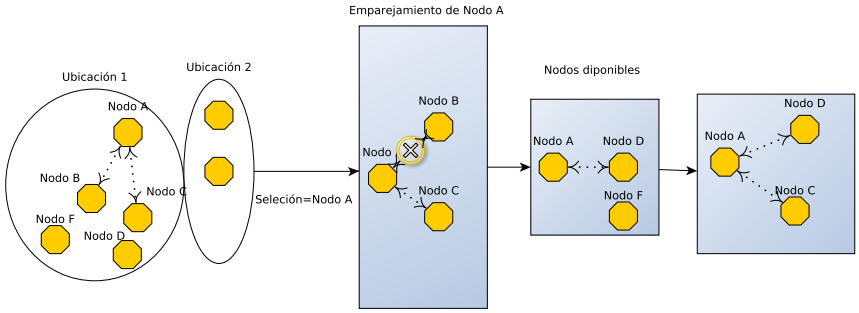

Figura D.7: Metodología para cambiar emparejamiento entre nodos.

D.7.3. Eliminar

Mediante esta función el usuario es capaz de eliminar un emparejamiento. La Figura G.3 muestra la metodología para realizar esta acción.

Se solicita al servidor las lista de los actuadores y sensores. El usuario selecciona el tipo de nodo (Actuadores y sensores), posteriormente selecciona un nodo A y se muestran todos los sensores conectados a él. Se selecciona un Nodo B el cual esta emparejado para eliminar el emparejamiento. Por último se arma la trama y se publica. Todo el proceso se muestra en la Figura G.4.

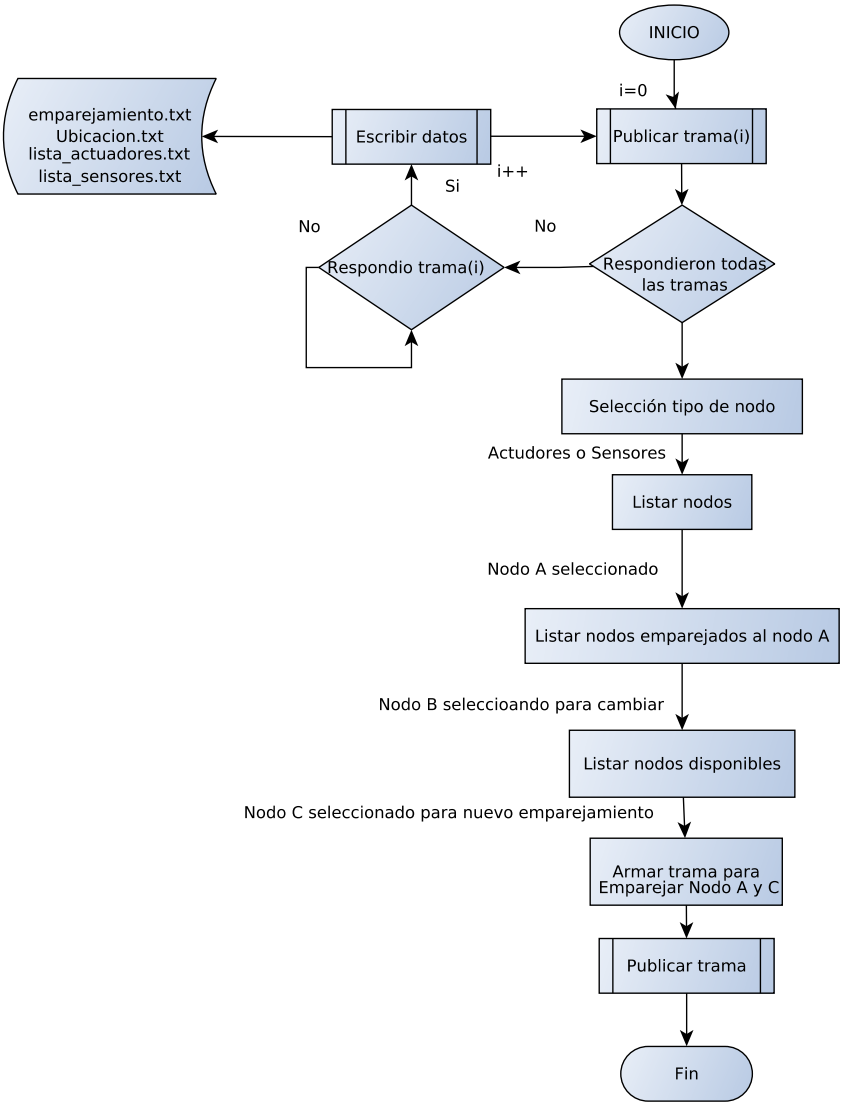

Figura D.8: Proceso para cambiar emparejamiento.

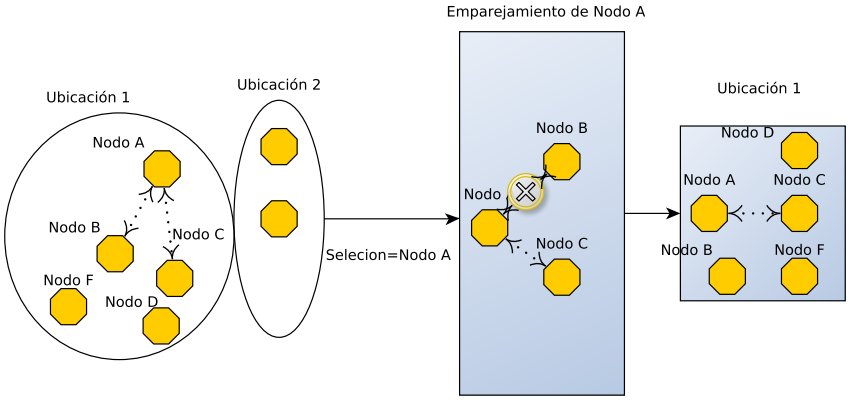

Figura D.9: Proceso para eliminar emparejamiento.

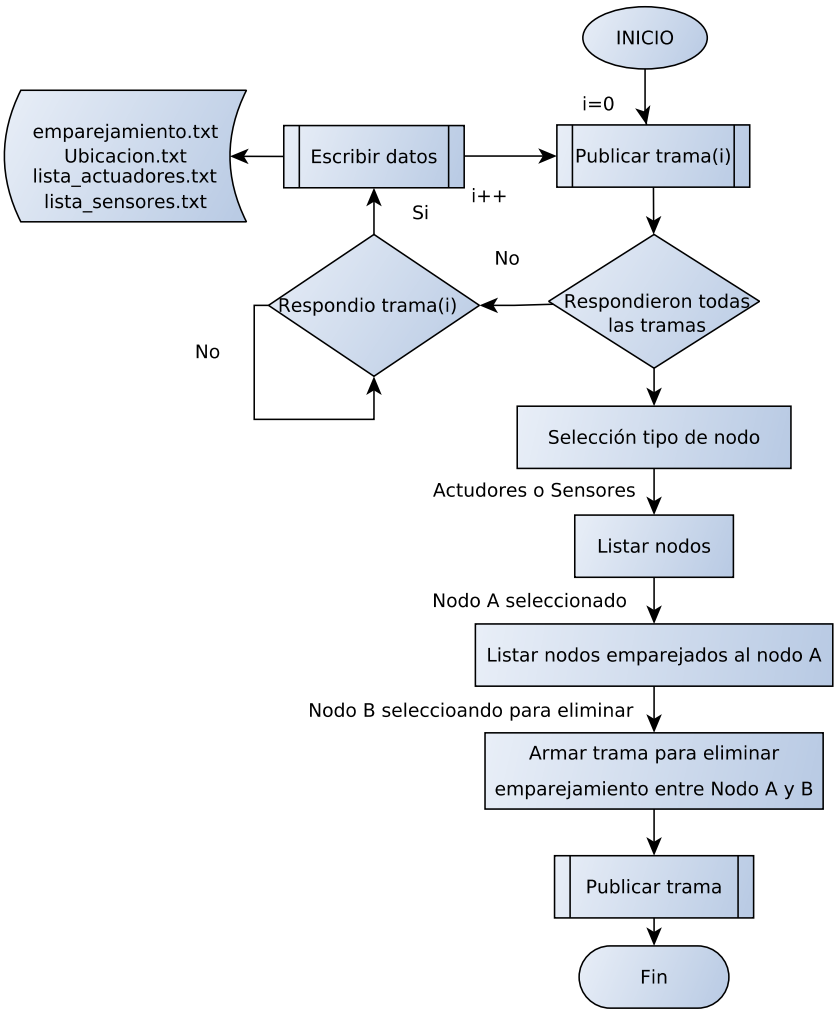

Figura D.10: Proceso para eliminar emparejamiento.

D.8. Control de actuadores

Mediante esta función el usuario acciona los actuadores que están dentro de la red. Se solicita el estado de los actuadores, se muestra el estado con un indicador en una lista, el usuario selecciona el actuador, se arma la trama y se publica; la aplicación espera la confirmación para cambiar de estado el indicador del actuador en la lista. Este proceso se muestra en la Figura D.11.

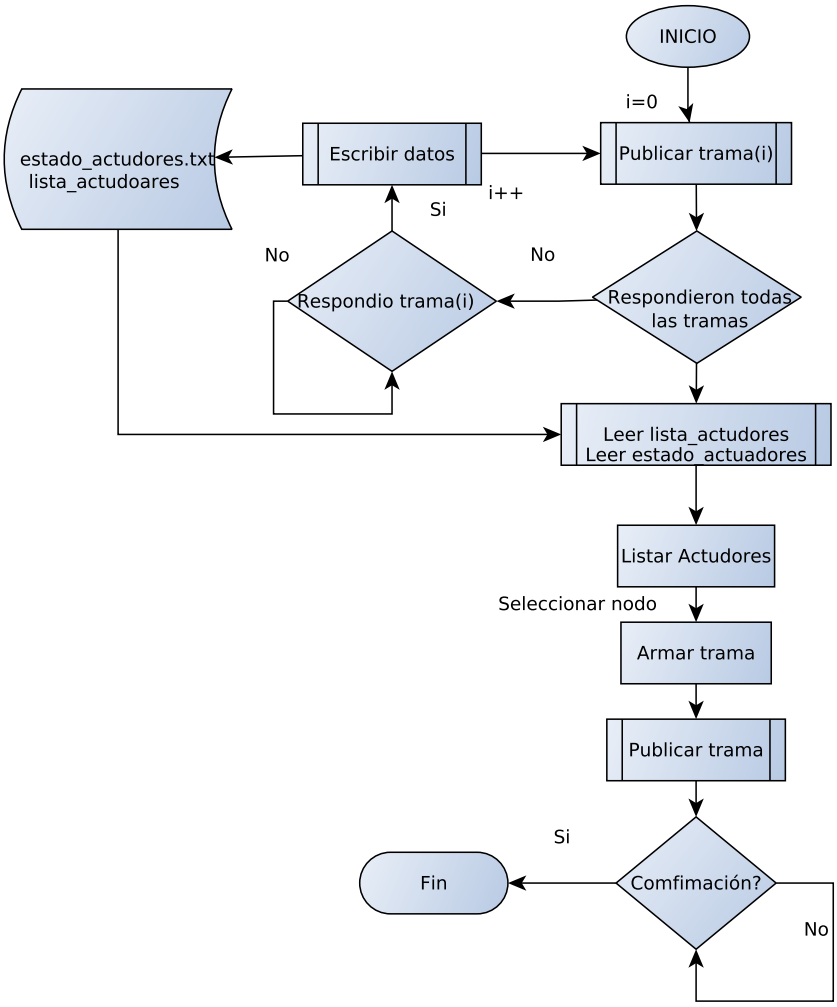

Figura D.11: Proceso para controlar los actuadores de la red.

D.9. Simulación de presencia

Este proceso activa o desactiva la simulación de presencia, mediante una trama con cabecera *2,10* para apagar y *2,9* para encender. El botón indica al usuario el estado del simulador. La Figura D.12 se muestra el proceso que se realiza.

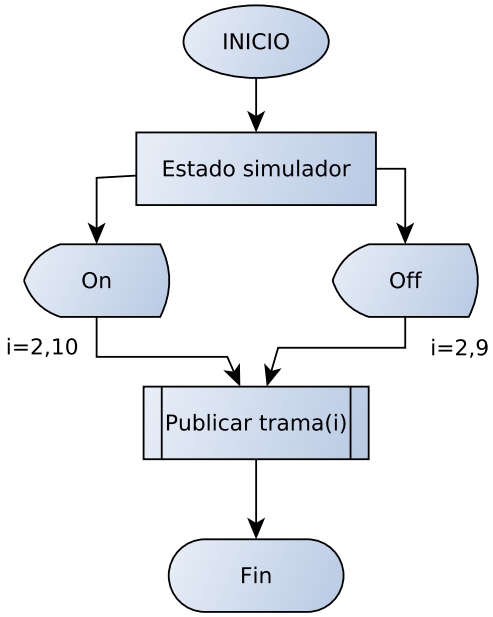

Figura D.12: Proceso para activar o desactivar la simulación de presencia.

Apéndice E

Instalación y configuración de la red neuronal

En este punto se detalla el procedimiento que se utilizó para instalar y configurar la red neuronal utilizada para la simulación de presencia.

El primer paso es instalar *NumPy*, es una biblioteca matemática para el lenguaje de programación *Python*, contiene una gran colección de funciones matemáticas de alto nivel para operar con matrices. El aprendizaje automático se trata de números y *Numpy* es una dependencia imprescindible para las bibliotecas y los *scripts* de aprendizaje automático. Esta librería se instala con el comando `sudo apt-get install python-numpy`

Luego se tiene que instalar *SciPy*, ésta es una biblioteca de código abierto de *Python* utilizada para informática científica e informática técnica. Se puede instalar *Scipy* usando los siguientes comandos:

Listado E.1: Dependencias de SciPy

```
1 sudo apt-get install libblas-dev
2 sudo apt-get install liblapack-dev
3 sudo apt-get install libatlas-base-dev
4 sudo apt-get install gfortran
5 sudo apt-get install python3-scipy
```

Theano es una biblioteca de Python que le permite definir, optimizar y evaluar expresiones matemáticas que involucran matrices multidimensionales de manera eficiente. Ésta es una herramienta de *deep-learning* que realiza todo el trabajo de aprendizaje detrás de *keras*. Se instala

ejecutando las líneas 1 y 2 del Listado E.2, mientras que la instalación de *keras* (es una API de redes neuronales de alto nivel, escrita en *Python* y capaz de ejecutarse sobre *TensorFlow* o *Theano*) es con la línea 3.

Listado E.2: Instalación Theano

```
1 sudo apt-get install python3-h5py
2 sudo pip install --upgrade --no-deps git+git://github.com/Theano/Theano.git
3 sudo pip install keras
```

Hay que asociar *Theano* con *keras*, y se lo hace asignando unas líneas de código al archivo `keras.json`, como se muestra a continuación:

Listado E.3: Asociación de Theano con Keras

```
1 "image_dim_ordering": "th"
2 "backend": "theano"
```

Apéndice F

Características del hardware

En esta Sección se detalla las características principales de los diferentes elementos de hardware utilizados en el proyecto.

F.1. Raspberry Pi

El Raspberry Pi 3 Modelo B es la tercera generación de Raspberry Pi. Esta poderoso computadora del tamaño de una tarjeta de crédito se puede usar para muchas aplicaciones. Mientras se mantiene el popular formato de *board*, el modelo Raspberry Pi 3B trae un procesador más poderoso, 10 veces más rápido que la primera generación Raspberry Pi. Además, agrega conectividad LAN inalámbrica y Bluetooth, por lo que es la solución ideal para diseños que requieran conexión [62].

F.1.1. Especificaciones

Las especificaciones del modelo 3B son:

- **Procesador:** Broadcom[1]BCM2387 chipset. 1.2GHz Quad-Core, ARM[2] Cortex-A53
- **Memoria:** 1GB LPDDR2
- **Conectividad:**

[1]https://www.broadcom.com/

[2]Procesador con arquitectura diseñada para utilizar menor número de transistores, con direccionamiento de 32 y 64 bits.

  - 802.11 b/g/n Wireless LAN.
  - Bluetooth 4.1.
  - 10/100 BaseT Ethernet socket
- **Acceso:** 40-pin 2.54 mm, 27 GPIO de +3.3 V, +5 V y GND.
- **Video y sonido:**
  - HDMI.
  - Salida de Audio 3.5mm jack, HDMI USB 4 x USB 2.0.
  - MIPI Camera Serial Interface (CSI-2).
- **Multimedia:**
  - Co-Processor.
  - Open GL ES 2.0.
  - hardware-accelerated OpenVG.
  - 1080p30 H.264 high-profile decode.
- **Soporte tarjeta SD:** Push/pull Micro SDIO.
- **Alimentación:** Micro USB socket 5V1, 2.5A

F.1.2. Beneficios y aplicaciones

Los principales beneficios son:

- Bajo costo
- Formato consistente de board.
- 10 veces más rápido.
- Conectividad.

Entre las principales aplicaciones estan:

- Laptop/tablet/PC de bajo costo
- Centro de medios
- Automatización Hogar/Industria.
- Servidores
- Cámara Web
- Puntos de acceso.
- Monitoreo de medio ambiente.
- Aplicaciones IoT
- Robótica.

F.2. Módulos WiF i: ESP8266-12E y ESP8266-07E

ESP8266EX ofrece una solución WiFi SoC altamente, integrada para satisfacer las necesidades de los usuarios; demandas continuas de uso de energía eficiente, diseño compacto y rendimiento confiable en el industria de Internet de las cosas. Con las capacidades de red WiFi completas e independientes, ESP8266EX puede realizar ya sea como una aplicación independiente o como el esclavo de una MCU anfitriona. Cuando ESP8266EX aloja la aplicación, se inicia inmediatamente desde el flash. El ESP8266EX se puede aplicar a cualquier diseño de microcontrolador como un adaptador de WiFi a través de Interfaces SPI / SDIO o I2C / UART [63].

ESP8266EX integra interruptores de antena, balún de RF, amplificador de potencia, recepción de bajo ruido amplificador, filtros y módulos de administración de energía. El diseño compacto minimiza el tamaño del PCB y requiere circuitos externos mínimos. La plataforma de conectividad inteligente (ESCP) de *Espressif Systems* permite funciones sofisticadas incluyendo el cambio rápido entre el modo de reposo y el de reactivación para un uso eficiente de la energía, radio adaptativa para operación de baja potencia, procesamiento de señal. En la Figura F.1 se presenta una tabla de las principales características del módulo ESP-12E [64] y ESP-07 [65].

CATEGORIAS	ITEMS	VALORES
Parametros wifi	Protocolos WIFI	802.11 b/g/n
	Rango de frecuenca	2.4GHz-2.5GHz (2400M-2483.5M)
PARAMETROS DE HARDWARE	Bus periferico	UART/HSPI/I2C/I2S/Ir
		GPIO/PWM
	Voltaje de Operación	3.0~3.6V
	Corriente de operación	80mA
	Rango de temp de operación	-40°~125°
	Rango de temp ambiente	25°
	Tamaño del dispositivo	16mm*24mm*3mm
	Interfaz externa	N/A
PARAMETROS DE SOFTWARE	Modo Wifi	station/softAP/SoftAP+station
	Seguridad	WPA/WPA2
	Encriptación	WEP/TKIP/AES
	Firmware	UART Download / OTA (via network) / download and write firmware via host
	Desarrollo de software	Cloud Server Development / SDK
	Protocolos de red	IPv4, TCP/UDP/HTTP/FTP
	Configuración del usuario	AT Instruction Set, Cloud Server, Android/iOS App

Figura F.1: Características de ESP8266 ESP-12 [6].

Apéndice G

Captura de las tramas de los mensajes con Wireshark

En este apartado se realiza la captura de las tramas de los mensajes de la Sub-sección 4.4.5 con un analizador de red, en específico utilizando *Wireshark*.

G.1. Mensajes de almacenamiento y configuración

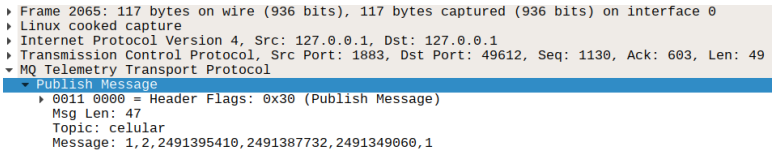

Figura G.1: Actualización del emparejamiento del nodo.

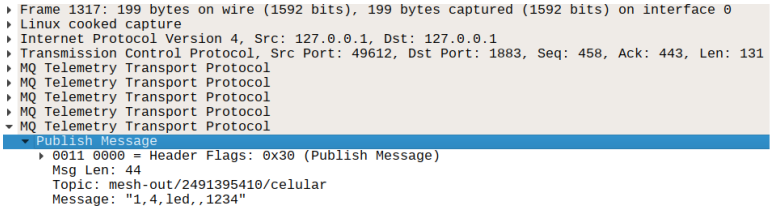

Figura G.2: Insertar nodo actuador a la red.

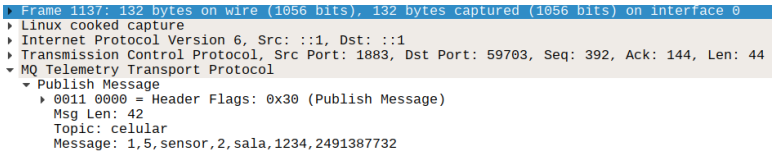

Figura G.3: Insertar nodo sensor a la red.

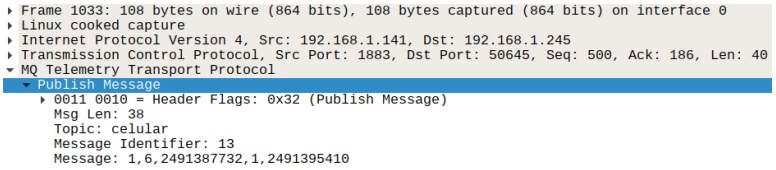

Figura G.4: Insertar emparejamientos de los nodos.

G.2. Mensajes de consulta

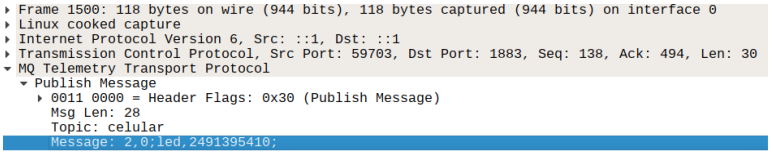

Figura G.5: Petición de lista de actuadores de la tabla estado.

Figura G.6: Petición de ubicaciones.

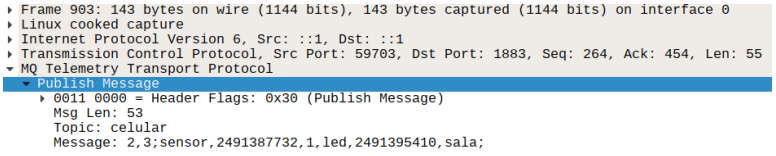

Figura G.7: Petición de información de emparejamientos existentes en la red.

Figura G.8: Petición de estado de actuadores.

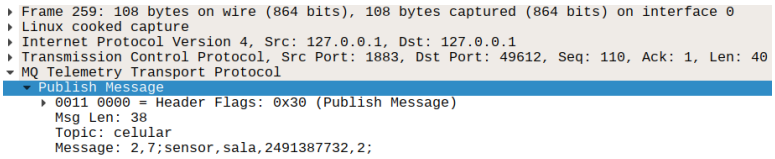

Figura G.9: Petición de información de sensores.

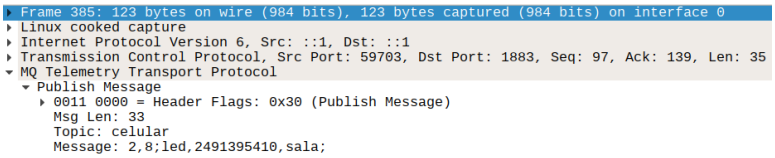

Figura G.10: Petición de información de actuadores.

G.3. Mensajes de control

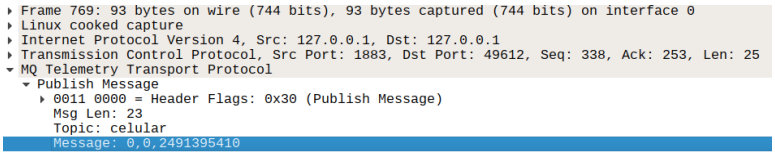

Figura G.11: Control de actuadores desde los sensores de la mesh.

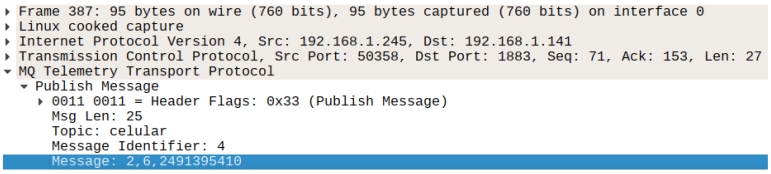

Figura G.12: Control desde el dispositivo móvil.

Bibliografía

[1] J. Haikara, "Publish-Subscribe Communication for CoAP Publish-Subscribe Communication for CoAP," KTH Royal Institute of Technology, Tech. Rep., 2017.

[2] T. Sauter, "The Three Generations of Field-Level Networks—Evolution and Compatibility Issues," *IEEE Transactions on Industrial Electronics*, vol. 57, num. 11, pp. 3585–3595, 2010.

[3] L. Á. Rocío, "Análisis e implementación de un sistema domótico Z-wave," Universidad de Sevilla, Tech. Rep., 2013.

[4] "Raspberry Pi Fundation," 2018. [En línea]. Disponible: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[5] A. Viktorov, "MQTT gateway for painlessMesh network," 2017. [En línea]. Disponible: https://github.com/latonita/painlessMeshMqttGateway

[6] "Datasheet: ESP-12E WiFi Module," Tech. Rep., 2015. [En línea]. Disponible: http://www.kloppenborg.net/images/blog/esp8266/esp8266-esp12e-specs.pdf

[7] P. A. Flores Siguenza, "Desarrollo de un bus propietario, para implementar una red en configuración maestro-multiesclavo." Master's thesis, Escuela Superior Politécnica de Chimborazo, 2017.

[8] S. S. R. Valenzuela, M. D. S. Laguna, y J. A. Holgado, "Control domótico del hogar a través de una plataforma de servicios distribuida basada en jxta," in *II Simposio en Desarrollo de Software*. Universidad de Granada, 2008, pp. 219–234, iSBN: 978-84-96856-71-4.

[9] V. Autores, "Empresa de domótica en madrid | diseñamos espacios inteligentes," urlhttp://www.masespacio.eu, 2018.

[10] ——, "Soluciones domoticas y electrónicas," urlhttp://www.domoticaecuador.com, 2018.

[11] ——, "Smart home quito - casas inteligentes - smarthome quito," urlhttp://www.smarthomequito.ec, 2018.

[12] M. Suyama, "Protocolos de comunicaciones," urlhttps://desarrolloweb.com/articulos/1617.php, 2004.

[13] J. A. Guerrero Meseguer *y otros*, "Diseño de una instalación domótica con tecnología lonworks," Universidad Politécnica de Cartagena, Tech. Rep., 2010.

[14] Zigbee.org, "Zigbee alliance," urlhttp://www.zigbee.org/, 2018.

[15] Z-Wave, "Safer, smarter homes start with z-wave," urlhttp://www.z-wave.com/, 2004.

[16] W. Kastner y W. Tumfart, "Remote control of eib systems based on virtual shared group objects," in *Factory Communication Systems, 2002. 4th IEEE International Workshop on*. IEEE, 2002, pp. 63–70.

[17] H. Merz, T. Hansemann, y C. Hübner, *Building Automation: Communication Systems with EIB/KNX, LON and BACnet*. Springer Science & Business Media, 2018.

[18] B. E. Markwalter, S. K. Fitzpatrick, P. Hargaden, y S. Appling, "Design influences for the cebus automation protocol," *IEEE transactions on consumer electronics*, vol. 37, num. 2, pp. 145–153, 1991.

[19] *BACnet Basics User's Guide*, CARRIER CORPORATION, 2013.

[20] J. Gutiérrez, "Las redes substitución-permutación y el aes (advanced encryption standard)," *Protocolos criptográficos y seguridad en redes*, p. 86, 2003.

[21] E. Niemela y T. Vaskivuo, "Agile middleware of pervasive computing environments," in *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*. IEEE, 2004, pp. 192–197.

[22] L. F. Herrera Quintero, "Viviendas inteligentes (domótica)," *Ingeniería e Investigación*, vol. 25, num. 2, pp. 47–52, 2005.

[23] E. Frontoni, D. Liciotti, M. Paolanti, R. Pollini, y P. Zingaretti, "Design of an interoperable framework with domotic sensors network integration," in *Consumer Electronics-Berlin (ICCE-Berlin), 2017 IEEE 7th International Conference on*. IEEE, 2017, pp. 49–50.

[24] B. A. B. Granda, L. A. B. Belduma, E. J. C. Gonzalez, y A. F. S. Sarango, "Designing a wireless sensor network for vehicular traffic and co2 pollution monitoring in an urban area," in *Communications (LATINCOM), 2017 IEEE 9th Latin-American Conference on*. IEEE, 2017, pp. 1–6.

[25] C. D. Ramirez, R. Sanabria Bocanegra, y M. Suarez Sierra, "Integración de sensores inálambricos y domótica," Corporación Universitaria Minuto de Dios, Tech. Rep., 2011.

[26] Dave Locke, "Mq telemetry transport (mqtt) v3.1 protocol specification." 2010. [En línea]. Disponible: https://www.ibm.com/developerworks/library/ws-mqtt/index.html

[27] M. Bani Yassein, M. Q. Shatnawi, S. Aljwarneh, y R. Al-Hatmi, "Internet of Things: Survey and open issues of MQTT Protocol," 2017, pp. 1–6.

[28] B.-y. Sung, K.-b. Kim, y K.-w. Shin, "An AES-GCM Authenticated Encryption Crypto-Core for IoT Security."

[29] J. Balamurugan y E. Logashanmugam, "High speed low cost implementation of advanced encryption standard on fpga," in *Current Trends in Engineering and Technology (ICCTET), 2014 2nd International Conference on.* IEEE, 2014, pp. 371–375.

[30] M. J. Dworkin, "Recommendation for block cipher modes of operation: Galois/counter mode (gcm) and gmac," Tech. Rep., 2007.

[31] K.-B. Kim, W.-L. Cho, Y.-C. Jang, y K.-W. Shin, "An efficient implementation of aria and aes block cipher algorithms supporting four modes of operation," *Conference on the Institute of Electronics Engineers of Korea*, pp. 711–713, 2017.

[32] D. Selent, "Advanced encryption standard," *Rivier Academic Journal*, vol. 6, num. 2, pp. 1–14, 2010.

[33] C. Hall y N. Ferguson, "Chapter 7 The Advanced Encryption Standard (AES)," num. November, pp. 58–73, 2001.

[34] Giovanni Blu Mitolo, "Cape 3.0." [En línea]. Disponible: https://github.com/gioblu/Cape

[35] J. C. Galende Díaz, "Principios básicos de la criptología," *Documenta & Instrumenta*, vol. 4, pp. 47–59, 2006.

[36] P. R. Lakhe, "A technology in most recent processor is complex reduced instruction set computers (crisc): A survey," *International Journal of Innovation Research and Studies*, vol. 2, num. 6, pp. 711–715, 2013.

[37] david Garcia, "Sistema de gestión domotica de una vivienda," Tech. Rep., 2010.

[38] "X10," 2017. [En línea]. Disponible: https://www.x10.com/x10-basics.html

[39] "ModBus," 2018. [En línea]. Disponible: http://www.modbus.org

[40] "BatiBus," 2018. [En línea]. Disponible: https://www.kunbus.com

[41] A. Guerrero, "Diseño de Una Instalación Domótica con Tecnología Lonworks," Tech. Rep., 2010.

[42] "Zigbee Alliance," 2018. [En línea]. Disponible: http://www.zigbee.org/

[43] "EnOcean," 2018. [En línea]. Disponible: https://www.enocean.com/en/

[44] "Safe Smarter Z-Wave," 2018. [En línea]. Disponible: http://www.z-wave.com/

[45] M. C. J. C, A. C. R. S, C. H. J. C, y V. T. P. E, "Mobile Applications Using TCP / IP-GSM Protocols Applied to Domotic." IEEE, 2015, pp. 1–4.

[46] M. T. Marginean y C. Lu, "SDOMO - A simple communication protocol for home automation and robotic systems," vol. 2015-August, 2015.

[47] C. A. M. Bolzani, C. Montagnoli, y M. L. Netto, "Domotics over ieee 802.15. 4-a spread spectrum home automation application," in *Spread Spectrum Techniques and Applications, 2006 IEEE Ninth International Symposium on*. IEEE, 2006, pp. 396–400.

[48] M. Rodríguez Cerezo, "Sistema de control remoto para aplicaciones domóticas a través de internet," B.S. thesis, 2014.

[49] R. P. Esteban, "Simulación y control mediante PLC de una vivienda," Tech. Rep., 2013.

[50] Lambda, "Simulador de presencia," 2016. [En línea]. Disponible: http://lambdadomotica.com/simulador-presencia/

[51] R. P. Foundation, "Raspbian," 2018. [En línea]. Disponible: https://www.raspberrypi.org/downloads/raspbian/

[52] "Arduino core for ESP8266 WiFi chip," 2018. [En línea]. Disponible: https://github.com/esp8266/Arduino

[53] G. Martin, "painlessMesh," 2018. [En línea]. Disponible: https://github.com/gmag11/painlessMesh

[54] N. Foundation, "Acerca de Node.js," 2018. [En línea]. Disponible: https://nodejs.org/es/about/

[55] *Introducción a las redes neuronales aplicadas*. Manual Data Mining, 2007.

[56] G. Hall, *El ojo: I. Óptica de la visión*, 1998, pp. 609–620. [En línea]. Disponible: http://ual.dyndns.org/biblioteca/fisiologia/Pdf/Unidad10.pdf

[57] A. Piper, "Mosca," 2013. [En línea]. Disponible: https://github.com/mqtt/mqtt.github.io/wiki/mosca

[58] M. E. Suquinagua R. (2018, jul) Sistema domótico con red de malla aplicando mqtt. [En línea]. Disponible: https://github.com/RaulSuquinagua/Domotic_Mesh_MQTT

[59] E. Foundation. (2018, feb) Eclipse paho java client. [En línea]. Disponible: https://github.com/eclipse/paho.mqtt.java

[60] E. Media, *Android Programming Cookbook*. P.C, 2016, vol. 1.

[61] J. Baptiste. (2009-2011, jan) Network discovery. [En línea]. Disponible: https://github.com/rorist/android-network-discovery

[62] R. P. Fundation. (2018, jan) Raspberry pi fundation. [En línea]. Disponible: http://docs-europe.electrocomponents.com/webdocs/14ba/0900766b814ba5fd.pdf

[63] (2018, jan) Esp8266ex datasheet. [En línea]. Disponible: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

[64] (2018, jan) Esp-12 wifi module. [En línea]. Disponible: https://www.kloppenborg.net/images/blog/esp8266/esp8266-esp12e-specs.pdf

[65] (2018, jan) Esp-07 wifi module. [En línea]. Disponible: https://www.mikrocontroller.net/attachment/338570/Ai-thinker_ESP-07_WIFI_Module-EN.pdf