UACM
Universidad Autónoma
de la Ciudad de México
Nada humano me es ajeno

COLEGIO DE CIENCIA Y TECNOLOGÍA

LICENCIATURA EN INGENIERÍA EN SISTEMAS ELECTRÓNICOS Y DE TELECOMUNICACIONES

Sistema de alarma automatizado.

TESIS

PARA OBTENER EL TÍTULO DE

LICENCIADO EN INGENIERÍA EN SISTEMAS ELECTRÓNICOS Y DE TELECOMUNICACIONES

P R E S E N T A

GUSTAVO DAMAZO GARCÍA

D I R E C T O R A

Mtra. Magali Cortez Vázquez

Ciudad de México, junio 2019

UACM
Universidad Autónoma de la Ciudad de México
Nada humano me es ajeno

CA Coordinación Académica

SISTEMA BIBLIOTECARIO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD AUTÓNOMA DE LA CIUDAD DE MÉXICO
COORDINACIÓN ACADÉMICA

RESTRICCIONES DE USO PARA LAS TESIS DIGITALES

DERECHOS RESERVADOS©

Contenido

ÍNDICE DE FIGURAS

CAPÍTULO Ⅰ

CAPÍTULO Ⅱ

CAPÍTULO Ⅲ

ÍNDICE DE TABLAS

CAPÍTULO Ⅱ

CAPÍTULO Ⅲ

Anexos

AGRADECIMIENTOS

Le agradezco a mi madre **Francisca García Martínez** por la ayuda durante estos años, de igual importancia le agradezco a dos personas muy importantes en mi vida, sin su motivación jamás hubiera decidido estudiar una carrera universitaria.

A mi directora de tesis **M. en C. Magali Cortez Vázquez** por haberme brindado su tiempo y el apoyo en la realización de este trabajo. Sin ella no lo hubiera logrado. Gracias.

A mis lectores **M. en C. José Ignacio Castillo Velázquez**, **M. en C. Joel Yazbet Buendía Gómez** y al **Ing. Ricardo Galindo Reyez**, los cuales brindaron su tiempo para ayudarme a mejorar y corregir mi trabajo. Gracias por el apoyo.

Por último, agradezco a la **Universidad Autónoma de la ciudad de México** por brindarme la oportunidad para continuar con mis estudios y cumplir una meta más en mi vida, así mismo agradezco a la institución por el apoyo para el empastado e impresión de este trabajo recepcional.

CAPÍTULO Ⅰ : INTRODUCCIÓN

Introducción

En esta sección se exponen los antecedentes, problemática, objetivos generales y específicos que sustentan el diseño e implementación de un sistema de alarma automatizado. Asimismo, se dará un breve vistazo a los materiales utilizados en el desarrollo de dicho sistema, materiales tales como lo son algunos sensores y tarjetas de desarrollo.

Este trabajo recepcional se centra en el diseño e implementación de un sistema de alarma automatizado implementado con una red de sensores, mismo que llevará un registro de actividades en una base de datos MySQL, todo esto vía una red de comunicación tipo Ethernet. Además, el sistema activará una alarma para alertar de una intrusión y ahuyentar a posibles invasores. Cabe señalar que los datos recabados por el sistema pueden ser utilizados de la forma que mejor disponga el dueño de la vivienda. La finalidad es únicamente diseñar un sistema de alarma que brinde el servicio de monitoreo automatizado.

En adición al punto anterior, en un futuro un sistema de este tipo podría ofrecer un mayor número de servicios que no solo hagan posible alertar de intrusiones, sino que también logré la automatización de una vivienda de una manera económica y, con ello, se pueda dar solución a posibles accidentes provocados por fugas de gas o derrames de agua, entre otros. Dicho lo anterior, en este trabajo recepcional se incluye en el sistema un sensor de calidad del ambiente, es decir, un sensor que pueda detectar cierto tipo de partículas en el aire, que, en caso de tener una lectura por parte del convertidor analógico digital (CAD) superior a un umbral definido, active un pequeño ventilador y con ello disminuya la probabilidad de que exista algún agente contaminante en el interior de la vivienda.

Para fines demostrativos, se presenta una maqueta de una vivienda, en la cual se realizan pruebas del sistema, una vez que todos sus sensores han sido probados, con la finalidad de darles el uso óptimo.

A continuación, se muestran los antecedentes de los sistemas de alarma con la finalidad de tener un mayor contexto de este trabajo.

Ⅰ.1 Antecedentes

Uno de los primeros sistemas de seguridad fue creado por Augustus Russell Pope, él creó una alarma utilizando pilas, una campana e imanes que reaccionaban a un circuito eléctrico controlado por las ventanas y puertas de una casa. Este sistema de alarma es el antecesor a lo que conocemos como alarma antirrobo.

Previo a la primera patente de un sistema de alarma lo que se solía emplear eran perros o algún sistema que hiciera ruido, todo esto se sigue utilizando actualmente y sigue siendo útil, pero la ventaja que tienen algunos sistemas de alarma actuales es que pueden alertar a las autoridades competentes.

Con el invento de Augustus Russell Pope los sistemas de alarma comenzaron a expandirse, todo esto gracias a empresarios, los cuales vieron un mercado, es decir, el producto es y era rentable. Una de las primeras empresas que ofrecía sistemas de alarma fue creada por Edwin Holmes, la *Holmes Electronic Protection*. La empresa brindaba un sistema de alarma eléctrico [1].

El siguiente paso fue crear un sistema de alarma que brindara una respuesta rápida a una intrusión no deseada. Edwar A. Calahan, quien fuera telegrafista de profesión, desarrolla un teletipo para la bolsa de valores para dar aviso oportuno a los inversionistas con respecto al cambio en la bolsa. La invención de Edwar A. Calahan llevó por nombre **caja de llamadas**. Gracias a esto y con la ayuda de Calahan se crea una de las primeras empresas dedicadas a responder al llamado de emergencia en caso de ser víctimas de un delito, la *American Distric Telegraphic* [2].

Al paso del tiempo los sistemas de alarma han ido evolucionando, en el año 1970 se implementan los sensores de movimiento, posteriormente se integraron los sistemas de seguridad inalámbricos. Una de las empresas que actualmente tiene cobertura a nivel mundial es la *Abus Security Tech Germany*, esta empresa hace la fusión entre sistemas mecánicos y eléctricos, todo esto en el año 2008. Dicha empresa, al igual que otras, cuenta con una pequeña línea de tiempo, la cual puede ser visitada en su página oficial y en la que muestran la historia de los sistemas de

alarmas [3]. En la figura Ⅰ.1 se muestra una línea de tiempo propuesta por dicha empresa con todo lo anteriormente mencionado.

Previo a los sistemas de seguridad se utilizaban perros guardianes o algún mecanismo que hiciera ruido.

1857: Edwin Holmes, fundador de la primera empresa de instalaciones de alarmas eléctricas, compra los derechos sobre el invento de Pope.

1871: Calahan ayuda a crear la empresa American Distric Telegraph, cuya finalidad era responder a la brevedad a las llamadas emergentes en caso de ser víctima de algún incidente.

Entre 1980 y 1990. Aparecen los primeros sistemas de seguridad inalámbricos.

1853: Augustus Russell Pope diseña y patenta el primer sistema de seguridad, primera alarma electromagnética del mundo.

1867: Edward A. Calahan, telegrafista de profesión, desarrolla el primer teletipo para el oro y la bolsa, esto permite a los inversionistas saber de forma más rápida de los cambios de divisas de Wall Street.

1970: Se integran los primeros detectores de movimiento a los sistemas de seguridad.

2008: Desarrolladores de productos de seguridad de Abus fusionan la mecánica con la electrónica para crear nuevos sistemas de alarma.

ABUS
Security Tech Germany

Figura Ⅰ 1. Línea de tiempo en donde se muestran pasos importantes respecto a los avances de sistemas de alarma.

Como se muestra en la figura Ⅰ.1, los sistemas de alarma han dado grandes pasos desde sus inicios con el uso de animales para dar alerta de intrusos, hasta la implementación de sistemas cada vez más complejos y eficientes. Dicho esto, se muestra cómo el monitoreo ha sido de gran ayuda para una ágil respuesta ante una intrusión.

Respecto al punto anterior, una de las tareas del sistema de alarma de este proyecto es el monitoreo de la vivienda en todo momento, esto apoyándose con la base de datos MySQL la cual guardará registro de las actividades, pero ¿qué es el monitoreo?

La Real Academia de la Lengua Española (RAE) define el monitoreo como la acción de vigilar mediante un monitor. Es de importancia resaltar el hecho de que el monitoreo nace de la automatización de inmuebles.

La idea de automatizar inmuebles viene de los años 80 cuando en Estados Unidos y Japón ya comenzaban a implementar los llamados edificios inteligentes con la finalidad de gestionar mejor los recursos de éstos. Actualmente, a la idea de automatizar los servicios se le conoce como Internet of Things (IoT). IoT, como una tecnología de vanguardia, presenta una revolución que busca la interconexión de todos los servicios que las personas puedan tener en su vivienda, tales como son: control de iluminación y calefacción, entre otros. Es por lo anterior, que cada vez un mayor número de ciudades y países alrededor del mundo se suman a la automatización para hacer de sus hogares y edificios en general, lugares mucho más cómodos y agradables de habitar.

Teniendo en cuenta lo anterior, el diseño del sistema de seguridad únicamente recoge algunos elementos empleados en sistemas de alarma y automatización, con ello y con la ayuda de una base de datos para llevar un monitoreo de las actividades de la vivienda, se propone una solución para que los usuarios que son víctimas de intrusiones puedan contar con un sistema que les alerte de éstas de forma eficiente.

Ⅰ.2 Situación problemática

Definitivamente existen diversos problemas relacionados con la delincuencia, por ejemplo, robos a transeúntes, secuestros, extorciones, etc. En concreto, es por lo anterior mencionado que el problema al cual se le brinda una posible solución con este trabajo es el de robo a casa habitación. Para ilustrar mejor esta problemática, a continuación, se muestran algunas estadísticas que evidencian el robo a casa habitación como un problema de gran magnitud.

El *Observatorio Nacional Ciudadano* muestra que, de febrero de 2016 a enero de 2017, el robo a casa habitación tuvo una variación de 3.54 % pasando de una tasa de 5.65 a 5.85 de carpetas de investigación, todos estos datos en tasas por cada 100 mil habitantes. Asimismo, el *Instituto Nacional de Estadísticas y Geografía* (INEGI) muestra que del 2015 al 2016 por cada 100 mil habitantes hubo un descenso en víctimas a casa habitación pasando de 2,496 a 2,437. De igual importancia en el año 2015 se muestra un porcentaje de robos sin violencia de 90.75 % y un 9.25 % de robos con violencia a casa habitación, cifras que prácticamente no cambiaron al 2017, en donde se muestra un 9.34 % de robos con violencia y un 90.66 % robos sin violencia. Por lo anterior mencionado, se puede deducir que por cada 100 habitantes 2 a 3 son víctimas de este modo de robo [4], [5]. Es por ello por lo que los sistemas de seguridad y alerta son necesarios.

Ⅰ.3 Objetivo general

Diseñar un sistema de alarma con monitoreo que permita a los usuarios vigilar su vivienda en todo momento cuando se encuentran fuera de ella, y que intimide y ahuyente a posibles asaltantes, para reducir el número de accidentes y prevenir robos a éstas.

Ⅰ.4 Objetivos específicos

- Implementar un sistema de alarma que alerte de intrusiones utilizando tarjetas de desarrollo y sensores.
- Estructurar un sistema de comunicación para poder enviar de manera continua un registro a una base de datos MySQL vía Ethernet, en la cual se almacenarán actividades de la vivienda.

- Crear un servidor para poder comunicar el sistema de alarma con la base de datos.

Ⅰ.5 Justificación

La problemática de los robos es un hecho, uno de los ideales de la Universidad Autónoma de la Ciudad de México (UACM) es formar profesionistas que puedan dar solución a los problemas de los ciudadanos. Este sistema de alarma busca la prevención, además de proporcionar tranquilidad a los usuarios.

Otro motivo por el cual se desea implementar este sistema, es que con ello se aplica gran parte de los conocimientos que un ingeniero en electrónica y telecomunicaciones debe poseer.

Ⅰ.6 Limitaciones

Al no contar con una vivienda de prueba se realizó una maqueta con la finalidad de emular el inmueble. Todo esto en el laboratorio B207 del Plantel San Lorenzo Tezonco de la UACM. Por otro lado, se utiliza una red de área local (Local Area Network - LAN), la cual tiene la misma función demostrativa que si se utilizará una red amplia (Wide Area Network - WAN).

Ⅰ.7 Viabilidad

Llevar a cabo un sistema que alerte de intrusiones y posibles problemas en el hogar es viable ya que todos los componentes necesarios se encuentran disponibles en la universidad. El ser viable también hace que el proyecto en un futuro pueda crecer, ser mejorado e implementado de forma que genere un beneficio social y económico.

Además de los materiales proporcionados por la UACM, el material extra como cableado y construcción de la maqueta no representó un costo grande.

Ⅰ.8 Metodología

La realización de este proyecto utilizó la siguiente metodología, ésta enumera los procedimientos realizados, los cuales son:

1.- Buscar información en noticieros y diarios para justificar la necesidad de este proyecto, así como de los actuales fabricantes de sistemas de seguridad, para identificar aspectos que podrían mejorarse.

2.- Consultar manuales y fichas técnicas de los diferentes dispositivos a emplearse en el sistema, a fin de poder diseñar un sistema competente.

3.-Diseñar el sistema como un todo, es decir, el sistema de sensores, el mecanismo para la concentración de datos, almacenamiento de la información e identificación de las características de los servidores adecuados para el proyecto.

4.-Realizar pruebas a los sensores del sistema en el laboratorio B207, para constatar que su rendimiento es óptimo. Realizadas todas las pruebas, los sensores se instalan en la maqueta de prueba para verificar su funcionamiento, por último, se realizan diversas pruebas de comunicación entre el sistema de alarma y el servidor en donde se almacenará toda la información recolectada por los sensores.

5.-Realización de pruebas en varios equipos de cómputo y sistemas operativos para comprobar la compatibilidad del sistema.

6-Generación de recomendaciones para el usuario.

A lo largo de este documento se describe cómo diseñar, instalar y configurar un sistema de alarma para poder salvaguardar la seguridad de los usuarios.

De igual importancia, en el marco teórico se muestra un poco de historia y el diseño del sistema y de todos sus componentes. Posteriormente, se muestran los resultados de una serie de pruebas y por último un análisis de los resultados y conclusiones.

CAPÍTULO Ⅱ : MARCO TEÓRICO

Marco teórico

En el presente capítulo se muestra el desarrollo del sistema de alarma haciendo énfasis en los recursos empleados para la creación de éste, desde los sensores, tarjetas de desarrollo, diseño e implementación.

Ⅱ.1 Introducción al monitoreo

El monitoreo tiene su origen en la palabra monitor, el cual no necesariamente es un monitor de computadora, es decir, monitorear algo significa darle seguimiento con alguna finalidad específica. Para este trabajo, éste es un sistema que sirve para tomar fotos, video o detectar alteraciones en una vivienda, para visualizar todo esto en una pantalla. Un sistema de monitoreo puede brindar estos servicios de forma autónoma, es decir, el sistema informa al usuario acerca de posibles intrusiones o cambios en el sistema.

Ⅱ.2 Definición de monitoreo

Una forma para definir el monitoreo, para este caso de una vivienda, es la siguiente, de acuerdo a la RAE *"sistema cuya finalidad es llevar un registro de las actividades en una vivienda y que a su vez alertará al propietario de ésta de forma autónoma cada que ocurra alguna anomalía".*

Por otro lado, los sistemas inteligentes, como se les suele llamar, son sistemas automatizados y no son recientes, tienen sus orígenes en la década de los setenta. Para entender mejor dichos sistemas, y cómo es que éstos surgieron, a continuación se explica cómo nacen los primeros edificios inteligentes.

Ⅱ.3 Edificios inteligentes

Para comprender cómo surgen los edificios inteligentes y la innovación que éstos trajeron se da a continuación una breve definición de **edificio** e **inteligencia**.

La definición de edificio según RAE es *"construcción estable, hecha con materiales resistentes, para ser habitada o para otros usos"*. Asimismo, la RAE define la palabra inteligente de varias formas, una de ellas, la cual es de interés para el presente trabajo, dice *"dicho de un sistema de un edificio, de un mecanismo, etc; que están controlados por computadoras y son capaces de responder a cambios*

del entorno para establecer las condiciones óptimas de funcionamiento sin intervención humana". El siguiente apartado explica brevemente cómo surgen los edificios inteligentes.

Ⅱ.3.1 Surgimiento de edificios inteligentes

La idea de edificios inteligentes surge en la década de los setenta como respuesta a la crisis energética que se da tras estallar la guerra del **Yon Kippur**, guerra entre árabes e israelíes [6]. Dicha guerra hizo que se pensara que el precio del petróleo aumentaría. Como consecuencia de la crisis los arquitectos e ingenieros de la época comenzaron el desarrollo de inmuebles automatizados, todo esto con la finalidad de que el inmueble ahorrará energía y operará de forma eficiente utilizando el mínimo de recursos [7].

Actualmente, la automatización de edificios es una labor muy recurrente, se emplea en casi todo tipo de inmuebles, desde una vivienda común hasta en instituciones de educación, un ejemplo de esto se encuentra en la UACM, la cual cuenta con detectores de presencia en casi todas las aulas de los edificios y cuya finalidad es regular el consumo de energía. La disciplina que se encarga de llevar a cabo todo lo anterior mencionado es la **domótica**.

Ⅱ.4 ¿Qué es la domótica?

La domótica proviene de la unión de dos palabras del latín: *domus* que significa casa y *tica* que significa automática, es decir, una casa automática. La RAE define la domótica *como un "conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda"*

Es decir, la domótica se encarga de que los diferentes dispositivos electrónicos destinados a tareas específicas se unan y formen un **sistema,** para que trabajen entre sí, pero ¿qué es un sistema? Un sistema es un conjunto de elementos trabajando entre sí de forma lógica para lograr llevar a cabo una meta. De este modo los diferentes dispositivos tales como los sensores, entre ellos, ultrasónico, de vibración, gas y proximidad además de controladores y sistemas de ventilación, riego e iluminación, serán aquellos que en conjunto formarán un único sistema. Lo

mencionado anteriormente trae consigo grandes beneficios al ser implementado para automatizar un edificio, destacando el ahorro de energía y recursos.

La siguiente es una breve lista de los beneficios que tiene la automatización de un edificio:

1.- Comodidad para el usuario: El usuario no tiene que hacer tareas que podrían ser tediosas ya que el sistema lo hará de forma automática.

2.-Sensación de modernidad para el usuario: El usuario experimentará la sensación de vivir en un entorno actual tecnológicamente hablando.

3.-Considerable ahorro de recursos: Los sistemas están programados para brindar servicio a usuarios cuando estos realmente hacen uso de ellos.

4.-Administración inteligente: la administración inteligente se encarga de la gestión de servicios para el usuario, de manera que el sistema autónomo encargado de la manutención del inmueble dé un buen mantenimiento al edificio, ejemplo de esto es un sistema de riego para las plantas, o sistemas de limpieza automáticos ya sea para ventanas o pisos.

De esta manera los edificios inteligentes son de gran ayuda ya que estos facilitan la vida de los usuarios. Para este trabajo se diseñó un mecanismo para la administración de los recursos.

Con la finalidad de aprovechar de mejor manera los recursos disponibles para el presente proyecto, como son sensores y tarjetas de desarrollo, se hizo una revisión de sus características, mismas que se presentan en la siguiente sección.

Ⅱ.5 Sensores

Los humanos contamos con diferentes receptores en todo nuestro cuerpo, un ejemplo de ello está en la lengua, en ella se encuentran las papilas gustativas, con ellas podemos saber si algo está caliente, frío, dulce o amargo. Asimismo, la piel está llena de sensores que nos alertan sobre cambios de temperatura, y presión, etc. Por lo tanto, un sensor sirve para realizar una medición, es decir *“un sensor se encarga de realizar una medición ya sea física o química, esto para poder convertirla*

en un voltaje legible por un microcontrolador, el cual a su vez responderá según su programación" [8].

Llegando a este punto, quizá muchas personas no lo notan, pero los sensores se encuentran presentes en gran parte de nuestra vida cotidiana, ejemplo de esto son los supermercados, los cuales cuentan con puertas automáticas que abren y cierran según detecten la presencia de una persona en la entrada, de igual manera, los elevadores son otro ejemplo ya que al detectar sobrepeso éstos no se desplazan puesto que hacerlo sería peligroso.

Dicho lo anterior, los sensores son muy utilizados actualmente en sistemas de seguridad, un ejemplo de esto son los sensores de presencia, los cuales son empleados para detectar movimiento, evitando la proximidad de peligro y alertando según se haya programado la respuesta: prendiendo luces, enviando un mensaje en un display o alertando con una sirena. Otro ejemplo de un sensor empleado en sistemas de seguridad es el sensor de vibración, el cual reacciona a impactos y con lo cual se puede dar aviso de la detección de intrusos irrumpiendo por la fuerza, ya sea en ventanas o puertas. Por consiguiente, la comprensión del funcionamiento de cada uno de los sensores empleados en este trabajo es importante ya que son parte fundamental del sistema de alarma, por lo que en los siguientes párrafos se explica el funcionamiento de los sensores utilizados, además de mostrar los resultados de las pruebas que se realizaron a éstos.

Por otro lado, en la actualidad las tarjetas de desarrollo han tomado gran popularidad para todos aquellos que buscan hacer proyectos de manera simple y rápida, por eso mismo gran parte de los sensores ya se pueden encontrar en pequeños módulos compatibles con dichas tarjetas de desarrollo, por ejemplo, el módulo relé de 5 V ya cuenta con relevador, diodo y transistor, este módulo con dos relevadores se puede adquirir a un costo aproximado de $ 48 pesos mexicanos. Por lo anterior mencionado, algunos sensores que se utilizan en este proyecto ya vienen en pequeños módulos, los cuales serán explicados a continuación.

Ⅱ.5.1 Módulo sensor de vibración SW-420

Este módulo está basado en un sensor SW-420 y un comparador LM393. El módulo cuenta con una abertura para poder fijarlo a una superficie, además de tener un potenciómetro para poder calibrar la sensibilidad del impacto de vibración, el módulo cuenta con tres pines de los cuales dos son para su alimentación y uno es para enviar una señal digital, es decir, envía 1 ó 0 lógico al sistema. La configuración del módulo enviará una señal nula en todo momento a menos que sea detectada una vibración.

La estructura de este sensor consta de un resorte y un poste que al hacer contacto cierra o abre un circuito, enviando un estado alto o bajo.

Por lo general, el módulo SW-420 suele ser empleado en sistemas de seguridad, para este proyecto se emplea precisamente para eso, su función es la de mandar un estado alto si se detecta que alguien irrumpió por la fuerza en una ventana.

Como parte del sistema de alarma también se emplea un módulo HC-SR501. Este módulo alertará si hay una presencia en la vivienda.

Ⅱ.5.2 Módulo HC-SR501

Este módulo, al igual que el SW-420, tiene otros usos además de la seguridad, como el control de luminarias y la detección de intrusos.

El módulo cuenta con un sensor piroeléctrico[1] (usualmente conocido como sensor PIR), además de dos potenciómetros con los cuales se regula el tiempo de espera y la distancia máxima de detección que tendrá el sensor.

El sensor PIR es muy seguro para los humanos ya que es un dispositivo que no emana radiación de ningún tipo, pero sí se encarga de medir cambios de radiación en los objetos cuando éstos alteran su posición, esto lo hace midiendo la luz infrarroja que radian los objetos, y con ello pueden detectar el movimiento.

Habría que decir también que la luz infrarroja es un tipo de radiación electromagnética que produce de forma natural cualquier cuerpo cuya temperatura

[1] Piroeléctrico: propiedad de ciertos materiales al ser sometidos a cambios de temperatura.

sea mayor al cero absoluto. Ésta no es peligrosa si se produce de manera natural. La luz infrarroja se encuentra entre la luz visible y las microondas, figura Ⅱ.1 [9].

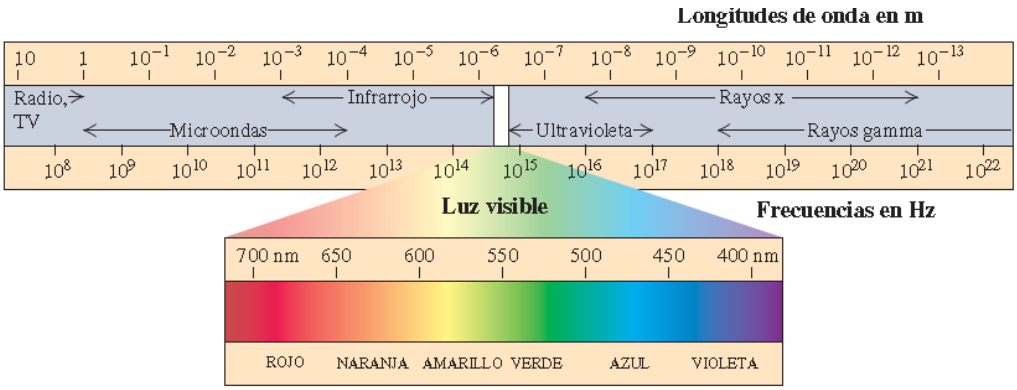


Figura Ⅱ 1. Espectro electromagnético.

El módulo PIR cuenta con tres pines de los cuales dos son para su alimentación de 5 V, además de un tercero para mandar la señal de salida. Su rango de alcance es aproximadamente de 6 metros, asimismo el tiempo de respuesta y la sensibilidad del módulo se pueden ajustar con ayuda de los potenciómetros. Un ejemplo de uso para el módulo PIR es: se requiere que un foco se apague al salir de la habitación, entonces el microcontrolador se debe de programar para que al realizar la acción el foco se apague. Para este proyecto el sensor PIR será utilizado para mandar un estado alto a uno de los puertos del microcontrolador y con ello activar una alarma y la iluminación, esto siempre y cuando se detecte alguna intrusión. Los sensores se instalarán en puntos que se consideren vulnerables, esto se mostrará más adelante.

Otra característica del PIR es que necesita ser calibrado para ser utilizado, de esto depende que el PIR funcione de manera óptima, para realizar esta acción se requiere un tiempo de entre 10 a 60 segundos y, de preferencia, que no haya nadie en el lugar donde se realizará la calibración ya que esto podría causar anomalías tales como malas lecturas y con ello una alarma ineficiente. En la figura Ⅱ.2 se muestra un escenario en donde el sensor PIR trabaja teniendo en cuenta su área

de detección, lente Fresnel[2] y un posible intruso que al atravesar dicha área activará una alarma. Las recomendaciones para el mejor uso de dicho sensor se darán más adelante.

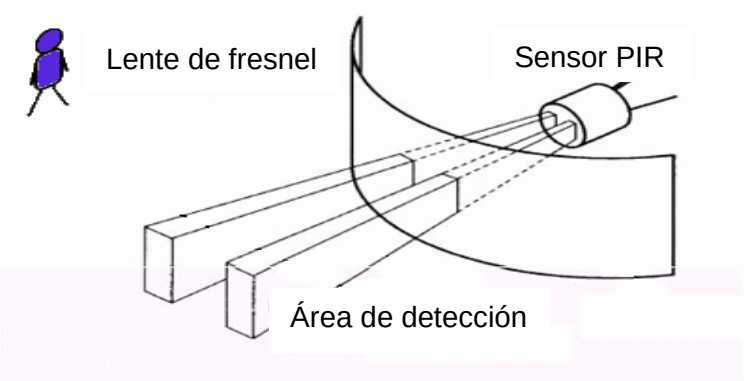


Figura Ⅱ 2. Escenario de prueba para el sensor PIR.

Otro sensor que también se puede emplear para un sistema de alarma es un sensor ultrasónico, el cual también se puede encontrar en un pequeño módulo.

Ⅱ.5.3 Módulo HC-SR04

Otro de los sensores que se utilizaron en este proyecto es el sensor ultrasónico cuyo módulo se llama HC-SR04, este sensor al igual que el sensor PIR no es nocivo para la salud del usuario. A continuación, se muestra por qué.

Los sensores ultrasónicos suelen ser empleados para medir distancia y evitar obstáculos, algunas personas los implementan en sus automóviles y con ello facilitan el estacionado del auto, también suele ser utilizado en proyectos de Arduino, por ejemplo, para implementar un carro evasor de obstáculos.

La forma en la que trabaja este sensor es mandando una señal ultrasónica que no se puede escuchar, esta señal es reflejada en una superficie y es cuando el sensor calcula el tiempo que tardó en ir y regresar la onda, al ser una onda mecánica de tan baja potencia no es dañina para el usuario.

[2] Las lentes de Fresnel son de vidrio o plástico cuya función es hacer que los rayos de luz se comporten al igual que si atravesaran lentes plano convexas.

La calibración del sensor ultrasónico considera la velocidad del sonido[3] tomando en cuenta todo el recorrido. Como la señal hace un recorrido de ida y vuelta entonces se tiene que dividir entre dos ese tiempo, además de realizar otros cálculos. En la figura Ⅱ.3 se muestra el funcionamiento del sensor ultrasónico, aquí se aprecia cómo el sensor manda una onda mecánica a una superficie plana cuya distancia es L, la onda se refleja y cuando completa su recorrido se hace un registro de la distancia a la que el objeto se encuentra.

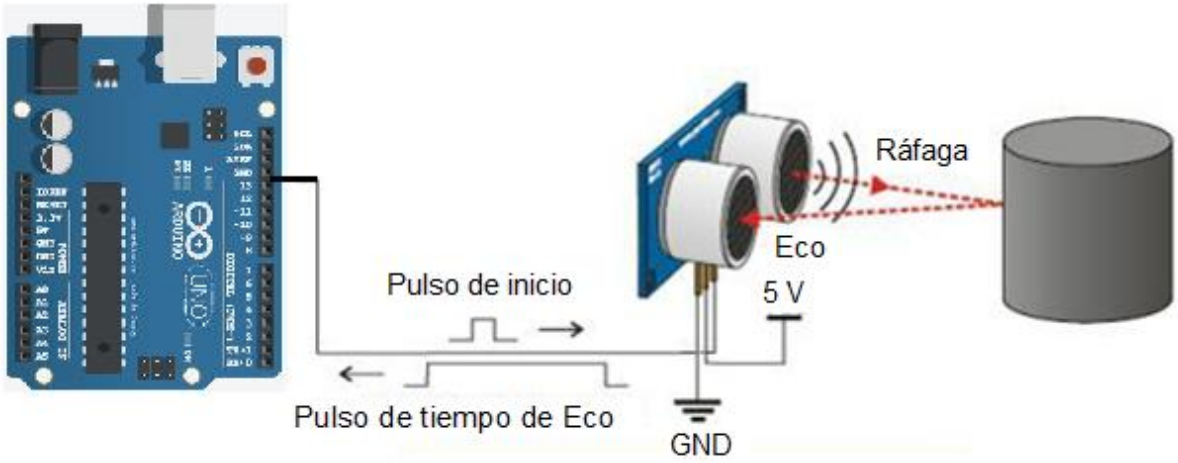


Figura Ⅱ 3. Diagrama que ejemplifica el funcionamiento del sensor ultrasónico.

La ecuación para calcular la distancia con este módulo es la siguiente:

$$L = \frac{1}{2}TV \qquad (1)$$

En donde:

L: Distancia

T: tiempo del recorrido (del emisor al receptor)

V: velocidad del sonido

[3] La velocidad del sonido en la atmosfera terrestre es de 343.2 m/s.

Actualmente existe una biblioteca que permite realizar todos los cálculos previamente mencionados, por lo que no se hace énfasis en la explicación de éstos [10].

Este sensor es de gran ayuda para la realización del sistema de seguridad ya que el módulo HC-SR04 se puede colocar en un escenario con mayor actividad que el sensor PIR, para tener un menor riesgo de activar una falsa alarma, por ejemplo, al pasar una ardilla en el jardín o aves. Si se coloca el sensor en un punto en donde se asegura que no pasará nada de lo anteriormente mencionado, entonces la probabilidad de falsas alarmas disminuye.

Para este proyecto el sensor se utilizó en la entrada de la casa y de esta forma detectará si alguien ingresa al hogar.

Otro problema que se considerá para este proyecto es la contaminación del ambiente. Para ello se emplea otro sensor que pueda verificar la calidad del ambiente.

Ⅱ.5.4 Módulo MQ-135

El sensor MQ-135 al igual que los anteriores viene en un módulo con el cual su uso se simplifica. La familia de sensores MQ sirve para detectar diferentes tipos de gases y alcohol. El motivo por el cual este sensor fue elegido es porque con él se puede detectar humo y gas butano, entre otros gases nocivos, de manera rápida y con ello evitar incendios o intoxicaciones.

Algunas de las sustancias que pueden ser detectadas con este sensor son: NH3, NOx, CO2 y humo, este último es muy subjetivo ya que dependerá del material que se esté consumiendo. El sensor MQ-135 es electroquímico y varía su resistencia al ser expuesto a los diferentes gases. Para comprobar esto último se realizaron dos pruebas, una con gas butano (CO2) y otra con humo. Las pruebas y resultados se muestran en el siguiente capítulo.

En adición y para comprender más a fondo el funcionamiento de este sensor, el MQ-135 cuenta con un calentador, el cual tiene una resistencia interna que varía según el tipo de gas con el que el sensor se encuentre en contacto. Si se desea

obtener una medición más precisa se debe revisar la hoja de especificaciones y ver la respuesta según el químico con el cual el sensor tenga contacto. Otro factor por considerar para una buena lectura es el tiempo que el sensor necesita para llegar a un punto estable, este tiempo puede ser de hasta 48 horas, pero no es necesario esperar ese largo periodo puesto que la lectura inmediata del sensor que lee el CAD no es mayor a una señal analógica de 150, es decir, si en la programación del microcontrolador se establece como umbral ese punto, mismo que no cambiará considerablemente, el sensor se puede utilizar de inmediato.

Todos los sensores mencionados en conjunto con una tarjeta de desarrollo formarán un sistema rentable que podrá responder de la mejor manera ante los aspectos que se consideraron para la realización de este trabajo. A continuación se habla de la tarjeta de desarrollo a la cual todos estos sensores fueron agregados.

Ⅱ.6 Microcontrolador

Un microcontrolador es un dispositivo programable, el cual se utiliza para la realización de procesos lógicos. En general, un microcontrolador está conformado por una memoria, una unidad central de procesamiento CPU, periféricos de entrada/salida y un convertidor analógico/digital. Para este proyecto todo esto está incluido en un microcontrolador llamado Atmega328P-PU.

Los microcontroladores no son recientes ya que tienen sus orígenes desde la década de los años 60. En 1969 trabajadores japoneses de la empresa *Busicom* tuvieron la idea de crear un dispositivo que sirviera para utilizar menos circuitos de los que se utilizaban en las calculadoras de dicha empresa. *Intel* acepta el proyecto y pone como responsable del desarrollo a Marcian Edward Hoff, él propone crear un circuito integrado cuya función estaría definida por un programa. Es así como comienza el desarrollo del primer microprocesador. Posteriormente, con ayuda de Federico Faggin, quien se une a *Intel* en 1970, es el encargado de diseñar e implementar chips con tecnología de compuertas de silicio y es así como se logra comercializar este producto. En 1971 *Intel* compra los derechos de *Busicom* y de

esta manera sale al mercado el primer microprocesador, el 4004[4] de *Intel* [11], [12], [13].

Algunos campos en los que se usan los microcontroladores son: la domótica, la ciencia médica, ingeniería agrónoma, automotriz, etc. Así mismo, los microcontroladores los podemos encontrar en productos de uso común, por ejemplo, un juguete o algún aparato electrodoméstico.

Con esta breve introducción de sensores y microcontroladores a continuación se hablará de las tarjetas de desarrollo, en particular de *Arduino*.

Ⅱ.7 Tarjetas de desarrollo

En este apartado veremos qué son las tarjetas de desarrollo además de que se explicará brevemente la función de ellas en el sistema de alarma.

Antes de continuar, una tarjeta de desarrollo cuenta con un microcontrolador, una fuente de alimentación y puertos de entrada y salida, así como un puerto USB con el cual se puede cargar a la tarjeta un programa. Además de las características mencionadas, las tarjetas se distinguen unas de otras por factores tales como: el tipo de memoria, arquitectura y capacidad (número de puertos analógicos y digitales).

Ⅱ.7.1 Arduino

Arduino es la plataforma que se empleó para realizar este proyecto. *Arduino* es una empresa open source[5] y open hardware[6] que desarrolla placas con microcontroladores. La finalidad de *Arduino* es proporcionar herramientas para crear proyectos multidisciplinarios ya sea por expertos o por principiantes, todo esto bajo la licencia de GNU[7] General Public License (GPL), que garantiza que todo software pueda ser compartido y modificado [14], [15].

[4] Procesador Intel 4004 de 4 bits con 6,000 operaciones por segundo.
[5] Open source hace referencia al código abierto, es decir, cualquiera es libre de utilizarlo.
[6] Open hardware quiere decir que toda la información y material puede ser utilizado por quien sea.
[7] Sistema operativo tipo UNIX formado por software libre, el nombre significa que no es un UNIX “Not Unix”.

Un punto de suma importancia implica el beneficio económico para el diseñador de productos basados en Arduino. Está permitido crear y distribuir productos a base de Arduino con una ganancia para quien se encargue de esta distribución y venta, siempre y cuando respete la licencia Creative Commons Attribution Share-Alike [16].

Otra ventaja que presenta Arduino, y de la cual es de suma importancia hacer mención, es que Arduino fue pensado para crear proyectos fáciles de implementar y a un menor costo. Por ejemplo, los PIC[8], familia de microcontroladores tipo RISC, necesitan de una programación basada en lenguaje ensamblador además de un programador especial. Aunque cabe mencionar que actualmente se pueden programar haciendo uso de lenguaje C y algunos compiladores, entre los cuales están: MPLAB-C18, MPLAB-C24 y MPLAB-32, en general, programar un microcontrolador Arduino requiere sólo de conocimientos básicos en lenguaje C y no requiere de un programador especial, sólo un puerto USB. De forma más clara, Arduino es un hardware basado en un microcontrolador, es utilizado en proyectos multidisciplinarios y es altamente recomendable para fines didácticos.

El Arduino uno, el cual fue uno de los empleados en este trabajo, cuenta con un microcontrolador Atmega328P-PU, este microcontrolador cuenta con una CPU, una memoria volátil que al desenergizar la tarjeta la información en la memoria se pierde, así mismo cuenta con una memoria no volátil la cual en gran parte almacena la programación previa para que al energizar la tarjeta ésta realice su trabajo, es decir cuenta con memoria RAM y ROM.

La tarjeta Arduino también cuenta con dos tipos de puertos, analógico y digital. Los puertos digitales de Arduino uno son 14, estos se pueden programar como entradas o salidas, en las cuales se puede recibir o mandar una señal de 0 y 5 V. Los puertos analógicos, los cuales son únicamente entrada, son 6, son utilizadas en su mayoría para monitorear el medio ambiente con ayuda de un sensor y el CAD y de esta forma obtener una señal analógica que se pueden traducir a un nivel de voltaje.

[8] Peripheral Interface Controller, controlador de interfaz periférico.

Arduino también cuenta con dos entradas las cuales sirven para transmisiones serie de señales TTL, estas entradas son la RX y TX, dichas entradas pueden ser utilizadas por el módulo Bluetooth, el cual es un estándar de comunicación inalámbrica.

Otro estándar utilizado en Arduino uno es el Serial Peripheral Interface (SPI). Para hacer uso de dicho estándar se necesitan utilizar los pines del 10 al 13, en conjunto permiten compartir información full dúplex entre dos dispositivos mediante una arquitectura maestro esclavo. Posteriormente se hablará más a fondo de dicho estándar puesto que es necesario para la realización de este trabajo.

De igual importancia el protocolo I²C también se puede utilizar en Arduino uno, éste para permitir la comunicación a través de un bus I²C.

Por último, la alimentación de Arduino puede ser de dos formas, utilizando un cable USB o con una fuente de alimentación, cuyos límites de voltaje se encuentren entre los 6 y 12 V. Si la alimentación es menor a 7 V la tarjeta funcionará, pero la salida de tensión de 5 V podría ser menor, por otro lado, si supera los 12 V la tarjeta podría dañarse. La tarjeta de desarrollo Arduino uno se puede ver en la figura Ⅱ.4.

Figura Ⅱ 4. Tarjeta de desarrollo Arduino uno.

Por ser una serie de tarjetas tan fáciles de utilizar, además de ser eficaces, la popularidad de Arduino a nivel mundial es tal que existen una gran cantidad de

tarjetas compatibles y copias, además de contar con soporte por parte de la comunidad de desarrolladores a nivel mundial, Arduino también cuenta con varios *shield* compatibles con sus tarjetas. Uno de los *shields* más relevantes para este trabajo se trata de la tarjeta *shield* ethernet, con este *shield* Arduino se convierte en un host y de esta forma se puede conectar a una red LAN.

Ⅱ.7.2 Arduino ethernet

Es necesario conocer qué es una tarjeta *shield* ethernet por lo que a continuación se hablará un poco de ésta.

Un *shield* en Arduino permite expandir las capacidades de la tarjeta Arduino ethernet aún más allá de las limitaciones físicas de fábrica, es decir, con un *shield* se puede lograr que una tarjeta Arduino se conecte a una red de datos, ya sea para mandar o recibir mensajes, y con ello realizar alguna función lógica. Para este trabajo la base es una tarjeta *shield* ethernet, con esta se puede lograr la conexión de Arduino a una red de datos, todo esto gracias al protocolo SPI, el cual permite la conexión entre microcontroladores.

El protocolo SPI es un protocolo que permite la comunicación serial entre dos dispositivos. Fue presentado por Motorola en 1982. Este protocolo permite la comunicación a muy corta distancia en full dúplex para comunicar el microcontrolador con otro dispositivo, en este caso el shield ethernet. Más adelante se retomará el tema de protocolos ya que éste no es el único utilizado en la realización de este proyecto.

La tarjeta *shield* ethernet se basa en un chip Wiznet Ethernet W5100. El W5100 proporciona una pila de protocolos soportados por Transmission Control Protocol (TCP) y User Datagram Protocol (UDP), es decir, permite la conexión a una red por medio de una pila de protocolos TCP/ Internet Protocol (IP). Dicho lo anterior, a continuación, se habla más a fondo de Arduino ethernet.

La tarjeta de desarrollo Arduino ethernet se empleó para poder entablar la comunicación con un servidor y de esta manera poder almacenar los datos del

proyecto en una base de datos MySQL. La tarjeta Arduino ethernet es la unión de un Arduino uno y una tarjeta *shield* ethernet para Arduino. Véase figura Ⅱ.5.

Figura Ⅱ 5. Arduino ethernet con módulo PoE.

A grandes rasgos, Arduino ethernet es la unión de un Arduino uno y un *shield* ethernet, pero con la diferencia de que se emplea en proyectos pequeños ya que cuenta con menos pines disponibles para usar dado que, como se ha hecho mención, los pines 10, 11, 12 y 13 son empleados para lograr la conexión con el módulo ethernet por medio del protocolo SPI y con ello lograr la conexión a una red de datos.

Arduino ethernet es una tarjeta que nos permite conectar un microcontrolador a una red de datos por medio de una serie de protocolos TCP/IP. Cuenta con un conector RJ45 y una ranura para una memoria SD, además de la opción de incorporar un módulo Power over Ethernet (PoE) con el cual la tarjeta Arduino podrá alimentarse directamente de la conexión Ethernet.

Una de las características de Arduino ethernet es que se puede utilizar para implementar una página web con etiquetas HTML, para el control de luminarias y otras funciones a las cuales tener acceso por medio de una red LAN, por ejemplo, conectarse al servidor por medio de un host, como podría ser un teléfono celular, y con ello ordenar alguna función al sistema. Dicho lo anterior, la tarjeta se empleará

como un servidor que se comunique a una base de datos, instalada en otro servidor, y con ello llenar en forma dinámica tablas con información de importancia del sistema de alarma.

El punto anterior es de interés ya que, como se ha hecho mención, el proyecto consiste en poder conectar esta tarjeta a una base de datos montada en otra tarjeta de desarrollo de nombre Intel Galileo generación 2.

Ⅱ.8 Programación en Arduino

Como ya se mencionó, el software de Arduino es de uso público, es decir, su programación puede ser compartida para su utilización o modificación dependiendo del usuario y aplicación. La programación de Arduino está basada en procesos, es decir, los programas se ejecutan línea a línea desde arriba hacia abajo. El lenguaje de Arduino está basado en lenguaje de programación C, se tienen constantes, variables, control de flujo, etc.

Asimismo, como todo lenguaje de programación, Arduino también cuenta con su entorno de desarrollo (IDE), éste lo podemos descargar directamente de la página oficial de Arduino. El IDE de Arduino está disponible para Macistosh, GNU/Linux y Windows, esto es otra ventaja que tiene Arduino sobre otras tarjetas de desarrollo ya que la mayoría no cuenta con un IDE multiplataforma. Los programas que son creados en Arduino son llamados bocetos o sketches.

Por otro lado, otra tarjeta empleada para este proyecto es la Intel Galileo generación 2. Esta tarjeta servirá como host para la base de datos. En esta tarjeta se montará un sistema operativo Linux de distribución Debian y se podrá tener acceso a este sistema por medio del protocolo Secure Shell (SSH).

Ⅱ.9 Intel Galileo generación 2

La tarjeta Galileo está basada en el procesador de aplicaciones Intel Quark SoC X1000. El procesador Intel SoC X1000 está diseñado para aplicaciones IoT, Wareables[9] y en general aplicaciones de control, pero todas de bajo consumo

[9] Sistemas incorporados al cuerpo humano para llevar un control de actividades, por ejemplo, la presión y ritmo cardiaco.

puesto que es un procesador que opera a 400 MHz. Además de lo anterior mencionado, dicho procesador es compatible con sistemas de cómputo X86, es decir, sistemas de 32 bits, asimismo también es compatible con el set de instrucciones Pentium. Otra característica es que la arquitectura de Galileo fue diseñada para ser compatible con el software y hardware de la tarjeta Arduino uno R3. Es decir, la tarjeta Galileo se puede programar con el mismo IDE de Arduino y es compatible con los *shields* de éste [17].

Un problema para aquél que quiera programar una tarjeta Galileo en el IDE de Arduino se presenta al querer utilizar una versión posterior a la arduino-1.6.0+Intel, esto ya que las versiones posteriores no cuentan con los controladores necesarios y esto provoca que la computadora no reconozca la tarjeta Galileo.

Al igual que las tarjetas Arduino, Galileo también cuenta con una serie de características, similares a las de una tarjeta Arduino, algunas de ellas son: procesador, memoria, puertos USB, ranura para una micro-SD, puerto RS-232, botones de reinicio, puertos de entrada y salida y la posibilidad de cargar un sistema operativo en una tarjeta micro-SD.

Dicho lo anterior, la tarjeta Galileo tiene la posibilidad de ser utilizada como un servidor al igual que la tarjeta Arduino ethernet, esto gracias a que en una memoria micro-SD es posible instalar un sistema operativo de distribución GNU/Linux, para este trabajo el sistema operativo es Debian. Por lo tanto, para este proyecto la tarjeta Galileo sirvió como un servidor portable, en el cual se almacena toda la información del sistema de alarma para que el usuario pueda dar seguimiento a los eventos que el sistema haya registrado. La finalidad de guardar los registros es que el usuario pueda hacer uso de ellos de la forma que le convengan.

Continuando con el tema, no solo se requiere que la tarjeta Galileo sea un host para guardar la información sino también se requiere de una serie de protocolos de comunicación y una arquitectura para poder hacer uso de ellos.

En adición a los puntos anteriores, la manera en que la tarjeta Galileo guarda la información es por medio de una base de datos conectada al servidor Arduino. Para

poder lograr esto se instaló una infraestructura LAMP en el sistema operativo Debian de la tarjeta Galileo, pero antes de llegar a ello se muestra una explicación de cómo será la comunicación entre el usuario y el sistema por medio de protocolos.

Ⅱ.10 Protocolos de comunicación

En general un protocolo es una serie de reglas estandarizadas para llevar a cabo una tarea específica de manera correcta. Para entender más a fondo el tema a continuación se muestra un ejemplo muy utilizado para ejemplificar los protocolos de comunicación.

Ejemplo: la comunicación entre dos personas necesita de un mensaje origen, un transmisor, un medio de transmisión, un receptor y un mensaje destino. Pero para poder lograr establecer esta comunicación entre dos o más individuos es necesario el uso de reglas, como ya se ha mencionado, las reglas son importantes ya que sin ellas la comunicación es imposible.

Para la conversación del ejemplo tenemos las siguientes reglas:

- Emisor y receptor identificados.- El mensaje no puede solamente ser enviado al azar, debe de existir un destinatario. No simplemente podemos hablar hacia un punto en el cual no exista un destinatario (por ejemplo, la pared).
- Método de comunicación.- Se debe de establecer un medio para comunicarse, por ejemplo, un teléfono o una carta.
- Idioma y gramática.- Se debe de hablar y escribir en la misma lengua, no podrían establecer una comunicación verbal dos personas cuyo idioma sea diferente.
- Mismo estado temporal en la entrega.- Se debe de llevar una concordancia en los mensajes, no puedes hablar sin un orden temporal.
- Confirmación de recibido.- El destinatario debe de constatar que la comunicación se estableció, esto puede ser con lenguaje verbal o físico.

En caso de un mensaje a distancia, por ejemplo, una carta, se debe de tener en cuenta básicamente la misma estructura de reglas, por ejemplo, el contenido de la carta, el destinatario, el remitente, la fecha de envío, etc. Asimismo, se debe de

considerar un medio para que el mensaje enviado llegue a su destino, en este caso al destinatario.

Lo anterior mencionado se podría considerar equivalente al encapsulamiento de un mensaje para poder enviarlo en tramas de host origen a su host destino. Finalmente, siempre se debería de recibir un mensaje de éxito o fracaso al recibir un paquete (mensaje).

En las redes de comunicación es exactamente lo mismo, no podemos enviar un mensaje sin:

- Un destinatario (destino)
- Un medio de comunicación
- Cumplir con requisitos en las tramas que se envían (segmentación)
- Mensaje del host destinatario (acuse de recibido)

Ahora veamos, en telecomunicaciones para enviar un mensaje lo hacemos por medio de tramas. Una trama es un segmento del paquete original, es decir, es una medida de datos con tamaño mínimo de 64 bytes y máximo de 1518 bytes para Ethernet II y Ethernet 802.3 respectivamente. Se debe de considerar el tamaño de la trama ya que, como se mencionó, los datos se envían por medio de diversas tramas y si éstas no cumplen con los requerimientos necesarios entonces son descartadas. Una trama usualmente se descarta si sufre de colisiones o ruido. Dos estándares que definen la arquitectura de una trama son el Ethernet II y el Ethernet 802.3 junto con sus actualizaciones. Por ejemplo, el Ethernet 802.3ac aumenta el tamaño máximo de la trama a 1522 bytes, todo esto por la inclusión de las VLAN[10] [18].

Una trama de Ethernet II se define por 6 campos, los cuales son:

1.-Preámbulo: señal de sincronía de 5 MHz generada por una secuencia de unos y ceros.

[10] Redes virtuales.

2.-Dirección MAC destino: consta de 6 campos y es única para cada host.

3.-Dirección MAC origen: consta de 6 campos y es única para cada host.

4.-Tipo: indica el protocolo de la capa superior.

5.-Datos: paquete proveniente de la capa de red en donde vienen almacenados los paquetes.

6.-Frame Check Secuence (FCS): verifica la trama y se encarga de encontrar errores para aceptar o descartar la trama.

La trama de Ethernet 802.3 cuenta con los mismos campos que Ethernet II, pero se le suma el delimitador de inicio de trama. En la tabla Ⅱ.1 se muestran los campos de ambas tramas.

Trama Ethernet II						
Preámbulo	Dirección de destino	Dirección origen	Tipo	Datos		Secuencia de verificación de trama
8 bytes	6 bytes	6 bytes	4 bytes	46 – 1500 bytes		6 bytes
Trama Ethernet 802.3						
Preámbulo	Delimitador de inicio de trama	Dirección destino	Dirección origen	Longitud	Encabezado y datos 802.3	Secuencia de verificación de trama
7 bytes	1 byte	6 bytes	6 bytes	2 bytes	46 – 1500 bytes	4 bytes

Tabla Ⅱ 1. Muestra el formato de las tramas Ethernet II y Ethernet 802.3.

Para que las tramas puedan ser enviadas se necesita que el host origen fragmente la información en *n* número de tramas iguales y las envíe. Posteriormente, el receptor ira recibiendo las tramas sin importar el orden en el que hayan llegado, el

receptor no podrá identificar el mensaje hasta que todas las tramas que lo compongan lleguen al host destino, en donde se desencapsulará la información para unirla y tener un mensaje legible. A todo esto, se le llama encapsulamiento y segmentación de datos.

Pero todo lo anterior no sirve sino tenemos un protocolo que haga viable la comunicación, por ejemplo, al conectar dos hosts por medio de un cable de Ethernet no sucederá nada a menos que los protocolos logren la comunicación entre ellos.

En redes de información un protocolo de comunicación es aquél que se encarga de la conexión y el correcto intercambio de datos entre diversos hosts, ya que sin ello no sería posible compartir información. Asimismo, existen diferentes protocolos cuya función va desde la transferencia de archivos (File Transfer Protocol – FTP) hasta la administración de errores en la transmisión de paquetes (Internet Control Message Protocol - ICMP).

Dicho lo anterior, es de gran importancia mencionar los modelos de comunicación, ya que estos permiten la comunicación entre dos hosts, es por ello que a continuación se muestran los modelos Open System Interconnection (OSI) y TCP/IP.

Ⅱ.11 Modelos de comunicación

El modelo OSI es un modelo únicamente de referencia que se utiliza para la docencia y sirve de ejemplo para entender cómo una computadora se comunica con otra sin importar su arquitectura.

Ⅱ.11.1 Estandarización

La primera versión del modelo OSI se creó en 1978 y la segunda se crea en 1979 por la International Organization for Standardization (ISO). La primera versión que se libera por parte de la ISO es la ISO 7498:1984, en 1984 [19].

Ⅱ.11.2 Modelo OSI y sus 7 capas

El modelo OSI consta de 7 capas, las cuales se muestran en la tabla Ⅱ.2. Aquí podemos ver las capas o niveles de manera jerárquica, mismas que se explican a continuación.

Capa modelo OSI	Nombre de la capa
7	Aplicación
6	Presentación
5	Sesión
4	Transporte
3	Internet
2	Enlace de Datos
1	Física

Tabla Ⅱ 2. Modelo OSI de 7 capas.

1. Física: encargada de establecer las características eléctricas, mecánicas, funcionales y procedimentales que se requieren para la transmisión de bits de datos (cables, conectores, voltajes y velocidades de transmisión).
2. Enlace de Datos: establece cómo se formatean los datos para su transmisión, da acceso al medio y cuenta con revisión o corrección de errores vía trama.
3. Red: esta capa provee los medios necesarios para que dos hosts, sin importar su ubicación geográfica, puedan establecer, mantener y finalizar la comunicación entre ellos. Aquí existe el tráfico de paquetes. En adición, el protocolo IP se apoya del ICMP para el control de posibles errores y los mensajes de control. Asimismo, el protocolo Address Resolution Protocol (ARP) es utilizado en las redes LAN/MAN para poder hacer un mapeo dinámico de las direcciones IP lógicas.
4. Transporte: esta capa proporciona los medios para que dos hosts puedan transmitir datos entre sí. Establece, mantiene y cierra los circuitos virtuales de forma lógica utilizando puertos, además, utiliza la segmentación de datos para poder transmitir la información evitando la pérdida de ésta.

5. Sesión: la capa de sesión se utiliza para sincronizar y controlar la transferencia de datos entre dos hosts. Además de esto se utiliza para que un usuario se conecte a un servidor remoto y pueda compartir información, por ejemplo, una sesión en la base de datos MySQL. Todo esto ya que la capa de sesión proporciona los medios para controlar dichas sesiones de forma adecuada, indicándolas, manteniéndolas y finalizándolas.
6. Presentación: esta capa se encarga de que la información enviada a la capa 7 de un sistema sea legible por la capa 7 de otro sistema sin importar que las dos máquinas tengan una representación diferente de caracteres. Es decir, si una máquina reconoce un tipo de caracteres diferente al de otra máquina, esta capa se encargará de que ambas puedan comunicarse. Esto es un ejemplo más claro de la estandarización que trae consigo este modelo ya que sin importar el fabricante y arquitectura los diferentes hosts se pueden comunicar.
7. Aplicación: esta capa sirve para que el usuario pueda acceder a las otras capas, pero no de forma directa, sino por medio de una interfaz, por ejemplo, al solicitar un recurso de un servidor utilizamos una aplicación (web browser), la cual no muestra cómo interactúan las capas entre sí para poder darnos el recurso. Uno de los protocolos que está en esta capa y que es de interés es el HyperTex Transfer Protocol (HTTP) además del HyperTex Transfer Protocolo Secure (HTTPS).

Ⅱ.11.3 Modelo TCP/IP de 5 capas

Este modelo a diferencia del modelo OSI cuenta únicamente con 5 capas. Tiene la misma función que el modelo OSI pero la diferencia es que este modelo es uno de los que realmente se utilizan en redes de comunicación y no solamente es un modelo didáctico [20]. La tabla Ⅱ.3 muestra la arquitectura del modelo TCP/IP y su equivalente en OSI.

Capa TCP/IP	Equivalente en OSI	Nombre de la capa
5	5, 6 y 7	Aplicación
4	4	Transporte
3	3	Internet
2	2	Enlace de datos
1	1	Física

Tabla Ⅱ 3. Modelo TCP/IP de 5 capas en comparación al modelo OSI de 7 capas.

La función de cada capa es la misma que la del modelo OSI, por lo que enumerar y mencionar nuevamente las capas es innecesario.

Ahora que se tiene una idea más clara de cómo es que dos hosts, sin importar su arquitectura, logran comunicarse y con ello intercambiar información, a continuación se hablará de un protocolo de suma importancia ya que es con el que el usuario (no el administrador) de la base de datos podrá acceder a los registros.

Ⅱ.12 Protocolo HTTP

El protocolo HTTP es un protocolo de la capa de aplicación, capa 5, 6 y 7 del modelo OSI y capa 5 del modelo TCP/IP, define la sintaxis para establecer la comunicación entre clientes y servidores, es decir, es un protocolo cliente-servidor.

El protocolo HTTP ha ido evolucionando desde su creación aproximadamente en 1991. La primera versión de este protocolo solamente servía para la trasferencia de archivos HTML, actualmente este protocolo sirve para transferir imágenes y videos.

Algunas de las primeras versiones del protocolo HTTP son:

1. **HTTP/0.9**: La versión inicial del protocolo HTTP no contaba con un número, pero posteriormente se le denominó como HTTP/0.9. Esta versión era conocida por ser muy simple, solamente utilizaba una petición con el método GET[11] más la página solicitada. Por ejemplo, GET /ejemplo.HTML. La respuesta de igual forma era simple:

[11] Método cuya función es enviar información codificada en una URL. El método GET está restringido a 2048 caracteres, no sirve para enviar información confidencial puesto que la información es visible. Con este método no se pueden enviar archivos.

<HTML>

Contenido de la página ejemplo

</HTML>

Además, como ya se mencionó, esta versión solamente servía para solicitar páginas HTML y no contaba con mensajes de error en la solicitud, pero cuando ocurría un error la página era regresada con una leyenda que decía cuál podría ser la posible causa del error.

2. **HTTP/1.0**: esta versión añade 1.0 a la petición GET, por ejemplo, GET/ejemplo2.html HTTP/1.0, además es posible saber si los datos se enviaron o no y de esta forma responder. Esta versión incluía cabeceras a diferencia de la HTTP/0.9, con ello se logró la transferencia de otros archivos además de los HTML. El RFC 1945 explica detalladamente este protocolo HTTP [21].

   Además de lo mencionado, el uso de cabeceras permitió incorporar los metadatos, los cuales son datos que describen el contenido de archivos o la información de estos, con lo cual el protocolo se vuelve mucho más útil.

3. **HTTP/1.1**: esta versión es mejor a las anteriores versiones en tanto a la velocidad al abrir una nueva página solicitada mientras se procesa una solicitud previa, puesto que la conexión era reutilizable. Las peticiones en subpartes permiten una segunda petición antes de que la respuesta a la primera petición finalice, además es posible alojar varios dominios en la misma IP y se cuenta con un mecanismo de control de cache. Todo esto se especifica en el RFC 2616 [22].

4. **HTTP/2**: esta es la versión más utilizada y mejora aspectos del protocolo HTTP/1.1. Una de sus mejoras es la multiplexación con la cual se permite hacer peticiones en paralelo con la misma sesión. Otra mejora es la compresión de las cabeceras con lo cual se eliminan los duplicados, es decir, las cabeceras con el mismo nombre, y se aumenta la velocidad de transmisión. También se permite almacenar datos en el cache del cliente, esto no se podía hacer con anterioridad

ya que el protocolo HTTP no almacenaba información y lo único que se tenía para ello eran las cookies. El RFC 7540 describe la sintaxis y optimización del protocolo HTTP versión 2 [23].

En retrospectiva, el WWW fue creado por Tim Berners-Lee con la colaboración de Robert Cailliua entre 1989 y 1990. Además de ello, diseñó el primer servidor httpd previo al servidor apache, y creó la primera versión del lenguaje de programación HTML.

Dicho brevemente, el protocolo HTTP es utilizado usualmente en la red informática mundial, es decir la Word Wide Web (WWW). Este protocolo fue creado por The World Wide Web Consortium (W3C) en colaboración con la comunidad Internet Engineering Task Force (IETF).

Posteriormente, Tim Berners-Lee fundó el consorcio W3C en los laboratorios de Ciencia Informática del Instituto de Tecnologías de Massachusetts.

La W3C se ha encargado de la creación de estándares web. Este consorcio tiene un alcance internacional, además de tener varios objetivos, cuya finalidad es compartir el conocimiento a todos los lugares y niveles del mundo sin importar la arquitectura que se utilice para acceder a los recursos en la Internet [24].

Asimismo, la IETF es una comunidad internacional creada en 1986. La IETF es una comunidad de diseñadores de redes para la evolución de la arquitectura de las redes de Internet. La asociación está compuesta por grupos, los cuales tienen cada uno una tarea específica. De igual manera cualquier persona interesada en el desarrollo de redes está invitada a la fundación [25].

Además de los puntos previamente mencionados, también es necesario el manejo de puertos de comunicación, ya que estos son los que nos permiten acceder a algún servicio.

Ⅱ.12.1 Puertos de comunicación

Un puerto físico se entiende como un puerto lógico que se localiza en la memoria de un host y permite la transferencia de información asociándola con un puerto

físico, por ejemplo, puerto USB, puerto en serie, puerto VGA o puerto HDMI, entre otros. Un puerto lógico es una salida de bits utilizada por las diferentes aplicaciones que se conectan a un host, es decir, se utiliza para distinguir la transferencia de información entre múltiples host. Los puertos lógicos están asociados a diferentes números y de esta manera se especifica la aplicación a la que se está enviando la información, por ejemplo, el puerto 80 está asociado al protocolo HTTP [26].

El protocolo HTTP trabaja con el puerto 80, y por medio de este puerto es que este protocolo establece la comunicación del cliente al servidor. A continuación se hablará un poco más a fondo de dicho puerto, además de otros dos puertos que también son de suma importancia para este proyecto.

Como se acaba de hacer mención, la función del puerto 80 es permitir el acceso vía protocolo HTTP, pero, ¿qué sucedería si tuviéramos más de un servidor en el host? Ya que el puerto 80 es utilizado por los servidores Internet Information Services (IIS) y Apache, entre otros, el puerto 8080 hace la suposición de que ya existen otros servidores en la computadora, con ello al redireccionar el servidor al puerto 8080 se evitan errores de comunicación. Asimismo, los puertos que utiliza el servidor Apache, el cual se empleó para la realización de este trabajo, son el 80 y el 443.

Utilizar estos puertos solamente es necesario si se desea utilizar el servidor con el administrador de base de datos PhpMyAdmin, esto lo veremos a continuación.

Para permitir la comunicación con la base de datos se requiere el uso de otro puerto de comunicación. El puerto utilizado para entablar acceso a la base de datos MySQL es el puerto 3306, con este puerto el servidor montado en la tarjeta Arduino ethernet podrá comunicarse con la base de datos MySQL.

Con base en lo anteriormente mencionado, es necesario hacer un correcto uso de los cortafuegos[12] ya que este es quien gestiona el tráfico de datos que entra o sale de un host, si el cortafuegos no está bien configurado la comunicación no se logrará.

[12] Hardware o software que administran el tráfico de datos en la red.

Hacer mención del protocolo HTTP es importante ya que el desarrollo de este trabajo se centra en el modelo cliente servidor, esto ya que el usuario podrá revisar los registros del sistema haciendo uso de un navegador. Otro factor importante es que en el capítulo 3 algunas pruebas se realizan utilizando este protocolo para comprobar lo antes mencionado.

Continuando con el tema, ya se ha hecho mención del modelo cliente-servidor, pero ¿qué es? Para esclarecer este modelo a continuación se explica un poco de su historia, usos, ventajas y desventajas.

Ⅱ.13 Modelo cliente servidor

La arquitectura cliente servidor es aquella que se encarga de cumplir todas las peticiones de un usuario a un servidor. El modelo cliente servidor nace en la década de los 70 y es Xerox-PARC quien se encarga de su desarrollo [27].

Como su nombre hace mención, esta arquitectura consta de un cliente y un servidor en donde el cliente es aquel que se encarga de solicitar algún recurso a los servidores, esto dependiendo de las necesidades del cliente ya que los recursos van desde la solicitud de una página web hasta la descarga de imágenes, videos, o aplicaciones, etc. Por otro lado, los servidores son aquellos que se encargan de alojar a los recursos para que el cliente pueda acceder a ellos cuando así lo requiera.

El método para solicitar algún recurso es mediante un navegador[13]. Esta aplicación nos permitirá poder visualizar el administrador de nuestra base de datos. Cabe resaltar que, según la página StatCounter, la cual es utilizada para analizar el tráfico en la red global, el navegador más utilizado en 2016 fue Google Chrome [28]. Véase figura Ⅱ.6.

[13] Aplicación cuya función es responder a las peticiones del usuario por medio de una interfaz gráfica.

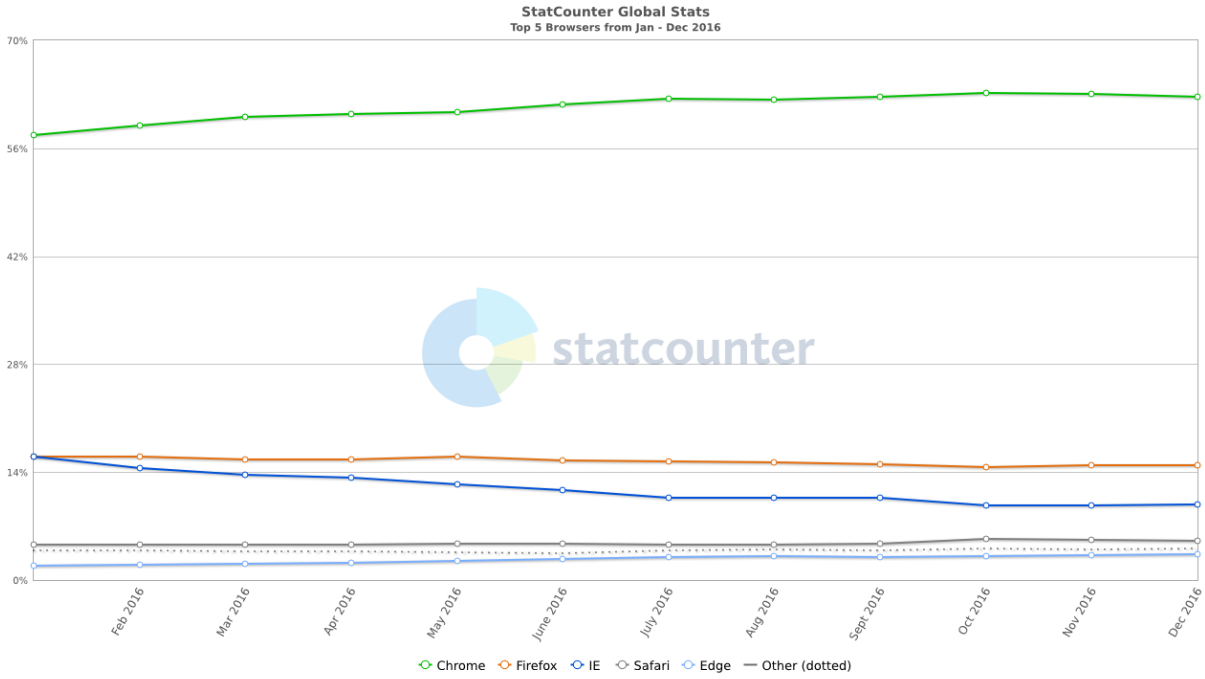


Figura Ⅱ 6. Grafica que muestra las estadísticas de navegadores de febrero a diciembre de 2016.

Como podemos ver, el uso de Google Chrome va en ascenso mientras uno de los navegadores más conocidos, pero menos empleados actualmente va en descenso, este navegador es Internet Explorer. Hacer mención de los navegadores es de importancia ya que algunos presentan problemas y simplemente no permiten la conexión a ciertos recursos. Más adelante se profundizará en el tema, esto en el siguiente capítulo.

Con lo anterior mencionado se entiende lo siguiente:

- Cliente. - Es quien solicita los recursos, es un consumidor de éstos.
- Servidor. - Es aquel que provee de los recursos a uno o varios clientes.
- Cliente/servidor. - Es una arquitectura de petición respuesta.

Un servidor puede atender varias peticiones, es decir, puede dar servicio a uno o más usuarios. Por ejemplo, en una vivienda hay una pareja de consumidores de recursos, cada uno con un *end system*[14], ambos buscando información en el mismo servidor, por ejemplo, uno de correo, el servidor atiende las peticiones de ambos

[14] Un end system puede ser un celular, Smart TV, iphone, etc.

clientes simultáneamente sin afectar la solicitud de ningún usuario. Del ejemplo anterior se entiende que sin importar si uno o más usuarios solicitan un recurso a un servidor, éste podrá responder a las peticiones siempre y cuando éstas estén dentro de las capacidades técnicas del servidor. De igual manera, el servidor o los servidores ya sean de una o varias compañías pueden estar situados en una o varias locaciones geográficas gracias a un modelo distribuido[15]. A continuación, se muestran algunas características del modelo cliente servidor de forma más sintetizada.

Características modelo cliente servidor:

- El cliente y el servidor pueden estar ubicados en diferentes redes.
- El servidor brinda servicio a varios clientes al mismo tiempo.
- El cliente y el servidor son independientes a menos que ambos se encuentren alojados en el mismo host.
- El cliente no necesita saber la ubicación del servidor, solamente hace peticiones.

Una vez explicado qué es el modelo cliente servidor, veamos lo que son las bases de datos. El conocimiento de bases de datos es de suma importancia para este proyecto ya que sin ellas recibir algún recurso o hacer algo tan simple como lo es el registrarse en una página sería una labor imposible.

Ⅱ.14 Bases de datos

Los bancos de datos o bases de datos (como se les conoce más comúnmente) se utilizan para el almacenamiento de la información y así poder hacer uso de ésta de la mejor forma posible.

Las bases de datos no son solamente digitales, también tenemos bases de datos físicas, como pueden ser: una hemeroteca, biblioteca, alguna lista impresa de precios o los deberes de la semana que alguien puede almacenar en un calendario,

[15] Modelo que permite la conexión de varias computadoras sin importar su ubicación física. Conexión realizada por una red.

etc. Es decir, una base de datos es un conjunto de archivos destinados a almacenar información [29].

De igual forma, una manera eficiente de llevar el control de una empresa es a través de una base de datos, por ejemplo, en una tienda de víveres podemos llevar un registro de toda la comida almacenada, ya sea cantidad y precio de ésta, asimismo, se puede llevar el control del personal y de las finanzas de la empresa.

Como ya se mencionó, las bases de datos sirven básicamente para almacenar información, lo que nos lleva a identificar a las bases de datos como de tipo estática o dinámica.

Ⅱ.14.1 Bases de datos dinámicas y estáticas

Una base de datos estática es aquella que puede seguir creciendo, pero la información en sus documentos no va a cambiar. Un ejemplo de esto sería una hemeroteca, las hemerotecas están conformadas por diarios los cuales solamente van llevando un registro del día de su publicación. A los diarios de la hemeroteca no le puedes modificar la información que ya existe en ellos, pero sí es una base de datos que puede seguir creciendo con la adición de nuevos diarios.

Una base de datos dinámica es aquella en la que se lleva un control de la información y ésta se puede actualizar según sea necesario o conveniente para la institución que la utilice. Por ejemplo una empresa del tipo de centro comercial debe de llevar un control de los precios, pero como todos sabemos los precios cambian bajo ciertas circunstancias, al tener precios que constantemente cambien con el mercado debemos tener una base de datos que aunque siempre marque *n* cantidades de latas de sopa también pueda actualizar sus precios, con esto entendemos que una base de datos dinámica es aquella que puede mantener cierta información estática, pero también se actualiza y cambia además de crecer.

Ahora bien, el almacenamiento es importante, pongamos como ejemplo los videojuegos de hace 20 años. Los videojuegos, previo a su distribución, se tenían que almacenar en grandes bodegas que separaban los títulos, les asignaban una fecha, un número de lote, etc. Actualmente un disco duro de 1 terabyte puede

almacenar entre 20,000 y 30,000 títulos dependiendo de su tamaño, variando éste entre los 16 MB y 64 MB. Además, se pueden catalogar por fecha de lanzamiento, costo, género, advertencias y restricciones de uso, y proporcionar información adicional a los usuarios del servidor en donde se aloje toda la información. Todo esto implica la digitalización de una base de datos para reducir costos y generar ganancias, es por ello que en la actualidad la digitalización de las bases de datos es algo esencial.

Ⅱ14.2 Tipos de bases de datos

Las bases de datos nos sirven para catalogar la información y no solo almacenarla, es por ello que es importante hacer mención de los tipos de bases de datos que existen, resaltando las siguientes.

Ⅱ.14.2.1 Bases de datos jerárquicas

Este tipo de base de datos consta de un orden jerárquico, el cual divide y subdivide la información en la base de datos como si se tratara de una estructura de árbol.

Los segmentos en los que se divide la base de datos tipo jerárquica constan de un segmento padre, hijos y gemelos. El segmento padre es aquel que se encuentra en la parte superior y del cual dependen los hijos. Los segmentos hijos son los subdirectorios que tienen como raíz el segmento padre. Por último, los gemelos son aquellos que comparten el mismo padre.

Todo lo anterior mencionado se ejemplifica en la figura Ⅱ.7, aquí se puede apreciar que la cabecera UACM es aquella llamada segmento padre, esto ya que es de donde parten los demás segmentos.

Los segmentos que siguen a la cabecera serían llamados hijos puesto que tienen como base el segmento padre.

Por último, los gemelos son aquellos que comparten la fuente del segmento, por ejemplo, dos estudiantes matriculados con el mismo profesor.

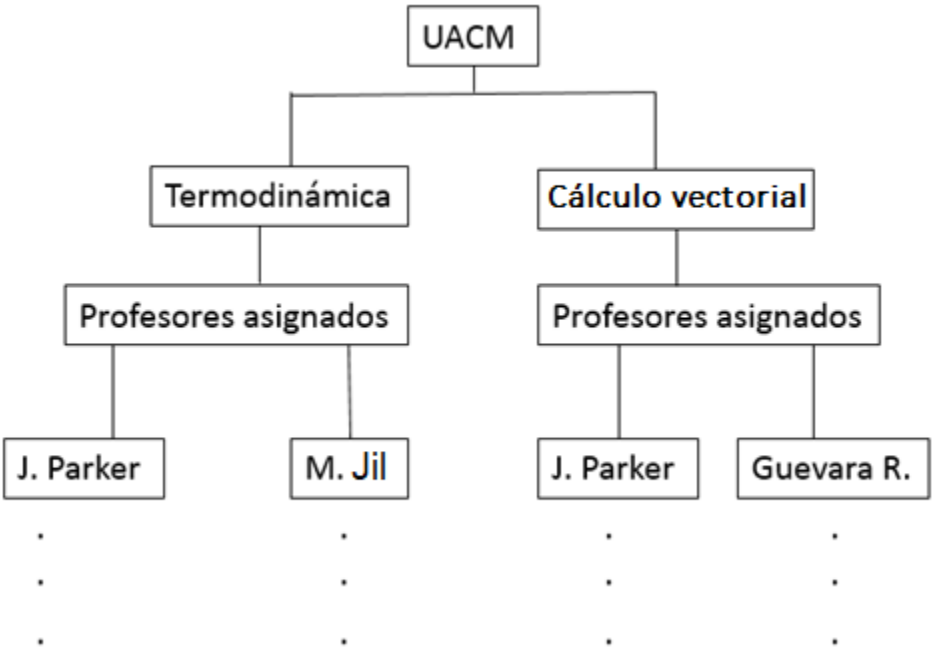


Figura Ⅱ 7. Estructura de una base de datos jerárquica.

Esta base de datos es poco práctica y casi no se emplea ya que presenta una gran posibilidad de redundancia, esto último se puede apreciar con el profesor J. Parker, él está asignado a dos cursos diferentes y esto podría provocar un error ya que, si modificamos la información del profesor en algún segmento, este cambio podría no verse reflejado en el siguiente segmento. Otra desventaja de la base de datos jerárquica es que tenemos que seguir toda la ruta para llegar a algún segmento en especial.

Ⅱ.14.2.2 Bases de datos de red

El modelo de base de datos de tipo red es similar al modelo de base de datos de tipo jerárquico, con la diferencia de que puede haber más de un registro padre. Este modelo busca la relación entre objetos. Otra ventaja que presenta el modelo de red es que al tener varios padres se evita la redundancia. La figura Ⅱ.8 muestra un ejemplo entre registros de una empresa de seguros y su distribución territorial.

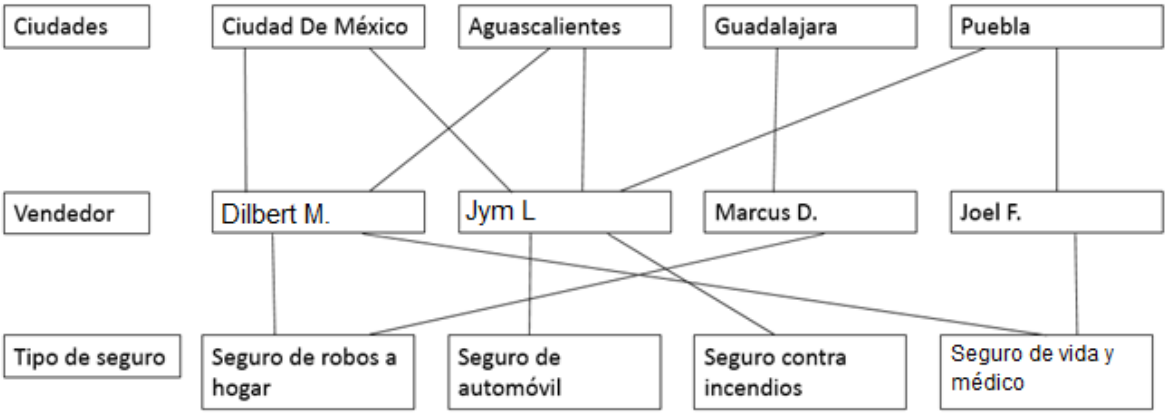


Figura Ⅱ 8. Estructura de base de datos de red.

En la figura Ⅱ.8 se tienen los registros ciudades, vendedor y tipo de seguro. Cada registro guarda información de su respectivo campo, la cual se relaciona entre sí con los demás campos. Por ejemplo:

Ciudad: Ciudad De México

Ubicación: América central

Número de clientes: 100,000

Vendedor en la zona: Dilbert M.

Jym L.

El siguiente tipo de base de datos es el que se empleó para la realización de este trabajo recepcional.

Ⅱ.14.2.3 Bases de datos tipo relacional

Este tipo de base de datos guarda los registros en tablas, es decir en una tabla se concentra toda la información de interés. Las tablas están conformadas por filas y columnas, cada fila lleva un registro de la actividad. Las filas también son llamadas duplas o registros.

En una base de datos se pueden almacenar varias tablas, las cuales podemos consultar seleccionando dicha tabla, también se puede trabajar con ellas para editarlas de forma manual o automática, es decir, podemos ingresar información de

forma manual por medio de comandos o haciendo un programa que llene los campos de la tabla según sea necesario. En la figura Ⅱ.9 se muestra un ejemplo de una tabla similar a la que se empleó para este trabajo además de un diagrama de una base de datos relacional.

Vendedor	Departamento	Producto
Itzel Marqués	Arte	Marco para pintar
Cesar Soria	Electrónica	Cargador de baterías
Julia Muro	Fármacos	Vitaminas
Felipe Barrera	Textiles	Rollo de tela negra

(a)

```
mysql> use data_iot;
Database changed
mysql> show tables;
+--------------------+
| Tables_in_data_iot |
+--------------------+
| configuracion      |
| distancia          |
+--------------------+
2 rows in set (0.00 sec)

mysql> select * from distancia;
+----+------------+-------+
| id | fecha      | valor |
+----+------------+-------+
|  1 | 2018-07-16 |    15 |
|  2 | 2018-07-16 |    11 |
|  3 | 2018-07-16 |    10 |
|  4 | 2018-07-16 |    19 |
|  5 | 2018-07-16 |    12 |
|  6 | 2018-07-16 |     5 |
|  7 | 2018-07-16 |     5 |
|  8 | 2018-07-16 |     5 |
|  9 | 2018-07-16 |     7 |
| 10 | 2018-07-16 |     7 |
| 11 | 2018-07-16 |     7 |
| 12 | 2018-07-16 |     8 |
| 13 | 2018-07-16 |     5 |
| 14 | 2018-07-16 |    18 |
| 15 | 2018-07-16 |    18 |
| 16 | 2018-07-16 |    21 |
```

(b)

Figura Ⅱ 9. (a) Estructura de una base de datos relacional, (b) ejemplo de base de datos relacional.

En la tabla de la Figura Ⅱ.9 (a) se tienen los registros de vendedor, departamento y producto. Debajo de cada registro está ordenado para así saber cuál vendedor de qué departamento ha vendido tal producto. Esta es la estructura que caracteriza una base de datos relacional. Por otra parte, en la tabla de la figura Ⅱ.9 (b), la cual fue utilizada para la primera prueba de guardado de registro del sensor ultrasónico, se pueden ver los campos id, fecha y valor, asimismo las columnas contienen registros, los cuales se llenaron de forma automática gracias a un programa que se encarga de detectar la distancia en centímetros de un sensor ultrasónico. La composición de la tabla de la figura Ⅱ.9 (b) es:

Id: Identificador de registro.

Fecha: muestra el día que se tomó la muestra.

Valor: muestra la distancia que fue registrada.

Este tipo de tablas puede parecer simple ya que pareciera que solamente es ir escribiendo en una hoja de papel la lista de víveres del súper mercado, pero es su simplicidad la que la hace una base de datos bastante práctica ya que este trabajo tiene como finalidad llevar un registro de las actividades que sucedan en una vivienda.

La base de datos que se utilizará para este trabajo es MySQL. Dicho lo anterior y continuando con el tema, dado que en este trabajo se tiene la finalidad de llevar un registro de sucesos en una vivienda, para ello se necesitará la implementación de varías herramientas, por suerte todas las que se emplearon para este trabajo son de licencia gratuita. A continuación, se describe la plataforma llamada LAMP que se usó en el servidor que aloja la base de datos y su administrador [30].

Ⅱ.15 LAMP

LAMP, WAMP, MAMP y XAMP son los acrónimos de una plataforma conformada por un sistema operativo, ya sea Linux, Windows, Macintosh o X sistema operativo (en el caso de XAMP), además de un servidor, un Data Base Manager (DBM) y un lenguaje de programación.

A lo largo de este apartado se hablará un poco de cada pieza del sistema en cuestión, la finalidad es explicar a fondo el sistema completo.

Ⅱ.15.1 Linux

La L en LAMP hace mención al sistema operativo en donde la plataforma es montada, en este caso L es por GNU/Linux o simplemente Linux.

Linux permite hacer lo mismo que un sistema operativo de Windows o Macintosh, es decir, se pueden instalar diferentes aplicaciones tales como son: editores de texto, video, imágenes, juegos o alguna aplicación que necesitemos. Las limitantes de Linux son pocas (refiriéndome específicamente al área de sistemas).

Otra cualidad con la que cuenta Linux es que es un software libre, cualquiera que así lo desee puede modificarlo y redistribuirlo, todo esto bajo la licencia GPL 3 (General Public License Version 3) de la GNU [31]. Todo esto ha motivado a los desarrolladores de software a realizar cada vez un mayor número de mejoras.

Ⅱ.15.1.1 Surgimiento de Linux

En 1969 empleados de los laboratorios *Bell de AT&T* desarrollan el sistema operativo llamado UNICS (Uniplex Information and Computing System), el cual posteriormente cambió de nombre a UNIX. Entre los responsables de la creación de UNIX tenemos a personajes tales como son Dennis Ritchie y Ken Thomson.

El surgimiento de UNIX nace con la necesidad de crear un sistema operativo destinado a una computadora central o mainframe, como también se le conocía. Entre las características de UNIX se tiene un sistema estable, multiusuario, multitarea y multiplataforma. Es por lo anterior mencionado que a Linux se le considera un UNIX para computadoras personales.

Posteriormente, en 1987 se creó el sistema operativo llamado Minix. Andrew Stuart Tanenbaum creó este sistema operativo con fines educativos ya que quería enseñar a sus estudiantes el sistema operativo UNIX pero las licencias de dicho sistema eran demasiado costosas, es por ello que desarrolló Minix utilizando lenguaje C y ensamblador.

Linus Benedict Torvalds, graduado de la maestría en ciencias de la computación de la universidad de Helsinki, en 1991 se encontraba trabajando en el desarrollo de un sistema operativo UNIX basado en Minix, es así como crea el primer Kernel[16] basado en UNIX. Posteriormente el kernel se puso a disposición por medio de un servidor FTP.

Con el tiempo la popularidad de Linux ha ido creciendo y con ello surgieron varias distribuciones. Una distribución de Linux es aquella que tiene como base el núcleo de Linux.

[16] El kernel es el encargado de que el software y hardware de nuestro equipo se comunique y pueda trabajar de forma correcta.

Para este trabajo la distribución que se utilizó es la *gcc versión 4.7.2 (Debian 4.7.2-5)*. Debian, al igual que Centos y Fedora, son software libre.

El motivo de utilizar Debian en este trabajo es que la tarjeta Intel Galileo es compatible con este sistema operativo.

De igual importancia, la paquetería que posee Debian tiene un tamaño aproximado de 51,000 paquetes, es decir, software pre compilado. Para este proyecto, los repositorios EPEL, los cuales son un conjunto de paquetería de software precargados en el sistema y que contienen paquetes adicionales, fueron necesarios al instalar el administrador de la base de datos, esto último es otra ventaja con la que cuenta Debian ya que, si su paquetería original no cuenta con los elementos necesarios en una instalación, es muy común que instalando los repositorios EPEL (es decir, paquetes adicionales) se obtenga lo necesario. Todo lo anterior puesto que Debian es un sistema estable el cual se sigue actualizando.

La siguiente letra en LAMP es la A, está hace mención a un servidor Apache, el cual debe estar instalado en nuestro sistema operativo Debian.

Ⅱ.15.2 Apache

Un servidor, como ya se mencionó, es aquel con el cual un cliente tendrá acceso a los diferentes recursos que necesite, tales como son archivos de texto, imágenes, video, etc. A continuación, se muestra brevemente cómo es que surge Apache server o servidor Apache.

Apache es un servidor de código abierto licenciado por GPL de la GNU, fue lanzado en 1995 y es un servidor multiplataforma, es decir, puede ser alojado en sistemas Linux, Microsoft o Macintosh. Apache puede estar escrito ya sea en PHP, HTML o Pearl. La comunicación con el servidor apache se logra gracias a la implementación del protocolo HTTP/1.1. Un dato cultural es que Apache debe su nombre a los Nativos Americanos.

La fundación que se encarga actualmente y desde 1999 es la ASF (*Apache Software Foundation*). Esta fundación sin fines de lucro se encarga de dar soporte a los servidores apache [32].

Cabe resaltar la importancia de Apache ya que es tal que actualmente es uno de los servidores más utilizados a nivel mundial. Según la página *Netcraft* [33], fundada en el año de 1987, Apache server, en marzo de 2009 contaba con el 66.65 % de dominios a nivel mundial y para marzo de 2018 Apache contaba con un total de 43 % de dominios. Aunque la cifra ha disminuido, Apache sigue siendo el preferido ya que el demás porcentaje se distribuye en servidores Microsoft, Sun, Google, entre otros.

La finalidad de Apache para este trabajo es alojar al administrador de base de datos PhpMyAdmin y de esta forma poder revisar las estadísticas de las bases de datos.

Es necesario el servidor ya que sin él no se tiene acceso al administrador de la base de datos, por ejemplo, si instalamos XAMPP (la cual es una aplicación de la que se hablara más adelante) y en el panel de control no iniciamos el servidor Apache entonces no tendremos un servidor para poder visualizar la base de datos con ayuda de PhpMyAdmin y el navegador nos mostrará un error, lo mismo sucede si iniciamos Apache, pero no MySQL en el panel de control de XAMPP. A continuación, se hablará de MySQL para explicar mejor la función del servidor Apache.

Ⅱ.15.3 MySQL

La base de datos es de suma importancia, anteriormente ya se mencionó lo qué es una base de datos y algunos tipos de éstas. MySQL es la letra M en LAMP, aunque en ocasiones se llega a utilizar Maria db.

El desarrollo de MySQL comenzó en 1994 y su primera versión fue liberada en 1995 siendo MySQL 1.0 la primera versión estable. Actualmente la versión más reciente es la 8.0.12 bajo la licencia GPL y a la cual podemos tener acceso visitando la página oficial MySQL. MySQL, al igual que el resto de nuestro sistema, es de código abierto bajo la licencia GPL de la GNU, además de ser multiplataforma.

MySQL fue desarrollada por la empresa *MySQL AB*, adquirida por *Sun Microsystems* en 2008 y posteriormente adquirida por *Oracle Corporation* en 2010; es por esta última absorción que MySQL cuenta con una licencia dual, ya que una

licencia es libre y una comercial por parte de *Oracle Corporation*. La empresa original fue fundada por David Axmark, Michael Widenius y Allan Larson.

El manejo de la base de datos MySQL está bajo el lenguaje de programación SQL, al cual podemos tener acceso descargando los diferentes manuales que existe en formato PDF en la red. La sintaxis de cada manual es de gran ayuda ya que para este trabajo existen dos formas de crear una base de datos, tablas y usuarios. Una manera de hacer esto es por medio de comandos SQL y otra es por una interfaz gráfica (PhpMyAdmin).

Ⅱ.15.4 MariaDB

En adición, MariaDB es una base de datos derivada de MySQL y al igual que ésta, cuenta con una licencia libre GPL V2. MariaDB fue desarrollada por uno de los fundadores de la *empresa MySQL AB*, Michael Widenius. El principal objetivo de MariaDB es ser un reemplazo directo de MySQL por lo que la compatibilidad entre ellas es total [34].

Habría que decir que al igual que MySQL, MariaDB también cuenta con muchas versiones a la fecha, entre las cuales destacan las estables, ya que también existen versiones de prueba. La primera versión estable publicada en 2010 es la versión MariaDB 5.1.42. La versión más actual de MariaDB es la perteneciente a la familia 10.3, esta es la versión estable MariaDB 10.3.10 lanzada en 2018. Otro punto relevante es que las versiones de MariaDB salen aproximadamente cada dos meses. Dicho lo anterior, si se requiere conocer más a fondo las versiones de MariaDB o MySQL se recomienda visitar la página de Oracle con los manuales de referencia de los servidores MySQL, además de la página MariaDB Foundation. Para este trabajo se utilizaron varias versiones, ya que las pruebas del sistema de alarma se realizaron en varios servidores.

Habría que decir también que para poder utilizar la base de datos MySQL no es necesario Apache ya que simplemente se puede descargar el servidor MySQL de su página oficial como un ejecutable. La importancia de Apache radica en que alojará el administrador PhpMyAdmin para nuestra base de datos. Asimismo, con PhpMyAdmin se pueden crear usuarios, bases de datos, tablas y dar permisos al o

los usuarios, todo esto de forma gráfica, más adelante se mostrará esto. A continuación, veamos que significa la P en LAMP.

Ⅱ.15.5 PHP

La letra P en LAMP puede hacer referencia a PHP (Hypertex Pre-Processor), Perl o Python. Para este proyecto se usó PHP5.

PHP es un lenguaje de programación de hipertexto fácil de utilizar si se tiene práctica en C, con PHP es como se logra la conexión a la base de datos MySQL una vez instalado PhpMyAdmin.

El lenguaje de programación PHP fue creado en 1994 por Rasmus Lerdorf. PHP originalmente fue creado para rastrear visitas de su currículum online, el conjunto de scripts tenía como nombre PHP Tools (Personal Home Page Tools). Con el paso del tiempo Rasmus Lerdorf mejoró PHP para que éste se pudiera comunicar con bases de datos, entre otras cosas. El código de PHP fue publicado para que los desarrolladores pudieran hacer uso de éste de forma gratuita y así mejorarlo.

PHP cambio de nombre en una ocasión, esto en su mismo año de creación en el mes de septiembre, haciéndose llamar FI[17], su aceptación fue poca ya que no fue bien recibido por todos, debido a su sintaxis en la programación. En 1995 se publicó una nueva versión llamada PHP con una sintaxis similar al lenguaje de programación C, con lo cual se vio bastante limitado a UNIX.

Tan solo un año después en 1996 el código fue reescrito una vez más incluyendo varias herramientas en las que son de interés el soporte a DBM, MySQL y bases de datos Progress95. Esta versión fue llamada PHP/FI. PHP es un lenguaje de programación el cual es sucesor de PHP/FI [35].

Los elementos de los que se ha venido hablando a lo largo de este capítulo son realmente útiles al trabajar en sistemas, ya sea para este proyecto o en una empresa en el mundo real, el conocer y manejar los servidores y la plataforma LAMP es de gran ayuda ya que facilita la implementación de páginas web y la creación de

[17] Forms Interpreter

formularios con ayuda de editores de texto y lenguajes de programación, además de la creación de bases de datos.

Una vez instalada toda esta infraestructura se necesitará de un software que pueda hacerse cargo de la administración, el que se utilizó es PhpMyAdmin, este ya se ha mencionado anteriormente pero ahora se explicará más a fondo en qué consiste dicha aplicación.

Ⅱ.16 PhpMyAdmin

PhpMyAdmin es un administrador de base de datos que nos permite crear bases de datos, tablas y editar los campos además de generar usuarios, así como la manipulación de sus privilegios, es decir, se puede limitar el uso de cada usuario sobre el contenido de las bases de datos.

PhpMyAdmin está escrito bajo el lenguaje de programación PHP y alojado en el servidor Apache. Otra característica de esta aplicación es que nos permite la manipulación de bases de datos siempre y cuando se esté conectado a una red y se tengan permisos del host.

PhpMyAdmin es un software libre bajo la licencia de GLP V2, es multiplataforma y la última versión es la 4.8.12, versión estable. Inicialmente PhpMyAdmin surge con el trabajo de Tobias Ratschiller, el cual por falta de tiempo deja el proyecto, actualmente *The PhpMyAdmin proyect* es quien se encarga del desarrollo de PhpMyAdmin [36], [37].

Ahora que sabemos qué es esta aplicación veamos cómo se instala. Existen dos formas de instalar PhpMyAdmin, una es por línea de comandos, la cual es la que se empleó para este proyecto, pero también existe una aplicación llamada XAMP, la cual es un ejecutable, que al ser instalada te da un panel de control con el cual se puede administrar la aplicación de una manera muy sencilla ya que con ella podemos acceder a toda su paquetería.

Concluyendo con la explicación, la imagen de la figura Ⅱ.10 muestra las mascotas y logos de la plataforma LAMP con la que se trabajó, además del administrador de base de datos PhpMyAdmin.

Figura Ⅱ 10. Logos de la plataforma LAMP sobre la que se trabajó.

En adición al punto anterior, este trabajo cuenta con una plataforma a la que cualquiera puede tener acceso, es lo que hace atractivo el proyecto ya que su simplicidad lo hace muy útil no solo a un nivel académico sino también a un nivel empresarial. Volviendo al tema que nos ocupa, a continuación se muestra el servidor en el cual se montará todo este conjunto de herramientas para de esta forma tener nuestro sistema de seguridad.

Ⅱ.17 Tarjeta de desarrollo utilizada como servidor de aplicaciones

La tarjeta de desarrollo que funge como host para toda la infraestructura de este proyecto es una Intel Galileo generación 2. Esta tarjeta cuenta con la cualidad de poder instalarle un sistema operativo con el mínimo de recursos, usualmente llamado minimal.

El sistema operativo en el cual se montó toda la plataforma LAMP es una distribución estable de GNU/Linux, Debian. El motivo de utilizar una tarjeta de desarrollo y no una computadora personal con algún sistema operativo Windows, Macintosh o GNU/Linux es que el servidor sea fácil de transportar además de ser discreto.

Cabe mencionar que, aunque la base del trabajo está sustentada por el servidor Debian instalado en la tarjeta Galileo, las pruebas también se realizaron en 3 sistemas operativos más, los cuales son: Centos 7 minimal, Windows 10 y Windows

7. Esto último con la finalidad de demostrar que el sistema de alarma es multiplataforma. Además, en cada una de las pruebas se utilizan métodos diferentes para la instalación de la base de datos y el servidor.

Finalmente, se muestra cómo es que el sistema recogerá la información según sea requerido.

Ⅱ.18 Almacenamiento de la información

La información del sistema de alarma será almacenada de forma automática cada que un evento ocurra y éste no esté contemplado como una situación segura, por ejemplo, dejar sola la vivienda y que se detecte presencia adentro de ésta o intentos de romper los cristales e inclusive forzar la puerta. Cabe destacar que la conexión del sistema será vía Ethernet y no inalámbrica. El propósito de que la conexión sea vía Ethernet y no inalámbrica es que de esta manera la conexión es segura y estable.

Una vez aclarado lo anterior, para guardar la información se descargó una biblioteca en la tarjeta Arduino ethernet, la cual permite al usuario enlazar su servidor Arduino con la base de datos MySQL instalada en el host Galileo. La biblioteca que se descargó es la MySQL_Connector_Arduino-master y la versión del IDE de Arduino que se empleó para este trabajo es la Arduino-1.8.5. Además, la biblioteca cuenta con un manual, el cual es de gran ayuda ya que éste contiene toda la información necesaria para crear y guardar información en una base de datos MySQL o MariaDB.

El proceso para guardar la información en la base de datos es el siguiente:

- Crear un usuario y contraseña. El usuario debe contar con todos los permisos. El usuario creado puede tener acceso solamente desde el host que le sea asignado en la red (puede asignarse local, algún host específico o cualquier host).
- Se debe de crear una base de datos en MySQL utilizando los comandos SQL, asignándole nombre a ésta.
- Una vez creada la base de datos se debe de seleccionar.

- Ya seleccionada la base de datos se debe crear una tabla y en ella crear los campos necesarios, como son: el incremento en el registro, nombre del evento (mensaje que se quiera guardar), fecha y hora del suceso.

Una vez cubiertos todos los puntos anteriores (ya sea en un host Debian, GNU/Linux, Windows o Macintosh), se procede a configurar la tarjeta Arduino ethernet para que sea utilizada como un servidor, a ésta se le debe de asignar la dirección de red y los datos de la base de datos, es decir: IP del host, nombre y contraseña.

El siguiente paso es habilitar la conexión remota, este método es diferente en Debian, Centos 7 y Windows 7 o 10, por lo cual se explicará en el siguiente capítulo.

Una vez concluidos estos pasos se debe proseguir a la realización de pruebas de conexión para comprobar que el sistema funcione. Todas estas pruebas se muestran en el siguiente capítulo.

Una vez que las pruebas resulten exitosas se procederá a crear un boceto, es decir, un programa con el cual el sistema de alarma guardará registro de las intrusiones y además activará la alarma de forma automática. Antes de proseguir, para poder programar todos los bocetos se debe de realizar un diagrama de flujo de cada uno, esto para posteriormente trasladar la idea a un programa que realice de forma lógica todas las instrucciones deseadas.

Ⅱ.19 Diagramas de flujo

Un diagrama de flujo es la representación gráfica de un algoritmo, un algoritmo es una serie de instrucciones ordenadas para realizar cualquier tipo de trabajo de forma eficiente y lógica.

El siguiente ejemplo muestra un diagrama de flujo para decidir si seguir acostado por la mañana o ya es hora de levantarse de la cama para ir a trabajar. Véase figura Ⅱ.11.

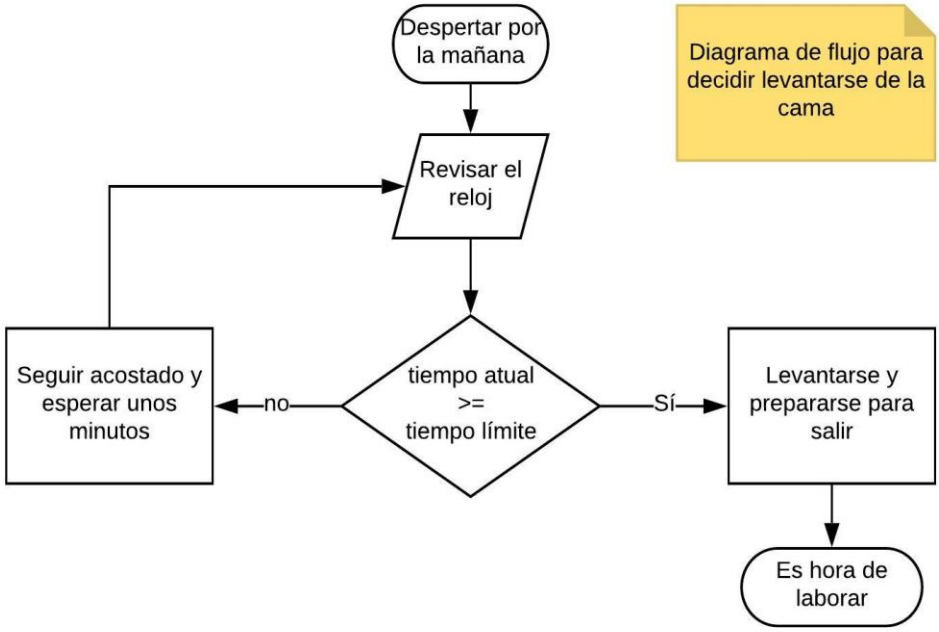


Figura Ⅱ 11. Diagrama de flujo que muestra de forma simple la decisión de levantarse por la mañana o no hacerlo.

El diagrama de flujo de la figura Ⅱ.11 muestra una toma de decisión basándose únicamente en el tiempo actual y el tiempo crítico que una persona tiene para prepararse y llegar a su trabajo.

Ⅱ.19.1 Simbología de los diagramas de flujo

Es de suma importancia que un diagrama de flujo explique de forma lógica los procedimientos para realizar una tarea, pero de igual importancia es conocer el tipo de simbología que se requiere al momento de crear un diagrama de flujo. La siguiente figura muestra parte de la simbología necesaria en la creación de un diagrama de flujo. Véase figura Ⅱ.12.

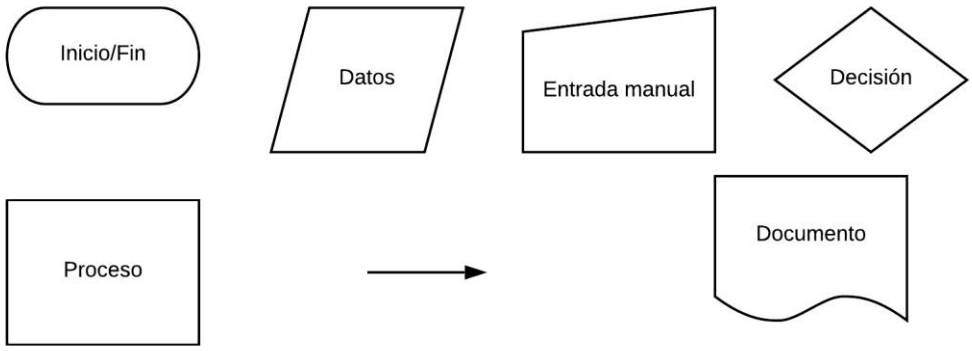


Figura Ⅱ 12. Símbolos utilizados en los diagramas de flujo realizados en este trabajo.

En la figura Ⅱ.12 se muestra la simbología utilizada para la creación de los diagramas de flujo que se emplearon en este trabajo, cabe aclarar que no son los únicos símbolos existentes, pero solamente se hará mención de éstos puesto que son los únicos empleados.

La función de cada símbolo es:

- Datos. – este campo se utiliza para describir entradas y salidas, es decir, lo que entra al sistema y lo que saldrá de él, por ejemplo, la configuración del sistema.
- Proceso. – aquí se describe la acción correspondiente que se debe de realizar, por ejemplo, prender un led o apagarlo.
- Decisión. – en este apartado se representa la toma de decisiones basándose en una condición, por ejemplo, avanzar o quedarse en un ciclo hasta que se cumple la condición necesaria para seguir el camino.
- Terminado. – este símbolo simplemente muestra el inicio y final del algoritmo.
- Entrada manual. – esta casilla se utiliza para representar la entrada de valores por medio de un teclado, por ejemplo, al utilizar un teclado matricial este símbolo representaría la tecla capturada al presionarla en el teclado.
- Flechas. – las flechas se utilizan para enlazar ideas, es decir, guia las instrucciones de forma ordenada y lógica.

- Documento. – con ayuda de él se representan de forma impresa resultados, por ejemplo, un mensaje.

Teniendo en cuenta la función que cumple un diagrama de flujo, a continuación, se hace mención de otra herramienta necesaria para la creación del sistema de alarma, es decir, el uso de un teclado matricial para el control del sistema.

Ⅱ.20 Teclado matricial

Un teclado matricial es un dispositivo cuya función es detectar cuando una tecla es pulsada y con ello dar órdenes a un sistema, por ejemplo, el teclado de la computadora está compuesto por varios teclados segmentados, es decir, una unión de teclados matriciales.

Con respecto a su arquitectura física, los teclados matriciales son arreglos de NxM conexiones, teniendo N filas y M columnas, se utilizan en base a coordenadas y cada punto de éstas está conectado por medio de un pulsador. Véase figura Ⅱ.13

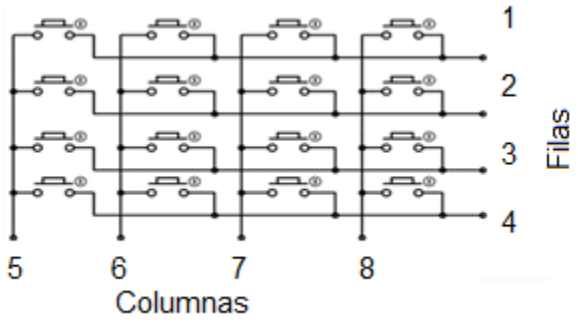


Figura Ⅱ 13. Arquitectura interna de un teclado matricial.

Para entender el funcionamiento del teclado es necesario conocer qué es una entrada pull up y pull dow.

Un resistor de pull up o down es un arreglo con un resistor que evita que un circuito tenga lecturas erróneas cuando se encuentra en reposo, es decir, elimina el ruido que podría introducirse al sistema.

Pull up es un arreglo que, como su nombre lo indica, manda un estado alto cuando el sistema está en reposo, análogamente, pull dow es un arreglo que envía un estado bajo en todo momento si el sistema está en reposo [38]. Véase figura Ⅱ.14.

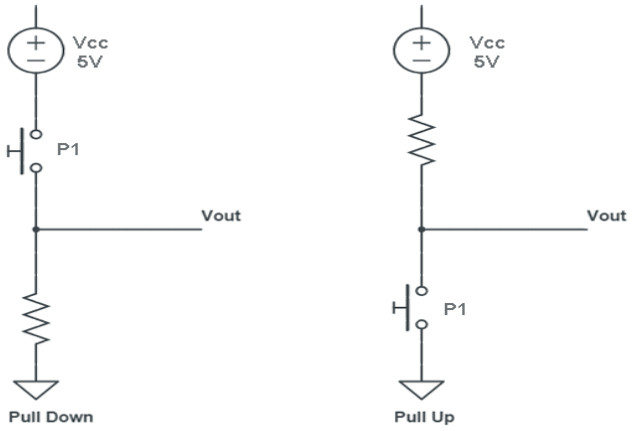


Figura Ⅱ 14. Resistencia de pull up y pull down.

En Arduino no es necesario utilizar dicha resistencia física, la cual equivaldría a 10 kΩ, simplemente se debe de hacer uso de la función de software incluida en el IDE de Arduino, INPUT_PULLUP, según lo requiera el proyecto.

Continuando con la explicación, la conexión de cada pulsador es una conexión pull up en serie, en donde cada una podría verse como un solo pulsador al cual se le asigna una respuesta, por ejemplo, un carácter. Es por lo anterior mencionado que para hacer uso del teclado se deben de configurar todos los pulsadores con resistencias ya sea de pull up o pull down según sea conveniente.

Ⅱ.20.1 Configuración del teclado matricial

Los anteriores conceptos se esclarecerán con la siguiente explicación que indica cómo se debe de configurar el teclado para su uso. Todo esto siguiendo las siguientes instrucciones:

- Para poder utilizar el teclado matricial es necesario crear variables, las cuales en conjunto permitirán el uso del teclado de forma correcta, entre estas variables se tiene que considerar la creación de filas y columnas de la matriz

correspondiente al teclado, la realización se lleva a cabo por medio de arreglos.

- Una vez que se ha creado la matriz, el siguiente paso es asignar un carácter a cada pin del Arduino, para realizarlo es necesario hacer un barrido entre filas y columnas asignando a cada coordenada un carácter de la matriz, posteriormente es necesario configurar todos los pines de las filas y columnas como entradas pull up.
- Una vez realizado todo lo anterior, simplemente se debe de crear un boceto en el IDE de Arduino que imprima cada carácter cuando un pulsador sea pulsado.

En el diagrama de flujo de la figura Ⅱ.15 se detallan los procesos para lograr la configuración del teclado. Cabe hacer mención de que existe una biblioteca llamada *keypad* con la cual también se puede hacer uso del teclado. Esta biblioteca es la que se emplea en este trabajo.

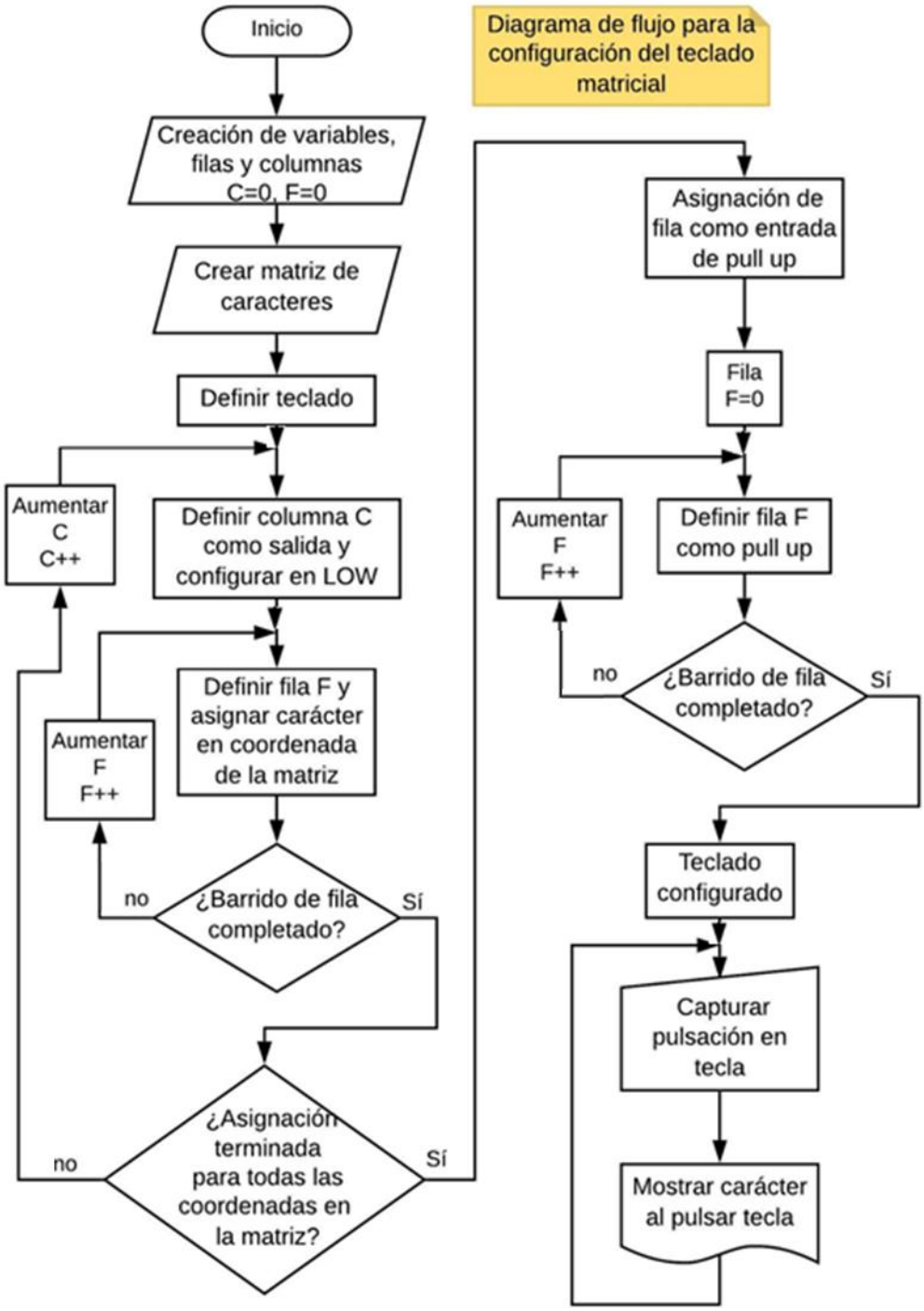


Figura Ⅱ 15. Diagrama de flujo que muestra los procesos para la configuración del teclado matricial.

Como ya se ha hecho mención, el teclado matricial se empleará para activar o apagar el sistema de alarma. A continuación, se explicará cómo es que se creó una contraseña de 4 dígitos con la cual se tendrá acceso al sistema de alarma por medio del teclado matricial.

Ⅱ.20.2 Algoritmo de contraseña del sistema de alarma

Para la realización de la contraseña de activación del sistema de alarma solamente es necesario crear arreglos y almacenar valores de cada uno de sus elementos para posteriormente compararlos con otro arreglo ya establecido (una contraseña). A continuación, se muestra cómo crear un algoritmo y su diagrama de flujo para con ello crear una contraseña, el programa realizado con apoyo de este algoritmo se puede visualizar en el **anexo A**.

- Como el Arduino ethernet y el Arduino uno se comunicarán entre sí, es necesario que el servidor Arduino ethernet dé la orden para que el teclado pueda inicializarse y con ello comience a recoger la información de las pulsaciones, es decir, el teclado esperará a que el Arduino ethernet termine de configurarse y conectarse a la base de datos MySQL.
- Una vez que se establezca la conexión con la base de datos, el teclado podrá comenzar a capturar las pulsaciones.
- Cada pulsación capturada por el teclado se almacenará en una variable inicializada en cero y se irá incrementando hasta llegar a cuatro pulsaciones.
- Cuando el conteo de pulsaciones llegue a cuatro se realizará una comparativa entre cada elemento del arreglo propuesto (contraseña) y las variables almacenadas.
- Dependiendo del resultado, el sistema de alarma realizará un proceso, ya sea como sistema interno, externo o como un sistema apagado.

El diagrama de flujo de la figura Ⅱ.16 explica los procesos mencionados. Prosiguiendo con el tema, una vez que sabemos cómo es que el teclado permitirá el acceso al servidor y con ello hacer uso de la base de datos por medio de una contraseña, es necesario explicar el funcionamiento del sistema de alarma y sus

diferentes funciones una vez que el servidor ha logrado conectarse a la base de datos, además de activar la red de sensores.

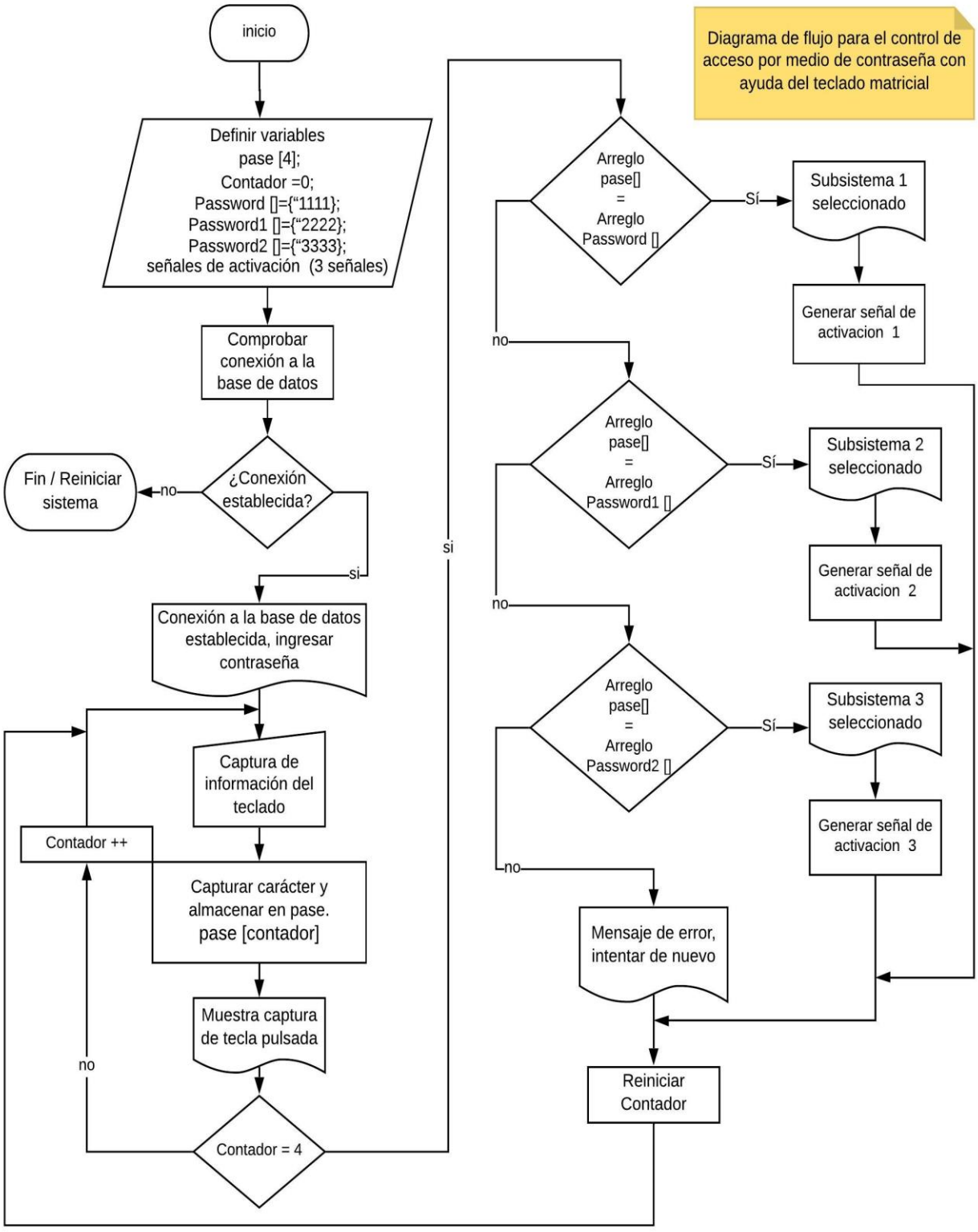


Figura Ⅱ 16. Diagrama de flujo que muestra el proceso para la creación de una contraseña para la selección de un sistema.

Ⅱ.21. Algoritmo que explica el funcionamiento del sistema de alarma

En este apartado se explica cómo es que trabaja el sistema de alarma, la conexión a la base de datos, el inicio del teclado y la selección de subsistema, es decir, se explica el algoritmo de funcionamiento del sistema de alarma (el programa realizado con apoyo de este algoritmo se puede visualizar en el **anexo B**):

- Iniciando el sistema lo primero que éste realizará es la configuración para lograr entablar una conexión con la base de datos MySQL. Esto implica realizar una conexión por medio de la IP de la base de datos, usando el nombre del usuario y una contraseña.
- Una vez que la configuración sea exitosa se procede a inicializar el teclado por medio de una señal digital.
- Una vez que el teclado se haya inicializado se procede a la captura de la señal proveniente de éste. Dependiendo de la contraseña, el sistema realizará determinada acción.
- Las acciones que pueden ejecutarse son: activar el sistema por completo, activar parte de él o mantener inactivo todo el sistema.
- Una vez activado, el sistema de alarma accede a la red de sensores, el sistema guardará un registro de la actividad detectada por dicha red en una base de datos determinada, todo esto con base en las lecturas proporcionadas por los sensores y las condiciones de uso.
- El sistema de alarma tendrá en funcionamiento un sistema externo que alertará si alguien quiere irrumpir por la ventana de la vivienda.

El siguiente diagrama de flujo ilustra el algoritmo propuesto. Véase figura Ⅱ.17, 18 y 19.

El diagrama de flujo está dividido en tres partes, cada una enlaza y continúa la idea por medio de globos y letras del alfabeto griego.

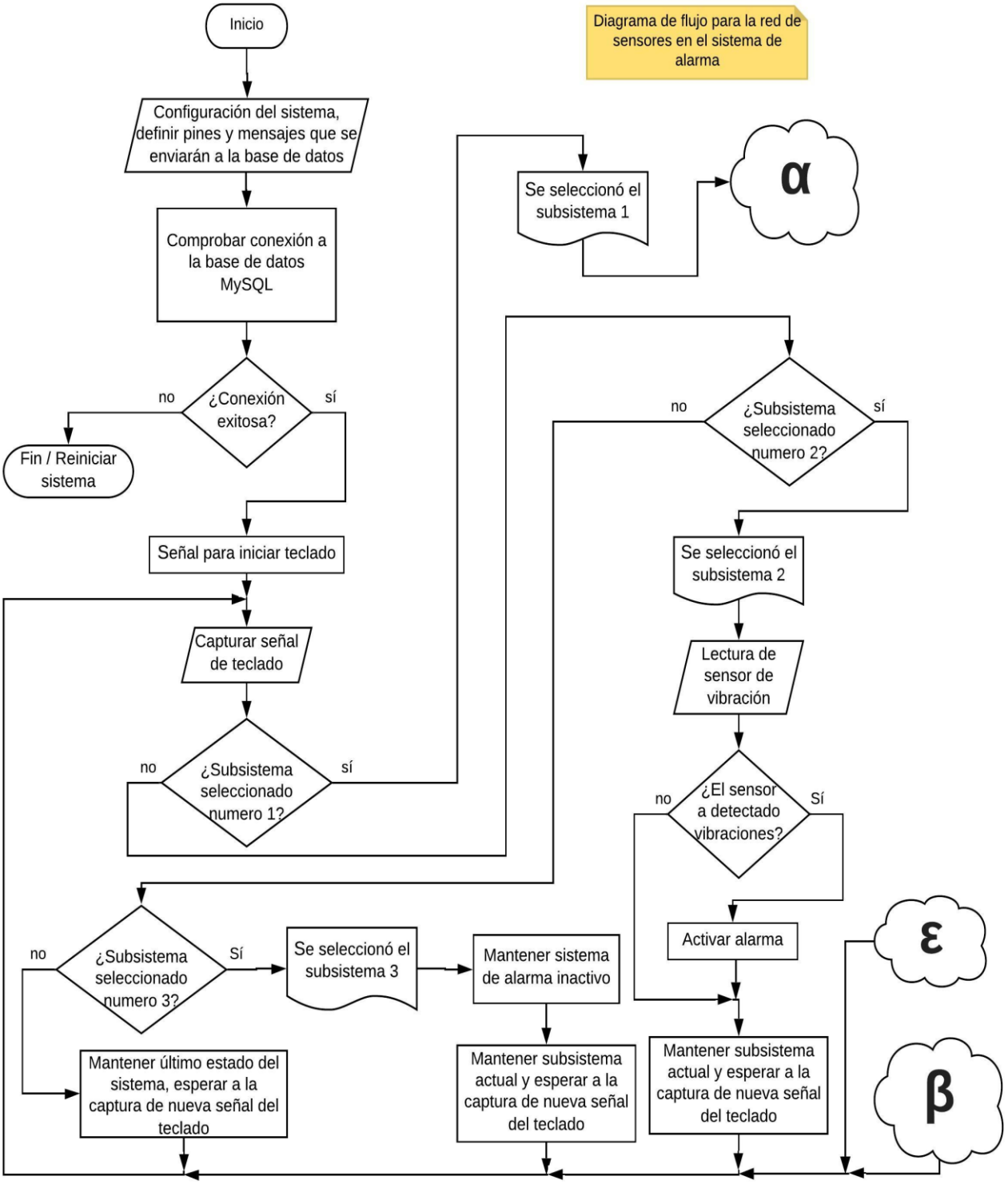


Figura Ⅱ 17. Diagrama de flujo que muestra el algoritmo propuesto para la implementación del sistema de alarma de este trabajo (parte a).

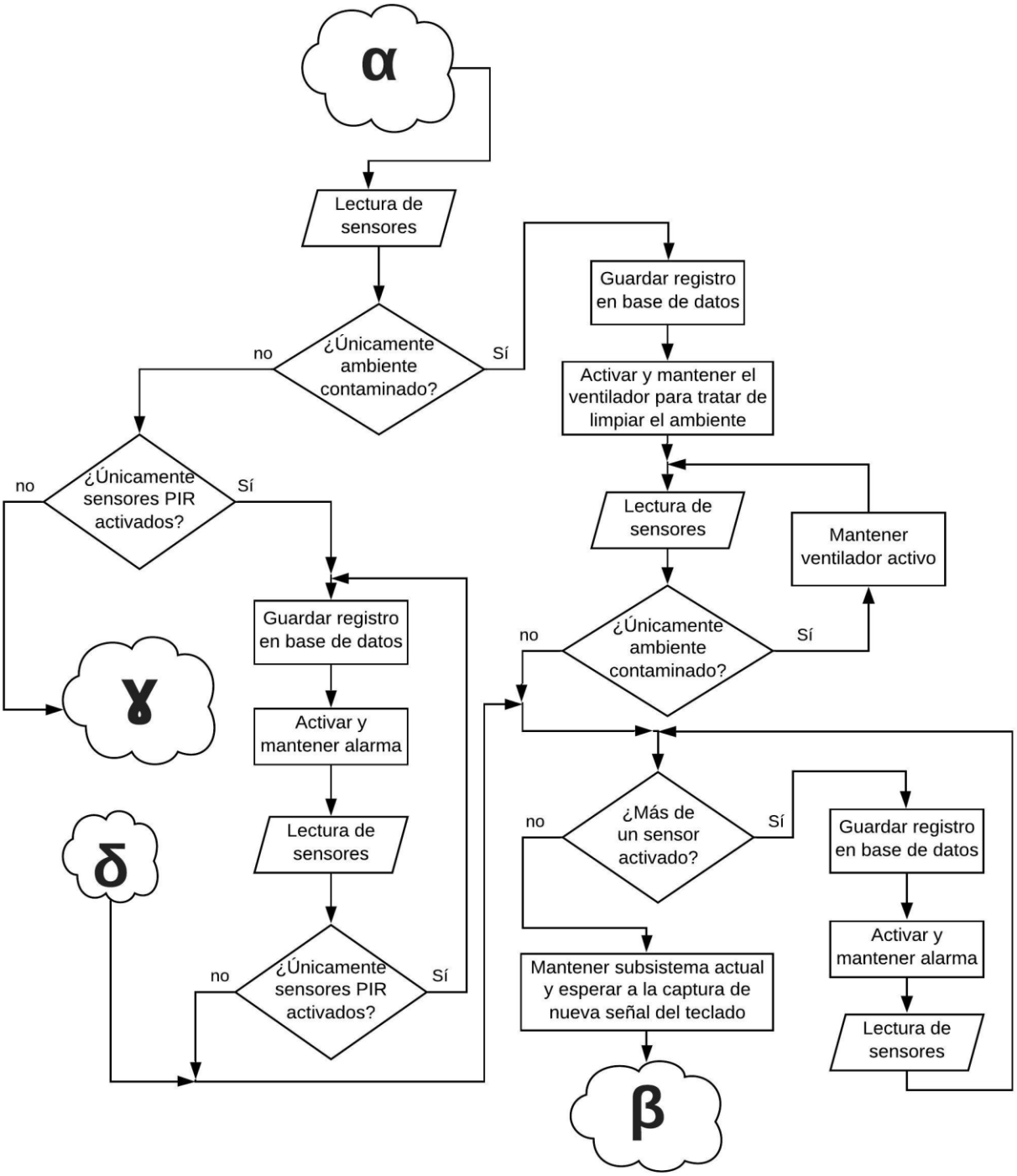


Figura Ⅱ 18. Diagrama de flujo que muestra el algoritmo propuesto para la implementación del sistema de alarma de este trabajo (parte b).

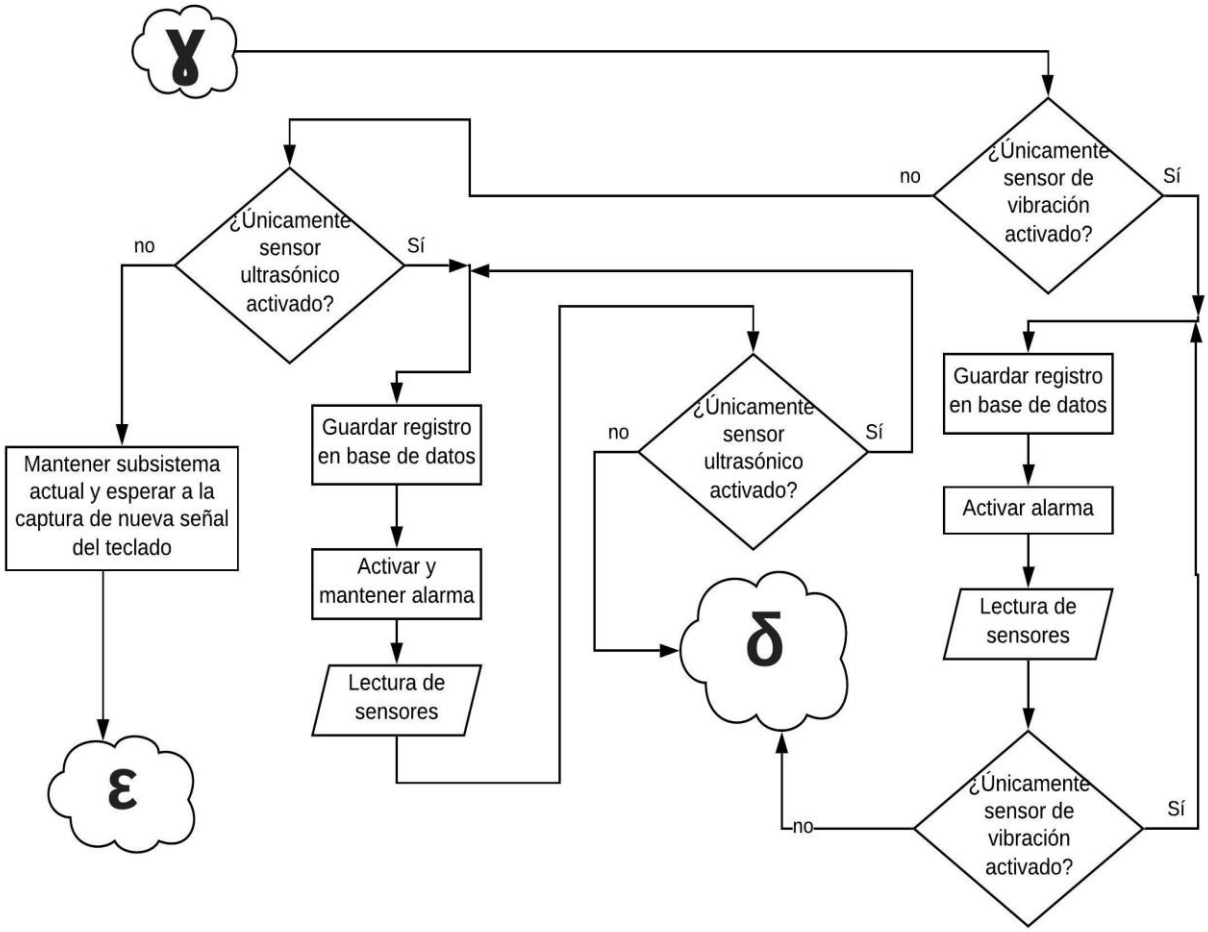


Figura Ⅱ 19. Diagrama de flujo que muestra el algoritmo propuesto para la implementación del sistema de alarma de este trabajo (parte c).

Para comprender mejor todo lo anterior mencionado a continuación se muestra un diagrama en donde se aprecian todos los elementos empleados para el desarrollo del Sistema de alarma automatizado, véase figura Ⅱ. 20.

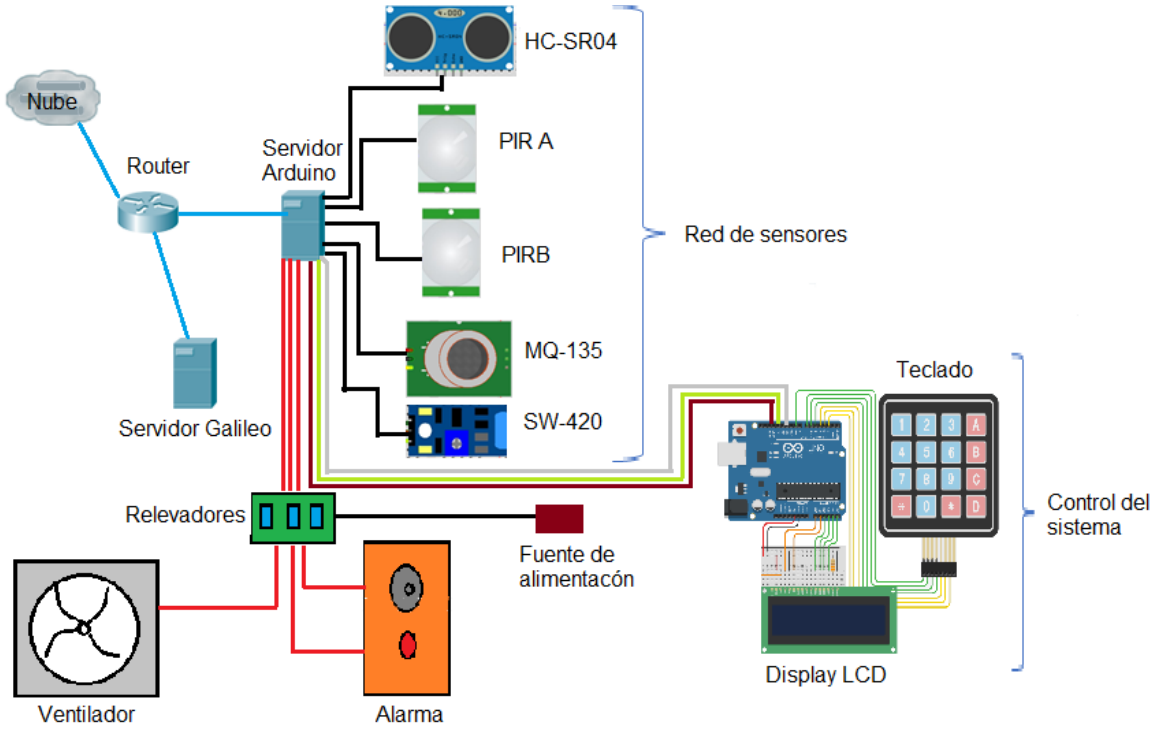


Figura Ⅱ 20. Diagrama de conexión del Sistema de alarma automatiza.

En la figura Ⅱ. 20 se muestran los servidores Arduino y Galileo. Cuando el sistema se inicia lo primero que debe de realizar es la conexión a la base de datos MySQL, una vez que se realiza la conexión el servidor Arduino manda una señal al microcontrolador y con ello activa el teclado matricial. El teclado matricial se utiliza para poder ingresar una contraseña y seleccionar un subsistema, de los que ya ha hecho mención. Con ayuda del display LCD se puede visualizar un mensaje de inicio del sistema y de captura de contraseña. El servidor Arduino es el encargado de recolectar la información extraída de cada sensor de la red de sensores. La información obtenida a través de la red de sensores se utiliza para mandar registros y llenar las bases de datos MySQL que se encuentran en el servidor Galileo. Asimismo, la información obtenida por la red de sensores se utiliza para activar con ayuda de módulos relé el sistema de alarma o el ventilador, según sea necesario.

Para concluir, una vez comprendido cómo es que cada parte de sistema de alarma funcionará, para de esta manera crear un sistema de alarma que guarde los registros de la actividad de forma automática si así se desea. Se da por finalizado este capítulo y da inicio el capítulo 3, en el cual se describen las pruebas al sistema que se llevaron a cabo, esto implica la configuración y prueba de cada sensor, tarjetas de desarrollo y los diferentes hosts, para demostrar que el sistema es multiplataforma.

CAPÍTULO Ⅲ: PRUEBAS Y RESULTADOS

Ⅲ.1 Pruebas del sistema de alarma

En esta sección se probará el funcionamiento del sistema para determinar la mejor configuración para un rendimiento óptimo de éste. El objetivo es probar todos los sensores por separado y posteriormente hacer pruebas de conexión entre el servidor Arduino ethernet y la base de datos MySQL.

Una vez que se hayan logrado todas las pruebas anteriormente mencionadas se proseguirá con la siguiente etapa, es decir, hacer las pruebas del sistema de alarma completo. Se probarán 3 cosas:

1. Verificar el funcionamiento del sistema de alarma con base en los diagramas de flujo de la sección anterior.
2. Determinar si la respuesta del sistema para guardar la información en la base de datos con base en la respuesta de los sensores es correcta.
3. Implementar el sistema de sensores en una maqueta que simulará un sistema de alarma ya instalado en una vivienda para determinar su adecuado funcionamiento.

Lo primero a realizar es probar los sensores. Cada sensor será puesto a prueba en condiciones específicas para su óptimo uso en este proyecto y con ello calibrarlos de la mejor manera para que su rendimiento sea el adecuado según su posición, es decir, cada sensor estará situado en la maqueta según sea conveniente para este proyecto.

Ⅲ.2 Calibración de los sensores

A partir de este punto se comprobará el funcionamiento de cada uno de los sensores de la red de sensores que constituye el sistema de alarma, en adición cada sensor será calibrado de tal forma que resulte un sistema de lo más eficaz posible.

Ⅲ.2.1 Configuración del sensor PIR

El sensor PIR cuenta con 3 pines y dos potenciómetros. Dos de los pines sirven para la alimentación del módulo PIR, el tercer pin manda una señal digital a un puerto del microcontrolador. El módulo PIR cuenta con dos potenciómetros, los cuales se utilizan para calibrar sensibilidad y tiempo de respuesta del módulo. En la

figura Ⅲ.1 se muestran todas las terminales del módulo PIR empleado para este proyecto.

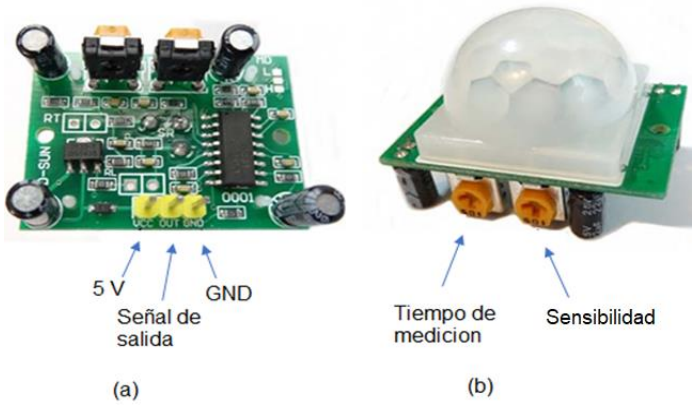


Figura Ⅲ 1. Sensor PIR: (a) Módulo donde se encuentran los tres pines de conexión, (b) Potenciómetros de calibración del módulo PIR.

Para calibrar este módulo basta con girar los potenciómetros que se muestran en la figura Ⅲ.1 (b), uno es para fijar el tiempo de medición y el otro es para establecer la distancia de detección (sensibilidad), en este caso se tomaron 3 niveles de sensibilidad los cuales dependen del potenciómetro, es decir, de cómo se manipule la perilla, los niveles se ajustaron con un valor mínimo, medio y alto.

Para el desarrollo del presente proyecto se usaron dos módulos PIR. El escenario de prueba en donde se realizó la calibración de los sensores fue un cuarto pequeño con los objetos pertenecientes a una habitación común y en el laboratorio B207 de la UACM. Dicho lo anterior, el circuito de prueba utilizado para la configuración de este módulo es el que se muestra en la figura Ⅲ.2.

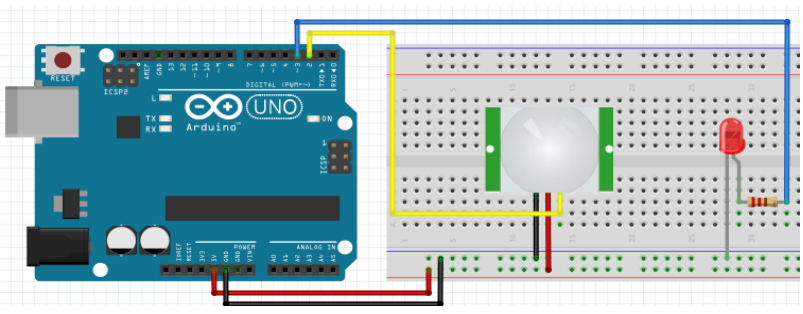


Figura Ⅲ 2. Circuito para calibrar el sensor PIR.

Ahora bien, para comprobar el funcionamiento del circuito se realizó un código de prueba en el IDE de Arduino (**anexo C**), para imprimir un mensaje de salida usando la herramienta monitor serie del IDE de Arduino cuando ocurra un evento. El evento mencionado es la lectura de movimiento en el cuarto, por ejemplo, una persona atravesando un pasillo, en este caso entrando a una habitación. El circuito junto con la tarjeta Arduino ejecutando el código de prueba muestra el mensaje de la figura Ⅲ.3 al detectar movimiento. Además de lo anterior, el circuito también envía una señal digital y con ello enciende un led en respuesta al evento.

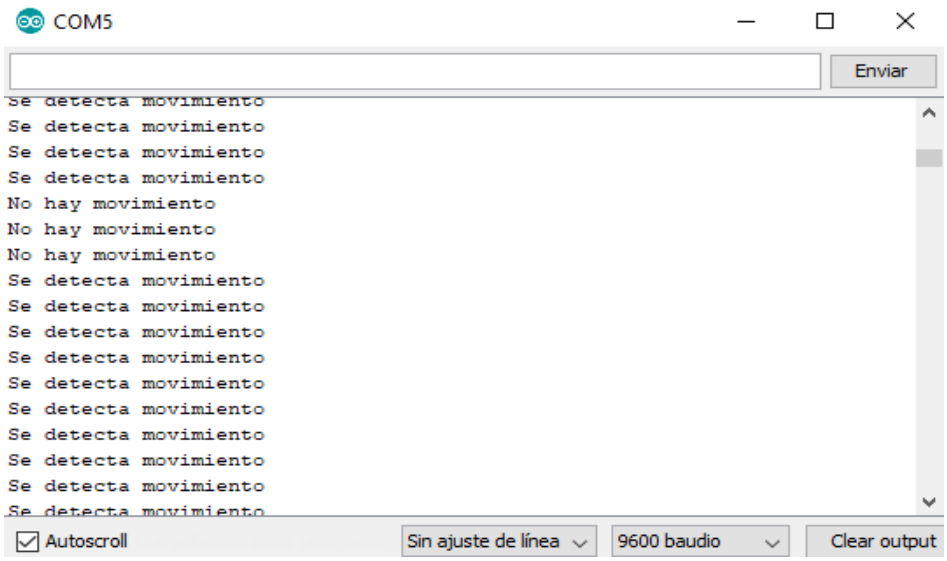


Figura Ⅲ 3. Mensaje al detectar movimiento: mensaje de salida capturado con la herramienta monitor serie del IDE de Arduino.

Las mediciones para lograr calibrar los sensores PIR, y con ello tener el mejor funcionamiento, se realizaron de dos maneras y en ambas los resultados fueron prácticamente los mismos, es por ello que para cada sensor se presenta la información recabada en una sola tabla.

1. Objetos a una distancia de entre 5 y 20 cm.
2. Objetos a una distancia de entre 1 y 3 metros.

El ángulo de medición fue aproximadamente de 120°, tomando de referencia una superficie plana para colocar el sensor.

Para fines prácticos se nombraron a los sensores como sensor A y sensor B. Las pruebas arrojaron los resultados de las tablas Ⅲ.1 Y Ⅲ.2.

Sensor A			
Potenciómetro para ajustar la distancia de detección	Potenciómetro para ajustar el tiempo de respuesta	Valor arrojado con movimiento (persona caminando)	Valor arrojado sin movimiento (persona inmóvil)
Mínimo	Mínimo	Voltaje lógico = 1	Voltaje lógico = 0
Medio	Mínimo	Voltaje lógico = 1	Voltaje lógico = 0
Alto	Mínimo	Voltaje lógico = 1	Voltaje lógico = 0
Mínimo	Medio	Voltaje lógico = 1	Voltaje lógico = 1
Medio	Medio	Voltaje lógico = 1	Voltaje lógico = 1
Alto	Medio	Voltaje lógico = 1	Voltaje lógico = 1
Mínimo	Alto	Voltaje lógico = 1	Voltaje lógico = 1
Medio	Alto	Voltaje lógico = 1	Voltaje lógico = 1
Alto	Alto	Voltaje lógico = 1	Voltaje lógico = 1

Tabla Ⅲ 1. Valores arrojados por el sensor A.

Sensor B			
Potenciómetro para ajustar la distancia de detección	Potenciómetro para ajustar el tiempo de respuesta	Valor arrojado con movimiento (persona caminando)	Valor arrojado sin movimiento (persona inmóvil)
Mínimo	Mínimo	Voltaje lógico = 1	Voltaje lógico = 0
Medio	Mínimo	Voltaje lógico = 1	Voltaje lógico = 0
Alto	Mínimo	Voltaje lógico = 1	Voltaje lógico = 0
Mínimo	Medio	Voltaje lógico = 1	Voltaje lógico = 1
Medio	Medio	Voltaje lógico = 1	Voltaje lógico = 0
Alto	Medio	Voltaje lógico = 1	Voltaje lógico = 0
Mínimo	Alto	Voltaje lógico = 1	Voltaje lógico = 1
Medio	Alto	Voltaje lógico = 1	Voltaje lógico = 1
Alto	Alto	Voltaje lógico = 1	Voltaje lógico = 1

Tabla Ⅲ 2. Valores arrojados por el sensor B.

Al calibrar los sensores se notaron diferencias. Entre estas diferencias se incluye que la sensibilidad del sensor A es mejor que la del sensor B, esto puesto que el sensor A da una respuesta más rápida ante el movimiento, tomando aproximadamente entre 1 y 2 segundos desde que un objeto comenzaba a moverse hasta que se enviaba la respuesta del sensor PIR y el led se encendía. El sensor B tardaba un poco más en responder al evento, además, el sensor B requería entre 5 y 7 segundos para tomar la muestra, mientras que el sensor A sólo requería entre 3 y 4 segundos desde que el mismo objeto comenzaba a moverse, para solucionar este problema se cambió el PIR B de tal forma que el nuevo sensor respondiera de forma efectiva, tal como lo hace el sensor A.

Otra característica que se presenta al calibrar los sensores es que cuando el sistema se inicializa, los sensores PIR mandan un 1 lógico. Esto se corrigió con un pequeño retardo de 5 segundos en la configuración inicial del programa de prueba, de manera que el sensor envíe un 0 lógico. La necesidad de que al inicializarse los sensores PIR manden un 0 lógico radica en que con ello se evitan lecturas erróneas.

Para la prueba de menor distancia solamente se utilizaron objetos de plástico, los objetos detectables se movían por delante y por detrás de un cristal. Una diferencia entre los objetos se presenta con los materiales de prueba, por ejemplo, con el plástico los sensores tardaban un poco más en realizar la detección (entre 1 y 2 segundos) mientras que en la detección de manos humanas la reacción de los sensores era inmediata, además al utilizar cualquier tipo de objeto y moverlo por detrás de un cristal el sensor no envía ninguna señal, esto indica que el módulo no puede ser activado con movimiento de una persona u objeto que se mueva detrás de una ventana.

Con lo anterior mencionado se tienen las siguientes recomendaciones al calibrar un sensor PIR:

- Calibrar el sensor en un escenario estático.
- Utilizar sensores cuyo espejo de Fresnel esté en óptimas condiciones.
- Calibrar el sensor considerando tiempos mínimos para la detección.

De igual importancia, el escenario en donde se instale un sensor PIR debe de cumplir con ciertos requisitos. Algunos de los requisitos al instalarlo son:

- No poner nada enfrente del sensor, por ejemplo, un ventilador o plantas ya que el movimiento de estas podría activar dicho sensor.
- El sensor no se debe de colocar de frente al intruso, es decir, no se debe de colocar en un pasillo o alguna zona en donde el intruso pueda caminar de frente al sensor ya que esto podría evitar su detección, la razón es que una persona que camina de frente casi no altera su movimiento respecto a un observador que ve cómo se va aproximando el sujeto hacia su posición.
- El sensor no debe de estar en línea de vista con alguna fuente de luminosidad, por ejemplo, junto a una ventana por donde entre la luz solar o incluso cerca de las lámparas ya que estas pueden activar el sensor por los cambios de radiación.

El sensor PIR es uno de los sensores que, para este proyecto, resultó ser difícil de calibrar, pero, es un sensor muy útil. Los siguientes sensores son más fáciles de calibrar y utilizar.

Ⅲ.2.2 Configuración del sensor de vibración

La calibración de este sensor es más sencilla puesto que solamente requiere de un pequeño golpe para ser activado.

El módulo SW-420 contiene, como ya se ha mencionado, un sensor SW-420 y un amplificador operacional LM393, el cual se utiliza como comparador de voltaje de precisión, en conjunto con algunos resistores y capacitores, los cuales forman un comparador de voltaje el cual manda una señal digital que es leída por un puerto del Arduino uno. Véase figura Ⅲ.4.

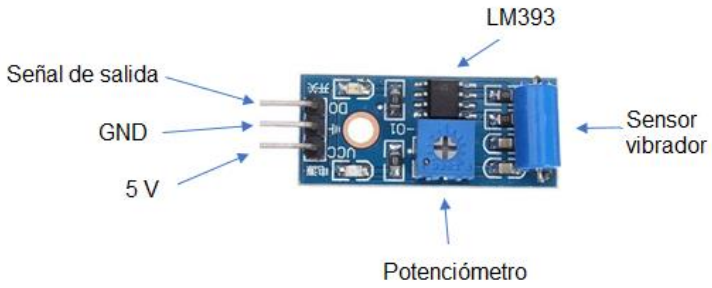


Figura Ⅲ 4. Módulo SW-420.

Al igual que la mayoría de sensores existentes, este sensor cuenta con un potenciómetro para calibrar su sensibilidad. Esta sensibilidad se tomó en tres niveles, los cuales son: mínimo, medio o alto, de acuerdo al ajuste mínimo, medio y máximo de la perilla del potenciómetro.

Para realizar las pruebas de funcionamiento, el sensor se colocó en una superficie de vidrio la cual es una pequeña ventana. El sensor funciona en todo momento como un comparador de voltaje, mandando un estado de 0 lógico a menos que el circuito se cierre a causa de una vibración, el sensor al recibir un golpe provocará que en su interior dos materiales hagan contacto y de esta forma envía una señal digital de 1 lógico a la salida del circuito. En la figura Ⅲ.5 se muestra el circuito de prueba para la calibración del módulo.

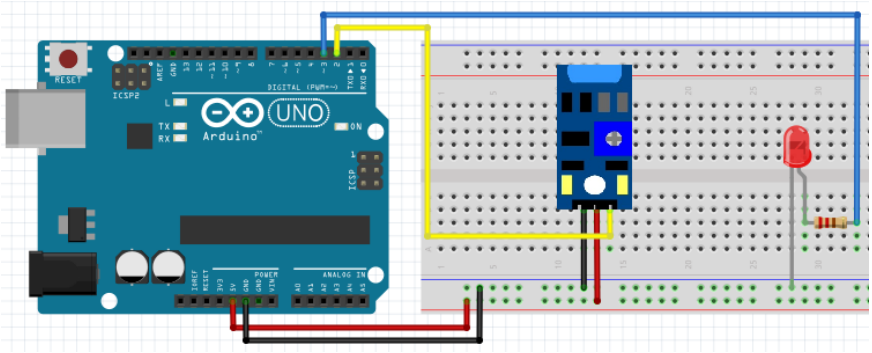


Figura Ⅲ 5. Circuito de prueba del sensor SW-420.

Continuando con las pruebas, los golpes proporcionados a la superficie fueron de pequeño impacto demostrando que, si alguien o algo intenta forzar la ventana o incluso si la rompe, el sensor se activará mandando una señal digital igual a un 1 lógico, cuando esto ocurra el led se prenderá indicando la existencia de alguna

fuerza no deseada. En caso contrario a lo mencionado, la señal se mantendrá con un 0 lógico y el led estará apagado.

Para probar lo anterior, se realizó un programa de prueba en el IDE de Arduino (**anexo D**). El programa de prueba mostrará un mensaje cuando el sensor detecte alguna alteración. Véase figura Ⅲ.6.

Figura Ⅲ 6. Mensaje al detectar vibraciones: mensaje de salida de la herramienta monitor serie del IDE de Arduino.

Este programa de prueba mostró que cuándo el cristal no es golpeado el mensaje “sin novedad” se mantiene como salida, pero por el contrario si es golpeado el mensaje se va alternando entre “el cristal fue golpeado” y “sin novedad”. Esto es porque el circuito no manda lecturas constantes durante un cierto periodo, a diferencia del sensor PIR que sí lo hace, es decir, este sensor lo que hace es mandar pulsos en un periodo de tiempo pequeño, es por lo que los mensajes se alternan. Haciendo un pequeño paréntesis, el periodo en el que la alarma sonará puede prolongarse con la programación del microcontrolador, para este proyecto solamente se programó para permanecer activa durante un momento mandando una señal digital a un puerto del Arduino uno. Lo anterior mencionado se verá reflejado como una señal intermitente que prenderá y apagará la sirena y las luces de la alarma.

Las pruebas realizadas tomando los tres niveles del potenciómetro arrojaron los resultados de la tabla Ⅲ.3.

Sensor SW-420		
Nivel del potenciómetro	Respuesta al golpe	Respuesta en reposo
Mínimo	Voltaje lógico = 1	Voltaje lógico = 1
Medio	Voltaje lógico = 1	Voltaje lógico = 0
Alto	Voltaje lógico = 0	Voltaje lógico = 0

Tabla Ⅲ 3. Valores arrojados por el sensor SW-420.

Las pruebas demostraron que la mejor combinación es la del potenciómetro en un nivel medio, ya que garantiza una correcta detección de un golpe.

Es de suma importancia hacer mención de que la sensibilidad es grande en un nivel medio ya que el sensor se activaba de forma rápida. Por otra parte, la sensibilidad en un nivel mínimo o alto hacen que el sensor o no responda o se mantenga con un estado alto todo el tiempo. Dicho lo anterior, al utilizar el sensor en un nivel medio se obtiene una buena respuesta ante una intrusión.

En adición a los puntos anteriores, el escenario para este sensor no requiere de tantos cuidados, únicamente requiere de una superficie estable.

El siguiente sensor que se probó fue el sensor ultrasónico. Para este proyecto este sensor se utiliza solamente para saber si alguien atraviesa una zona no deseada.

Ⅲ.2.3 Configuración del sensor ultrasónico

El sensor ultrasónico sirve principalmente para detectar objetos a cierta distancia. Para calibrarlo se realizó un programa de prueba de tal manera que éste detecte cuando algo o alguien se atraviesa en su línea de vista. Cabe mencionar que para este trabajo se hace uso de la biblioteca *New ping*, compatible con Arduino, ya que facilita su calibración.

Este sensor cuenta con 4 terminales, dos de ellas para su alimentación y otras dos para mandar y recibir una señal ultrasónica que funciona para detectar objetos. Véase figura Ⅲ.7.

Figura Ⅲ 7. Módulo del sensor HC-SR04.

Para realizar las pruebas se utilizó una pared y objetos planos tales como cartón o plástico, ésto simulando un muro. El sensor ultrasónico se fijó a una base que se podía desplazar ya sea hacia adelante o hacia atrás, con la finalidad de que la detección del objeto fuera eficiente al atravesar objetos en su línea de vista, mientras el sensor estaba fijo o en movimiento.

Cabe mencionar que el módulo no cuenta con potenciómetros, como los sensores anteriores, para ser calibrado de forma analógica, ya que toda la configuración se lleva a cabo mediante una serie de instrucciones en su programación en donde se involucra el tiempo y la distancia de respuesta, para poder imprimir con la herramienta monitor serie o algún otro display las lecturas que realiza. Es por lo anterior que se crearon dos programas para comprobar el funcionamiento del sensor ultrasónico.

Programa 1: este programa consiste en una serie de instrucciones, previamente mencionadas en el capítulo 2, para calcular la distancia en centímetros tomando en cuenta el tiempo de viaje de la onda mecánica que produce el sensor y un ciclo *while,* el cual mantiene un led encendido mientras el valor detectado se mantiene en un rango propuesto (**anexo E**).

Programa 2: para este programa fue necesario utilizar una biblioteca de Arduino, la cual tiene por nombre *New ping*. Esta biblioteca simplifica mucho la programación para poder utilizar el sensor ultrasónico. Con ella se creó un programa que hace lo mismo que el programa 1, es decir, utilizando un ciclo *while* se prende un led al cumplir con ciertas condiciones. Cabe aclarar que la condición para este caso es una distancia preestablecida, la misma para ambos (**anexo F**).

En el programa 1, al entrar al ciclo *while,* el led se quedaba encendido y la distancia de detección no cambiaba, incluso al mover el objeto de detección. Al contrario, al utilizar el programa 2, éste funcionó de una forma correcta gracias a la biblioteca *New ping*. Es por esto que las pruebas del sensor se realizaron únicamente utilizando el programa 2. Para realizar las pruebas con el sensor ultrasónico se creó el siguiente circuito. Véase figura Ⅲ.8.

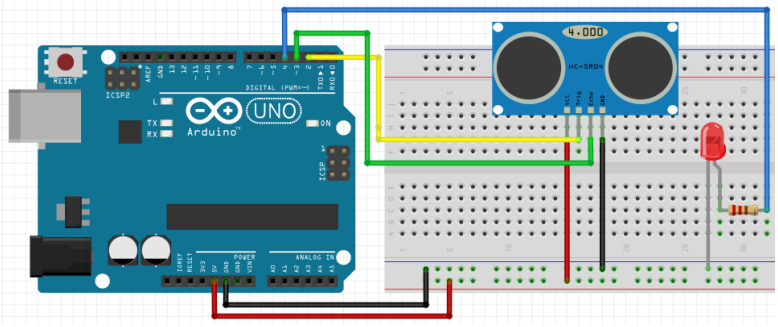


Figura Ⅲ 8. Circuito de prueba del sensor HC-SR04.

Utilizando el programa 2 se realizaron pruebas de distancia, las pruebas consistieron en ir variando la distancia entre el sensor y la pared, esto con la finalidad de comparar distancias reales medidas con una cinta métrica con las que arrojaba el sensor. Los resultados obtenidos se muestran en la tabla Ⅲ.4.

Sensor ultrasónico					
Distancia real	20 cm	40 cm	60 cm	80 cm	1 m
Distancia calculada con el sensor	21 cm	41 cm	62 cm	81 cm	1.02 m

Tabla Ⅲ 4. Valores de distancias reales y calculadas con el sensor ultrasónico.

Los resultados de la tabla Ⅲ.4 muestran que el sensor ultrasónico tiene un rango de error muy pequeño puesto que en su mayoría el error era de 1 cm. Además de las mediciones de la tabla Ⅲ.4 también se calculó un umbral de detección mínimo, casi libre de error, de 12 cm teniendo el sensor en el centro, es decir, teniendo como referencia una línea de 12 cm como marca y el sensor en el centro de ésta.

Por otro lado, el sensor muestra errores considerables si se interrumpe el viaje de la onda mecánica, ya sea tapando el receptor o el emisor del módulo. También

muestra problemas en la medición si el sensor es sacudido. Dicho lo anterior, es de suma importancia instalar este sensor en un lugar fijo con un escenario estático, como por ejemplo un pasillo, ya que de otra manera es muy fácil que marque mediciones erróneas.

Para comprobar lo que sucede si alguien o algo atraviesan la línea de vista del sensor se realizó una segunda prueba.

La siguiente prueba es para comprobar la reacción del sensor ultrasónico al obstruir su línea de vista. Los resultados se muestran en la tabla Ⅲ.5.

Sensor ultrasónico			
Distancia al obstáculo	20 cm	40 cm	60 cm
Respuesta del sensor	entre 21 y 25 cm	entre 37 y 41 cm	entre 61 y 63 cm

Tabla Ⅲ 5. Comparación entre valor real y valor arrojado si algo se atraviesa en la línea de vista del sensor ultrasónico.

La tabla Ⅲ.5 muestra que al interferir un objeto con la línea de vista del sensor, éste muestra una respuesta casi igual a la distancia real. Esto es adecuado ya que los resultados indican que el sensor es útil en un sistema de alarma.

Dicho lo anterior, las recomendaciones para utilizar este sensor son pocas, simplemente se debe de buscar una superficie estable en donde colocar el sensor ultrasónico ya que sacudirlo o tapar parte de su línea de vista hacen que el sensor muestre lecturas analógicas con un error muy grande, que va desde un 0 hasta 3,339. Dicho lo anterior, el sensor se colocará enfrente de la puerta ya que al abrirla la lectura analógica podrá activar la alarma.

Ⅲ.2.4 Sensor de calidad del ambiente

El próximo sensor que se calibrará es el MQ-135. Este sensor al igual que el sensor PIR necesita de un lapso para estabilizarse, para lo cual el sensor requiere de aproximadamente 48 horas. Después de estas 48 horas, el sensor estará en un estado estable con lo que brindará lecturas más precisas, pero si se requiere de su

uso inmediato, el sensor se puede utilizar al momento, simplemente estableciendo un rango en la lectura analógica que interprete el sistema.

Este sensor cuenta con 4 pines de los cuales dos son para alimentación y dos para producir señales, una analógica y otra digital. Una característica de éste es que cuenta con un potenciómetro para hacer uso de la señal digital, es decir, el potenciómetro se puede calibrar para que al tener un nivel de voltaje específico se produzca o deje de producirse una señal digital, pero para este proyecto no es necesario el uso del potenciómetro ya que la respuesta analógica es la que se desea. En la figura Ⅲ.9 se muestra el módulo MQ-135.

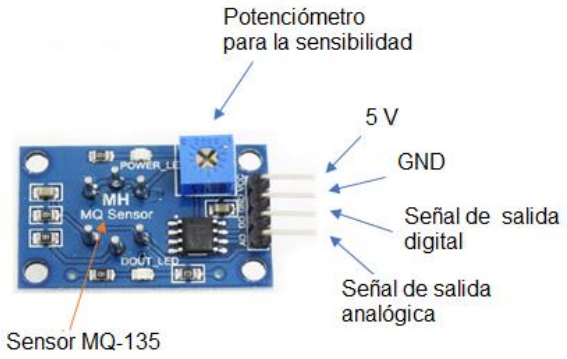


Figura Ⅲ 9. Módulo del sensor MQ-135.

Para este módulo se realizaron dos pruebas, las cuales se apollan de un programa realizado en el IDE de Arduino (**anexo G**), las cuales consisten en exponer al sensor al gas butano y al humo. Con cada uno se establecerá un umbral y con ello, cuando el módulo se utilice en el sistema de seguridad, éste podrá sugerir una posible contaminación del medio ambiente. El circuito de la figura Ⅲ.10 es el que se implementó para poder realizar los experimentos mencionados anteriormente. Los resultados obtenidos de ambas pruebas se muestran en la tabla Ⅲ.6 y Ⅲ.7.

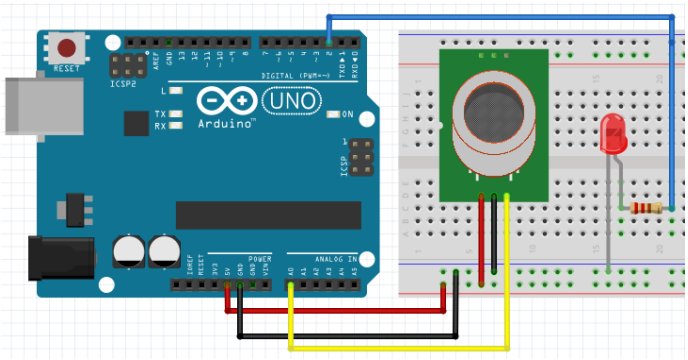


Figura Ⅲ 10. Circuito de prueba del sensor MQ-135.

La tabla Ⅲ.6 muestra las lecturas obtenidas ante la exposición directa de humo. La tabla únicamente muestra las lecturas de la salida analógica. Si se desea saber con exactitud la concentración de gas butano o de humo es necesario realizar cálculos para obtener los valores de las resistencias R0, Rs y RL, además de otras constantes, las cuales dependerán directamente del tipo de gas al que sea sometido el sensor. Todo esto se puede obtener de la hoja de especificaciones del producto [39]. Dicho lo anterior, saber qué tipo de gas se encuentra en el medio ambiente es realmente difícil con un sensor MQ-135, es por ello que únicamente con base en las lecturas analógicas y a las respuestas obtenidas se imprimirá un mensaje de alerta a causa de un ambiente contaminado. En el sistema de alarma, además del mensaje también se activará un ventilador para poder disipar el agente contaminante.

Sensor MQ-135				
Lectura analógica de ambiente limpio	Medida mínima de lectura analógica de ambiente con humo	Medida máxima de lectura analógica de ambiente con humo	Tiempo de respuesta ante la exposición del sensor al humo	Tiempo para estabilizar la lectura del sensor una vez que el humo se disipa
97	150	297	Entre 1 y 2 segundos	Aproximadamente 1 minuto máximo dependiendo de la penetración del humo en el sensor

Tabla Ⅲ 6. Lecturas del CAD generadas por el sensor ante la exposición de humo.

En la tabla Ⅲ.6 se muestran los campos pertenecientes al experimento de exposición al humo, se tiene una lectura con el ambiente limpio y las mediciones mínimas y máximas obtenidas al exponer el sensor al humo, además el tiempo de respuesta a esta exposición, así como el tiempo que tarda en estabilizarse nuevamente el sensor una vez disipado el humo. La siguiente prueba es para determinar un umbral para detectar una fuga de gas. Para llevar a cabo esta prueba se utilizó un encendedor para liberar cantidades concentradas de gas butano a una distancia no mayor a 2 centímetros del sensor, posteriormente también se alejó un poco el encendedor para comprobar que el sensor siguiera trabajando si la fuga de gas no se daba en cantidades cuya concentración es menor. Los resultados se muestran en la tabla Ⅲ.7.

Sensor MQ-135 prueba de una exposición de gas butano a 2 cm aproximadamente				
Lectura analógica de ambiente limpio	Medida mínima de lectura analógica del ambiente con gas butano	Medida máxima de la lectura analógica del ambiente con gas butano	Tiempo de respuesta ante la exposición del sensor al gas butano	Tiempo para estabilizar la lectura del sensor una vez que el gas butano se disipa
105	342	927	Entre 1 y 2 segundos	Aproximadamente 5 minutos máximo dependiendo de la penetración del gas en el sensor
Sensor MQ-135 prueba de una exposición de gas butano a 10 cm aproximadamente				
Lectura analógica de ambiente limpio	Medida mínima de lectura analógica del ambiente con gas butano	Medida máxima de la lectura analógica del ambiente con gas butano	Tiempo de respuesta ante la exposición del sensor al gas butano	Tiempo para estabilizar la lectura del sensor una vez que el gas butano se disipa
108	312	398	Entre 4 y 6 segundos	Aproximadamente 1 minuto

Tabla Ⅲ 7. Lecturas analógicas obtenidas ante la exposición del sensor a gas butano a una distancia aproximada de 2 cm y 10 cm.

La tabla Ⅲ.7 indica los resultados cuando el gas presenta una fuga y el sensor lo detecta a 2 cm de distancia del sensor y cuando el gas se encuentra un poco más disperso en el medio ambiente, a 10 cm de éste. Asimismo, se aprecia una lectura analógica mínima y máxima que supera la que se produjo con la exposición al humo, y por último el tiempo de respuesta para llegar a dichos valores y el tiempo que le toma al sensor bajar su nivel cuando el gas se ha disipado.

En adición a los puntos anteriores, los resultados de estas pruebas indican que el sensor da una respuesta rápida ante la presencia de humo y gas butano. De igual manera se comprobó que el potenciómetro solamente es útil si se desea hacer uso de la salida digital.

Por otro lado, la eficiencia de este sensor radica en su ubicación dentro de la vivienda. Es recomendable instalar el sensor un poco retirado del suministro del gas, además de aislar bien las conexiones eléctricas de este, ya que la exposición al gas puede provocar un chispazo y con ello una explosión. Otro punto a considerar es el tiempo que le toma al sensor estabilizarse una vez disipado el gas o humo, aunque usualmente las lecturas analógicas iniciales son de 100 a 120, es por ello que no es necesario esperar horas a que se estabilice, basta con darle la instrucción al controlador para que imprima un mensaje y active el ventilador cuando se exceda un rango considerable, para este trabajo la lectura a superar es de 170.

Tomando en cuenta los puntos anteriores la única recomendación que se puede dar es que el sensor no se instale cerca de lugares en donde se concentren sustancias tales como desodorantes o insecticidas.

Llegando a este punto, las pruebas indican que todos estos sensores de los que se ha venido hablando a lo largo de este documento pueden ser utilizados para el desarrollo de este proyecto. Es por ello que la siguiente fase es realizar pruebas de conexión entre el servidor Arduino ethernet y la base de datos en diferentes sistemas operativos. Una vez realizado esto se proseguirá a mostrar los resultados de las pruebas del sistema completo ya montado en una maqueta.

Ⅲ.3 Pruebas de comunicación con el servidor del sistema de alarma

En el siguiente apartado se muestran las pruebas que fueron realizadas para conectar el servidor Arduino ethernet a la base de datos MySQL alojada en diversos sistemas operativos como son: GNU/Linux, Windows 7, Windows 10 y Debian. Al ser MySQL y Apache multiplataforma, todas estas pruebas deben demostrar que el sistema es multiplataforma.

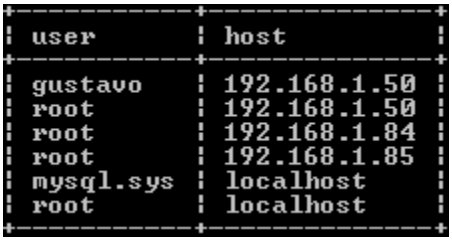
La prueba consiste en comunicar el servidor Arduino ethernet con la base de datos en los diferentes sistemas operativos y con ello mostrar un mensaje que certifique la conexión. Dicho lo anterior, el primer sistema operativo con el que se llevará a cabo la prueba es Windows 7, en donde está instalada la base de datos.

Ⅲ.3.1 Comunicación utilizando MySQL Community Server 5.7

La instalación de la base de datos en Windows 7 solamente requiere del archivo MySQL Community Server 5.7, que puede descargarse de la página de MySQL. Cabe mencionar que el servidor Arduino ethernet solamente es compatible con la versión MySQL Community Server 5.7 ya que una versión posterior, al intentar hacer la comunicación del servidor Arduino ethernet con la base de datos, mostrará un mensaje en el cual se avisa que se requiere actualizar el servidor Arduino ethernet.

La configuración de la base de datos en Windows 7 solamente requiere aceptar términos y condiciones, ya que por defecto viene la configuración que permite comunicar el servidor y la base de datos. Esta configuración incluye: acceso por el puerto 3306 de TCP, creación de una contraseña root, superusuario con privilegios de administrador, y la inclusión de todos los complementos necesarios para que la base de datos funcione correctamente.

Una vez instalada la base de datos, se debe de crear un usuario con su contraseña, además se le deben de dar todos los permisos para que éste pueda realizar los cambios correspondientes en las tablas, como, por ejemplo, escribir y borrar en una tabla. El usuario debe tener la dirección IP del servidor Arduino ethernet ya que desde él se realizará la conexión. Todo esto puede realizarse usando instrucciones CMD desde el intérprete de comandos Windows. En la figura Ⅲ.11 se muestran los usuarios existentes, además del usuario Gustavo con el host con dirección IP 192.168.1.50.



user	host
gustavo	192.168.1.50
root	192.168.1.50
root	192.168.1.84
root	192.168.1.85
mysql.sys	localhost
root	localhost

Figura Ⅲ 11. Usuarios en la base de datos MySQL instalada en una computadora con sistema operativo Windows 7.

En este punto, de acuerdo a mi experiencia, considero que es mejor crear un usuario distinto al usuario root para trabajar de manera más fácil con la base de datos, la razón es que el usuario root por defecto ya viene instalado y configurado en el servidor.

Con el fin de lograr la comunicación entre el servidor y la base de datos, se debe de crear un programa de prueba en el IDE de Arduino y posteriormente cargarlo al microcontrolador de la tarjeta Arduino ethernet, esto siguiendo el manual de la biblioteca *MySQL connectors*. En dicho manual se indica cómo entablar la comunicación entre Arduino ethernet y la base de datos. Asimismo, el manual de dicha biblioteca también indica cómo guardar información en las tablas que se crean en la base de datos MySQL, esto último se mostrará más adelante.

En el programa de prueba se debe de incluir la configuración adecuada, la cual incluye direcciones IP, DNS y Gateway (DNS y Gateway solamente si la conexión es WAN y no LAN), nombre de usuario, contraseña y activar mensajes de notificación para saber si se ha logrado entablar una conexión. Cabe aclarar que si se requieren ver mensajes enviados por la tarjeta Arduino ethernet se puede utilizar la herramienta monitor serie del IDE de Arduino o un display LCD que puede conectarse a la tarjeta. Dicho lo anterior, para saber la dirección IP de la base de datos solamente se hace uso del comando *ipconfig* en el host de la base de datos, dicho comando arrojará información de los adaptadores de red. Cabe mencionar que para lograr esta conexión se requieren de varias condiciones, las cuales son:

- Los hosts deben estar en la misma red.
- La dirección IP del servidor Arduino ethernet debe ser única y exclusiva para el servidor ya que causaría conflicto si otro host ya tiene dicha dirección IP asignada. Para saber si la IP que se utiliza está siendo ocupada se recomienda probar dos opciones, las cuales son: utilizar la aplicación Nmap para hacer un escaneo de la red e identificar los host que están siendo utilizados, la otra opción es simplemente, por medio del comando ping, hacer un ping indicando la dirección IP de interés para comprobar si ésta es accesible, sino hay respuesta quiere decir que o hay un problema con ese host o simplemente no está siendo

utilizado. La dirección IP que se asignó al Arduino ethernet es la misma que se muestra en la figura Ⅲ.11.

Una vez que la base de datos y el servidor Arduino ethernet fueron configurados y agregados en la misma red LAN, se prosiguió a realizar una prueba de conexión. La figura Ⅲ.12 muestra la salida de la herramienta monitor serie con un mensaje en el cual se indica que se logró la conexión, además de indicar el servidor al cual se conectó, incluyendo versión.

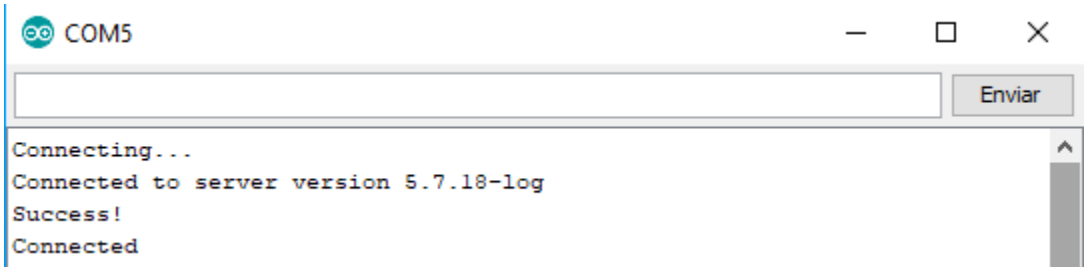


Figura Ⅲ 12. Mensaje de éxito en la prueba de conexión entre el servidor Arduino ethernet y el host con sistema operativo Windows 7.

Cabe mencionar que la aplicación MySQL Community Server 5.7 no es la misma que se usa en la infraestructura LAMP ya que ésta solamente provee la base de datos MySQL y no cuenta con el administrador de base de datos PhpMyAdmin ni con el servidor Apache que LAMP sí provee.

Ⅲ.3.2 Comunicación utilizando una infraestructura XAMP

La siguiente prueba de comunicación entre el servidor y la base de datos se realizó utilizando un host con Windows 10 y la infraestructura XAMP, la cual es casi lo mismo que la LAMP, pero con la diferencia en el sistema operativo. De la misma manera que en Windows 7, en Windows 10 solamente se necesita descargar un ejecutable que instalará esta infraestructura, el ejecutable lleva por nombre XAMPP, que se puede descargar de la página apachefiends.org. XAMPP es un sistema multiplataforma, es decir, se puede instalar en sistemas tales como son: Windows, GNU/Linux y Macintosh. La principal diferencia entre XAMPP y LAMP es que la primera es únicamente un ejecutable que se configura de una manera muy fácil e

incluye todas las herramientas, a diferencia de LAMP, la cual es una infraestructura que se descarga parte por parte, es decir, se debe descargar Apache server en un sistema GNU/Linux, la base de datos MySQL o MariaDB y el lenguaje de programación requerido (PHP o Pearl).

La configuración del puerto, usuario y contraseña es la misma mencionada para MySQL Community Server 5.7, es decir, no se debe alterar la configuración, ya que ésta brinda la facilidad para comunicar el servidor con el administrador de la base de datos.

Además de contar con la base de datos, XAMPP instalará Apache, PHP y el administrador PhpMyAdmin para la base de datos. Para esta prueba se puede utilizar el administrador PhpMyAdmin mediante un navegador y con ello manipular y crear una base de datos, en lugar de acceder por medio del intérprete de comandos cmd. Además de lo anterior, XAMPP cuenta con un panel de control el cual sirve para activar y desactivar Apache y MySQL, además del resto de las herramientas instaladas con la aplicación.

Cabe aclarar que para acceder al administrador PhpMyAdmin se necesita escribir en el navegador la dirección IP del host en donde se haya instalado XAMPP, de igual manera, para saber la IP que se requiere, solamente se tiene que escribir en el intérprete de comandos cmd del host el comando *ipconfig*. Teniendo la dirección IP del host se prosigue a poner el nombre del administrador en el navegador de la siguiente manera “dirección_IP/PhpMyAdmin”.

Una vez que se accede al administrador PhpMyAdmin se prosigue a crear un usuario con su respectiva contraseña e indicar el host del cual se conectará, además de otorgarle todos los permisos de administrador. De la misma manera que con el servidor MySQL Community Server 5.7, el usuario creado para esta prueba se llama arduino y se le brindaron todos los permisos de escritura para la base de datos. Además de lo anterior, también se realizó una prueba más para verificar si en efecto el usuario podría conectarse desde diferentes hosts: desde el host con la dirección IP del servidor, desde cualquier host o desde un host local. Véase figura Ⅲ.13. Para

ello se crearon tres usuarios de diferente nombre, pero con el mismo nombre del servidor.

Vista global de las cuentas de usuario

	Nombre de usuario	Nombre del servidor	Contraseña	Privilegios globales	Conceder	Acción
☐	arduino	%	Sí	ALL PRIVILEGES	Sí	Exportar
☐	arduino	192.168.1.50	Sí	ALL PRIVILEGES	Sí	Exportar
☐	arduino	localhost	Sí	ALL PRIVILEGES	Sí	Exportar
☐	pma	localhost	No	USAGE	No	Exportar
☐	root	127.0.0.1	No	ALL PRIVILEGES	Sí	Exportar
☐	root	::1	No	ALL PRIVILEGES	Sí	Exportar
☐	root	localhost	No	ALL PRIVILEGES	Sí	Exportar

Figura Ⅲ 13. Usuario arduino creado mediante PhpMyAdmin, así como sus privilegios y el nombre del servidor desde el cual se conectará.

Continuando con la prueba, se modificó el programa de prueba de Arduino para comprobar si en efecto se puede entablar una conexión con la base de datos en cada caso. En el programa de prueba únicamente se cambian la dirección IP de la base de datos, usuario y contraseña. Los resultados de la comunicación se muestran en la siguiente figura. Véase figura Ⅲ.14.

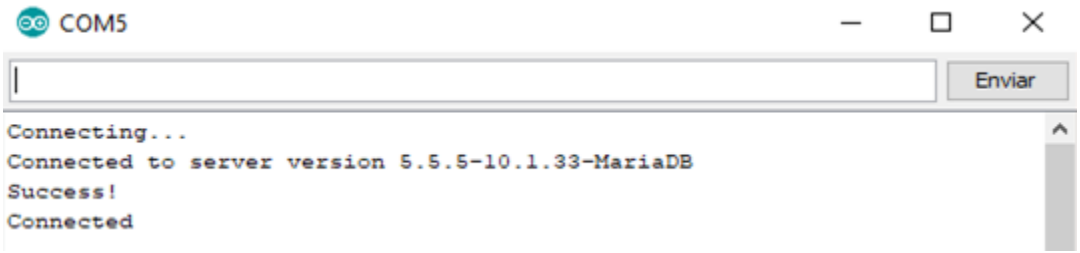


Figura Ⅲ 14. Mensaje de éxito en la prueba de conexión entre el servidor Arduino ethernet y el host con sistema operativo Windows 10.

Al verificar la respuesta al intentar entablar comunicación con la base de datos desde: el host local, cualquier host, y un host con la IP asignada, el único inconveniente se presenta al querer conectarse a la computadora en donde se instaló la infraestructura XAMP, la razón es que el host local no busca conectarse a esta computadora, puesto que el host local es el mismo que el host en donde se

instaló la infraestructura XAMP, la consecuencia es que el host local en ocasiones no se conecta.

Este mensaje se muestra únicamente con dos hosts del usuario arduino, el que tiene asignado la IP del servidor y el que indica que cualquier host puede entablar conexión, al parecer solamente no se conecta desde el servidor local ya que este pertenece a la misma maquina en donde está instalada la infraestructura XAMP, y esto es lógico ya que este host no busca conectarse.

Ⅲ.3.3 Comunicación utilizando LAMP en Centos 7 minimal

Continuando con las pruebas, para llevar a cabo la tercera prueba de comunicación se utilizó un host con un sistema operativo Centos 7 minimal. A este host se le instaló toda la infraestructura LAMP, además del administrador de base de datos PhpMyAdmin. Como Centos es un sistema mínimo, solamente cuenta con un intérprete de comandos, y por medio de éste se instaló toda la infraestructura, el administrador de base de datos, y se configuró para poder realizar una conexión exitosa.

Como se ha hecho mención, este sistema no cuenta con la facilidad de contar con un archivo ejecutable para realizar la instalación de LAMP, por lo que es necesario tener conocimientos acerca de diferentes comandos, ya sea para instalar o configurar alguna aplicación usando un intérprete de comandos.

Al igual que en la prueba de conexión con XAMPP también se crearon tres usuarios, los cuales llevan por nombre arduino01, dichos usuarios cuentan con tres hosts, los cuales son iguales que en la prueba de la infraestructura XAMPP. Véase figura Ⅲ .15.

Figura Ⅲ 15. Usuario arduino01 creado mediante PhpMyAdmin, así como sus privilegios y el nombre de servidor desde el cual se conectará.

Al realizar la prueba de conexión se obtuvieron los mismos resultados que en el caso realizado con la infraestructura XAMPP. Antes de continuar, es de suma importancia mencionar que con Centos 7 es necesario permitir el acceso a los puertos 80, 443 y el 3306, ya que esto no se hace en automático en una infraestructura LAMP. Una forma fácil, pero no la más recomendable, para permitir este acceso es desactivar el cortafuegos del sistema.

Por otro lado, si se busca visualizar el administrador PhpMyAdmin simplemente se deben de dar los permisos necesarios en la configuración de éste, de esta manera se podrá acceder desde un equipo remoto. Esto último solamente si se quiere utilizar el administrador de la base de datos ya que es posible conectarse simplemente teniendo la base de datos MySQL sin la necesidad de tener instalada la aplicación PhpMyAdmin. Dicho lo anterior. Para comprobar que se puede conectar a la base de datos con y sin el administrador PhpMyAdmin se realizaron dos pruebas de conexión.

La prueba sin el administrador PhpMyAdmin arrojó resultados positivos ya que se logró la conexión, cuyo mensaje de salida muestra la versión del servidor. Véase figura Ⅲ.16.

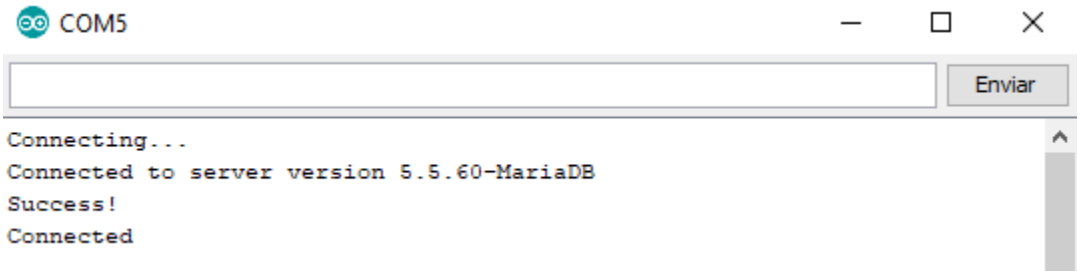


Figura Ⅲ 16. Mensaje de éxito en la prueba de conexión entre el servidor Arduino ethernet y el host con sistema operativo Centos 7 sin administrador PhpMyAdmin.

La siguiente figura muestra la prueba de conexión a la base de datos en donde sí se tenía instalado PhpMyAdmin como administrador, demostrando que PhpMyAdmin es opcional para utilizar una base de datos MySQL, más adelante se mostrará como también es opcional para crear y guardar información en una base de datos. Véase figura Ⅲ.17.

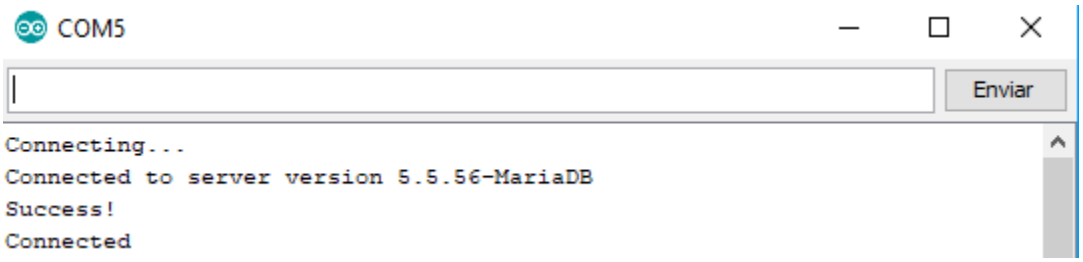


Figura Ⅲ 17. Mensaje de éxito en la prueba de conexión entre el servidor Arduino ethernet y el host con sistema operativo Centos 7 con administrador PhpMyAdmin.

Como se acaba de demostrar no es necesario utilizar el administrador, aunque es útil si el usuario final no tiene manejo en bases de datos MySQL ya que con PhpMyAdmin podrá acceder de forma fácil y eficiente a su información. A continuación, se muestra la última prueba, la cual consiste en establecer la conexión entre el servidor Arduino ethernet y la base de datos MySQL, pero esta vez utilizando una tarjeta Galileo.

Ⅲ.3.4 Comunicación utilizando LAMP en Debian

La siguiente prueba consiste en verificar la comunicación entre el servidor Arduino ethernet y la base de datos MySQL instala en la tarjeta Galileo la cual tiene un sistema operativo Debian. Cabe mencionar que, en los casos anteriores, la base de datos fue instalada en computadoras de escritorio.

Esta prueba es la de mayor interés ya que en la tarjeta Galileo es donde se guardará toda la información de los sensores, por ello, la base de datos debe interactuar de manera eficiente con el servidor Arduino ethernet.

Al igual que en las pruebas anteriores, al sistema operativo Debian se le instaló la infraestructura LAMP, además del administrador de base de datos PhpMyAdmin. De igual manera, se creó un usuario con la finalidad de probar la comunicación entre el servidor y la base de datos. El usuario creado lleva por nombre gustavo y puede conectarse desde la IP del servidor Arduino ethernet, desde el host local y desde cualquier servidor. Véase figura Ⅲ.18.

Figura Ⅲ.18. Usuario gustavo creado mediante PhpMyAdmin, así como sus privilegios y el nombre de servidor desde el cual se conectará.

Los resultados de estas pruebas arrojaron los mismos que en todas las pruebas anteriores. Es decir, la comunicación desde el servidor con IP 192.168.1.50 y de cualquier host se llevan a cabo sin problemas, la conexión desde el localhost por otra parte, no se logra establecer o tarda en establecerse, esto por los mismos motivos que ya se han explicado. Véase figura Ⅲ.19.

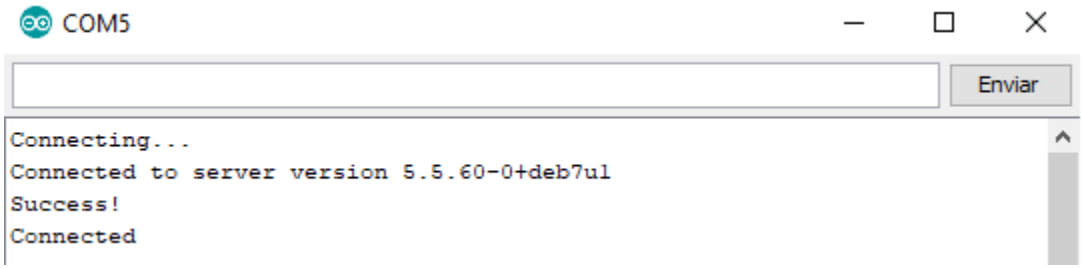


Figura Ⅲ 19. Mensaje de éxito en la prueba de conexión entre el servidor Arduino ethernet y el host con sistema operativo Debian.

Para lograr que la prueba de comunicación entre el servidor Arduino ethernet y el host con sistema operativo Debian sea un éxito, se debe de modificar un archivo llamado *my.cnf*, la ruta del directorio es etc/MySQL. Utilizando algún editor de texto (nano para este proyecto) se debe de cambiar en el archivo *my.cnf* el campo **bind-address = 127.0.0.1** por **bind-address =192.168.1.162**, esto para este proyecto, pero en general se debe de cambiar por la dirección IP del host Debian. La razón para realizar este cambio en las direcciones IP es porque en sistemas operativos Debian y Ubuntu la configuración de MySQL está abierta únicamente para la interfaz localhost, cambiando la dirección IP localhost por la del host Debian se puede lograr entablar una conexión remota.

La finalidad de realizar estas pruebas es demostrar la viabilidad del proyecto ya que, como se demostró, no solamente se puede utilizar con el software y el hardware inicialmente planteado, sino que también es posible usar una infraestructura LAMP, XAMP o solamente el servidor MySQL. Además, utilizar una tarjeta Galileo es eficiente dado que se trata de una tarjeta discreta que fácilmente puede esconderse en la vivienda.

La siguiente etapa tiene la finalidad de realizar pruebas para guardar la información proporcionada por los sensores utilizados. Es importante mencionar que en el resto de las pruebas se usará la base de datos instalada en Debian en la tarjeta Galileo, esto puesto que ya se ha comprobado que el sistema es multiplataforma.

Ⅲ.4 Guardado de la información de los sensores en la base de datos

En esta sección se muestra cómo guardar información en la base de datos directamente desde el servidor Arduino ethernet utilizando como guia el manual de la biblioteca *MySQL Conector*.

Ⅲ.4.1 Conexión y configuración a la base de datos

Anteriormente se explicó cómo es posible conectar el servidor a la base de datos MySQL, ahora haciendo uso de la base de datos, ya sea accediendo a ella directamente o por medio del administrador PhpMyAdmin, se creará una base de datos con una tabla con la finalidad de guardar un mensaje desde el servidor Arduino ethernet. A continuación, se describe el procedimiento para ello.

Para poder utilizar la base de datos de la tarjeta Galileo es necesario acceder a ella de forma remota, existen varias maneras, pero en este proyecto, por fines prácticos, se utilizará una máquina virtual con sistema operativo Centos 7, cuya conexión remota a la tarjeta Galileo será por medio del protocolo SSH (Secure Shell). La máquina virtual se utilizó puesto que no se contaba con una maquina real con sistema operativo GNU/Linux instalado.

Con el protocolo SSH es posible conectarse a un host de manera remota de forma segura y así acceder a su información únicamente conociendo el usuario y la contraseña del host al cual se va a conectar, dicho protocolo facilita la comunicación utilizando una arquitectura cliente servidor, el comando utilizado es **ssh root@Dir_IP** para este proyecto, pero en general se debe de utilizar el comando de la siguiente manera **ssh usuario@Dir_IP** [40]**.**

Si no se conoce el host, únicamente es necesario hacer uso de la aplicación Nmap. Anteriormente ya se ha hecho mención de esta aplicación, hablando un poco más a detalle, Nmap es una herramienta con la cual podemos escanear redes y saber qué host están conectados a la red LAN, en adición Nmap puede mostrar información tal como las direcciones IP, el nombre del host, la dirección MAC, los puertos que se están utilizando y la topología. Por otro lado, Nmap es una aplicación

multiplataforma. A continuación, se muestra cómo se utiliza Nmap para saber el host dónde se aloja la base de datos.

Utilizando el comando **Nmap –sP dirección_IP/mascara_de_subred** (para esta prueba la dirección utilizada fue **192.168.1.162/24**), Nmap realiza un escaneo de ping, es decir, escanea cada dirección IP de la LAN que contenga los mismos 24 bits, esto con ayuda del comando **sP**. Al realizar el escaneo Nmap, arroja como resultado la dirección IP asociándola a una dirección MAC, así como el tiempo que le toma realizar el escaneo y el número de host en la red LAN. Véase figura Ⅲ.20.

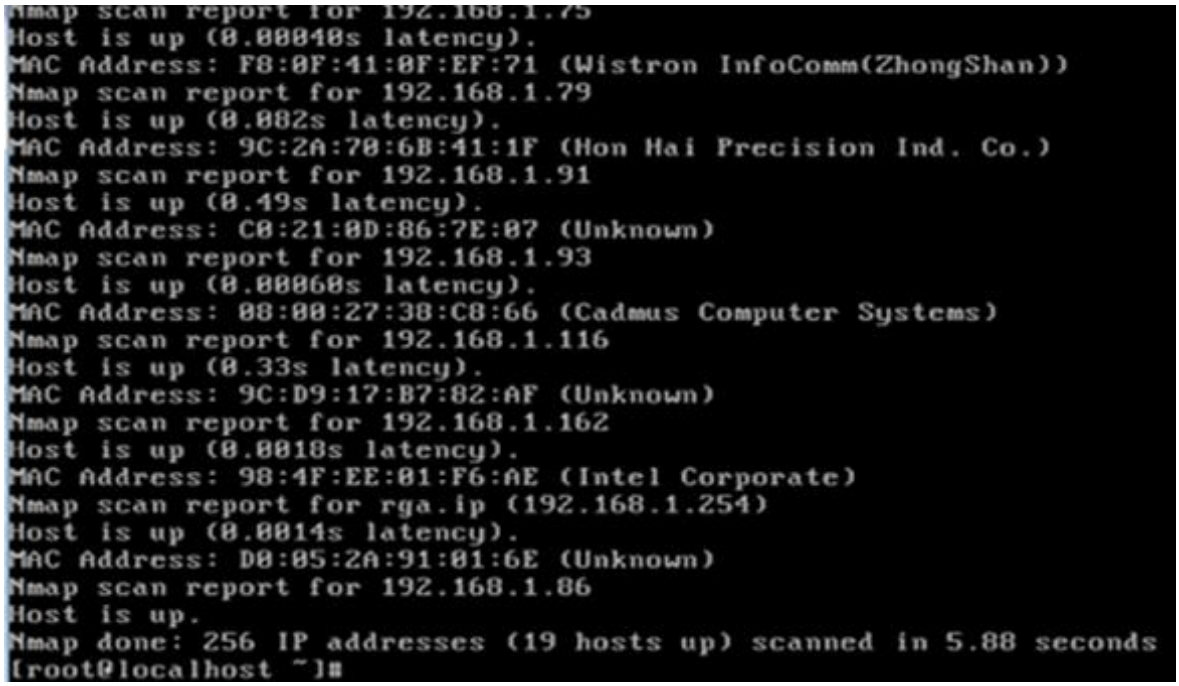


Figura Ⅲ 20. Mensajes de salida al ejecutar la instrucción nmap -sP dirección_IP/mascara_de_subred.

La figura Ⅲ.20 muestra el escaneo realizado por medio del comando previamente. Los resultados del escaneo muestran la dirección MAC de la tarjeta Galileo la cual es 98:4F:EE:01:F6:AE, esta dirección está asociada al host **Intel Corporate** y a la dirección IP 192.168.1.162. Todos los datos recopilados de dicho escaneo muestran únicamente los dispositivos conectados a la red LAN y dan la información necesaria para realizar la conexión remota.

En la figura Ⅲ.21 se muestra el uso del protocolo de comunicación SSH y cómo se logra entablar la conexión a la tarjeta Galileo.

Figura Ⅲ 21. Conexión a la tarjeta galileo por medio del protocolo SSH.

Una vez que se tiene acceso a la base de datos instalada en el sistema operativo Debian de la tarjeta Galileo, se creó una tabla con el nombre Prueba_de_guardado_de_informacion, y en ella se creó la tabla Mensaje1. Posteriormente se creó un programa de prueba en el IDE de Arduino con el cual el servidor Arduino pudiera comunicarse y grabar un mensaje de forma automática. Véase figura Ⅲ.22.

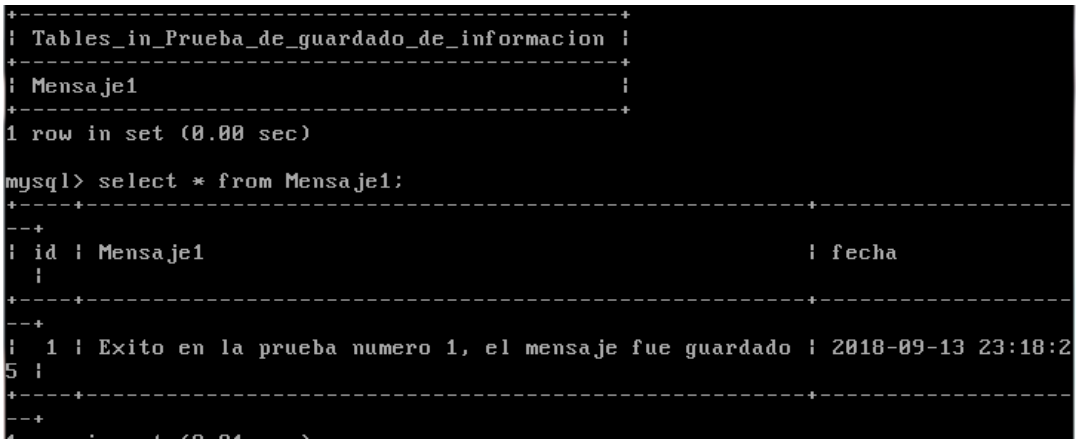


Figura Ⅲ 22. Base de datos y tabla creada para guardar el mensaje: El mensaje ha sido guardado con éxito.

En la figura Ⅲ.22 se muestra el nombre de la base de datos, la cual fue nombrada Prueba_de_guardado_de_informacion, además de la tabla creada llamada Mensaje1, asimismo, al utilizar el comando *select * from* se puede visualizar el contenido de la tabla. El contenido de la tabla Mensaje1 muestra el id, el cual es un

campo que se va incrementando de manera automática, este tipo de mensaje es una variable de tipo entero, todo esto para poder llevar un conteo de las veces que un mensaje se ha insertado en la tabla. El segundo campo es un mensaje de tipo char con una longitud de 100, es decir, es un mensaje de texto el cual puede contener hasta 100 caracteres. Por último, se muestra una variable llamada timestamp, con la cual se puede llevar un registro exacto del momento en el que se inserta un mensaje en la tabla Mensaje1, es decir, con timestamp se muestra fecha y hora del evento.

Con base en toda la información recopilada hasta este punto, se ha demostrado cómo se deben de utilizar los sensores, se ha comprobado la conexión del servidor a la base de datos desde cualquier host y se ha explicado cómo guardar información en una base de datos MySQL.

En la siguiente sección se describe la implementación de la red de sensores apoyándose en los diagramas de flujo presentados en el capítulo 2. A continuación, se muestran las pruebas y resultados al instalar el sistema de alarma en una pequeña maqueta.

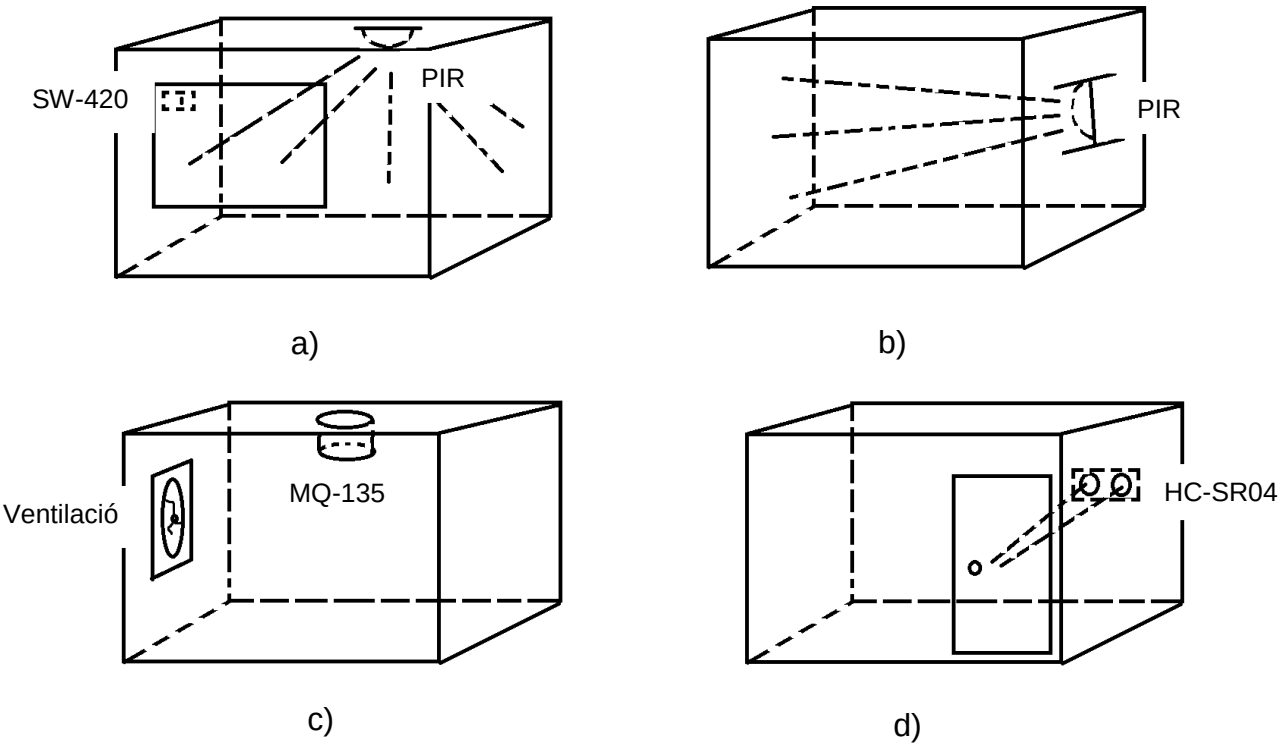
Ⅲ.5 Implementación del sistema de alarma

Una vez que todos los sensores han sido probados para un uso óptimo y además se ha logrado entablar la comunicación entre el servidor Arduino ethernet y la base de datos, instalada en una tarjeta Galileo, el siguiente paso es crear una maqueta en donde se pueda demostrar el funcionamiento del sistema de alarma.

Ⅲ.5.1 Maqueta demostrativa

Crear una maqueta tiene como finalidad visualizar y comprobar cómo es que todos los dispositivos electrónicos pueden trabajar en conjunto de forma eficiente.

La maqueta consta de 4 habitaciones: cocina, cuarto principal y sala, además de la entrada principal. Véase figura Ⅲ.23.



e)

Figura Ⅲ 23. La ubicación de las habitaciones en la maqueta es la siguiente: a) sala, b) cuarto principal, c) cocina, d) entrada, e) maqueta empleada para la demostración del sistema.

Como ya se ha mencionado cada sensor tiene una finalidad. A continuación, se habla de las habitaciones y la función de cada sensor en ellas.

Para la cocina se utilizó el sensor MQ-135 y un ventilador que disipa cualquier contaminante en el ambiente, similar a los ventiladores en las cocinas industriales. El ventilador solamente se probó con humo y gas butano, pero en general el sensor responde a cualquier contaminante en el medio, incluso a las partículas que deja el uso del desodorante en aerosol. Todos los experimentos previos comprobaron un buen funcionamiento al interactuar con humo y gas butano.

El cuarto principal se utilizó para comprobar el funcionamiento del sensor PIR. Las pruebas, algunas realizadas para su configuración, se realizaron con obstáculos transparentes y juguetes que atraviesan el escenario y una combinación de ambas.

Un experimento al utilizar el sensor PIR consistió en lo siguiente: poniendo una pequeña pecera delante del sensor y realizando movimientos con muñecos y con la mano, todo esto dentro de la pecera, el sensor no detecta movimiento alguno. Esto indica que este sensor no detecta movimiento a través de una superficie transparente, ya que el sensor detecta solamente lo que tenga movimiento en su línea de vista, es por ello que las recomendaciones de no utilizar nada que pueda moverse por sí solo (plantas, cortinas, etc.) deben de seguirse al pie de la letra ya que los sensores se podrían activar.

De igual forma, la sala cuenta con un sensor PIR, además la sala cuenta con una pequeña ventana que registra si alguien la intenta romper, esto con ayuda del módulo SW-420. Colocar estos dos sensores en una sola habitación tiene como finalidad lograr un sistema con redundancia, es decir, si se intenta forzar la ventana la alarma se encenderá, pero solamente cuando la ventana reciba un impacto. Por otra parte, si el intruso logra entrar a la sala, un sensor PIR detectará esto de inmediato y activará la alarma.

La puerta al igual que la ventana es de gran interés, ya que es uno de los accesos inmediatos a la vivienda. El sensor ultrasónico tiene como finalidad custodiar la actividad en la puerta, esto lo realiza encendiendo la alarma si alguien la atraviesa. La forma en que el sensor ultrasónico realiza lo anterior es comprobando la distancia de su ubicación y la puerta, tiene que mantenerse siempre la misma distancia. Si la

puerta se abre se produce un cambio en la distancia, lo cual indica que alguien ingresó por ella y con ello activará la alarma.

La ubicación de cada uno de los sensores empleados en cada habitación se muestra en la figura Ⅲ.23.

Retomando el diagrama de flujo de la figura Ⅱ.17, 18 y 19 el sistema de alarma debe de grabar cada suceso en una base de datos. Dicho esto, se crearon tablas en una base de datos llamada *sistema*.

Ⅲ.5.2 TABLAS DINÁMICAS

Las tablas dinámicas en MySQL son de gran ayuda ya que se pueden llenar de forma automática apoyándose de programación en PHP, un ejemplo de esto son los formularios que solemos llenar al registrarnos en algún servicio en línea.

En este caso, los sensores al ser activados guardan un registro. En la base de datos *sistema* se crearon tablas, las tablas guardan un registro de forma dinámica, es decir, se actualizan de forma automática cada que un evento tiene lugar.

Todas las tablas que se crearon cuentan con los mismos campos, los cuales son: id, mensaje del evento y fecha del evento. La diferencia entre cada una de las tablas es el mensaje que se registra al momento de suscitarse un evento. Cada mensaje se programó con ayuda del IDE de Arduino para que el microcontrolador pueda enviarlo a la base de datos cuando uno o más sensores sean activados. Antes de continuar, el manejo de los comandos SQL es esencial para acceder a las tablas de las diferentes bases de datos. Un comando básico que cualquier administrador debería conocer es el comando ***show,*** acompañado de *databases* o *tables*, con ellos el administrador visualiza las bases de datos y las tablas que el servidor MySQL tenga creadas, de igual forma se puede visualizar el contenido de las tablas creadas, es decir, se puede ir un nivel más abajo y ver los registros guardados. Asimismo, para seleccionar una base de datos se debe utilizar el comando ***use*** *nombre_de la base_de_datos,* véase figura Ⅲ.24. En esta figura se pueden apreciar las bases de datos y las tablas que se crearon para la realización de este trabajo.

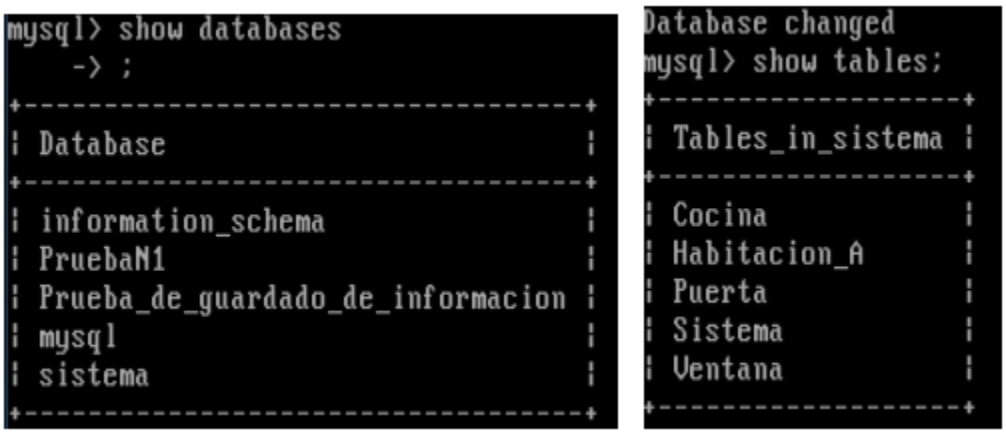


Figura Ⅲ 24. Bases de datos y tablas en el servidor MySQL.

Para poder visualizar todos los registros en una base de datos se debe de utilizar el comando ***select * from*** *nombre_de_la_tabla,* con este comando se puede acceder a cualquier tabla de la lista. En las tablas se encontrará toda la información del evento capturado por la red de sensores, la información es la que se comenta a continuación.

Las tablas guardarán un ID, el cual es un número de identificador, este irá incrementándose con cada registro, también guardarán un mensaje y una fecha con el día, mes, año y hora en el que se guardó la información. Aquí es necesario hacer un pequeño paréntesis, puesto que la hora y la fecha se guardan de forma automática, es primordial tener el reloj de la tarjeta Galileo configurado de forma correcta, para hacerlo solo se debe de configurar el reloj en Debian. Por ejemplo, en las pruebas realizadas a la cocina se obtuvieron las siguientes muestras, véase figura Ⅱ.25.

Figura Ⅲ 25. Registros con el año equivocado.

Como se puede apreciar en la figura Ⅱ.25, el año ni siquiera es el correcto ya que todas las pruebas se realizaron en 2018. Este tipo de problemas no se presentaron en los demás servidores instalados en una PC ya que éstos pueden mantener la hora gracias a una batería, la cual energiza un Real Time Clock (RTC), aun cuando están apagados, pero el sistema Debian de la tarjeta Galileo no lo puede hacer porque no cuenta con un RTC. Es por ello que la configuración de la hora se debe de llevar a cabo cada que el sistema se inicia.

El método para configurar el reloj en Debian por medio de un intérprete de comando es el siguiente:

howclock –set “Año-Mes-Día Hora:Minutos”

Para configurar el reloj del BIOS[18] se utiliza el siguiente comando:

howclock --set --date=” Año-Mes-Día Hora:Minutos”

Por último, para visualizar la hora solamente se debe de utilizar el comando *date.* Una vez que la hora está bien configurada la siguiente prueba es realizar la conexión entre el servidor y la base de datos, para tomar una muestra de las actividades de los sensores y que la base de datos muestre la hora exacta del evento.

Cabe mencionar que aunque el servidor Apache y la base de datos se inician junto con el sistema completo, gracias al comando *sistemctl start*, al intentar realizar la conexión a la tarjeta Galileo, aunque el sistema indicaba el estatus “**running**”, la conexión no se realizaba en algunas ocasiones. Esto solamente ocurrió en el host Debian. En caso de que esto ocurra se debe de reiniciar la base de datos. El error que aparece es el número 2002, el cual indica que no se puede acceder a MySQL, una posible causa es que se haya corrompido o que no se haya cerrado bien en la última sesión.

Otro de los problemas al utilizar varios sistemas operativos con la base de datos MySQL es que en sistemas operativos Windows, al crear la base de datos y las tablas, estos pueden escribirse indistintamente en mayúscula o minúscula, pero

[18] Basic Input Output System. Estándar que define la interfaz del firewall en computadoras compatibles IBM.

específicamente en Debian deben ser escritos de manera idéntica en las tablas y en el programa del microcontrolador. Al realizar las pruebas y verificar los datos recopilados por el sistema, todos los sensores recolectaban bien la información, exceptuando el sensor ultrasónico que se encuentra en la puerta. Este error se corrigió cambiando la palabra “puerta” por “Puerta” en la programación del microcontrolador.

Una vez que se corrigieron todas las fallas en el sistema, el siguiente paso fue la captura de la información que revela el evento que la produjo, es decir, las alarmas y el guardado de los avisos en la base de datos, cada uno con la hora y fecha sin corregir y corregida, esto para demostrar la correcta configuración del reloj. Véase figura Ⅲ.26.

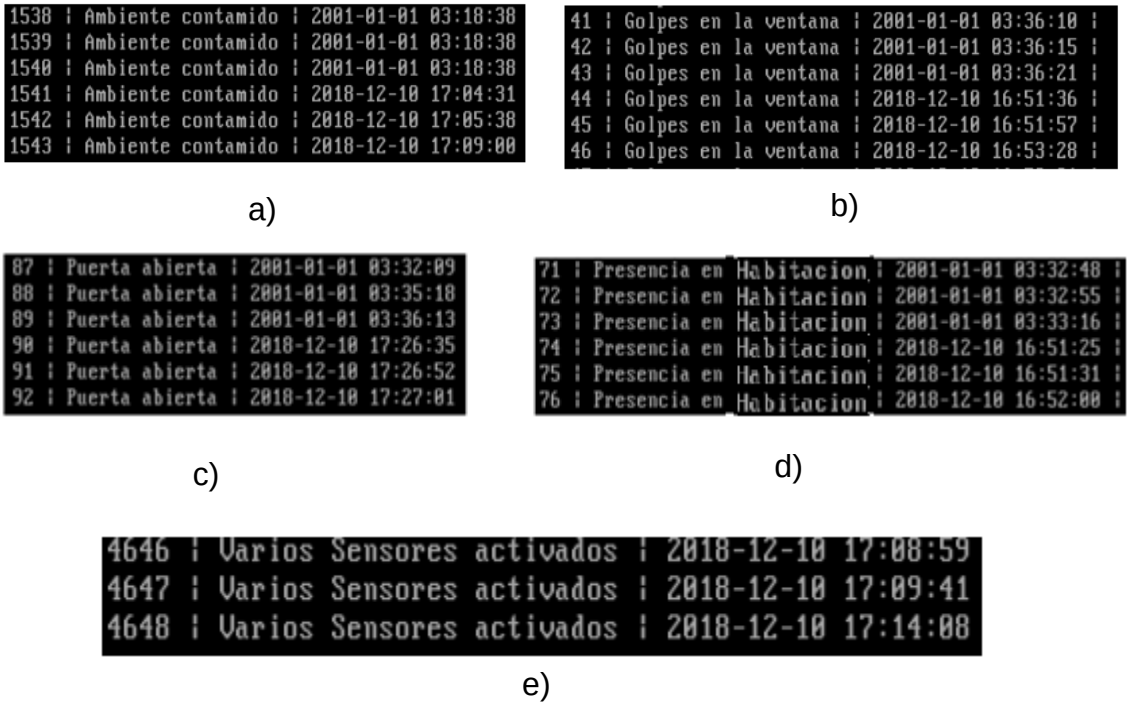


Figura Ⅲ 26. Mensajes almacenados en la base de datos al activarse a) el sensor MQ-135, b) el sensor SW-420, c) el sensor HC-SR04, d) los sensores PIR y e) más de un sensor.

El motivo de utilizar una fecha correcta es para que los datos se puedan utilizar como prueba del siniestro en caso de ser necesario. Una vez dicho esto proseguimos con la prueba de acceso al sistema de alarma.

Ⅲ.6 Acceso al sistema de alarma

El acceso al sistema de alarma está dado por medio de una contraseña. Haciendo uso del diagrama de flujo Ⅱ.3 se creó un programa que pudiera proporcionar pulsos digitales como señal de entrada para manipular el servidor Arduino ethernet, todo esto por medio de un teclado matricial.

La función del teclado es proporcionar una contraseña, que es, una combinación de señales digitales. Utilizando dos pulsos se crearon 3 respuestas a 3 contraseñas diferentes las cuales tienen la siguiente función:

1. Contraseña 1: inicializa el sistema en su totalidad. Es decir, el sistema de alarma está encendido y toda actividad es guardada.
2. Contraseña 2: el sistema de alarma solamente está preparado para responder a una intrusión por la ventana o la puerta. Similar a una alarma antirrobo para autos, el sistema de alarma sonará si alguien quiere irrumpir en la vivienda y hay gente en su interior. En este caso no se guardará ningún registro puesto que solo se considera usar esta opción cuando la vivienda esté ocupada.
3. Contraseña 3: el sistema de alarma se apagará. Este modo es el predeterminado para el sistema y mantiene todo apagado.

Cabe mencionar que la tarjeta Arduino uno y Arduino ethernet compartían una tierra común. Dicho esto, al iniciar el servidor Arduino ethernet, este no habilitará el teclado hasta que la conexión a la base de datos se haya concretado. El tiempo que suele tardar en entablarse una conexión a la base de datos puede ser de entre 10 a 30 segundos.

Para saber si la conexión tuvo éxito se pueden realizar dos pruebas. Una prueba consiste en utilizar el monitor serie del IDE de Arduino, pero este método no es recomendable ya que requiere de un host para verificar esto, además de que es ineficiente al no permitir que el sistema de alarma sea un sistema autónomo el cual se puede conectar a una red. El siguiente método es utilizar una señal digital que envíe el servidor Arduino ethernet al Arduino uno para indicar al teclado que ya se

ha entablado la conexión y se puede utilizar. Lo mencionado en los dos puntos anteriores se muestra en la figura Ⅲ.27.

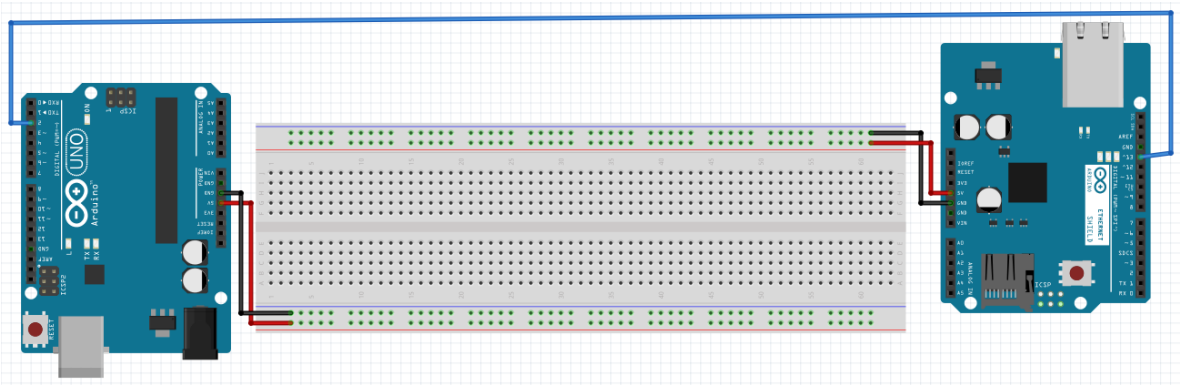


Figura Ⅲ 27. Conexión entre Arduino uno y Arduino ethernet cuya finalidad es poder alimentar toda la red de sensores. Por medio del cable azul se manda una señal que inicializa el teclado.

Para hacer un correcto uso del sistema fue necesario utilizar una entrada pull up que mantuviera la señal del teclado en alto, evitando con ello lecturas erróneas, ya que sin la entrada pull up, el teclado en muchas ocasiones se activaba sin que se realizara una conexión a la base de datos.

Con esto finaliza el capítulo 3, en el cual se describió el funcionamiento del sistema, además de la corrección de una serie de problemas con los cuales un ingeniero en sistemas se suele tropezar, eso va desde la configuración hasta la detección y resolución de errores, además de la administración de la red datos. Dicho lo anterior, en el próximo y último capítulo, se dan las conclusiones a las que se llegó al realizar el sistema de alarma.

CAPÍTULO Ⅳ: CONCLUSIONES

El tema principal de este trabajo recepcional es el desarrollo de un sistema de alarma automatizado que pueda guardar en una base de datos registros de los eventos que causan la activación del sistema de alarma. En los capítulos anteriores se explican los elementos y la función de ellos para poder crear dicho sistema. En resumen, se explicó de forma sintética la comunicación entre las partes del sistema, además de hacer mención de una serie de herramientas y métodos que son necesarios para dicha comunicación.

Con base en lo expuesto anteriormente, se puede afirmar que se logró comprobar que un sistema de alarma creado con ayuda de tarjetas desarrollo es viable y se puede implementar.

Otro punto que se logra comprobar con éxito es que el sistema de alarma es un sistema de bajo costo al implementarse totalmente con una serie de herramientas de uso abierto.

Señalando los resultados obtenidos se tienen los siguientes puntos:

1. Se corroboró que el sistema de alarma creado es multiplataforma.
2. Con la realización de diversas pruebas se comprobó que el sistema de alarma da una respuesta eficiente ante sucesos que podrían interpretarse como una intrusión a la casa habitación del usuario del sistema de alarma, lo que comprueba su funcionalidad.
3. La forma en que se corroboro que el sistema es multiplataforma no fue solamente utilizando computadoras personales, el hacer uso de la tarjeta Intel Galileo demostró que, mientras se tenga un host con la base de datos MySQL, se puede implementar este sistema de alarma.

Por otro lado, la implementación de la alarma en una maqueta muestra que este sistema es viable para ser instalado en una vivienda, ya sea en modo completo, es decir, que pueda guardar registro de los eventos que detecten los sensores o que simplemente sirva para encender una alarma de forma automática cada que se presente un evento sin llevar un registro de ello.

Cabe resaltar que, aunque este tipo de sistemas ya existe, el utilizar todos los elementos mencionados hace de este proyecto una posible respuesta para aquellos que no cuentan con los recursos económicos suficientes para pagar por un sistema proporcionado por alguna empresa privada, lo que implica ayudar a que la sociedad pueda proteger de intrusiones su vivienda de una manera más económica.

La conclusión general a la que se llegó es con respecto a la viabilidad de crear un tipo especial de sistema de alarma. Es decir, se concluye que con poco dinero se puede crear algo útil y funcional que ayude a la solución de un problema real.

Otro hecho importante es que con este proyecto se utilizaron casi todos los conocimientos enfocados a la carrera de Ingeniería en Sistemas Electrónicos y de Telecomunicaciones.

Este proyecto únicamente tiene fines demostrativos y educativos, es por ello que se empleó únicamente una red privada la cual es la 192.168.1.x/24. No se hace uso de un hosting gratuito ya que una de las condiciones para hacer uso de uno de ellos implica el total control de mi información.

Finalmente, este proyecto aporta las bases necesarias para un sistema cuya función no sea solamente de alertar acerca de posibles peligros, sino que también pueda modificarse y emplearse más a fondo teniendo como meta el IoT. De igual forma este trabajo ejemplifica de forma sencilla la automatización de procesos de la cual ya somos parte.

Este sistema puede servir como base para el desarrollo de mejores y más económicos sistemas de seguridad, de ser así espero que en un futuro la sociedad pueda gozar de una mayor sensación de seguridad.

REFERENCIAS

1. Karen C. S. Donnelly, “Domestic Security: The Holmes Burglar Alarm Telegraph, 1853 – 1876”, Master thesis, University of Pennsylvania, Philadelphia, PA, 1992.
2. ADT Security system (2018, Dec 15). Our history [En línea]. Disponible en: https://www.adt.com/about-adt/history
3. Abus Security Tech Germany. (2018, Dec 15). Historia de los sistemas de alarma [En línea]. Disponible en: https://www.abus.com/es/Guia/Proteccion-antirrobo/Sistemas-de-alarma/Historia-de-los-sistemas-de-alarma
4. INEGI. (2010 – 2017). Tasa de incidencia delictiva por entidad federativa de ocurrencia por cada cien mil habitantes [En línea]. Disponible en: http://www.beta.inegi.org.mx/temas/incidencia/
5. Observatorio Nacional Ciudadano. (2017, Abr 25). Tag Archives: Robo a casa habitación [En línea]. Disponible en: http://onc.org.mx/tag/robo-a-casa-habitacion/
6. History.com Editors. (2018, Aug 21). Yom Kippur War [En línea]. Disponible en: https://www.history.com/topics/middle-east/yom-kippur-war#FWNE.fw..yo007800.a
7. P. V. Felipe, C. D Adrian, C. Díaz, R. C Sardiñas y M. H Tanayi, “Edificios Inteligentes”, *Cubasolar*, Articulo 09 [En línea]. Disponible en: http://www.cubasolar.cu/biblioteca/Energia/Energia60/HTML/Articulo09.html
8. C. Ricardo. Sensores para Arduino. Aplicaciones con microprocesadores y microcontroladores, Colegio de Ciencia y Tecnología. Universidad Autónoma de la Ciudad de México. Junio del 2017.
9. Young, Hugh D. y Roger A. Freedman. *Física universitaria, con física moderna volumen 2*. 12ª. Ed. México. Pearson, 2009. P 1054.
10. T. Eckel. (2017, Feb 24). NewPing Library for Arduino. [En línea]. Disponible en: https://playground.arduino.cc/Code/NewPing
11. Intel. (2018, May 24). Intel en sus 50 años: el microprocesador 8080 (English only) [En línea]. Disponible en: https://newsroom.intel.la/news-releases/intel-en-sus-50-anos-el-microprocesador-8080/?wapkw=microprocesador+4004#gs.gRQ4xc0

12. Stanford University. (2018, Dec 15). Marcian “Ted” Hoff [En línea]. Disponible en: https://engineering.stanford.edu/about/heroes/ted-hoff
13. Computer History Museum. (2018, Dec 15). Federico Faggin 2009 fellow [En línea]. Disponible en: http://www.computerhistory.org/fellowawards/hall/federico-faggin/
14. R. Palomino. “GNU Lesser General Public Licence (traducción). Free Software Fundation. Versión 2.1 (Feb, 1999). [DOI].
15. Free Software Fundation, Int. “GNU Lesser General Public Licence”, Free Software Fundation, Inc. Tec. version 3 (Jun 29, 2007).
16. Creative Commons. (2018, Dec 15). Attribution-ShareAlike 4.0 International [En línea]. Disponible en: https://creativecommons.org/licenses/by-sa/4.0/legalcode
17. Intel Corporate, “Intel Quark SoC X1000”, Intel, Datasheet, 329676-005US, August 2015.
18. José Ignacio Castillo Velázquez, *Redes de datos Contexto y evolución*. 2ª. Ed. México: Samsara, 2016.
19. José Ignacio Castillo Velázquez, *Switching & Routing Introducción*. 1ª. Ed. México: Samsara, 2016.
20. Luis Miguel Crespo Martínez y Francisco A. Cadelas herías*, Introducción A TCP/IP*. Edición Electrónica. Universidad De Alicante: Publicaciones de la Universidad de Alicante.
21. RFC 1945, *Hypertext Transfer Protocol - - HTTP/1.0*, 1996.
22. RFC 2616, *Hypertext Transfer Protocol - - HTTP/1.1*, 1999.
23. RFC 7540, *Hypertext Transfer Protocol -Version 2 (HTTP/2), 2015.*
24. W3C. (2018, Dec 15). About w3c [En línea]. Disponible en: https://www.w3.org/Consortium/
25. S. Bradner (traducción de M. Bagnulo). (2018, Dec 15). Estrucutura del IETF y procesos de estandarización en Internet [En línea]. Disponible en: https://www6.ietf.org/edu/tutorials/IETF-newcomers-esp.pdf
26. C. Villagómez (2017, Sep 21). Puerto/Puertos TCP/IP [En línea]. Disponible en: https://es.ccm.net/contents/272-puerto-puertos-tcp-ip

27. José Ignacio Castillo Velázquez, *Redes de datos Contexto y evolución*. 2ª. Ed. México: Samsara.
28. Statcounter. (2018, Dec 15). Company Mission [En línea]. Disponible en: https://statcounter.com/about/mission/
29. T. C Miguel Ángel, *Bases de datos para la administración. Access*. 1ª Ed. México: Mc Graw Hill.
30. G. D Alejandro, *Bases de datos*.Clave-Mis 308. Centro de cultura itálica S.C [En línea]. Disponible en: https://www.aiu.edu/cursos/base%20de%20datos/pdf%20leccion%201/lecci%C3%B3n%201.pdf
31. Free Sofware Fundation, Inc. "GNU General Public License". Free Software Foundation, Inc. Tec. Version 3, (Jun 29, 2007).
32. The Apache Software Foundation. (2018 Dec 15). What is the asf? [En línea]. Disponible en: https://www.apache.org/foundation/
33. Netcraft. (2018 Dec 15). About Netcraft [En línea]. Disponible en: https://www.netcraft.com/about-netcraft/
34. MariaDB. (2018 Dec 15). About us [En línea]. Disponible en: https://MariaDB.com/about-us/
35. PHP. (2001-2018). Historia de PHP [En línea]. Disponible en: http://php.net/manual/es/history.php.php
36. PhpMyAdmin. (2018 Dec 15). History [En línea]. Disponible en: https://www.PhpMyAdmin.net/about/
37. EcuRed. (2011 oct 18). PhpMyAdmin [En línea]. Disponible en: https://www.ecured.cu/PhpMyAdmin
38. Diseño Digital Instituto Tecnológico de Querétaro. Departamento de Ingeniería Eléctrica y Electrónica Guía de Práctica de Diseño Digital (Feb 23)
39. Hanwel Electronics CO. LTD, "Technical Data MQ-135 gas sensor". Hanwel Electronics CO. LTD [En línea]. Disponible en: https://www.electronicoscaldas.com/datasheet/MQ-135_Hanwei.pdf
40. RFC 4253, *The Secure Shell (SSH) Transport Layer Protocol*, 2006

Anexos

Anexo A

El siguiente código se escribió para poder manejar el sistema por medio de contraseñas. Con la ayuda de dos bibliotecas de apoyo, Keypad y LiquidCrystal, se ingresan caracteres y se visualizan respectivamente. Este código lo que realiza es una comparación entre dos arreglos para seleccionar un subsistema del sistema de alarma.

```
#include <Key.h> // biblioteca del teclado
#include <Keypad.h>
#include <LiquidCrystal.h> // biblioteca del display LCD
int VO = 10;
int RS = A0;
int E = A1;
int D4 = A2;
int D5 = A3;
int D6 = A4;
int D7 = A5;
LiquidCrystal lcd (RS, E, D4, D5, D6, D7);
int a=14;
char password[]="1111";
char password1[]="2222";
char password2[]="3333";
char pass[4];
int counting=0;
int led12=12;
int led11=11;
int pin10=10;
const byte ROWS = 4; // filas 4
const byte COLS = 4;  // columnas 4
char keys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
};
byte rowPins[ROWS] = {9,8,7,6}; //pines de las filas en arduino
byte colPins[COLS] = {5,4,3,2,}; // pines de las columnas en arduino
// crear el keypad
```

```
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS); // crear el
teclado
void setup(){
Serial.begin(115200);
  lcd.begin(16, 2);
  pinMode(A0, OUTPUT);
  pinMode(A1, OUTPUT);
  pinMode(A2, OUTPUT);
  pinMode(A3, OUTPUT);
  pinMode(A4, OUTPUT);
  pinMode(A5, OUTPUT);
pinMode(12,OUTPUT);
digitalWrite(12,LOW);
pinMode(11,OUTPUT);
digitalWrite(11,LOW);
pinMode(10,INPUT_PULLUP );
digitalWrite(10,HIGH);
delay(3000);
while(digitalRead(10)==HIGH)
{
  lcd.setCursor(0,0);
  lcd.print("Starting system");
  lcd.setCursor(0,1);
  lcd.print("please wait");
}
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Connneted  to");
  lcd.setCursor(0,1);
  lcd.print("server");
  delay(1000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("hello, please");
  lcd.setCursor(0,1);
  lcd.print("your password");
}
void loop(){
  char key=keypad.getKey();
  if (key!=0)
  {
pass[counting]=key;
Serial.print(pass[counting]);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Entering");
  lcd.setCursor(a,1);
  lcd.print("_***************");
```

```
a--;
counting++;
if (counting==4)
{
  if (pass[0]==password[0] && pass[1]==password[1] && pass[2]==password[2] &&
pass[3]==password[3])
{
  lcd.clear();
   lcd.setCursor(0,0);
   lcd.print("Password correct");
   lcd.setCursor(0,1);
  lcd.print("for system 1");
 delay(1000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Password correct");
  lcd.setCursor(0,1);
  lcd.print("system 1 on");
  digitalWrite(12,HIGH);
 digitalWrite(11,HIGH);
 a=14;
}
else if(pass[0]==password1[0] && pass[1]==password1[1] && pass[2]==password1[2] &&
pass[3]==password1[3])
{
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("Password correct");
 lcd.setCursor(0,1);
 lcd.print("for system 2");
 delay(1000);
 lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Password correct");
  lcd.setCursor(0,1);
  lcd.print("system 2 on");
  digitalWrite(12,LOW);
  digitalWrite(11,HIGH);
 a=14;
}
 else if(pass[0]==password2[0] && pass[1]==password2[1] && pass[2]==password2[2] &&
pass[3]==password2[3])
{
  lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("Password correct");
 lcd.setCursor(0,1);
 lcd.print("for system 3");
```

```
 delay(1000);
 lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Password correct");
  lcd.setCursor(0,1);
  lcd.print("system 3 off");
 digitalWrite(12,LOW);
 digitalWrite(11,LOW);
 a=14;
}
else if(pass!=password || pass!=password1)
{
   lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("Password error");
 lcd.setCursor(0,1);
 lcd.print("XXXX");
  delay(1000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Error, please");
  lcd.setCursor(0,1);
  lcd.print("try again");
  a=14;
}
counting=0;
}
  }
  }
```

Anexo A. Código para la creación de contraseña y selección de subsistema.

Anexo B

En este código convergen todos los diagramas de flujo y todas las pruebas de los sensores. Con ayuda de este código un Arduino ethernet se puede comunicar a una base de datos MySQL y guardar registros siempre y cuando la red de sensores detecte alguna actividad.

```
#include <SPI.h>
#include <Ethernet.h>
#include <MySQL_Connection.h>
#include <MySQL_Cursor.h>
// libreria new ping y configuracion para el sensor ultrasonico
#include <NewPing.h>
#define TRIGGER_PIN  2
#define ECHO_PIN     3
#define MAX_DISTANCE 200
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
// finaliza new ping
byte mac[]={0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};
IPAddress servidorMySQL(192,168,200,154);
IPAddress IPLocal(192,168,200,150);
IPAddress DNS(192,168,1,254);
IPAddress IPGateway(192,168,1,254);
IPAddress IPSubnet(255,255,255,0);
EthernetClient objRed;
MySQL_Connection cn((Client *)&objRed);
const int pin4=4; // encendido/apagado
const int pin5=5; // interno/externo
const int pin6=6; // inicar el teclado
const int pin7=7; // ventilador
const int pin8=8; // sirena
const int pin9=9; // foco
char userMySQL[]="gustavo";
char passwordMySQL[]="aozaki01";
// mensaje para la tabla ambiente
char INSERT_SQL1[] = "INSERT INTO sistema.Habitacion_A (Actividad) VALUES ('Presencia en
abitacion')";
// char INSERT_SQL2[] = "INSERT INTO sistema.Habitacion_B (Actividad) VALUES ('Presenscia
en abitacion')";
char INSERT_SQL3[] = "INSERT INTO sistema.Puerta (Actividad) VALUES ('Puerta abierta')";
char INSERT_SQL4[] = "INSERT INTO sistema.Ventana (Actividad) VALUES ('Golpes en la
ventana')";
char INSERT_SQL5[] = "INSERT INTO sistema.Sistema (Actividad) VALUES ('Varios Sensores
activados')";
```

```
char INSERT_SQL6[] = "INSERT INTO sistema.Cocina (Actividad) VALUES ('Ambiente
contamido')";
// char INSERT_SQL7[] = "INSERT INTO sistema.Cocina (Actividad) VALUES ('Contaminacion
grave')";
void setup() {
pinMode(4, INPUT);
pinMode(5, INPUT);
pinMode(6, OUTPUT);
pinMode(7, OUTPUT);
pinMode(8, OUTPUT);
pinMode(9, OUTPUT);
digitalWrite(6,HIGH);
digitalWrite(7,LOW);
digitalWrite(8,LOW);
digitalWrite(9,LOW);
Ethernet.begin(mac, IPLocal,DNS, IPGateway, IPSubnet);
Serial.begin(115200);
delay(1000);
Serial.println(F("Connecting..."));
 if(cn.connect(servidorMySQL, 3306, userMySQL, passwordMySQL)){
delay(1000);
Serial.println(F("Success!"));
digitalWrite(6,LOW); // señal de inicio al teclado, con esto el teclado sabe que ya puede
meter contraseñas.
Serial.println(F("Connected"));
}else {
Serial.println(F("Connection failed."));
}
}
void loop() {
char var1=digitalRead(4); // lectura del teclado 11 teclado
char var2=digitalRead(5); //  lectura del teclado   12 teclado
if(var1==HIGH && var2==HIGH){ // if del encendido
if ((analogRead(A0)>130) && (analogRead(A1)<400) && (analogRead(A2)<400) &&
(sonar.ping_cm()<=13)) // sensor de gas
{// primer if
MySQL_Cursor *cur_mem = new MySQL_Cursor(&cn);
cur_mem->execute(INSERT_SQL6); // ambiente contaminado
delete cur_mem;
delayMicroseconds(100);
while((analogRead(A0)>130) && (analogRead(A1)<400) && (analogRead(A2)<400) &&
(sonar.ping_cm()<=13))
{
digitalWrite(7,HIGH); // se encienden los ventiladores
digitalWrite(8,LOW);  // foco
digitalWrite(9,LOW);  // pitido
delayMicroseconds(100);
Serial.println(analogRead(A0));
```

```
}
}  // primer if
else if ((analogRead(A0)<=130) && (analogRead(A1)>=400) && (analogRead(A2)<400) &&
(sonar.ping_cm()<=13))
{// segundo if pir habitacion_A
MySQL_Cursor *cur_mem = new MySQL_Cursor(&cn);
cur_mem->execute(INSERT_SQL1); // presencia en la vivienda
delete cur_mem;
delayMicroseconds(100);
while((analogRead(A1)>=400))
{
digitalWrite(7,LOW); // se encienden los ventiladores
digitalWrite(8,HIGH);  // foco
digitalWrite(9,HIGH);  // pitido
}
}// fin segundo if pir habitacion_A
else if((analogRead(A0)<=130) && (analogRead(A1)<400) && (analogRead(A2)>=400) &&
(sonar.ping_cm()<=13))
{ //tercer if vibrador
Serial.println(F("vibrador"));
MySQL_Cursor *cur_mem = new MySQL_Cursor(&cn);
cur_mem->execute(INSERT_SQL4); // ventana golepada
delete cur_mem;
delayMicroseconds(100);
{
digitalWrite(7,LOW); // se encienden los ventiladores
digitalWrite(8,HIGH);  // foco
digitalWrite(9,HIGH);  // pitido
delay(1000);
}
} // fin tercer if
else if ((analogRead(A0)<=130) && (analogRead(A1)<400) && (analogRead(A2)<400) &&
(sonar.ping_cm()>13))
{ // curto if puerta
Serial.println(sonar.ping_cm());
MySQL_Cursor *cur_mem = new MySQL_Cursor(&cn);
cur_mem->execute(INSERT_SQL3); // puerta abierta
delete cur_mem;
delayMicroseconds(100);
{
digitalWrite(7,LOW); // se encienden los ventiladores
digitalWrite(8,HIGH);  // foco
digitalWrite(9,HIGH);  // pitido
delayMicroseconds(50);
}
} // fin cuarto if
else if((analogRead(A0)<=130) && (analogRead(A1)<400) && (analogRead(A2)<400) &&
(sonar.ping_cm()<=13)) //sistema apagado
```

```
{ // quito if
 //  Serial.println(F("sistema apagado"));
digitalWrite(7,LOW); // se encienden los ventiladores
digitalWrite(8,LOW);  // foco
digitalWrite(9,LOW);  // pitido
} // fin quinto if
else { // todos los sensores activos
MySQL_Cursor *cur_mem = new MySQL_Cursor(&cn);
cur_mem->execute(INSERT_SQL5); // varios sensores activos
delete cur_mem;
delayMicroseconds(100);
{
digitalWrite(7,LOW); // se encienden los ventiladores
digitalWrite(8,HIGH);  // foco
digitalWrite(9,HIGH);  // pitido
delayMicroseconds(100);
}  }
}// fin primer if de encendido
else if (var1==HIGH && var2==LOW) // segundo if, sistema externo
{
analogRead(A2);
if (analogRead(A2)>400){
// Serial.println("vibrador HIGH");
digitalWrite(9,HIGH);
digitalWrite(8,HIGH);
digitalWrite(7,LOW);
delay (2000);
}
else
{
//Serial.println("vibrador: LOW");
digitalWrite(9,LOW);
digitalWrite(8,LOW);
digitalWrite(7,LOW);
}}
else if (var1==LOW && var2==LOW)
{
//Serial.println("vibrador LOW");
digitalWrite(9,LOW);
digitalWrite(8,LOW);
digitalWrite(7,LOW);
}} // loop
```

Anexo B. Código para la implementación del sistema de alarma automatizado.

Anexo C

Código utilizado para comprobar el funcionamiento del sensor PIR. El código hace que cada vez que el sensor PIR detecte movimiento se muestre un mensaje del evento con la herramienta monitor serie del IDE de Arduino.

```
const int pin8=8;
void setup() {
pinMode(8,INPUT);
Serial.begin(9600);
delay(5000);
}
void loop() {
if(digitalRead(8)==HIGH)
{
Serial.println("Se detecta movimiento");
Serial.println(analogRead(A0));
}
else
{
Serial.println("No hay movimiento");
}
delay(200);
 Serial.println(analogRead(A0));
}
```

Anexo C. Código para probar el funcionamiento del sensor PIR.

Anexo D

Código utilizado para comprobar el funcionamiento del sensor SW-420. El código hace que cada vez que el sensor SW-420 sea golpeado se muestre un mensaje del evento con la herramienta monitor serie del IDE de Arduino.

```
int pin9=9;
int pin4=4;
void setup() {
pinMode(9, INPUT);
pinMode(4, OUTPUT);
Serial.begin(9600);
}
void loop() {
if(digitalRead(9)== HIGH)
{
digitalWrite(4, HIGH);
Serial.println("el cristal fue golpeado");
}
else
{
digitalWrite(4, LOW);
Serial.println("sin novedad");
}
delay(100);
}
```

Anexo D. Código para probar el funcionamiento del sensor SW-420.

Anexo E

Código utilizado para comprobar el funcionamiento del sensor HC-SR04 sin biblioteca. El código hace que muestre la distancia entre el sensor y algún objeto en su línea de vista, la distancia se muestra en centímetros, todo esto con la herramienta monitor serie del IDE de Arduino.

```
const int Trigger = 2;  //Pin digital 2 para el Trigger del sensor
const int Echo = 3;   //Pin digital 3 para el Echo del sensor
int led4=4;
void setup() {
Serial.begin(9600);//iniciailzamos la comunicación
pinMode(2, OUTPUT); //pin como salida
pinMode(3, INPUT);  //pin como entrada
pinMode(4,OUTPUT);
digitalWrite(2, LOW);//Inicializamos el pin con 0
}
void loop()
{
long t; //timepo que demora en llegar el eco
long d; //distancia en centimetros
digitalWrite(2, HIGH);
delayMicroseconds(10);          //Enviamos un pulso de 10us
digitalWrite(2, LOW);
t = pulseIn(3, HIGH); //obtenemos el ancho del pulso
d = t/59;             //escalamos el tiempo a una distancia en cm
while(d<=10)
{
long t; //timepo que demora en llegar el eco
long d; //distancia en centimetros
digitalWrite(2, HIGH);
delayMicroseconds(10);          //Enviamos un pulso de 10us
digitalWrite(2, LOW);
t = pulseIn(3, HIGH); //obtenemos el ancho del pulso
d = t/59;             //escalamos el tiempo a una distancia en cm
}
digitalWrite(4,LOW);
Serial.print("Distancia: ");
Serial.print(d);    //Enviamos serialmente el valor de la distancia
Serial.print("cm");
Serial.println();
delay(100);          //Hacemos una pausa de 100ms
}
```

Anexo E. Código para probar el funcionamiento del sensor ultrasónico sin hacer uso de una biblioteca.

Anexo F

Código utilizado para comprobar el funcionamiento del sensor HC-SR04 con el apoyo de la biblioteca NewPing. El código hace que cada que se abra la puerta se muestre un mensaje del evento, todo esto con la herramienta monitor serie del IDE de Arduino.

```
#include <NewPing.h>
#define TRIGGER_PIN  2
#define ECHO_PIN     3
#define MAX_DISTANCE 200
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
const int pint4=4;
void setup() {
Serial.begin(115200);
pinMode(4, OUTPUT);
digitalWrite(4,LOW);
}
void loop() {
delay(50);
// Serial.print("Ping: ");
// Serial.println(sonar.ping_cm());
// Serial.println("cm");
while(sonar.ping_cm()<12)
{
digitalWrite(4,HIGH);
Serial.print(sonar.ping_cm());
Serial.print(" cm");
Serial.println(" sirve");
delay(100);
}
digitalWrite(4,LOW);
Serial.print(sonar.ping_cm());
Serial.print(" cm");
Serial.println(" puerta habierta ");
delay(100);
}
```

Anexo F. Código para probar el funcionamiento del sensor ultrasónico haciendo uso de una biblioteca.

Anexo G

Código utilizado para comprobar el funcionamiento del sensor MQ-135. Con este código se puede visualizar la lectura del sensor MQ-135 utilizando el CAD del Arduino ethernet, todo esto con apoyo de la herramienta monitor serie del IDE de Arduino.

La finalidad de esta prueba es obtener rangos en los que se detecte algún contaminante como lo son el humo y el gas butano.

```
void setup() {
Serial.begin(9600);
}
void loop() {
Serial.println(analogRead(A0));
}
```

Anexo G. Código para probar el funcionamiento del sensor MQ-135.