

SEP

SES

TNM

INSTITUTO TECNOLÓGICO DE CHIHUAHUA II



**“IMPLEMENTACIÓN DE SOFTWARE SCADA PARA
MONITOREO Y CONTROL DE EDIFICIO INTELIGENTE”**

TESIS

PARA OBTENER EL GRADO

MAESTRO EN SISTEMAS COMPUTACIONALES

PRESENTA

TOMÁS DANIEL PORTILLO OGAZ

DIRECTOR DE TESIS

DR. RAFAEL SANDOVAL RODRÍGUEZ

CO- DIRECTOR DE TESIS

M.C. ARTURO LEGARDA SÁENZ

CHIHUAHUA, CHIH. FEBRERO 2019

Dictamen

Chihuahua, Chih., 04 de marzo del 2019

LIC. OLGA REBECA CASTILLO CRUZ
JEFA DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
Presente.-

Por medio de este conducto el comité tutorial revisor de la tesis para obtención de grado de Maestro en Sistemas Computacionales, que lleva por nombre "IMPLEMENTACIÓN DE SOFTWARE SCADA PARA MONITOREO Y CONTROL DE EDIFICIO INTELIGENTE", que presenta el (la) **C. TOMÁS DANIEL PORTILLO OGAZ**, hace de su conocimiento que después de ser revisado ha dictaminado la **APROBACIÓN** del mismo.

Sin otro particular de momento, queda de Usted.

Atentamente

La Comisión de Revisión de Tesis.



M.C. RAFAEL SANDOVAL RODRÍGUEZ

Director de Tesis



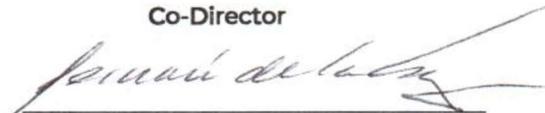
M.C. ARTURO LEGARDA SÁENZ

Co-Director



M.I.S.C. JESÚS ARTURO ALVARADO GRANADINO

Revisor



DR. HERNÁN DE LA GARZA GUTIÉRREZ

Revisor



SECRETARÍA DE
EDUCACIÓN PÚBLICA
INSTITUTO TECNOLÓGICO
DE CHIHUAHUA II
DIVISIÓN DE ESTUDIOS DE
POSGRADO E INVESTIGACIÓN

DEDICATORIA

Agradezco a los familiares y amigos que me apoyaron durante mis estudios de maestría, a los maestros de los que aprendí tanto.

Principalmente quiero dedicar esta tesis a aquellos que nunca duraron de mi capacidad para poder completar esta etapa.

RESUMEN

La presente tesis documenta el desarrollo de un proyecto de automatización de un edificio en el Instituto Tecnológico de Chihuahua II, dicho proyecto está compuesto de un sistema de control SCADA para el monitoreo y control de los sensores y actuadores, el cual es un sistema Web con características de un STR (Sistema de tiempo real). Otro componente es la red de domótica, que está compuesta por los sensores, actuadores, microcontroladores que los manejan y las interconexiones entre los mismos. El tercer componente que se describe en la tesis es el Gateway, el cual es un dispositivo con múltiples puertos de conexión capaz de comunicar al sistema SCADA y a la red de domótica de una manera transparente.

Para maximizar la compatibilidad con tecnologías de comunicación y fabricantes, el sistema se desarrolló haciendo uso de protocolos de comunicación abiertos. Gracias a la implementación de este sistema el Instituto Tecnológico de Chihuahua II tendrá una eficiente manera de medir el uso de la energía y de este modo reducir el desperdicio de la misma.

ABSTRACT

This thesis describes the development of a project of automation of a building in the Technological Institute of Chihuahua II, this project is composed of a SCADA control system for the monitoring and control of the sensors and actuators, which is a Web system with characteristics of a STR (Real-time system). Another component is the home automation network, which is composed of the sensors, actuators, microcontrollers that handle them and the interconnections between them. The third component described in the thesis is the Gateway, which is a device with multiple connection ports responsible of the communication with the SCADA system and the home automation network in a transparent way.

To maximize compatibility with different communication technologies and manufacturers, the system was developed using open communication protocols. Because of the implementation of this system, the Technological Institute of Chihuahua II will have an efficient way to measure the use of energy and in this way reduce the waste of it.

ÍNDICE

1. I. INTRODUCCIÓN	1
1.1. Introducción.....	1
1.2. Planteamiento del problema	2
1.3. Alcances y limitaciones	10
1.4. Justificación.....	11
1.5. Objetivos.....	12
2. II. ESTADO DEL ARTE.....	13
2.1. Una revisión sobre sistemas para edificios inteligentes sustentables	13
2.2. Sensores y actuadores inteligentes basado en WSN.....	16
2.3. Diseño de sistema SCADA con HTML5	17
2.4. Sistema SCADA Completo de Bajo Costo para Casa Inteligente.....	18
2.5. IoT (internet de las cosas) basado en micro-servicio para edificios inteligentes	19
2.6. Construyendo Servicios de IoT para adultos mayores	20
2.7. Conclusión del estado de al arte	22
3. III. MARCO TEÓRICO	23
3.1. Domótica	23
3.2. Inmótica.....	24
3.4. Edificio inteligente	25
3.5. Componentes para la automatización de un edificio	25
3.5.1. Sensores	25
3.5.2. Acondicionadores de señal	27
3.5.3. Actuadores	27
3.5.4. Relés o relevadores.....	28
3.5.5. Contactores	28
3.5.6. Regulador o <i>dimmers</i>	29
3.6. SCADA.....	29
3.7. <i>Mean stack</i>	29

3.7.1. MongoDB	30
3.7.2. Express	30
3.7.3. AngularJS	31
4. IV. DESARROLLO	32
4.1. Análisis	32
4.1.1. Análisis de los requisitos	32
4.1.2. Diagramas de caso de uso.....	33
4.2. Diseño	34
4.2.1. Diseño arquitectónico	35
4.2.2. Definición de protocolo de comunicación.....	36
4.2.3. Diseño de navegación.....	38
4.2.4. Diseño de la interfaz	39
4.2.5. Diseño de base de datos.....	39
4.2.6. Características de hardware y software para desarrollo	40
4.3. Codificación	43
4.3.1. Codificación de la red de domótica	43
4.3.2. Codificación del software para el Gateway.....	46
4.3.3. Codificación del servidor.....	49
4.4. Pruebas	50
4.5. Implementación	51
5. V. RESULTADOS Y DISCUSIÓN	55
6. VI. CONCLUSIONES	57
7. VII. BIBLIOGRAFÍA	59
VIII. ANEXOS	61
Anexo 1. Código fuente del MCU maestro en red de domótica	61
Anexo 2. Código fuente de MCU esclavo en red de domótica	62
Anexo 3. Código fuente de dispositivo Gateway	67
Anexo 4. Código fuente de servidor.....	68
Package.json	68
App.js.....	69

bin\www	72
App_server\routes\index.js	75

ÍNDICE DE FIGURAS

Figura 1.1. Cierre de la brecha entre el consumo y la producción nacional de energía.	3
Figura 1.2. La producción no puede aumentar si no se perforan más pozos.	4
Figura 1.3. La demanda de gas natural no puede ser abastecida.	5
Figura 1.4. Consumo energético per cápita promedio por hogar.	8
Figura 1.5. Gastos y proporciones en electricidad por decena de habitantes	9
Figura 1.6. Porcentaje de consumo promedio de electricidad en un hogar.	9
Figura 2.1. Gráfica de frecuencia y objetos de estudio.	14
Figura 2.2. Investigaciones realizadas en distintos países.	15
Figura 2.3. Porcentajes según el sector.	15
Figura 2.4. Diagrama funcional de bloques del sistema.	16
Figura 2.5. Arquitectura de la aplicación.	17
Figura 2.6. Sistema SCADA plano de visualización.	18
Figura 2.7. Dispositivos y sus conexiones.	19
Figura 2.8. Diagrama de bloques del servicio.	21
Figura 3.1. Muestra de domótica aplicada.	24
Figura 3.2. Modulo detector de movimiento PIR sensor infrarrojo pasivo HCSR501	26
Figura 3.3. Ejemplo de topología de red de edificio inteligente.	26
Figura 3.4. Acondicionamiento de señal.	27
Figura 3.5. Acondicionamiento de señal de la interfaz	28
Figura 3.6. Relé de 5V modelo XC4419.	28
Figura 4.1. Caso de uso principal.	34
Figura 4.2. Diseño arquitectónico del sistema.	35
Figura 4.3. Ejemplo de comando.	37
Figura 4.4. Estructura de navegación de sitio web.	38
Figura 4.5. Diseño de interfaz gráfica.	39
Figura 4.6. Diagrama entidad relación de la base de datos.	40
Figura 4.7. Diagrama de flujo del funcionamiento del MCU maestro.	43
Figura 4.8. Diagrama de flujo del funcionamiento del MCU esclavo.	44
Figura 4.9. Topología de red de domótica.	45
Figura 4.10. Diagrama de interacción entre los elementos de la red de domótica.	46
Figura 4.11. Pseudocódigo de Gateway.	47
Figura 4.12. Diagrama de flujo de funcionamiento de Gateway.	48
Figura 4.13. Diagrama de flujo de funcionamiento del servidor.	49
Figura 4.14. Aplicación Visual Studio Code.	50
Figura 4.15. Aplicación Arduino IDE.	51
Figura 4.16. Diagrama de conexión de subsistema de iluminación.	52

Figura 4.17. Conexión de circuito en un ambiente de pruebas.....	53
Figura 4.18. Subsistema de iluminación con luz encendida.....	53
Figura 4.19. Capturas de interfaz web.....	54
Figura 4.20. Consola de Gateway.....	54

ÍNDICE DE TABLAS

Tabla 2.1. Escenarios de servicio.	20
Tabla 4.1. Características de servidor.....	40
Tabla 4.2. Características de Gateway.....	41
Tabla 4.3. Características de MCU maestro.	41
Tabla 4.4. Características de MCU esclavos.	42
Tabla 4.5. Características de clientes.....	42

INTRODUCCIÓN

I. INTRODUCCIÓN

1.1. Introducción

La automatización de edificios, aunque tiene sus orígenes desde principios de los años setentas, en la actualidad es una práctica muy frecuente en países desarrollados como Estados Unidos y algunos países europeos. Una de las principales ventajas de este tipo de edificios es la reducción del desperdicio energético, mientras que la principal desventaja es que se utilizan protocolos propietarios (los cuales no son compatibles con protocolos de otras compañías) y con ello se complica la escalabilidad y el mantenimiento de los proyectos de automatización de edificios.

La presente tesis documenta el desarrollo de un proyecto de automatización de un edificio en el Instituto Tecnológico de Chihuahua II, dicho proyecto está compuesto de un sistema de control SCADA para el monitoreo y control de los sensores y actuadores, el cual es un sistema Web con características de un STR (Sistema de tiempo real). Otro componente es la red de domótica, que está compuesta por los sensores, actuadores, microcontroladores que los manejan y las interconexiones entre los mismos. El tercer componente que se describe en la tesis el Gateway, el cual es un dispositivo con múltiples puertos de conexión capaz de comunicar al sistema SCADA y a la red de domótica de una manera transparente.

Para maximizar la compatibilidad con tecnologías de comunicación y fabricantes, el sistema se desarrolló haciendo uso de protocolos de comunicación abiertos. Gracias a la implementación de este sistema el Instituto Tecnológico de Chihuahua II tendrá una eficiente manera de medir el uso de la energía y de este modo reducir el desperdicio de la misma.

INTRODUCCIÓN

1.2. Planteamiento del problema

Según la Secretaría de Energía (SENER), en colaboración con la Agencia Internacional de Energía (AIE), en un estudio realizado en el año 2008, el sector residencial fue el tercer consumidor de energía, tanto nivel nacional (16.7% de energía consumida en México) y así como a nivel internacional (24% de energía consumida a nivel mundial).

De la energía que se consume en los hogares mexicanos, el calentamiento del agua representa el 47% del consumo, seguidos por la cocción de alimentos y la refrigeración con consumos del 27.5% y 9.9% respectivamente. (Secretaría de Energía (SENER) en colaboración con la Agencia Internacional de Energía (AIE), 2011).

En la presente tesis se planea combatir activamente el desperdicio de energía, mediante la automatización de varios procesos cotidianos, tales como; control de clima, control de iluminación, horas de uso de los aparatos electrónicos, entre otros.

La incorporación de sensores y elementos de control dentro de un edificio tiene otras ventajas que se desean explorar en este proyecto, tal es el caso del aumento de la seguridad y la integración del Internet de las Cosas (*Internet of Things, IoT*) en algunos dispositivos dentro del edificio.

Otro aspecto importante que se incluye en el desarrollo del proyecto es la capacidad del sistema de proveer información confiable, ya sea en tiempo real o por medio de un registro histórico, acerca de lo que ocurre en el edificio, de este modo se agilizará la toma de decisiones.

Problema del consumo descuidado de la energía eléctrica

Sobre la *oferta y la demanda de energía* nos habla el CIDAC:

En 10 años México podría carecer de la energía que requiere para su desarrollo económico. Frente a una prospectiva de crecimiento de la economía a tasas superiores al 3% anual durante los siguientes 6 años, la cantidad de insumos energéticos es insuficiente: de 2000 a 2011 la sociedad mexicana incrementó su consumo de energía a una tasa promedio anual de 2.08%;

INTRODUCCIÓN

pero durante el mismo periodo la producción de energía primaria disminuyó a una tasa anual de 0.3%. Esto en un contexto en el que México tiene una limitada capacidad de importación de hidrocarburos básicos como el gas natural.

Como se muestra en la gráfica (ver Figura 1.1) obtenida de (Serra, Ch, Lujambio, Toro, & Aguilar, 2013) (1 watt = 1 joule/s), el consumo de energía ha aumentado paulatinamente, mientras que la generación ha tenido variaciones, pero se encuentra estable, es decir, solo se produce un poco más de lo que se consume. Por lo que La Secretaría de Energía (SENER) estima que para el 2020 México carecerá de la capacidad para asegurar el suministro nacional energético.

Los problemas que incrementan el riesgo de no tener la energía necesaria en el futuro son:

Agotamiento de las reservas y debilitamiento en la producción de crudo

Debido a la caída en pozos exploratorios perforados y una gran proporción de campos en proceso de declinación.

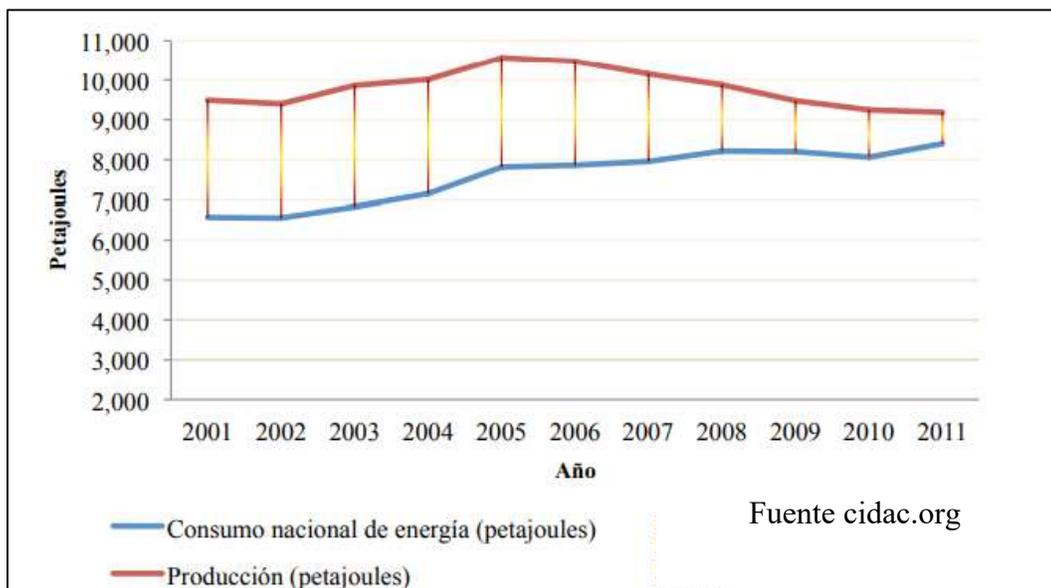


Figura 1.1. Cierre de la brecha entre el consumo y la producción nacional de energía.

INTRODUCCIÓN

En México existe un desfase entre la incorporación de reservas y la producción de crudo. Mientras que la producción de petróleo crudo de 1960 a 2009 aumentó 838%, la incorporación de reservas se desplomó un 3,725% de 1980 a 2010. Ante tal situación, también en 2008 se ordenó en ley a la Comisión Nacional de Hidrocarburos (CNH) procurar que la regulación de los proyectos de exploración y extracción de Pemex se realice considerando siempre la reposición de las reservas de hidrocarburos, como garantes de la seguridad energética del país (ver Figura 1.2) obtenida de (Serra, Ch, Lujambio, Toro, & Aguilar, 2013).

Exceso de demanda de gas natural debido a una capacidad de transporte y red de distribución insuficientes

México es uno de los países con mayores reservas de gas natural en el mundo. Sin embargo, la producción de gas del país no es capaz de satisfacer su demanda, la cual ascendió en el país en 2012 a 246.3 millones de metros cúbicos diarios. Para satisfacer tal demanda, a finales de 2012 se llegó a importar poco más de 73 millones de metros cúbicos de gas natural por día. La demanda continuará creciendo y la capacidad de transporte del recurso a nivel nacional no es suficiente (ver Figura 1.3) obtenida de (Serra, Ch, Lujambio, Toro, & Aguilar, 2013).

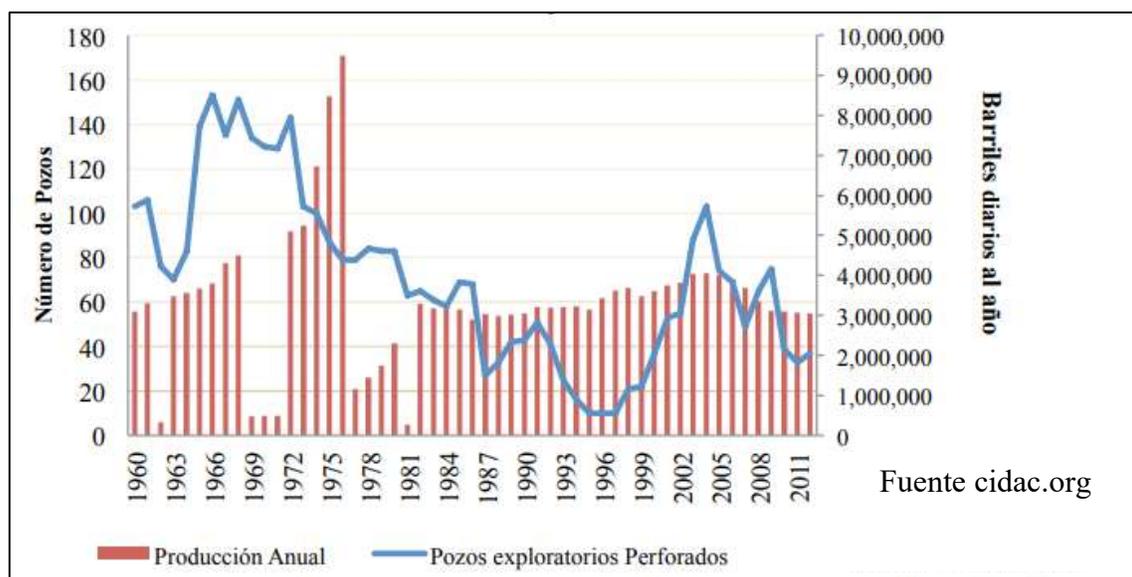


Figura 1.2. La producción no puede aumentar si no se perforan más pozos.

INTRODUCCIÓN

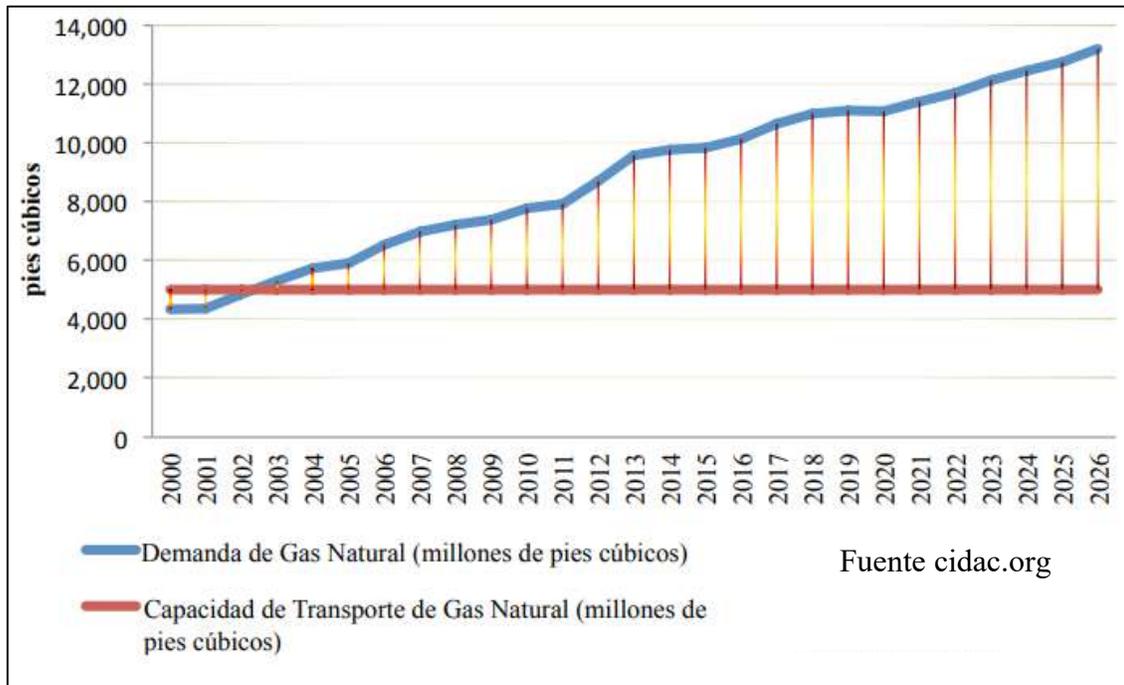


Figura 1.3. La demanda de gas natural no puede ser abastecida.

INTRODUCCIÓN

Ineficiencias de mercado ocasionadas por la simbiosis que prevalece entre el Estado y Petróleos Mexicanos (Pemex) que, por un lado, merman el patrimonio de la paraestatal, y, por el otro, envían señales equivocadas al mercado respecto a los precios de algunos de sus combustibles;

Obstáculos en el sector eléctrico: una baja interconexión geográfica en el Sistema Interconectado Nacional, red de transmisión en proceso de envejecimiento, aumento en pérdidas no técnicas y un Margen de Reserva Operativo decreciente y por debajo de estándares internacionales; e. Subsidios energéticos ineficientes y regresivos con efectos perversos.

Finalmente, una matriz energética altamente dependiente de hidrocarburos susceptibles a fuertes variaciones en precios.

Adicionalmente, México ha procurado el agravamiento de estos problemas por cuatro motivos:

- 1) Mantener una visión de corto plazo que concentra la producción de energía en hidrocarburos y asigna a Pemex el rol de equilibrador fiscal y macroeconómico;
- 2) Promover un portafolio de inversión en hidrocarburos inadecuado que privilegia la maximización de extracción de petróleo en el presente por encima del manejo sustentable de sus pozos;
- 3) No reconocer el costo de oportunidad de hidrocarburos no explotados;
- 4) Actores que asocian la posesión del petróleo a la defensa de la soberanía nacional y, en consecuencia, elevan el costo político de llevar a cabo las reformas que requiere el sector.

Tomando esto en cuenta, el Centro de Investigación para el Desarrollo, A.C. (CIDAC) desarrolló cuatro recomendaciones puntuales para que México pueda alcanzar, en el largo plazo, un abasto de energía suficiente que habilite el crecimiento y desarrollo económico esperado:

INTRODUCCIÓN

1. Generar en la Oficina de la Presidencia de la República, así como en la SENER, la Secretaría de Economía (SE) y la Secretaría de Hacienda y Crédito Público (SHCP), una visión de largo plazo que, por un lado, fomente la sustentabilidad del sector al diversificar la composición de la matriz energética nacional, y, por el otro, permita a Pemex reinvertir su renta petrolera en investigación y desarrollo, así como en un Sistema Nacional de Gasoductos capaz de satisfacer las necesidades productivas del país;
2. Contar con un portafolio de inversión en hidrocarburos adecuado a las fortalezas de exploración y producción de Pemex;
3. Reconocer el costo de oportunidad de los hidrocarburos no explotados; y,
4. Asociar el concepto de soberanía nacional a la seguridad energética, de tal manera que Pemex pueda adquirir la estructura de una empresa eficiente en beneficio del país (Serra, Ch, Lujambio, Toro, & Aguilar, 2013).

Consumo energético en los hogares mexicanos

En su artículo la autora nos da información acerca del consumo de energía en los hogares mexicanos (Sanchez Peña, 2012):

Los datos de la ENIGH muestran que el consumo energético per cápita de los hogares aumentó entre 1992 y 2008 (ver Figura 1.4) obtenida de (Sanchez Peña, 2012), aunque con importantes fluctuaciones a lo largo del periodo. A nivel nacional, el consumo per cápita de los hogares pasó de 2530 en 1992, a 2926 *megajoules* al trimestre en 2008. También se observan importantes diferencias por tipo de localidades: los hogares urbanos consumen poco más del doble que los hogares rurales en 2008 y aunque estos últimos han aumentado su consumo promedio en el tiempo, la brecha es todavía amplia. De hecho, para 2008, los hogares rurales muestran una reducción en su consumo energético en contraste con los hogares urbanos. La gráfica (ver Figura 1.4) también hace evidente el peso que estos últimos tienen sobre las tendencias nacionales, tanto por su peso numérico (representan alrededor de 78% de los hogares del país) como por el nivel de su demanda energética (Sanchez Peña, 2012).

INTRODUCCIÓN

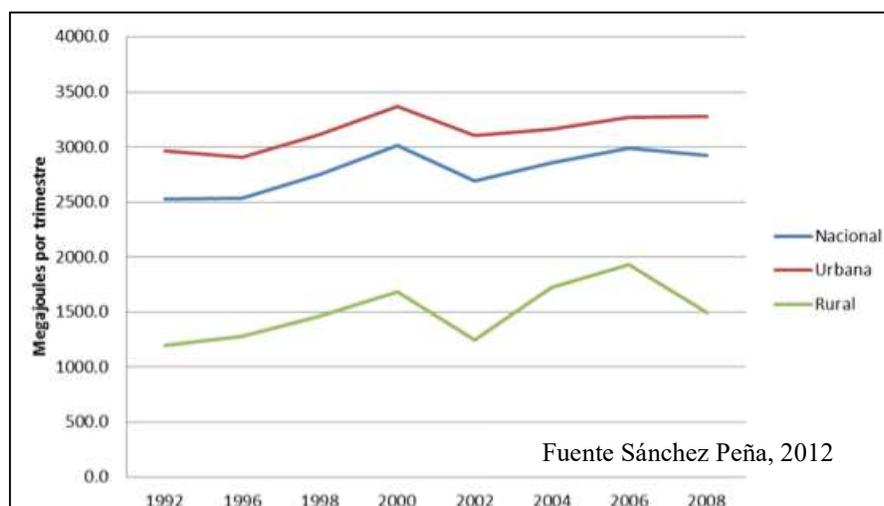


Figura 1.4. Consumo energético per cápita promedio por hogar.

En su boletín electrónico *Brújula de compra* la Profeco menciona:

Según la Encuesta Nacional de Ingresos y Gastos de los Hogares de 2006 los hogares gastan 2.8% de su ingreso en energía eléctrica. Cada mes las familias destinan \$411 en promedio para adquirir electricidad. En la gráfica se observa que el gasto en este energético se incrementa cuando el nivel de ingreso es mayor (ver Figura 1.5) obtenida de (PROFECO, 2007).

La electricidad es uno de los energéticos que más utilizan los hogares mexicanos. De acuerdo con la Comisión Nacional de Ahorro de Energía (CONAE), ésta se consume principalmente para iluminar, hacer funcionar al televisor y refrigerar alimentos, como se observa en la gráfica (ver Figura 1.6) obtenida de (PROFECO, 2007).

Consejos para reducir el desperdicio de energía eléctrica

- Ahorro en iluminación:
 - Utilizar la luz necesaria y apagarla cuando no se necesite.
 - Mantener abiertas las cortinas y persianas durante el día para aprovechar al máximo la luz natural.

INTRODUCCIÓN

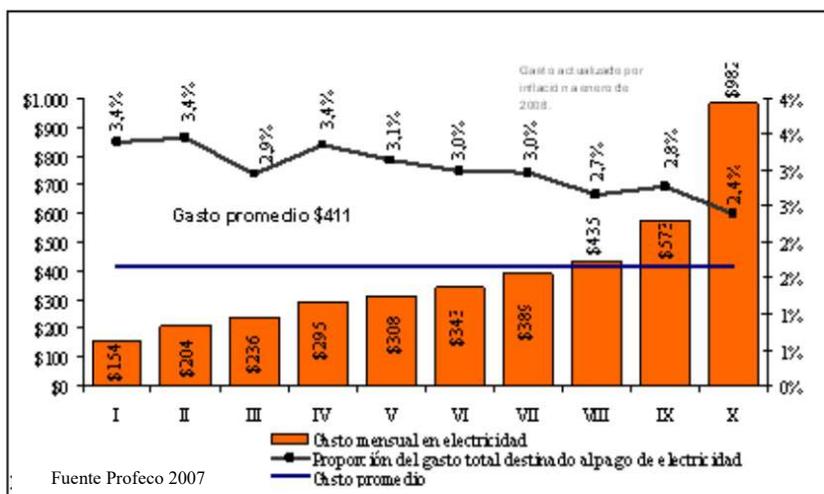


Figura 1.5. Gastos y proporciones en electricidad por decena de habitantes

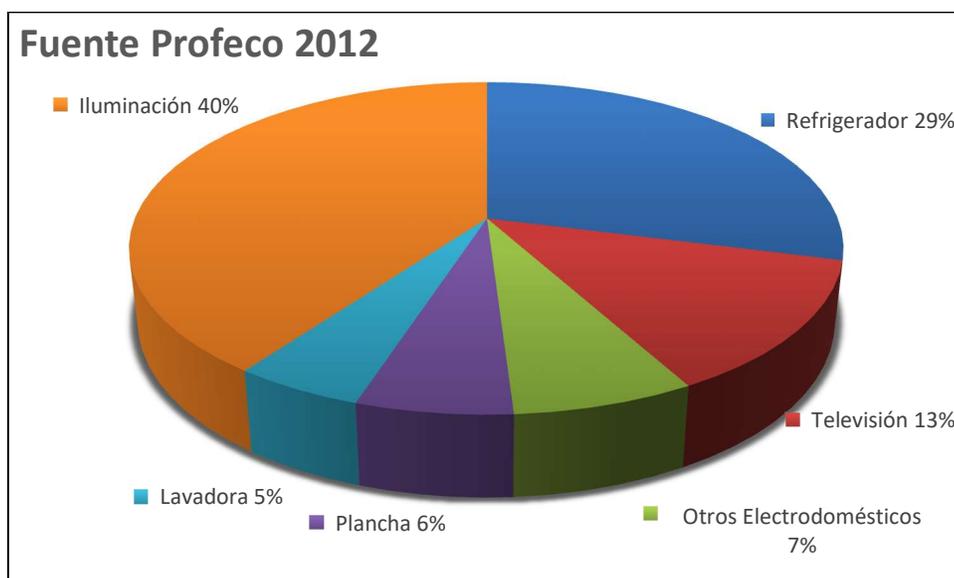


Figura 1.6. Porcentaje de consumo promedio de electricidad en un hogar.

- Pintar las paredes del hogar con colores claros para aprovechar mejor la luz, tanto la natural como la artificial.

INTRODUCCIÓN

- Realizar el mayor número de actividades posibles en el día, como lavar, planchar y el aseo.
- Sustituir focos incandescentes y halógenos por lámparas ahorradoras (fluorescentes compactas), que llegan a consumir cuatro veces menos energía y duran hasta diez veces más.
- Limpiar focos y lámparas para que el polvo no obstruya la emisión de luz.
- Los focos ahorradores
- Seguir las instrucciones de uso de los aparatos electrodomésticos
- Conocer lo que se consume para saber si se está desperdiciando la energía (PROFECO, 2007).

Con base en la información anterior, esta tesis se pretende dotar de inteligencia a un edificio del Instituto Tecnológico de Chihuahua II, utilizando un sistema SCADA (*Supervisory Control and Data Acquisition*) para el monitoreo y control de variables como iluminación, climatización, consumo de agua y energía eléctrica. Con la finalidad de comprobar la hipótesis de que un edificio inteligente es capaz reducir el uso descuidado de los recursos energéticos, se instalaron sensores y elementos de control (relevadores) en el edificio y se interconectaron con protocolos cableados y/o inalámbricos, integrándose al sistema SCADA.

1.3. Alcances y limitaciones

Alcances

- Desarrollar un sistema SCADA.
- Capaz de interactuar con los usuarios y el edificio.
- Automatizar el uso de un edificio.
- Dar la capacidad a un edificio de interactuar con usuarios a través Internet.
- El medio de comunicación será al menos mediante una red cableada y una inalámbrica.

INTRODUCCIÓN

Limitaciones

- Tiempo y mano de obra escaseaban para la realización del proyecto.
- El sistema es una aplicación Web, por lo que el usuario deberá estar conectado a la red para poder acceder.
- Será necesario un análisis de seguridad informática para lograr que la aplicación esté protegida de amenazas.

1.4. Justificación

Actualmente el número de dispositivos que pueden estar conectados a una red de computadoras ha aumentado de una manera impresionante, gracias a la reducción de costos y tamaño de los componentes que permiten dicha funcionalidad, permitiendo la automatización, monitoreo y control de virtualmente cualquier entorno.

Gracias a ello nace el concepto de edificio inteligente, es decir, una red de sensores y actuadores interconectados entre sí en una construcción arquitectónica. La implementación de este tipo de construcciones trae consigo ventajas como las que se mencionan a continuación:

- Uso eficiente de recursos.
- Aumento de la comodidad y seguridad de los usuarios.
- Análisis de información (Agiliza toma de decisiones).
- Capacidad de agregar nuevas funcionalidades.

INTRODUCCIÓN

1.5. Objetivos

Objetivo general

Implementar un sistema SCADA en un edificio inteligente, capaz de controlar una red de sensores y actuadores, para aumentar la seguridad y comodidad de los usuarios, hacer uso más eficiente de los recursos, además de brindar información histórica y/o en tiempo real.

Objetivos específicos

- Instalar sensores y actuadores en un edificio.
- Interconectar los componentes de la red a través de protocolos abiertos.
- Reducir el consumo de electricidad gracias al monitoreo y control proporcionado por el sistema.
- Crear una base de datos para tener información confiable con respecto al uso de los recursos del edificio.

II. ESTADO DEL ARTE

En la sección anterior (introducción) se habló de la problemática en cuanto al uso descuidado de energía eléctrica en que existe México y se describe los objetivos, metas, alcances y limitaciones que tiene la presente tesis.

Este capítulo es una revisión de literatura que le dará al lector un panorama sobre el trabajo que se ha documentado por diversos autores, sobre los temas relacionados con la tesis (automatización de edificios, sistemas SCADA, control de edificios por web).

Para conseguir la información se usó principalmente buscador del CONRICYT, de los artículos encontrados en esta plataforma se seleccionaron seis, los cuales son los más relacionados con el tema de tesis.

2.1. Una revisión sobre sistemas para edificios inteligentes sustentables

A review on optimized control systems for building energy and comfort management of smart sustainable buildings.

Los autores de este artículo (Shaikh, Mohd Nor, Nallagownden, Elamvazuthi, & Ibrahim, 2014) investigaron en diversas fuentes como *Web of Knowledge*, *IEEE explore digital library* y Google Académico, un total de 128 artículos relevantes al tema de SEB (*Smart Energy Buildings*) o edificios de energía inteligente. Los distintos artículos tienen diferentes parámetros de control, algoritmos y métodos con varios esquemas de control. Se ha observado un notable crecimiento en el área de gestión de energía y confort en edificios a lo largo de las últimas décadas, la mayoría de estos estudios tiene como objetivo hacer un uso eficiente de la energía, mientras que los objetivos referentes al confort son diversos (ver Figura 2.1) obtenida de (Shaikh, Mohd Nor, Nallagownden, Elamvazuthi, & Ibrahim, 2014).

ESTADO DEL ARTE

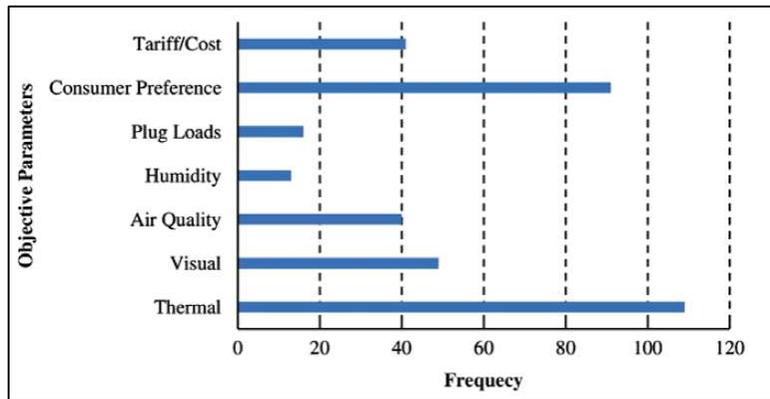


Figura 2.1. Gráfica de frecuencia y objetos de estudio.

Se encontró en dicha literatura que los estudios se llevan a cabo con una gran variedad de sistemas de control, siendo los más utilizados MPC (15%), MAST (14%) y Fuzzy (13%).

De acuerdo con la literatura revisada, Estados Unidos es el país que más publica información con respecto al tema de edificios inteligentes. Varios países en Europa (Suiza, Finlandia, Francia, Grecia, Reino Unido) también muestran gran interés en el tema (ver Figura 2.2) obtenida de (Shaikh, Mohd Nor, Nallagownden, Elamvazuthi, & Ibrahim, 2014).

Los edificios para oficinas son los involucrados en lo que respecta al uso eficiente de energía (ver fFigura 2.3) obtenida de (Shaikh, Mohd Nor, Nallagownden, Elamvazuthi, & Ibrahim, 2014).

ESTADO DEL ARTE

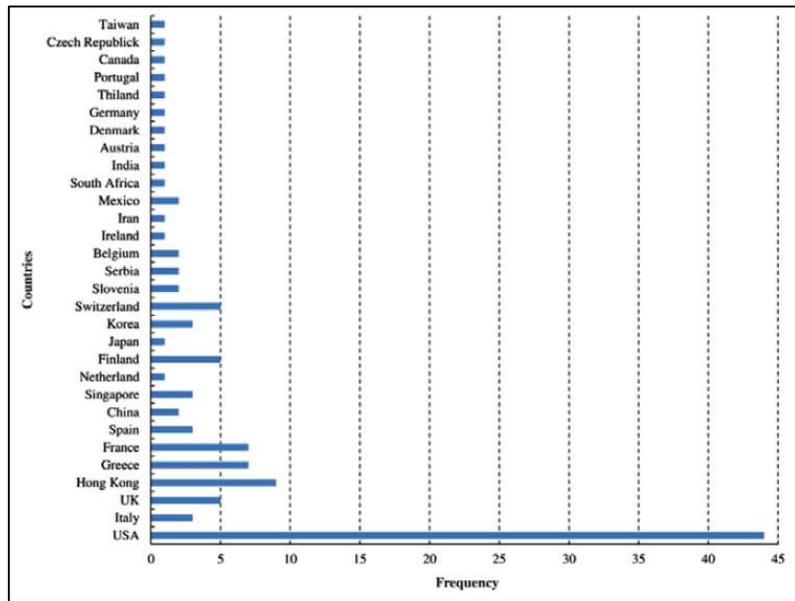


Figura 2.2. Investigaciones realizadas en distintos países.

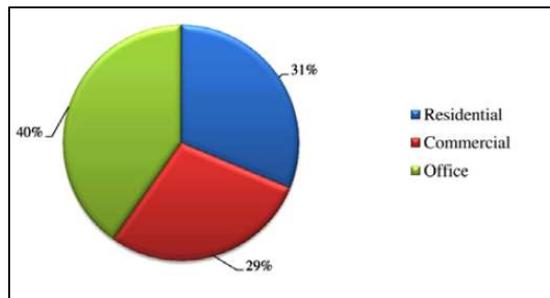


Figura 2.3. Porcentajes según el sector.

Sistema de control en edificios inteligentes

Los sistemas de control de edificios inteligentes (*building control system BCS*, *building automation system BAS*, *building management system BMS*, etc.) son pieza fundamental para el funcionamiento del edificio inteligente. Dichos sistemas generalmente son sistemas centralizados que integran redes de hardware y software a través del edificio.

Normalmente esos controles están de alguna manera conectada a los controles mecánicos, eléctricos y de plomería (MEP).

2.2. Sensores y actuadores inteligentes basado en WSN

WSN-Based Smart Sensors and Actuator for Power Management in Intelligent Buildings.

En dicho artículo (Tebje Kelly, Mukhopadhyay, Suryadevara, & Singh Gill, 2015) se describe un sistema para la medición de parámetros eléctricos de varios electrodomésticos, la Figura 2.4 obtenida de (Tebje Kelly, Mukhopadhyay, Suryadevara, & Singh Gill, 2015), muestra el diseño de la solución propuesta:

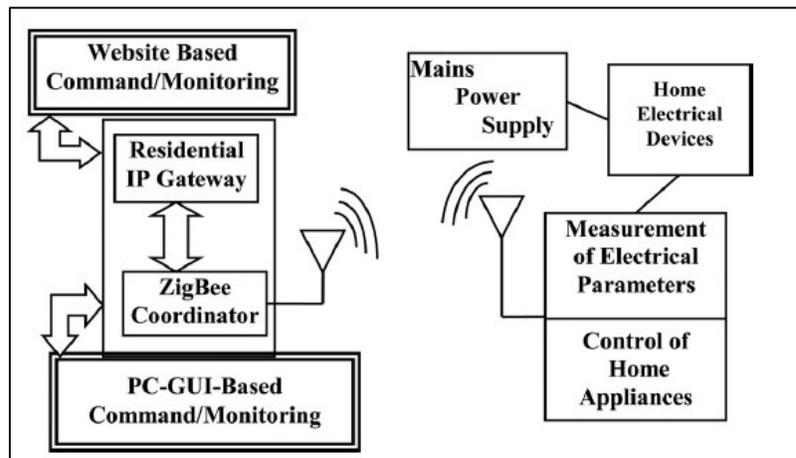


Figura 2.4. Diagrama funcional de bloques del sistema.

Para poder medir la cantidad de energía eléctrica que cada electrodoméstico consume se fabricaron unos dispositivos especiales, que están integrados con un módulo XBee, por medio de cual se interconectan a una red MESH (malla) para poder enviar información al coordinador ZigBee, que está conectado a una PC, en la cual se almacena información en una base de datos que utiliza el gestor MySQL. La información recolectada se envía a un Gateway a través de Internet para el monitoreo y control remoto. La interfaz entre el usuario y el sistema fue desarrollada con el lenguaje C#.

2.3. Diseño de sistema SCADA con HTML5

Design of a HTML5 SCADA System.

En este documento los autores (Maccarana, Cologni, & Previdi, 2016), nos hablan de la implementación de un Sistema SCADA para controlar y monitorear remotamente un PLC de la marca B&R, la arquitectura propuesta se compone de tres capas; cliente, servidor y PLC (ver Figura 2.5) obtenida de (Maccarana, Cologni, & Previdi, 2016).

Las tecnologías involucradas para este desarrollo son:

- Servidor Web IIS Express.
- SDK Unificado de Automatización.
- AJAX y Servicios Web.
- C#, .NET 2012 (lado del servidor).
- Javascript (lado del cliente).
- Servidor OPC-UA para controlar PLC.

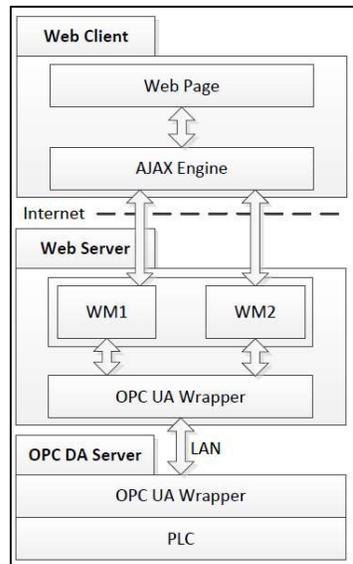


Figura 2.5. Arquitectura de la aplicación.

2.4. Sistema SCADA Completo de Bajo Costo para Casa Inteligente

Complete Low Cost SCADA System of the Intelligent House

En este trabajo el autor (Krejcar, 2012) describe el desarrollo de un sistema SCADA completo utilizado en ambientes de viviendas inteligentes (domótica) haciendo especial énfasis en la reducción del costo del mismo.

2.4.1. Características

El SCADA se implementó en una PC, dicho sistema fue desarrollado en C# y con AtouchX (Para controlar PLC), tiene una interfaz gráfica amigable (ver Figura 2.6) obtenida (Krejcar, 2012) y es capaz de procesar información en tiempo real.

El PLC es el encargado de controlar y monitorear los sensores y actuadores del sistema (hasta el momento cuando se publicó el artículo controlaban la calefacción y aire acondicionado), además de reportar información y recibir instrucciones del administrador del sistema, el cual se comunica únicamente con la PC.



Figura 2.6. Sistema SCADA plano de visualización.

2.5. IoT (internet de las cosas) basado en micro-servicio para edificios inteligentes

Microservice-based IoT for Smart Buildings

En el presente trabajo (Salikhov, Khanda, Gusmanov, Mazzara, & Mavridis, 2016) se explica cómo los autores implementaron un sistema de domótica sencillo para el monitoreo de tres habitaciones en la universidad de Innopolis (Federación Rusa).

La topología del sistema consta de computadora con z-stack (Software para interconexión con Zigbee), un sensor para puertas, una cámara web y una Raspberry Pi la cual se conecta vía Bluetooth a un SensorTag (ver Figura 2.7) obtenida de (Salikhov, Khanda, Gusmanov, Mazzara, & Mavridis, 2016).

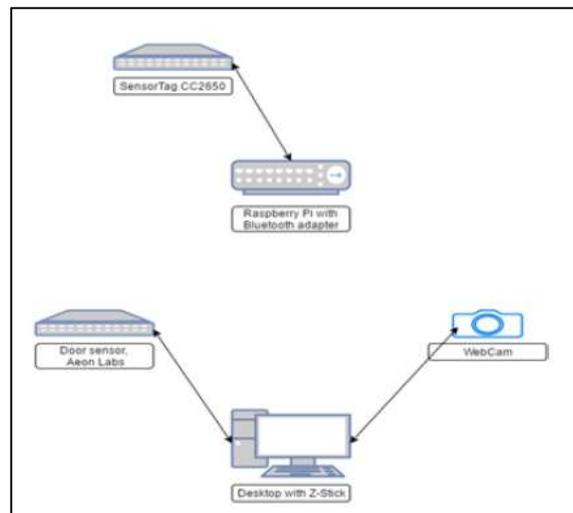


Figura 2.7. Dispositivos y sus conexiones.

Se utilizó lenguajes de alto nivel, como son Java y Jolie, además se trabajó con un sistema operativo beta llamado HomeOS, el cual está especializado para desarrollo de sistemas domóticos.

Gracias esta sencilla implementación se pudo recolectar información como; temperatura de las habitaciones, humedad, luz, presión. La información de la temperatura en el exterior se obtuvo de Internet, también se obtiene información acerca de la ocupación del laboratorio debido al uso del sensor para puertas instalado allí.

2.6. Construyendo Servicios de IoT para adultos mayores

Building IoT Services for Aging in Place Using Standard-Based IoT Platforms and Heterogeneous IoT Products

Con el objetivo de mejorar la calidad de vida de las personas de la tercera edad los autores (Fattah, Sung, Ahn, Ryu, & Yun, 2017), desarrollaron el proyecto para permitir a los adultos mayores vivir en la comodidad de sus propias casas reduciendo el costo de cuidado, esta tecnología permite el cuidado remoto y/o automático.

En el artículo se presenta un caso práctico del uso de los servicios creados en el proyecto, además de una herramienta permite a los usuarios crear nuevos según los requiera.

Servicio implementado

Los escenarios para el servicio de medicación definido por los autores se muestran en la siguiente tabla (ver Tabla 2.1) obtenida de (Fattah, Sung, Ahn, Ryu, & Yun, 2017).

Tabla 2.1. Escenarios de servicio.

Scenario 1	Scenario 2	Scenario 3	Scenario 4
A senior brings a pillbox having the specific amount of drugs from a pharmacy	The senior comes back home and put the pillbox on its container	The senior can get medication directives and schedule via his/her smartphone app	Medication reminder service will alert the senior at the right time for taking medication
Requirements	Requirements	Requirements	Requirements
<ul style="list-style-type: none"> Information about pharmacy, pharmacist, and prescription is stored in the pillbox 	<ul style="list-style-type: none"> The pillbox holder reads the stored information from the pillbox 	<ul style="list-style-type: none"> The smart app can retrieve all the information about medication routine from AIP platform 	<ul style="list-style-type: none"> If suggested usage is 'taking with food', monitor when having food and give the senior proper alarms
<ul style="list-style-type: none"> Medication directives and schedule is also stored in the pillbox 	<ul style="list-style-type: none"> The collected information will be sent to the AIP platform 	<ul style="list-style-type: none"> 	<ul style="list-style-type: none"> Give the senior alarms in appropriate ways according to locations: smartphone, light, speaker, wrist band

Proceso

1. Los medicamentos entregados a la persona de la tercera edad llevan pegada una etiqueta NFC la cual debe ser escaneada por un Smartphone en la farmacia, usando la aplicación definida por los autores para cargar la información de paciente, la información del medicamento, los horarios y dosis.
2. Cuando llega a casa el adulto mayor coloca las píldoras en la caja de pastillas (*pillbox holder*), automáticamente el *Raspberry PI 3*[®] lee la etiqueta NFC y carga la información.
3. Gracias la información proporcionada por la caja de medicamentos y la aplicación de la farmacia, el sistema es capaz de recordar al adulto mayor en que momento debe consumir los medicamentos (se utiliza sonido, luz y vibraciones). Dicho sistema también cuenta con sensores ultrasónicos para detectar donde se encuentra la persona y sensores en la arrocera para detectar si ha comido y de este modo poder calcular la dosis de medicamento suministrada.

El diagrama de bloques (*overall diagram*) se presenta a continuación (ver Figura 2.8) (Fattah, Sung, Ahn, Ryu, & Yun, 2017).

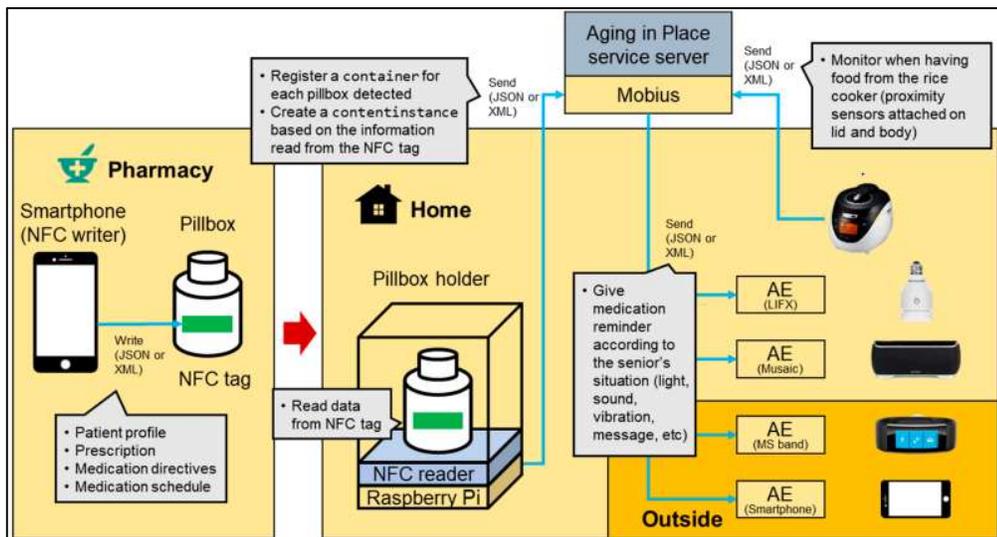


Figura 2.8. Diagrama de bloques del servicio.

2.7. Conclusión del estado de al arte

Como ya lo vimos en este capítulo es muy amplia la cantidad de soluciones a las que se puede llegar mediante la aplicación de la domótica/inmótica, para conseguir la automatización de un edificio.

Las soluciones tienen diversos objetivos, tales como: la reducción de costos, aumento del confort, control de aire acondicionado, monitoreo de temperatura, control de iluminación. La mayoría de la información recabada proviene de Estados Unidos y se puede apreciar que la automatización de edificios es una tendencia en países de primer mundo. El uso que se da a los edificios de los estudios es principalmente empresarial (oficinas), seguido del residencial y por último el comercial.

Un punto muy importante es el uso de sistema de sistemas informáticos para el control del edificio, siendo principalmente los sistemas de tipo SCADA (*Supervisory Control and Data Acquisition*) y BMS (*Building Management System*) los más elegidos para dicha tarea.

Se utilizan diversas tecnologías para la transmisión de datos, siendo las más destacadas: Zigbee, Wifi, Ethernet y RFID. Se emplean principalmente equipos (hardware) como: *Raspberry Pi*®, teléfonos inteligentes, *Arduinos*®, *PLC*® y computadoras de escritorio.

Todos los estudios coinciden en que es necesario una interfaz para que el usuario pueda interactuar con el sistema de control, ya sea una aplicación de escritorio, una app móvil, una web o mediante comandos de voz.

III. MARCO TEÓRICO

3.1. Domótica

La domótica es un concepto relativamente nuevo, “Los diccionarios franceses incorporaron el término domotique a partir de 1998. Esta palabra se introdujo en España por los Pirineos como Domótica, que procede del latín domus (casa, domicilio) y del griego αὐτόματοϛ, automática (aunque existen autores que opinan que deriva de informática, como defiende el Diccionario de la RAE, o incluso de robótica)” (Domínguez & Sáez Vacas, 2006). En otras palabras la domótica, la automatización de la vivienda, es responsable con el medio ambiente, teniendo en cuenta los siguientes aspectos (García-S. Matachana & Lopez Peñalver, 2004):

- Aumentar la comodidad (confort) de los habitantes de la vivienda.
- Incrementar y mejorar las comunicaciones.
- Facilitar el control de la casa.
- Gestionar la energía, controlando su gasto y ahorrando dinero.
- Ofrecer nuevos servicios.

MARCO TEÓRICO

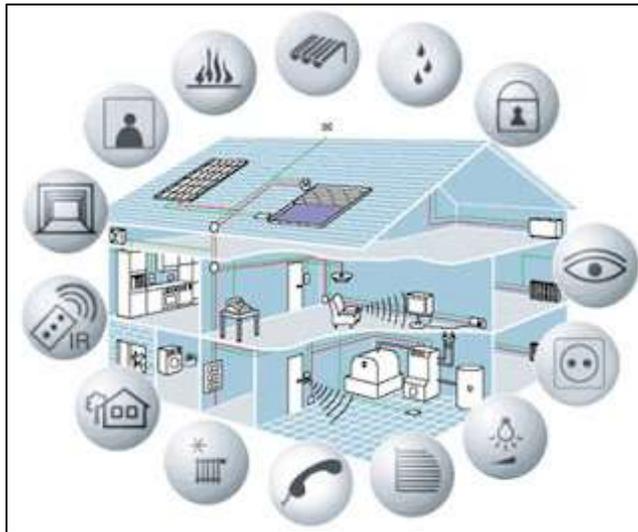


Figura 3.1. Muestra de domótica aplicada.

Como se muestra en el diagrama (ver Figura 3.1) se pueden integrar servicios como el teléfono, la vigilancia, la iluminación, entrenamiento entre otros, a través de protocolos cableados o inalámbricos.

3.2. Inmótica

A diferencia de la domótica, es la automatización de edificios no destinados a la vivienda (hoteles, centros comerciales, escuelas, universidades, hospitales, etc.), es decir, la automatización de edificios donde existe un mayor consumo de recursos y donde la convivencia de personas es mucho mayor que en una vivienda familiar típica.

Además de los objetivos de la domótica, la inmótica agrega otros más específicos enfocados a aumentar la calidad del trabajo:

- Ahorro de trabajo; simplificación del trabajo.
- Facilidad de uso.
- Control de tiempo.
- Control Remoto de los dispositivos.
- Reducción de ruido/Ahorro de energía.
- Impacto Ambiental.

MARCO TEÓRICO

- Reducción de Costos.

3.4. Edificio inteligente

Un edificio inteligente es un conjunto de sensores y actuadores interconectados en la estructura de alguna construcción arquitectónica con la finalidad de hacer un uso más eficiente de los recursos, aumentar la comodidad de los usuarios, mejor mejorar la seguridad y agregar nuevas funcionalidades.

En este tipo de edificaciones (ver Figura 3.3) pueden converger muchos servicios, generalmente conectados a través de la red de computadoras local (LAN).

3.5. Componentes para la automatización de un edificio

3.5.1. Sensores

Los sensores son dispositivos capaces de convertir una magnitud de un tipo a otro, por ejemplo, magnitud térmica a magnitud eléctrica. Estas magnitudes pueden ser físicas, químicas, biológicas, entre otras. Generalmente en un edificio inteligente cuentan con sensores para monitorear la temperatura, humedad, presencia, iluminación, etc.

Un ejemplo de sensor es el PIR (*Passive infrared sensor*), el cual es utilizado para detectar el movimiento de presencia (ver Figura 3.2).

Los sensores se pueden clasificar de la siguiente manera:

- Activos: Deben ser alimentados eléctricamente, son más habituales.
- Pasivos: No necesitan alimentación eléctrica.

MARCO TEÓRICO



Figura 3.2. Modulo detector de movimiento PIR sensor infrarrojo pasivo HCSR501

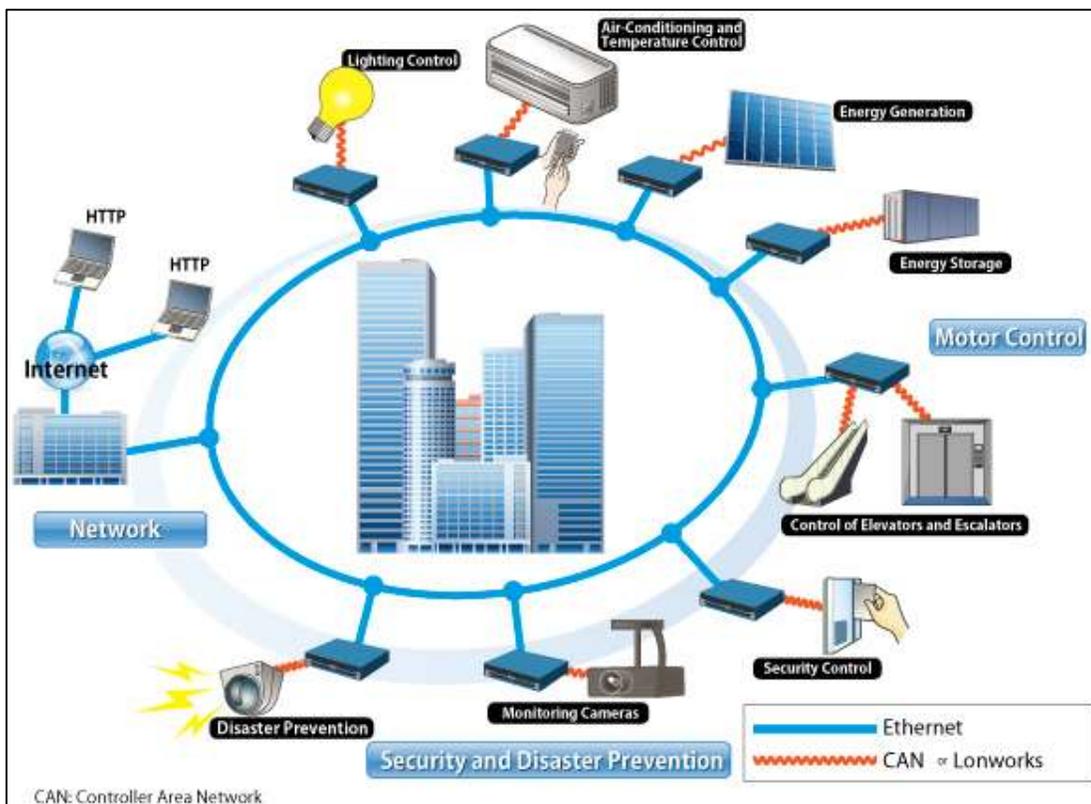


Figura 3.3. Ejemplo de topología de red de edificio inteligente.

También se clasifica por tipo de señal:

- Continuos: su señal está disponibles en cualquier instante
- Discretos: su señal está disponibles en ciertos tiempos.

Sensores según la magnitud medida:

MARCO TEÓRICO

- Detectores de humo.
- Detector de gas.
- Detector de monóxido de carbono.
- Detectores de presencia.
- Detectores magnéticos para puertas o ventanas.
- Detectores de rotura de cristal.
- Barrera de infrarrojos.
- Sensores de luminosidad.
- Sonda y detector de inundación.
- Sensor de temperatura.
- Sensor de viento.
- Sensor de lluvia.

3.5.2. Acondicionadores de señal

Las señales que envían los sensores generalmente deben ser acondicionadas para que el sistema o controlador que recibe sea capaz de procesarla (ver Figura 3.4).

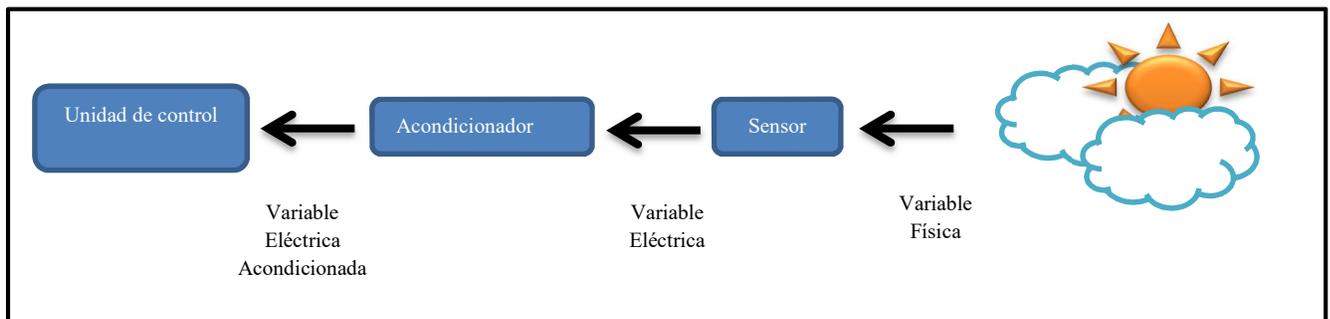


Figura 3.4. Acondionamiento de señal.

3.5.3. Actuadores

Son dispositivos electromecánicos que interactúan y afectan físicamente el edificio, es decir, convierten una magnitud eléctrica en una de otro tipo (mecánica, térmica, etc.). Existen actuadores con niveles de salida continuos o discretos. Algunos ejemplos de actuadores son; motor de persianas, lámparas y sirenas.

MARCO TEÓRICO

Entre los actuadores y el dispositivo que los controla se encuentran las interfaces, cuya función es acondicionar la señal de entrada del actuador (ver Figura 3.5).

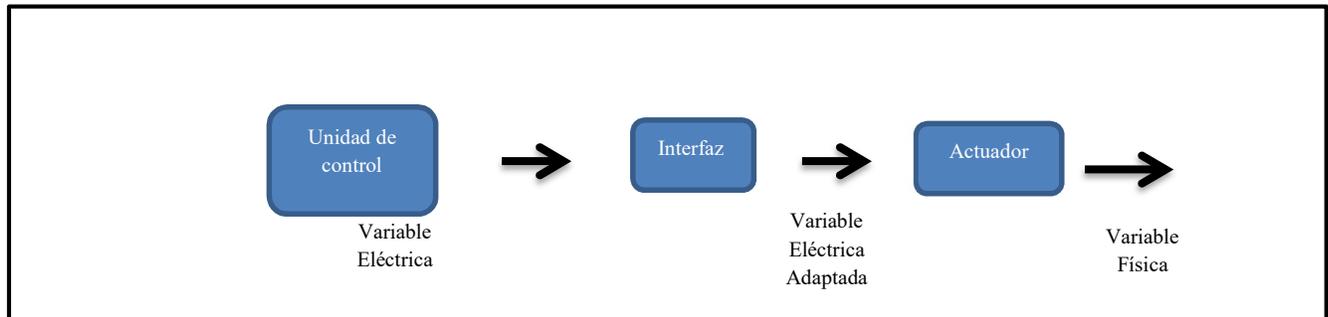


Figura 3.5. Acondicionamiento de señal de la interfaz

3.5.4. Relés o relevadores

Dispositivos electromagnéticos cuya función principal es la de abrir o cerrar circuitos eléctricos independientes, están compuestos por un electroimán y una bobina. Se puede considerar un amplificador de señal eléctrica gracias a su capacidad de controlar un circuito de mayor potencia de salida que el de su entrada, un ejemplo de estos dispositivos muestra en la Figura 3.6.

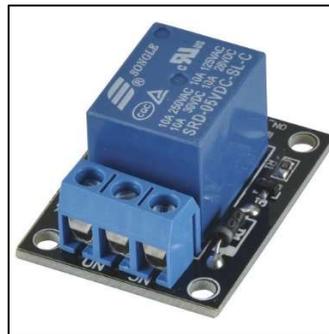


Figura 3.6. Relé de 5V modelo XC4419.

3.5.5. Contactores

También conocidos como relés de potencia, permiten el flujo de más o menos corriente.

MARCO TEÓRICO

3.5.6. Regulador o *dimmers*

Permiten regular la potencia que llega una carga, un ejemplo de aplicación de estos actuadores en domótica es la regulación de luminosidad en las lámparas.

3.6. SCADA

Por sus siglas en inglés *Supervisory Control and Data Acquisition*,

- Es un sistema informático capaz de monitorear (gracias al uso de sensores).
- Controlar (uso de actuadores).
- Almacenar información (uso de bases de datos).
- En un proceso en el mundo real.
- Generalmente los SCADA son usados en procesos industriales, como fundición de acero y generación de energía (Daneels & Salter, 1999).

3.7. *Mean stack*

El *mean stack* es un conjunto de tecnologías web que permiten desarrollar aplicaciones web de principio a fin, sin la necesidad de contar con un experto en *front-end* (interfaz de usuario), un experto en *back-end* (lado del servidor) y un experto en bases de datos, ya que todas las tecnologías que componen al *stack* son similares y se trabaja bajo el mismo lenguaje de programación.

El *stack* está compuesto cuatro tecnologías:

- MongoDB: El manejador de bases de datos.
- Express: El entorno de trabajo web de lado del servidor (*back-end*).
- Angular JS: El entorno de trabajo de la interfaz (*front-end*).
- Node.js: El servidor web. (Holmes & Harber, 2017).

3.7.1. MongoDB

“MongoDB es una base de datos orientada a documentos. Esto quiere decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON”.

Ventajas de MongoDB:

- Documentos JSON: Ya que se trabaja con JavaScript no se debe tratar la información antes de cargarla a la base de datos, esto ayuda a aumentar el rendimiento de la aplicación en general.
- Velocidad: MongoDB está diseñado para trabajar consumiendo muy pocos recursos por lo que en una máquina de gran capacidad lo hace muy rápido.
- Volumen: Puede manejar una gran cantidad de información simultáneamente.
- Variabilidad: los documentos en NoSQL no deben de cumplir con una estructura como lo hacen los registros en las tablas de SQL, es decir, los datos registrados por la aplicación pueden registrarse sin que tengan que agregar campos o hacer cualquier modificación a la base de datos. (Genbeta, 2018).

3.7.2. Express

Express es el framework web más popular de Node, y es la librería subyacente para un gran número de otros frameworks web de Node populares. Proporciona mecanismos para:

- Escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes caminos URL (rutas).
- Integración con motores de renderización de "vistas" para generar respuestas mediante la introducción de datos en plantillas.
- Establecer ajustes de aplicaciones web, como qué puerto usar para conectar, y la localización de las plantillas que se utilizan para renderizar la respuesta.
- Añadir procesamiento de peticiones "middleware" adicional en cualquier punto dentro de la tubería de manejo de la petición.

MARCO TEÓRICO

- A pesar de que Express es en sí mismo bastante minimalista, los desarrolladores han creado paquetes de middleware compatibles para abordar casi cualquier problema de desarrollo web. Hay librerías para trabajar con cookies, sesiones, inicios de sesión de usuario, parámetros URL, datos POST, cabeceras de seguridad y muchos más. Puedes encontrar una lista de paquetes middleware mantenida por el equipo de Express en Express Middleware (junto con una lista de algunos de los paquetes más populares de terceros). (MDN web docs, 2018).

3.7.3. AngularJS

“AngularJS (comúnmente llamado Angular.js o AngularJS 1), es un framework de JavaScript de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles “ (Wikipedia, 2018).

IV. DESARROLLO

En esta sección se definen los materiales y métodos utilizados para el desarrollo de la aplicación. Este proceso se llevó a cabo haciendo uso la metodología de cascada.

4.1. Análisis

La aplicación será implementada en el edificio de posgrado del Instituto Tecnológico de Chihuahua II, por lo que la etapa de análisis se llevó a cabo en este mismo lugar. La información recabada se consiguió mediante entrevistas y observación de los procesos actuales.

Gracias a la información recabada se especificaron las siguientes características con las que debe cumplir el sistema:

- Debe ser fácil de usar.
- Debe ser multiplataforma.
- Debe ser económico, ya que no se cuenta con mucho presupuesto para la compra de equipo y/o licencias.
- Debe responder de manera ágil.

4.1.1. Análisis de los requisitos

Este paso es de vital importancia para el éxito de proyecto, ya que es aquí donde se analizan las necesidades de los usuarios finales para definir las características con las que debe cumplir la aplicación.

La información recabada en esta fase será de gran importancia para las fases de diseño y codificación.

DESARROLLO

Requisitos funcionales (RF)

- El sistema web debe registrar el estado de cada componente (actuador o sensor).
- La interfaz web debe reflejar los cambios del sistema de manera inmediata.
- Se podrá agregar información al catálogo de componentes para garantizar la escalabilidad del sistema.
- La aplicación web debe capturar los comandos generados por el usuario y enviarlos a los componentes
- Los componentes deben enviar sus estados a la aplicación web para que esta actualice la interfaz
- El usuario podrá revisar la información histórica del sistema mediante reportes.

Requisitos no funcionales (RNF)

- El usuario podrá acceder al sistema desde cualquier dispositivo con navegador web
- La interfaz web debe ser intuitiva y fácil de usar para el usuario.
- El sistema deberá responder de manera ágil

4.1.2. Diagramas de caso de uso

Los diagramas de uso nos muestran la relación entre actores, y los casos de uso del sistema representan la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. A continuación, se detalla la interacción que tendría el usuario con la aplicación.

Caso de uso general del sistema

Como se puede ver en caso de uso (ver Figura 4.1), el usuario podrá interactuar de cuatro maneras con el sistema:

1. **Ingresar al sistema.** Para ello será necesario ingresar un nombre de usuario y una contraseña, con la finalidad de garantizar la seguridad del sistema.

DESARROLLO

2. **Monitorear y controlar el edificio.** Esta es la principal función del sistema, desde este módulo el usuario podrá accionar los actuadores y monitorear los sensores que se encuentren distribuidos por el edificio.
3. **Generar reportes.** El usuario podrá generar reportes con la información que se encuentra en la base de datos del sistema.
4. **Consultar y editar catálogos del sistema.** Para darle escalabilidad al sistema, el usuario podrá agregar nuevos componentes y/o usuarios.

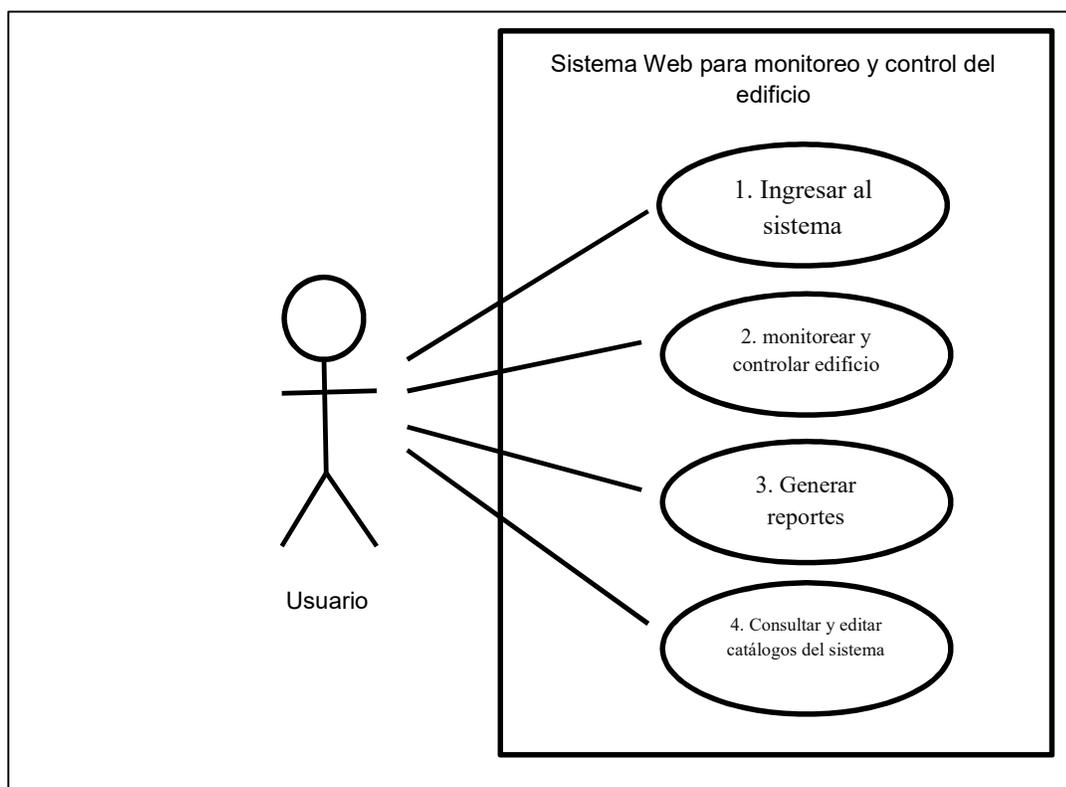


Figura 4.1. Caso de uso principal.

4.2. Diseño

En esta etapa se toman decisiones sobre la organización lógica del software, de tal manera que se cumpla con los requerimientos definidos en la etapa de análisis. En esta etapa es donde se definen las herramientas de software que se usarán en la fase de codificación.

4.2.1. Diseño arquitectónico

Como se muestra en el siguiente diagrama (ver figura 4.2), la estructura del proyecto se divide en cinco elementos fundamentales:

Red de domótica: es un conjunto de sensores, actuadores y microcontroladores interconectados entre sí con el propósito de recabar información y ejecutar los comandos que son enviados desde otros componentes.

Gateway: debido a sus diferentes interfaces de interconexión es capaz de enviar información desde el servidor a la red de domótica y viceversa, además es un dispositivo con un sistema operativo de alto nivel, el cual le permite ejecutar software más complejo que el que ejecuta la red de domótica.

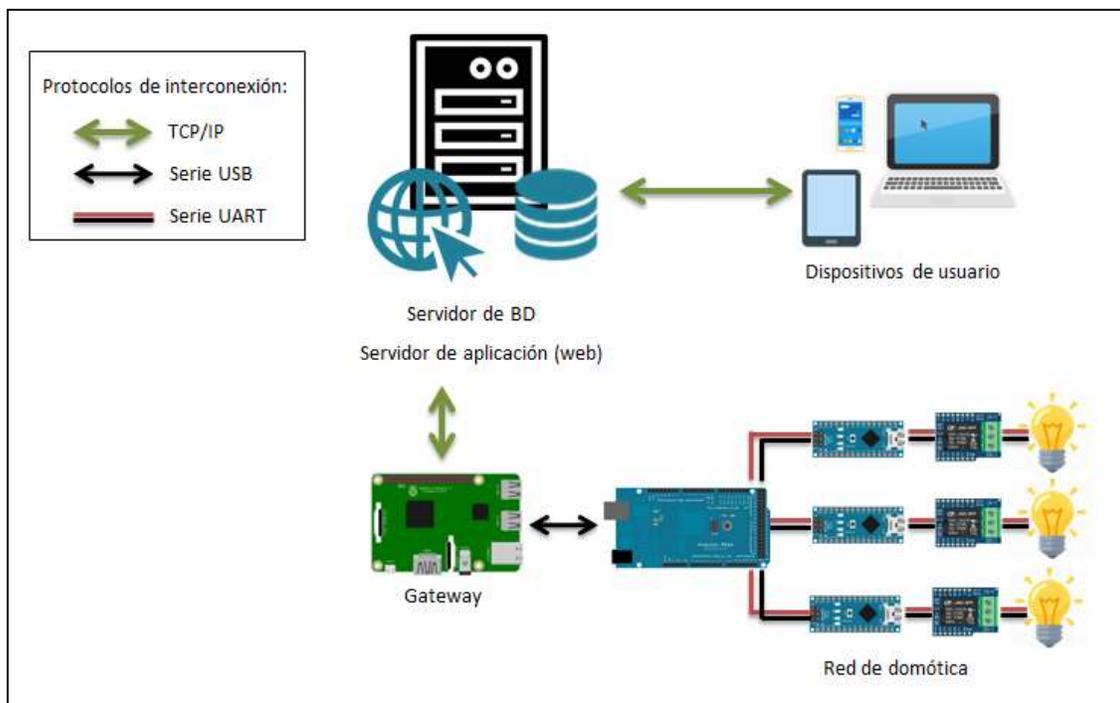


Figura 4.2. Diseño arquitectónico del sistema.

Servidor de Base de Datos y Servidor Web: es un equipo de cómputo con menos interfaces de red que el Gateway, pero con mayor poder de procesamiento y almacenamiento, también

DESARROLLO

cuenta con un sistema operativo de alto nivel, en él se hospedan el software servidor de base de datos (BD) y el software Servidor Web.

Dispositivos de usuario: puede ser cualquier equipo capaz de conectarse a la red mediante un navegador web, como lo son: computadoras personales, teléfonos inteligentes, tabletas, televisiones inteligentes, etc.

Protocolos de interconexión: para conectar todo el proyecto se utilizan tres protocolos:

- **El protocolo TCP/IP.** El cual permite la comunicación con el servidor mediante la web.
- **El protocolo serie USB.** El cual permite la comunicación desde el Gateway a la red de domótica y viceversa.
- **Protocolo serie UART.** El cual permite la comunicación entre los dispositivos dentro de la red de domótica.

4.2.2. Definición de protocolo de comunicación

Dirección

Para identificar los microcontroladores (MCU) esclavos, los cuales son encargados de ejecutar los comandos que envían los usuarios se definió un protocolo sencillo de direccionamiento para evitar confusiones entre los mismos.

- La dirección del MCU esclavo cuenta con las siguientes características:
- Longitud de dos dígitos decimales, ejemplo; 01, 04, 10.
- No debe repetirse en toda la red, sin importar que los MCU esclavos estén conectados a distintos MCU maestro.

Comando

Cuando el usuario cambie el estado de un componente en la interfaz web (cambiar de encendido a apagado o viceversa), se generará un comando el cual cuenta con las siguientes características:

DESARROLLO

- Longitud de seis caracteres, cada uno un número decimal.
- Los dos primeros caracteres corresponden a la dirección del MCU esclavo que ejecutará el comando.
- Los caracteres tres y cuatro corresponden al sub sistema al que van dirigidos, por ejemplo, iluminación.
- Los caracteres quinto y sexto corresponde al circuito al que va dirigido el comando dentro del MCU (ya que un MCU puede tener varios circuitos del mismo tipo de subsistema, por ejemplo, un MCU esclavo con dos circuitos de iluminación).
- Los caracteres séptimo y octavo corresponden a la instrucción que se deberá ejecutar, por ejemplo, encender o apagar.

A continuación, se muestra el ejemplo de la estructura de un comando (ver fFigura 4.3).

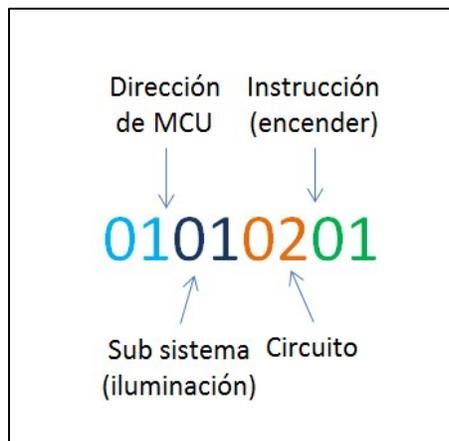


Figura 4.3. Ejemplo de comando

Proceso de emisión / recepción de comandos

1. El usuario acciona un control dentro de la interfaz web.
2. El servidor genera el comando con las características correspondientes.
3. El servidor envía el comando al Gateway.
4. El Gateway envía el comando a los MCU maestros.
5. Los MCU maestros envían el comando a los MCU esclavos.
6. Los MCU esclavos comparan su dirección con la que está incluida en comando

DESARROLLO

7. Si las direcciones son iguales se ejecuta el comando y se envía la respuesta al Gateway, en caso contrario solo se ignora el comando.
8. El Gateway envía la respuesta al servidor.
9. El servidor actualiza la interfaz web con la información recibida.

4.2.3. Diseño de navegación

Es importante delimitar la información que se encuentra disponible en el sistema, por lo que se debe separar en páginas según su uso para no saturar al usuario y/o confundirlo, en el presente desarrollo se definieron las páginas de la siguiente manera (ver Figura 4.4):

- Tablero.
- Usuarios.
- Componentes.
- Reportes.
- Cerrar sesión.

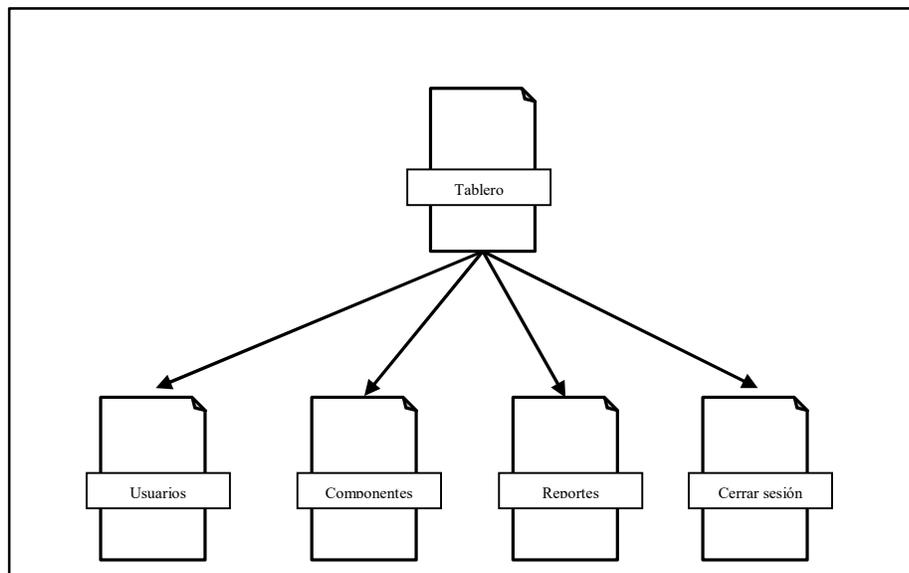


Figura 4.4. Estructura de navegación de sitio web.

DESARROLLO

4.2.4. Diseño de la interfaz

La interfaz web es una página interactiva con una imagen de un mapa, la cual cuenta con botones que representan los subsistemas dentro de la red de domótica, los cuales serán activados por el usuario y esto generará un comando en el sistema (ver Figura 4.5).

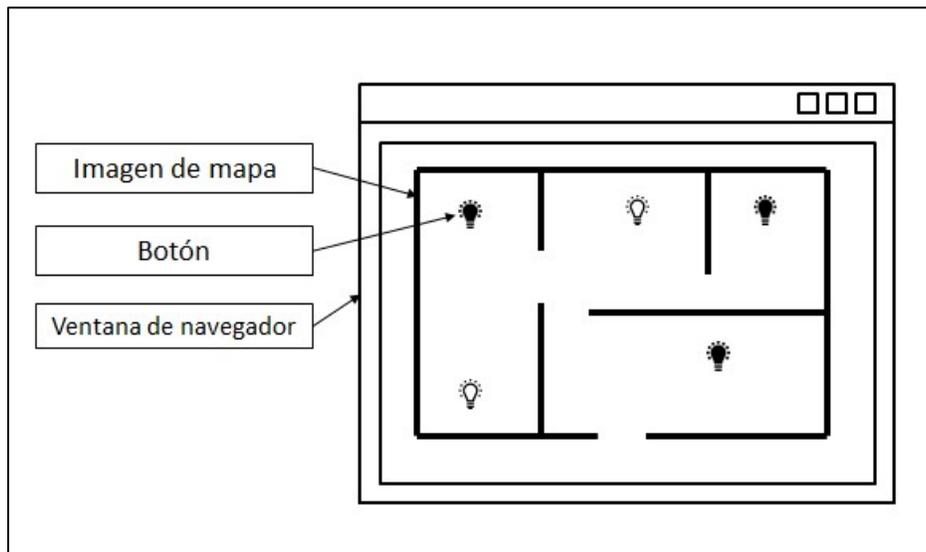


Figura 4.5. Diseño de interfaz gráfica.

4.2.5. Diseño de base de datos

Se tienen planeadas las siguientes 5 tablas para almacenar la información referente al sistema, se evitó el uso de la tilde en algunas palabras para prevenir problemas de compatibilidad:

- **Componente:** Contiene los datos de los sensores y actuadores del edificio.
- **Ubicacion:** Contiene los datos de las ubicaciones de los sensores y actuadores.
- **Piso:** contiene la información referente a los pisos del edificio.
- **Usuario:** Cuenta con la información de los usuarios que interactúan con el sistema.
- **Transacción:** Cuenta con la información referente a los registros del sistema, cada vez que se ejecuta un comando los datos son almacenados en esta tabla.

A continuación, se muestra el diagrama de entidad relación con las tablas del sistema y sus respectivos campos, además de las relaciones entre las mismas (ver Figura 4.6).

DESARROLLO

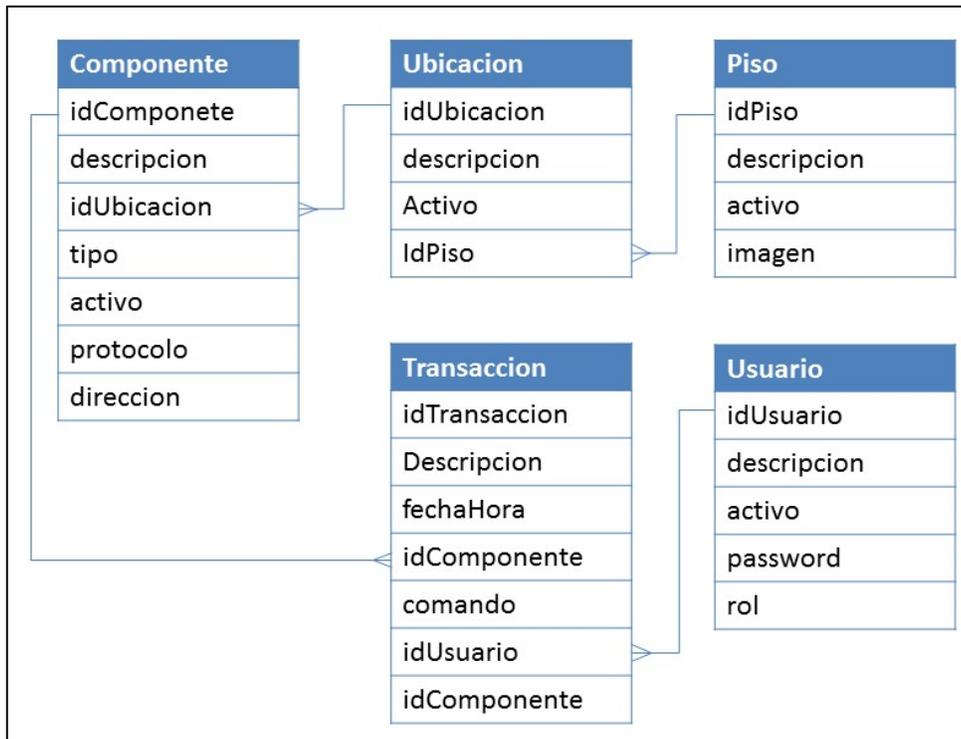


Figura 4.6. Diagrama entidad relación de la base de datos.

4.2.6. Características de hardware y software para desarrollo

El servidor web y de base de datos deberá contar con las siguientes características (ver Tabla 4.1):

Tabla 4.1. Características de servidor.

Hardware		Software	
Procesador	Intel i5	Sistema Operativo	Windows 10
Memoria RAM	8 GB	Servidor Web	Node.js
Conexión de red	Ethernet	Gestor de base de	Mongo DB
Almacenamiento	500 GB	datos	

DESARROLLO

El Gateway debe contar con las siguientes características (ver Tabla 4.2):

Tabla 4.2. Características de Gateway.

Hardware		Software	
Procesador	Quad Core 1.2GHz	Sistema Operativo	Linux Raspbian
Memoria RAM	1 GB	Servidor Web	Node.js
Conexión de red	Ethernet, WiFi y GPIO		
Almacenamiento	4 GB		

Los microcontroladores maestros deben contar con las siguientes características (ver Tabla 4.3):

Tabla 4.3. Características de MCU maestro.

Hardware		Software	
Procesador	ATmega2560	Programación	Lenguaje C en arduino IDE
Memoria FLASH	256k		
Conexión de red	UART conexión serie		
Entradas o Salidas digitales	54 (14 de ellos son salidas PWM).		
Canales análogos	16		

DESARROLLO

Los microcontroladores esclavos deben contar con las siguientes características (ver Tabla 4.4):

Tabla 4.4. Características de MCU esclavos.

Hardware		Software	
Procesador	ATMega328	Programación	Lenguaje C en arduino IDE
Memoria EEPROM	1KB		
Conexión de red	GPIO		
Entradas o Salidas digitales	14		
Canales análogos	6		

Los

dispositivos cliente deben contar con las siguientes características (ver Tabla 4.5):

Tabla 4.5. Características de clientes.

Hardware		Software	
Procesador	1.2 Ghz	Sistema Operativo	Windows 7 o Android
Memoria RAM	2 GB	Navegador Web	Google Chrome
Conexión de red	Ethernet o WiFi		
Almacenamiento	1 GB		

4.3. Codificación

Al igual que el diseño de código se divide en 3 partes (red de domótica, Gateway y servidor).

4.3.1. Codificación de la red de domótica

Los microcontroladores se programaron con Arduino IDE en lenguaje C. La red de domótica trabaja bajo el modelo de maestro-esclavo, es decir, un microcontrolador maestro envía comandos y coordina la comunicación con el resto de la red. Mientras que los microcontroladores esclavos son los encargados de ejecutar los comandos y enviar las respuestas al maestro. A continuación, se muestra el diagrama de flujo del funcionamiento de programa codificado para el funcionamiento los MCU maestros en la red de domótica (ver Figura 4.7), el código fuente se encuentra en el anexos 1 de este documento.

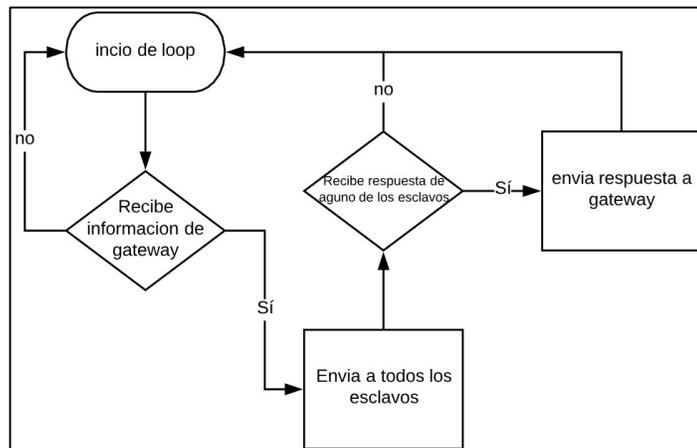


Figura 4.7. Diagrama de flujo del funcionamiento del MCU maestro.

DESARROLLO

El diagrama de flujo que explica el funcionamiento de los MCU esclavos se muestra a continuación (ver Figura 4.8), el código fuente se encuentra en el anexos 2 de este documento.

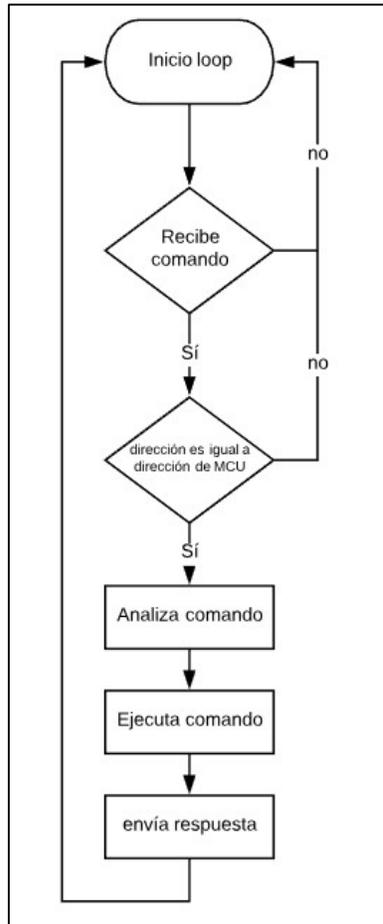


Figura 4.8. Diagrama de flujo del funcionamiento del MCU esclavo.

DESARROLLO

Toda la red está conectada de manera cableada, como se muestra en la Figura 4.9.

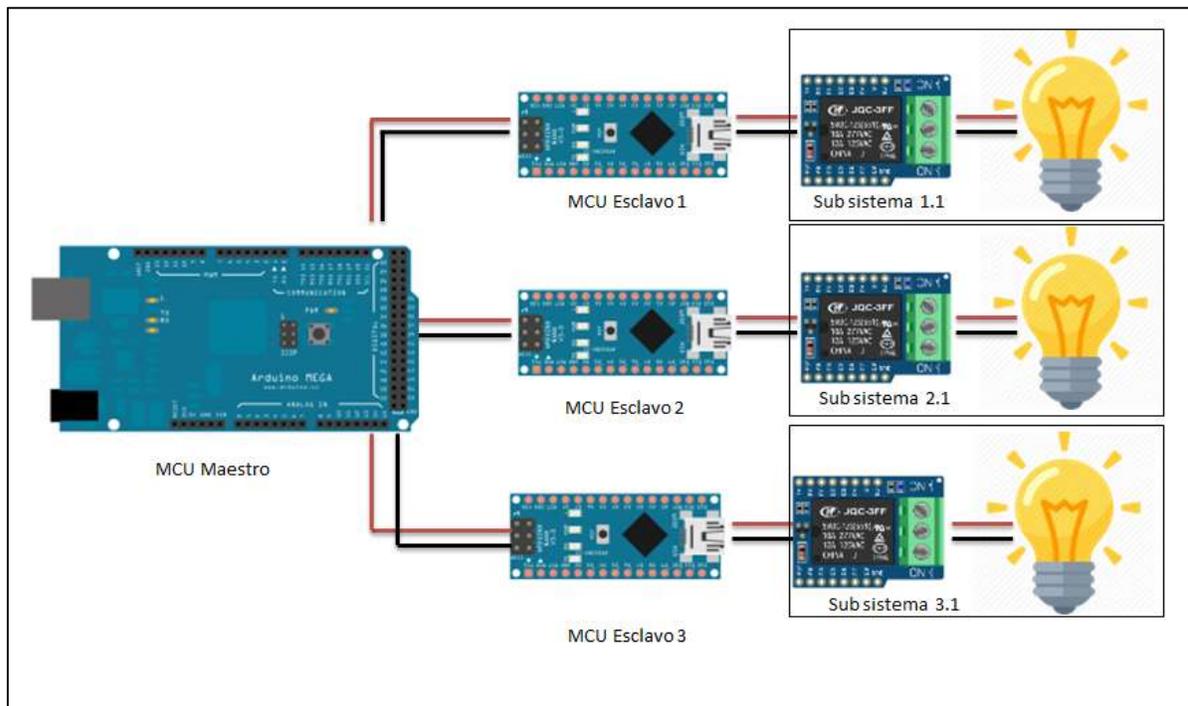


Figura 4.9. Topología de red de domótica.

El maestro envía los comandos a todos los esclavos mediante un broadcast y ellos comparan la dirección del comando con la propia para saber si deben, o no, ejecutar el comando. El microcontrolador maestro es el encargado de enviar y recibir la información al Gateway. A continuación, se muestra un diagrama de interacción entre los elementos de la red de domótica (ver Figura 4.10).

DESARROLLO

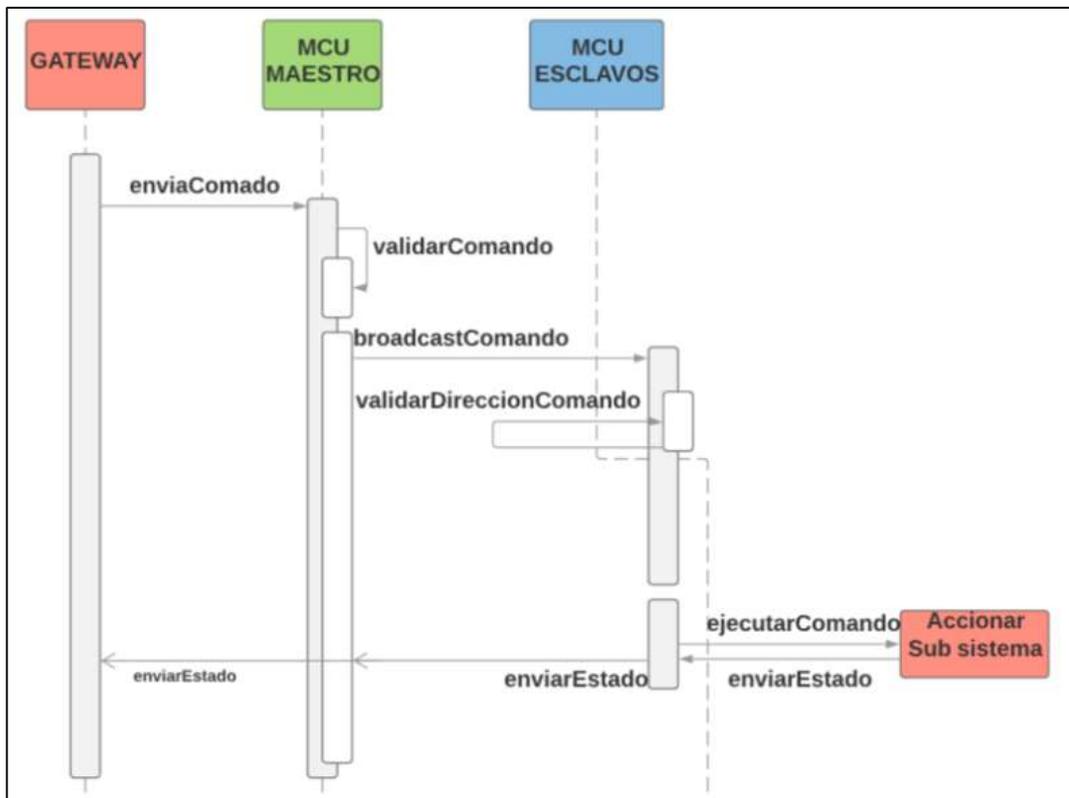


Figura 4.10. Diagrama de interacción entre los elementos de la red de domótica.

4.3.2. Codificación del software para el Gateway

Se programó en JavaScript gracias a la instalación de un servidor node.js y los módulos socket.io (para comunicación con el servidor) y Serialport (para comunicación con la red de domótica). Su función principal es la de transformar los datos recibidos en el protocolo web socket y empaquetarlos en el protocolo serie que se definió previamente. El comportamiento del Gateway está definido por el siguiente algoritmo (ver Figura 4.11).

DESARROLLO

```
//Primera Parte
Si ConectarSocket() Entonces
    "Conexión exitosa"
Si no
    "Error de conexión con servidor"
//Segunda Parte
EscucharSocket("Comando"){
    transmitirSerie("Comando"){
        ConectarPuerto()
        EscribirEnPuerto("Comando")
    }
}
//Tercera Parte
escucharSerie(){
    Si leerSerie() = ComandoValido Entonces
        TransmitirSocket("Comando")
    }
}
```

Figura 4.11. Pseudocódigo de Gateway.

DESARROLLO

En la primera parte del pseudocódigo el programa únicamente establece la conexión con el servidor de sockets y lo notifica en la consola. En la segunda parte el programa está escuchando si el servidor transmite un comando, en caso de ser así, el Gateway lo empaqueta y lo envía a la red de domótica mediante el protocolo serie. En la tercera parte el programa está esperando que la red de domótica envíe una respuesta para el servidor, en caso de ser así, esta será empaquetada y enviada al servidor mediante los web sockets.

A continuación, se muestra el diagrama de flujo del funcionamiento del programa que ejecuta el Gateway (ver Figura 4.12), el código fuente se encuentra en el anexos 3 de este documento.

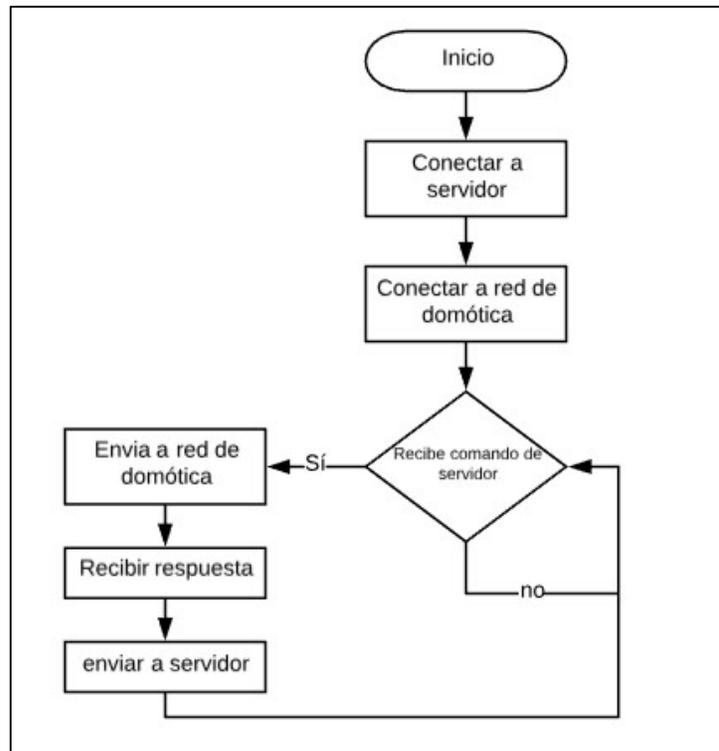


Figura 4.12. Diagrama de flujo de funcionamiento de Gateway.

DESARROLLO

4.3.3. Codificación del servidor

El servidor web corre sobre un servidor de node.js (en lenguaje JavaScript) y el manejador de base de datos es un servidor mongoDB. El servidor web tiene alojado una interfaz web con la cual el usuario interactúa para enviar los comandos, como la que se describe en el apartado de diseño. Una vez capturado el comando el servidor transmite el comando mediante los web sockets, el cual será escuchado por el Gateway. Cuando el servidor tenga respuesta del Gateway, se actualizará la interfaz web con los estados correspondientes. Los comandos y los estados también serán almacenados en la base de datos para su posterior consulta.

A continuación, se muestra el diagrama de flujo que explica el funcionamiento del software del servidor (ver Figura 4.13), el código fuente se encuentra en el anexo 4 de este documento.

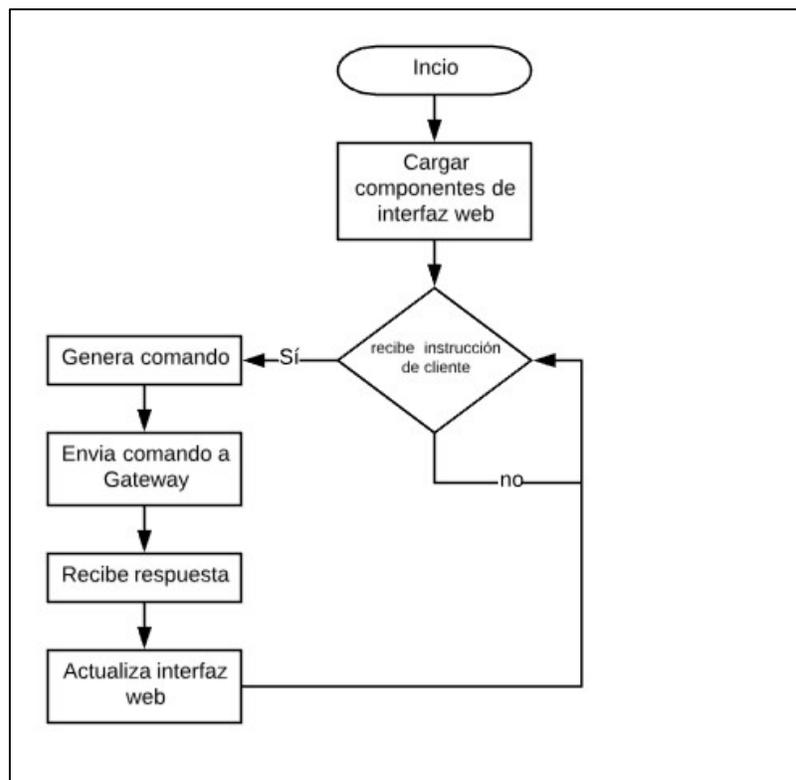


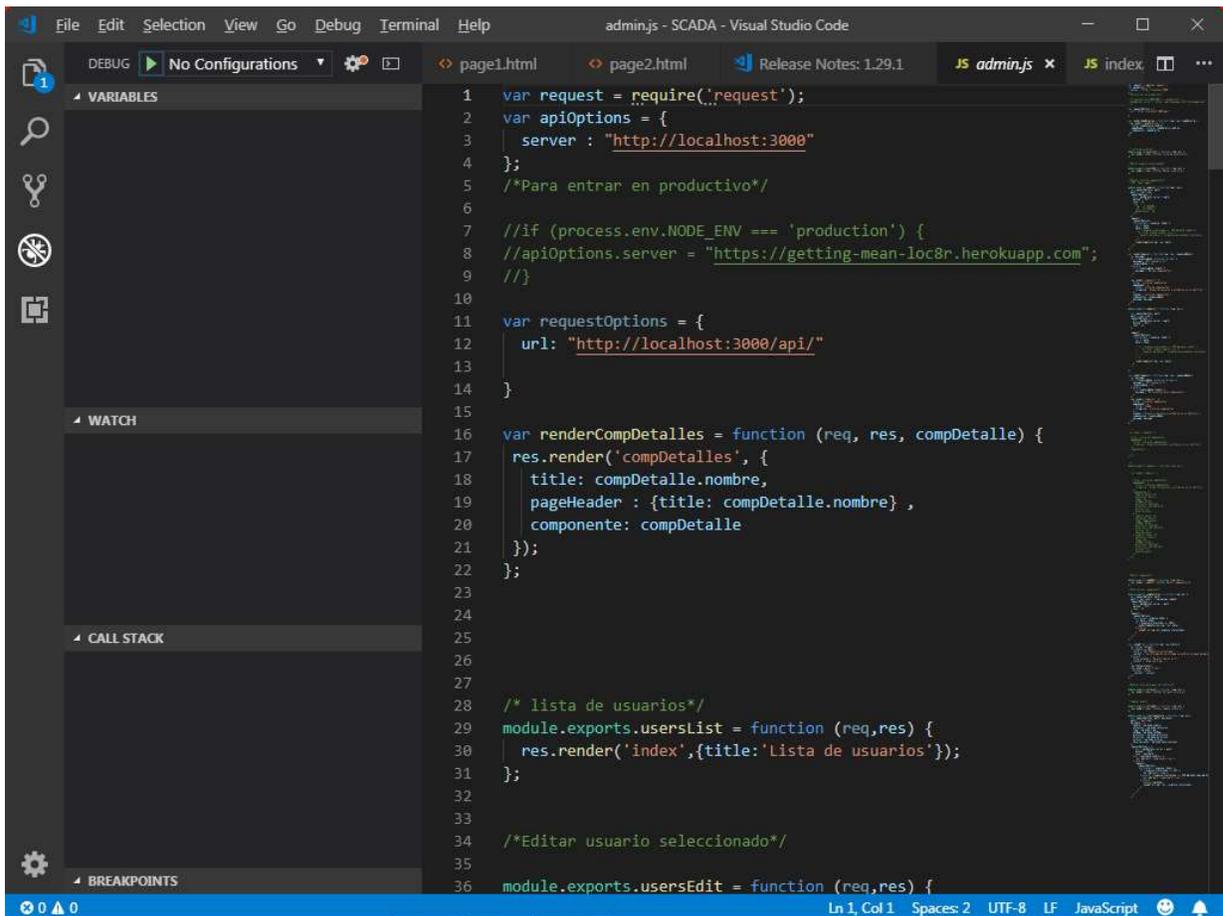
Figura 4.13. Diagrama de flujo de funcionamiento del servidor.

DESARROLLO

4.4. Pruebas

Para hacer las pruebas de funcionamiento del software, se utilizaron las herramientas Visual Studio Code© y el Arduino IDE©.

Visual Studio Code se utilizó para depurar el código del servidor y del Gateway (ver Figura 4.14). Se utilizó el módulo de depuración para inspeccionar el código un corregir algunos errores en la interfaz de usuario.

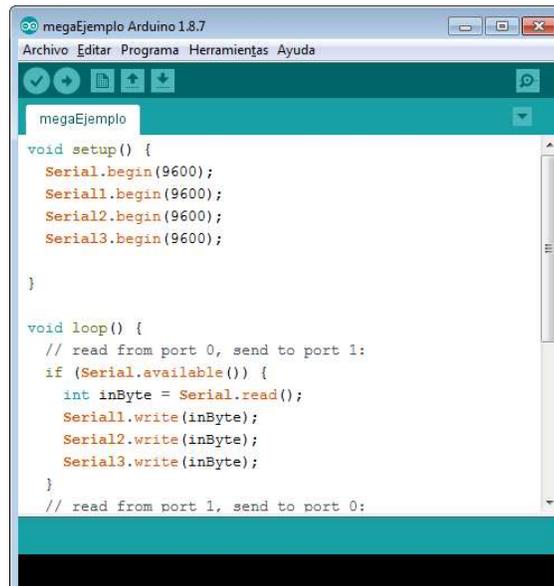


```
admin.js - SCADA - Visual Studio Code
DEBUG No Configurations
page1.html page2.html Release Notes: 1.29.1 JS admin.js JS index
VARIABLES
WATCH
CALL STACK
BREAKPOINTS
1 var request = require('request');
2 var apiOptions = {
3   server : "http://localhost:3000"
4 };
5 /*Para entrar en productivo*/
6
7 //if (process.env.NODE_ENV === 'production') {
8 //apiOptions.server = "https://getting-mean-loc8r.herokuapp.com";
9 //}
10
11 var requestOptions = {
12   url: "http://localhost:3000/api/"
13 }
14 }
15
16 var renderCompDetalles = function (req, res, compDetalle) {
17   res.render('compDetalles', {
18     title: compDetalle.nombre,
19     pageHeader : {title: compDetalle.nombre} ,
20     componente: compDetalle
21   });
22 };
23
24
25
26
27
28 /* lista de usuarios*/
29 module.exports.usersList = function (req,res) {
30   res.render('index',{title:'Lista de usuarios'});
31 };
32
33
34 /*Editar usuario seleccionado*/
35
36 module.exports.usersEdit = function (req,res) {
```

Figura 4.14. Aplicación Visual Studio Code.

DESARROLLO

Mientras que arduino IDE fue empleado para depurar el código de la red de domótica (ver Figura 4.15), gracias a ello se logró mejorar los tiempos de respuesta de la aplicación

The image shows a screenshot of the Arduino IDE window titled 'megaEjemplo Arduino 1.8.7'. The menu bar includes 'Archivo', 'Editar', 'Programa', 'Herramientas', and 'Ayuda'. The toolbar contains icons for opening files, saving, and running. The main editor area shows the following code:

```
void setup() {  
  Serial.begin(9600);  
  Serial1.begin(9600);  
  Serial2.begin(9600);  
  Serial3.begin(9600);  
}  
  
void loop() {  
  // read from port 0, send to port 1:  
  if (Serial.available()) {  
    int inByte = Serial.read();  
    Serial1.write(inByte);  
    Serial2.write(inByte);  
    Serial3.write(inByte);  
  }  
  // read from port 1, send to port 0:
```

Figura 4.15. Aplicación Arduino IDE.

4.5. Implementación

El primer uso práctico del sistema solo utiliza el subsistema de iluminación. Dicho subsistema se encarga de controlar las luces de una sala mediante el uso de sensores (fotorresistencia) y actuadores (relevadores). Como se mencionó en el diseño, la interfaz web consta de un mapa del edificio y los botones que representan los sensores y actuadores (ver Figura 4.19). Haciendo clic sobre las luces en la pantalla, el usuario será capaz de accionar los relevadores y los MCU esclavos enviarán la información de la fotorresistencia al servidor, para que este actualice la interfaz web.

El subsistema de iluminación cuenta con las siguientes características:

- **MCU esclavos** los cuales se encargan de controlar al relevador e interpretar las señales de la fotorresistencia.

DESARROLLO

- **MCU maestro** es el encargado de la comunicación entre el Gateway y los esclavos
- **Fotorresistencia**, sensa la luz ambiente en la habitación y con esta información se determina si la lámpara está encendida o apagada.
- **Switch escalera**, se cambian los switches simples por los switches escalera o de dos vías para que los usuarios del edificio puedan encender las luces a través del interruptor de pared (switch escalera) o a través de la red.
- **Relevador**, dada su similitud de funcionamiento (un tiro dos polos) con el switch de dos vías es la contraparte de este y funciona como el actuador que enciende o apaga la luz por medio de la red domótica.

En la Figura 4.16 se muestra como se conectó el circuito

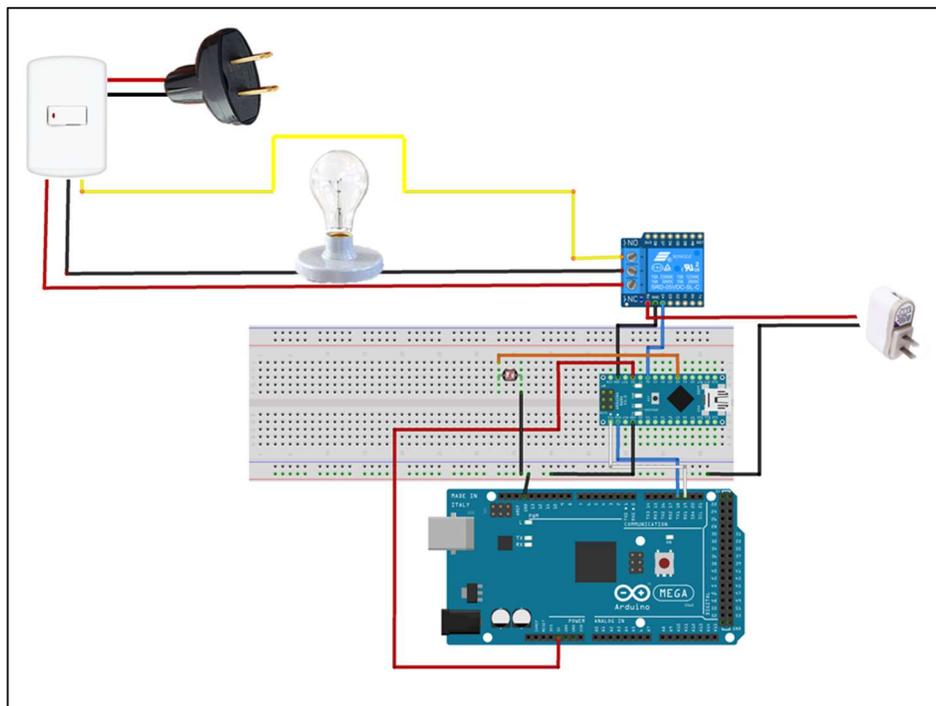


Figura 4.16. Diagrama de conexión de subsistema de iluminación.

DESARROLLO

En la figura Figura 4.17 se muestra la conexión del subsistema en un ambiente de pruebas y en la figura Figura 4.18 se muestra el mismo subsistema pero con la luz encendida.

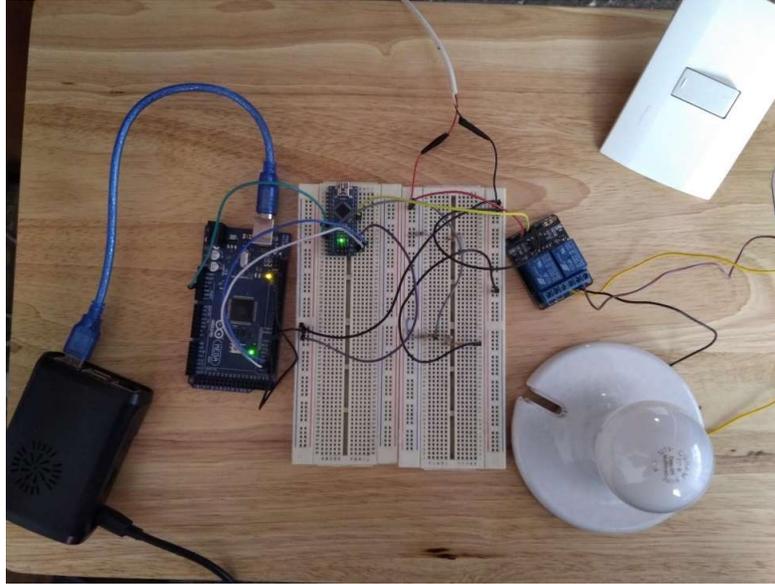


Figura 4.17. Conexión de circuito en un ambiente de pruebas.

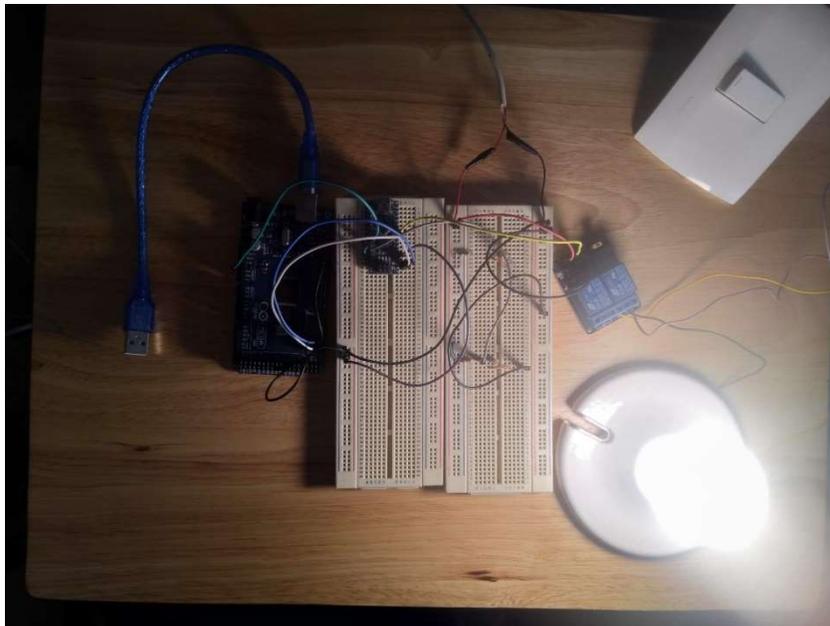


Figura 4.18. Subsistema de iluminación con luz encendida.

RESULTADOS Y DISCUSIÓN

V. RESULTADOS Y DISCUSIÓN

En este capítulo se analizan y se discuten los resultados obtenidos después de haber terminado la investigación.

Antes del desarrollo del sistema SCADA el Instituto tecnológico de chihuahua II no tenía manera de medir y/o controlar el desperdicio de recursos energéticos en ninguno de sus edificios, por lo que la implementación del sistema en el edificio de posgrado representa una ventaja para combatir el problema antes mencionado.

Gracias al sistema es posible controlar y monitorear las luces de edificio y controlarlas remotamente. Lo cual ayuda a los administrativos del edificio a tomar decisiones sobre las medidas preventivas y correctivas sobre el desperdicio energético.

Para lograr los objetivos del proyecto fue necesario hacer una investigación sobre las diversas tecnologías disponibles y determinar cuáles son las más adecuadas para el desarrollo del mismo, como resultado se dedujo lo siguiente:

- El tipo de sistema más apropiado es el de tiempo real debido a que se interactúa con el mundo real y está distribuido en varios dispositivos, por lo que es importante que todos los dispositivos vean las actualizaciones al momento de que estas son efectuadas, garantizando la congruencia de los datos.
- El sistema se desarrolló usando tecnologías web para permitir que sea compatible con el mayor número de dispositivos, sin importar el sistema operativo, el tamaño de la pantalla. Otra ventaja de los sistemas web es la fácil actualización gracias que todo el software está en el servidor.
- El sistema usa protocolos de comunicación abiertos, gracias a ello la compatibilidad de hardware (sensores y actuadores) es mayor. El uso de protocolos propietarios limita al

RESULTADOS Y DISCUSIÓN

sistema en base a las decisiones comerciales del fabricante, para evitar este riesgo se evitó el uso de estos últimos protocolos.

- Para hacer más eficiente la comunicación entre los dispositivos se desarrolló un protocolo sencillo el cual permite hacer el envío y recepción de la información de una manera eficiente y ordenada. Dicho protocolo está preparado para ser compatible con la escalabilidad del sistema.

Para probar el funcionamiento del sistema se agregó el subsistema de iluminación el cual permite encender o apagar las luces del edificio a través de la interfaz web y también permite hacer uso de los apagadores convencionales para hacer más fácil la transición de los usuarios.

Una de las características más interesantes del sistema es su escalabilidad, gracias que se emplea una base de datos no relacional (MongoDB) la cual permite agregar documentos totalmente diferentes sin necesidad de modificar las estructuras de la base de datos.

CONCLUSIONES

VI. CONCLUSIONES

Actualmente existen varias tecnologías para interactuar con edificios inteligentes, aunque la mayoría de ellas no son compatibles unas con otras, por lo que resulta muy complicado interconectar un edificio completo.

Se desarrolló una aplicación multiplataforma gracias al uso de las tecnologías web, lo cual permite a los usuarios conectarse desde equipos móviles como tabletas electrónicas o teléfonos inteligentes, equipos de cómputo de escritorio o portátiles. Gracias a las tecnologías usadas para el desarrollo de las aplicaciones será muy sencillo desarrollar interfaces para otros dispositivos como pantallas inteligentes.

La codificación del software para los micro procesadores maestros y esclavos para la red de domótica se realizó haciendo uso de Software y Hardware libre, lo cual facilita el mantenimiento correctivo y preventivo.

Se desarrolló un programa en JavaScript para el control del Gateway y se programó el software para el control del servidor web.

- Dicha aplicación es capaz de contactarse a una gran diversidad de sensores y actuadores, gracias al uso de protocolos de comunicación abiertos.
- Se desarrolló un protocolo de comunicación sencillo pero potente capaz de interconectar a una gran cantidad de sensores y actuadores en una misma red de domótica.
- Se desarrolló una interfaz de usuario responsiva por lo que es útil tanto en dispositivos móviles como en computadoras personales.

Tanto el desarrollo como la implementación del sistema SCADA fueron muy importantes en mi desarrollo académico y profesional, ya que para cumplir con los objetivos planteados fue necesario

CONCLUSIONES

estudiar más acerca de la programación de diferentes tecnologías web y sus protocolos de comunicación, así como el manejo de bases de datos no relacionales por medio de protocolos web. También fue necesario aprender e investigar sobre los diferentes plataformas de sistemas embebidos, de los cuales al final se optó por Raspberry Pi debido a que es parte de la comunidad de software libre.

Gracias al desarrollo de este proyecto, ahora el Tecnológico de Chihuahua II tiene un sistema de control de edificios que no está casado con los componentes de ningún fabricante, por lo que se pueden usar sensores y actuadores de cualquier costo y además de la posibilidad de incluir diversos protocolos libres.

Los resultados obtenidos en la presente tesis servirán como base para trabajos futuros, como lo son:

- Análisis de seguridad completo para descartar vulnerabilidades y reforzar las debilidades que se puedan encontrar.
- Agregar subsistemas y con ellos más protocolos, sensores y actuadores.
- Mejorar la interfaz de usuario para que pueda visualizarse en pantallas de gran tamaño como lo son las Smart TV.

BIBLIOGRAFÍA

VII. BIBLIOGRAFÍA

- Domínguez, H. M., & Sáez Vacas, F. (2006). *Domótica: Un enfoque sociotécnico*. Madrid: Fundación Rogelio Segovia para el Desarrollo de las Telecomunicaciones.
- Fattah, M. M., Sung, N.-M., Ahn, I.-Y., Ryu, M., & Yun, J. (2017). Building IoT Services for Aging in Place Using Standard-Based IoT Platforms and Heterogeneous IoT Products. *Sensors*, 1-29.
- García-S. Matachana, E., & Lopez Peñalver, F. J. (2004). EL ESTADO DEL ARTE DE LA TECNOLOGÍA AL SERVICIO DE LA CONSTRUCCIÓN. *Informes de la Construcción*, Vol. 56, n° 494, 19-26.
- Genbeta. (15 de Enero de 2018). *MongoDB: qué es, cómo funciona y cuándo podemos usarlo (o no)*. Obtenido de Genbeta: <https://www.genbeta.com/desarrollo/mongodb-que-es-como-funciona-y-cuando-podemos-usarlo-o-no>
- Holmes , S. D., & Harber, C. (2017). *Getting MEAN with Mongo, Express, Angular, and Node, Second Edition*. Nueva York: Manning.
- Krejcar, O. (2012). Complete Low Cost SCADA System of the. *10th IEEE Jubilee International Symposium on Applied Machine Intelligence and Informatics*, 339-343.
- Maccarana, Y., Cologni, A., & Previdi, F. (2016). *Design of a HTML5 SCADA System*. Bergamo: IEEE.
- MDN web docs. (25 de Agosto de 2018). Obtenido de Introducción a Express/Node: https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction
- PROFECO. (2007). *Brújula de compra*. Ciudad de México: PROFECO.

BIBLIOGRAFÍA

- Salikhov, D., Khandia, K., Gusmanov, K., Mazzara, M., & Mavridis, N. (2016). Microservice-based IoT for Smart Buildings. *innopolis*, 23-31.
- Sanchez Peña, L. (01 de Octubre de 2012). *Revista Digital Universitaria UNAM*. Obtenido de Hogares y consumo energético en México: <http://www.revista.unam.mx/vol.13/num10/art101/index.html>
- Secretaria de Energía (SENER) en colaboración con la Agencia Internacional de Energía (AIE). (2011). *Indicadores de eficiencia energética en México: 5 sectores 5 retos*. México.
- Serra, L., Ch, R., Lujambio, J. M., Toro, M. Á., & Aguilar, S. (7 de Agosto de 2013). *CIDAC*. Obtenido de México inseguro... energéticamente: <http://cidac.org/mexico-inseguro-energeticamente/>
- Shaikh, P. H., Mohd Nor, N. B., Nallagownden, P., Elamvazuthi, I., & Ibrahim, T. (2014). A review on optimized control systems for building energy and comfort management of smart sustainable buildings. *Renewable and Sustainable Energy Reviews*, 409-429.
- T. S., Mukhopadhyay, S. C., Suryadevara, N. K., & Singh Gill, S. P. (2015). WSN-Based Smart Sensors and Actuator for Power Management in Intelligent Buildings. *IEEE/ASME TRANSACTIONS ON MECHATRONICS*, 564 - 571.
- Wikipedia*. (12 de Ago de 2018). Obtenido de AngularJS: <https://es.wikipedia.org/wiki/AngularJS>

VIII. ANEXOS**Anexo 1. Código fuente del MCU maestro en red de domótica**

```
void setup() { //Se inicializan los puertos serie
  Serial.begin(9600);
  Serial1.begin(9600);
  Serial2.begin(9600);
  Serial3.begin(9600);
}
void loop() {
  // lee de Puerto 0, Envía a los otros puertos (1 2 3):
  if (Serial.available()) {
    int inByte = Serial.read();
    Serial1.write(inByte);
    Serial2.write(inByte);
    Serial3.write(inByte);
  }
  // lee de Puerto 1,2 o 3 y Envía a puerto 0:
  if (Serial1.available()) {
    int inByte = Serial1.read();
    Serial.write(inByte);
  }
  if (Serial2.available()) {
    int inByte = Serial2.read();
    Serial.write(inByte);
  }
  if (Serial3.available()) {
    int inByte = Serial3.read();
    Serial.write(inByte);
  }
}
```

Anexo 2. Código fuente de MCU esclavo en red de domótica

```

#define CMDBUFFER_SIZE 8
//#define n 3
int led = 13;
char DIRECCION[3] = "01"; //Direccion del esclavo

typedef struct{
  char id[3];
  char subsistema[3];

  int pin1;
  int pinMode1;
  int pin2;
  int pinMode2;
  int pin3;
  int pinMode3;
  int pin4;
  int pinMode4;
  int pin5;
  int pinMode5;
  int pin6;
  int pinMode6;

} circuito;

circuito circuitos[3] = {"01","01",A5,0,6},{ "02","01",13,0},{ "03","01",10,0}};
void setup() {
  Serial.begin(9600);
  pinMode(led, OUTPUT);
  // initialize digital pin 13 as an output.
  pinMode(6, OUTPUT);
}

void loop() {}

```

ANEXOS

```
void serialEvent(){

static char cmdBuffer[CMDBUFFER_SIZE] = "";
char c;
while(Serial.available())
{
c = processCharInput(cmdBuffer, Serial.read());
if (c == '\n') {
//Full command received. Do your stuff here!
//Dividir el comando
char direccion [3];
memcpy(direccion,cmdBuffer,2);
direccion[2]='\0';
//
char subsistema [3];
subsistema[0] = cmdBuffer[2];
subsistema[1] = cmdBuffer[3];
subsistema[2] = '\0';
//
char circuito [3];
circuito[0] = cmdBuffer[4];
circuito[1] = cmdBuffer[5];
circuito[2] = '\0';
//
char instruccion [3];
instruccion[0] = cmdBuffer[6];
instruccion[1] = cmdBuffer[7];
instruccion[2] = '\0';

if (strcmp( DIRECCION , direccion) == 0)
{
exeSubsistema(subsistema,circuito,instruccion);
}
}
```

ANEXOS

```
cmdBuffer[0] = 0;
}

}
delay(1);
}

char processCharInput(char* cmdBuffer, const char c)
{
//Store the character in the input buffer
if (c >= 32 && c <= 126) //Ignore control characters and special ascii characters
{
if (strlen(cmdBuffer) < CMDBUFFER_SIZE) {
strncat(cmdBuffer, &c, 1); //Add it to the buffer
}
else {
return '\n';
}
}
else if ((c == 8 || c == 127) && cmdBuffer[0] != 0) //Backspace
{
cmdBuffer[strlen(cmdBuffer)-1] = 0;
}

return c;
}

void exeSubsistema(char* subsistema, char* circuito, char* instruccion){

char respuesta[10];

//Recorrer lista de circuitos para ver cual coincide con el comando Subsistema y circuito
int i;
char estado[2];
```

ANEXOS

```
for (i=0; i<3;i++){
if (strcmp( circuito , circuitos[i].id) == 0 && strcmp( subsistema , circuitos[i].subsistema) == 0){
  if (strcmp( subsistema , "01") == 0) {
    if(strcmp( instruccion , "01") == 0 && sensorLuz(circuitos[i].pin1)== 0 ){
      cambio(circuitos[i].pin2); // Enciende el led
      strcpy(respuesta,DIRECCION);
      strcat(respuesta,subsistema);
      strcat(respuesta, circuito);
      strcat(respuesta, instruccion);
      estado [0]='1';
      estado [1]='\0';
      strcat(respuesta,estado);
      Serial.println(respuesta);
    }

    if(strcmp( instruccion , "00") == 0 && sensorLuz(circuitos[i].pin1) == 1){
      cambio(circuitos[i].pin2); // Apagar el led
      strcpy(respuesta,DIRECCION);
      strcat(respuesta,subsistema);
      strcat(respuesta, circuito);
      strcat(respuesta, instruccion);
      estado [0]='0';
      estado [1]='\0';
      strcat(respuesta,estado);
      Serial.println(respuesta);
    }
  }
}
}
if (strcmp( subsistema , "02") == 0){
  Serial.println("\r\nSub Ventilacion");
}
}
}
```

ANEXOS

```
int sensorLuz (int entradaAnalog){
  int sensorValor = analogRead(entradaAnalog);
  if(sensorValor <= 100){
    //SIGNIFICA QUE LA LUZ ESTÁ APAGADA
    return 1;
  } else {
    //significa que la luz esta encendida
    return 0;
  }
}

void cambio (int pin){
  int estado = digitalRead(pin);
  if (estado == 1 ){
    digitalWrite(pin, LOW);
  } else {
    digitalWrite(pin, HIGH);
  }
}
```

Anexo 3. Código fuente de dispositivo Gateway

```
var SerialPort = require('serialport');
var socket = require('socket.io-client')('http://192.168.1.100:3000', { 'force a new connection': true
}).connect();

var Readline = SerialPort.parsers.Readline;
var serialPort = new SerialPort('/dev/ttyACM0', {
  baudRate: 9600,
  dataBits: 8,
  parity: 'none',
  stopBits: 1,
  flowControl: true
});

var parser = new Readline();
serialPort.pipe(parser);

parser.on('data', function (data) {
  console.log('data received: ' + data);
  socket.emit('respuesta', data);
});

socket.on('exeComando', function (data) {
  comando = data;
  console.log(comando);
  serialPort.write(comando);
});
```

Anexo 4. Código fuente de servidor**Package.json**

```
{
  "name": "scada",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "body-parser": "~1.18.2",
    "cookie-parser": "~1.4.3",
    "debug": "~2.6.9",
    "express": "~4.15.5",
    "jade": "~1.11.0",
    "mongoose": "^5.0.10",
    "morgan": "~1.9.0",
    "request": "^2.85.0",
    "serve-favicon": "~2.4.5",
    "socket.io": "^2.1.0"
  }
}
```

App.js

```
var express = require('express');
var path = require('path');
var favicon = require('serve-favicon');
var logger = require('morgan');
var cookieParser = require('cookie-parser');
var bodyParser = require('body-parser');
require('./app_api/models/db');
require('./app_api/models/componentes');
var routes = require('./app_server/routes/index');
var routesApi = require('./app_api/routes/index');
var users = require('./app_server/routes/users');
var app = express();
var server = require('http').Server(app);
var io = require('socket.io')(server);
// view engine setup
app.set('views', path.join(__dirname, 'app_server', 'views'));
app.set('view engine', 'jade');
var clients = {}
io.on('connection', function(client) {
  clients[client.id] = client;

  io.on('disconnect', function() {
    delete clients[client.id];
  });
  var address = client.request.connection.remoteAddress;
  console.log("New connection from " + address);

  // console.log(client.id);
  client.on('getComando',function(data){
    io.emit('exeComando',data);
    console.log("Comando Enviado "+ data);
  });
  client.on('respuesta',function(data){
```

ANEXOS

```
    console.log("respuesta " + data);
    io.emit('updateInterfaz', data);
  });
});
app.use(function(req, res, next){
  res.io = io;
  next();
});
// uncomment after placing your favicon in /public
//app.use(favicon(path.join(__dirname, 'public', 'favicon.ico')));
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use('/', routes);
app.use('/api', routesApi);

app.use('/users', users);
// catch 404 and forward to error handler
app.use(function(req, res, next) {
  var err = new Error('Not Found');
  err.status = 404;
  next(err);
});

// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err : {};

  // render the error page
```

ANEXOS

```
res.status(err.status || 500);  
res.render('error');  
});  
module.exports = {app: app, server: server};
```

bin\www

```
#!/usr/bin/env node

/**
 * Module dependencies.
 */

var app = require('./app').app;
var debug = require('debug')('scada:server');
var http = require('http');
/** Get port from environment and store in Express.*/

var port = normalizePort(process.env.PORT || '3000');
app.set('port', port);

/**
 * Create HTTP server.
 */
var server = require('./app').server;

/**
 * Listen on provided port, on all network interfaces.
 */

server.listen(port);
server.on('error', onError);
server.on('listening', onListening);

/**
 * Normalize a port into a number, string, or false.
 */

function normalizePort(val) {
```

ANEXOS

```
var port = parseInt(val, 10);

if (isNaN(port)) {
  // named pipe
  return val;
}

if (port >= 0) {
  // port number
  return port;
}

return false;
}

/**
 * Event listener for HTTP server "error" event.
 */

function onError(error) {
  if (error.syscall !== 'listen') {
    throw error;
  }

  var bind = typeof port === 'string'
    ? 'Pipe ' + port
    : 'Port ' + port;

  // handle specific listen errors with friendly messages
  switch (error.code) {
    case 'EACCES':
      console.error(bind + ' requires elevated privileges');
      process.exit(1);
      break;
```

ANEXOS

```
case 'EADDRINUSE':
    console.error(bind + ' is already in use');
    process.exit(1);
    break;
default:
    throw error;
}
}

/**
 * Event listener for HTTP server "listening" event.
 */

function onListening() {
    var addr = server.address();
    var bind = typeof addr === 'string'
        ? 'pipe ' + addr
        : 'port ' + addr.port;
    debug('Listening on ' + bind);
}
```

App_server\routes\index.js

```
var express = require('express');

var router = express.Router();
var ctrlDashboard = require ('../controllers/dashboard');
var ctrlAdmin = require ('../controllers/admin');
var ctrlReports = require ('../controllers/reports');
var ctrlOthers = require ('../controllers/others');
/* GET home page. Obtener pAgina principal*/

router.get('/', ctrlDashboard.index);
router.get('/comp', ctrlAdmin.compList);
router.get('/comp/edit', ctrlAdmin.compEdit);
router.get('/comp/:compid', ctrlAdmin.compDetalles);
router.get('/infra', ctrlAdmin.infraList);
router.get('/infra/edit', ctrlAdmin.infraEdit);
router.get('/reports', ctrlReports.reportsList);
router.get('/reports/details', ctrlReports.reportsDetails);
router.get('/about', ctrlOthers.about);
router.post('/comp/edit', ctrlAdmin.crearComponente);
module.exports = router;
```