UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA CIVIL EN COMPUTACIÓN

EcoLight: Interruptor inteligente para el uso eficiente de la luz eléctrica

SEBASTIÁN FELIPE DÍAZ ÁVILA

Profesor Guía: BENJAMIN INGRAM

Memoria para optar al título de
Ingeniero Civil en Computación

Curicó – Chile
Noviembre, 2016

CONSTANCIA

La Dirección del Sistema de Bibliotecas a través de su encargado Biblioteca Campus Curicó certifica que el autor del siguiente trabajo de titulación ha firmado su autorización para la reproducción en forma total o parcial e ilimitada del mismo.

Curicó, 2019

Dedicado a mi familia, gracias por su apoyo

AGRADECIMIENTOS

A mi madre y mi padre por cada una de esas largas conversaciones donde me apoyaron y me aconsejaron seguir adelante ya que siempre estarían ahí para apoyarme, siempre estaré agradecido de sus palabras, de su confianza, preocupación y por siempre buscar que estuviera cómodo para que solo me preocupara de estudiar, a mi hermana por todos tus consejos, cariño y preocupación por saber cómo iba, gracias por siempre estar ahí cuando los he necesitado, son la mejor familia que yo podría tener.

A Roxana por todos los años de apoyo, por la paciencia de enseñarme a pesar de mi torpeza con los números, gracias por nunca rendirte por mis sueños y por siempre estar cuando lo necesite. A su familia por darme un hogar y entregarme los mejores momentos durante mi estadía en Curicó, su preocupación y cuidados me entregaron un amor de familia del cual siempre estaré agradecido.

A los integrantes de casa amarilla por entregarme un lugar donde abundaron las risas y los buenos ratos, cada uno de ustedes me entregaron las mejores partidas de MOHAA y de PES que siempre recordare, momentos épicos como el salto del águila me entregaron tardes de risas y buenos momentos.

A Alberto, Liliana y Karen por entregarme lo más parecido a un hogar donde la tranquilidad y el excelente ambiente me permitió sentirme en casa, un lugar donde el apoyo y preocupación y el trabajo en equipo dieron sus frutos y hoy nos permitió avanzar de forma sólida con lo que nos pusieran al frente.

Al Team MNKK por cada uno de los buenos momentos que hemos vivido y seguiremos viviendo, a Nicolás por ser un amigo de la vida que siempre ha estado y estará, al Felipe y su destreza que nos matan de la risa, al Matías por armar una casa para todos y al Javier por su motivación para apañar en cada de nuestras locuras, gracias por todo.

RESUMEN

Hoy la eficiencia en el uso de la energía eléctrica está destinado a la conciencia de los usuarios, y existen pocas soluciones integrales en el mercado que permitan controlar el uso de las luces, evitar que éstas se mantengan encendidas innecesariamente, produciendo un consumo que podría evitarse.

El sobreconsumo de los hogares es algo que, a nivel país resulta relevante, ya que Chile es un país en vías de crecimiento y por tanto, tiene una alta demanda de electricidad, llevando a considerar nuevas estrategias de producción de energía para acoger la demanda que el país requiere.

El uso eficiente de la luz en los hogares se puede solucionar con interruptores que puedan ser capaz de determinar presencia humana, y encender las luces por un tiempo determinado, terminando así con un consumo de luz innecesario. Además se puede tener un control a través de plataformas móviles que permita administrar la luz del hogar sin la necesidad de estar físicamente.

En el presente documento se revisan todos los componentes necesarios para poder crear un prototipo de interruptor inteligente de bajo costo que permita tener reglas para el uso eficiente de la luz en los hogares. Primero analizaremos las razones de por qué el uso eficiente de la luz es tan importante para nuestro país. Una de estas razones es debido a la alta dependencia que se tiene de fuentes hídricas para la generación de energía en el país. Luego se describen los componentes que formaron parte del interruptor; algunos componentes destacables son las placas de Arduino, una dedicada a obtener información de los sensores y otra dedicada al procesamiento de la información que proviene de la WEB. Para utilizar servicios web y obtener información de estos, se utilizó una placa ESP8266 que permite conexiones WIFI y realizar consultas HTTP para mantener la información del interruptor en un servidor. Estos componentes fueron seleccionados para crear un dispositivo de bajo costo, de modo que pudiera competir en precio con otras soluciones ya existentes.

Una vez descrito los componentes, entenderemos como estos fueron integrados dentro de un circuito único, para este objetivo se utilizó placas de aprendizaje y no un circuito definitivo, una vez hecho esto, destacaremos los fragmentos más importantes del código que permite comunicar cada una de estas piezas de hardware antes descritas.

Para lograr un uso más intuitivo del interruptor se desarrolló una aplicación móvil que se conecta a través de internet con el interruptor y a través de la cual se programan reglas para el encendido y el apagado automático.

Al final del proyecto se pudo concluir que es posible realizar un prototipo de bajo costo de interruptor para la automatización de las luces del hogar, esto no garantiza el eficiente uso de esta y por consecuencia un ahorro en las cuentas, pero presenta una base sólida para poder conectar placas y sensores a través de internet.

TABLA DE CONTENIDOS

página

ÍNDICE DE FIGURAS

página

ÍNDICE DE TABLAS

página

1. Introducción

1.1. Descripción de la propuesta

Los avances tecnológicos de hoy en día nos permiten tener en la palma de la mano microcontroladores conectados a Internet con un alto poder de cálculo y recopilación de datos a través de sensores, esto permitió reconsiderar las capacidades que podrían llegar a tener los artículos de uso diario. Este tipo de desarrollo ha llevado a crear conceptos como la *Internet de las cosas (Internet of things)*. Este concepto plantea la creación de un ecosistema en donde las conexiones entre todas las cosas sean posibles, es decir, no sólo entre Smartphones o electrodomésticos sino también entre carreteras, automóviles, plantaciones, etc, en pocas palabras, entre todo.

Aprovechando estos avances tecnológicos es que se busca crear un dispositivo que permita el uso eficiente de la luz eléctrica en el hogar, entregando la posibilidad de automatizar las luces del hogar.

Para facilitar la automatización del interruptor se creó una aplicación móvil que permite programar el interruptor para ajustar las condiciones en las cuales encenderá o apagará la luz. Esto permitirá al usuario tener el control sobre las reglas de automatización.

Estos dispositivos podrán estar conectados a través de Internet permitiéndole al usuario el control de todos los interruptores del hogar desde el lugar que se encuentre. Esto podrá aumentar la capacidad de administrar el gasto de luz en el hogar sin la necesidad de estar presente en el lugar.

1.1.1. Contexto del proyecto

En nuestro país la energía es un tema que siempre ha estado en la palestra de la discusión. En estudios realizados por la empresa Quiroz & asociados a petición del

CDEC SIC(Centro de Despacho Económico y Carga Sistema Interconectado Central), en su documento "Estudio de previsión de demanda 2015-2035(2050)" [7], se analiza la demanda hasta el 2035, en la cual concluyen que del SIC aumentará en un 63 % y el SING en un 50 % la demanda, además de estos datos, analizan la posibilidad de que ocurran cambios de envergadura en la demanda. En particular, se revisan los casos de una mayor eficiencia energética en el consumo, un incremento en la penetración del auto eléctrico y por último, de una masificación de la auto generación. Al respecto, se concluye que la eficiencia, entre otros factores, podría reducir el crecimiento anual del consumo desde el 2,7 % en 2015 hasta apenas un 1,2 %.

Según el anuario estadístico de energía [9] el 41 % de las energía generada para el SIC proviene de una fuente hídrica, lo cual deja a Chile en una alta dependencia de esta materia prima, de la cual no tiene control.

1.1.2. Trabajo Relacionado

La gama de productos que existen hoy en día, que buscan dar una solución al problema, puede dividirse en 2 categorías:

- Ampolletas inteligentes
- Interruptores inteligentes

La primera categoría se encarga del control remoto de una ampolleta, es decir, tener el control a través de un dispositivo móvil, pudiendo controlar el encendido, apagado, intensidad, color y las horas en las cuales deben ocurrir estos eventos. Una de las desventajas que tiene este tipo de dispositivos es el precio que podría llegar a costar cambiar las luces de todo el hogar, ya que actualmente obtener una de estas ampolletas resulta más caro que obtener una de bajo consumo.

La segunda categoría se acerca más a los objetivos que se busca obtener, ya que controla el encendido y apagado de la luminaria a través del interruptor. La ventaja que tiene este dispositivo es lograr el control de más de una luz por dispositivo, lo que permite un ahorro de dinero en caso de querer obtener este tipo de tecnología para el control de la luz en el hogar.

Estas dos soluciones están pensadas para aumentar la accesibilidad del control de las luces del hogar, ya que en caso de quedar encendida una luz, éste podrá cambiar su estado desde el lugar que se encuentre, solo con una conexión a Internet.

1.1.3. Definición del problema

Uno de los grandes problemas que existen hoy en día con respecto al uso eficiente de la energía, es que siempre se recurre a la conciencia social para lograr un cambio. Esto se lleva a cabo a través de campañas para el ahorro energético, además de la inclusión de nuevas tecnologías que permiten un menor consumo. Pero esperar a que toda la población se acostumbre a ocupar de manera eficiente la luz eléctrica es una solución que puede tardar mucho tiempo, o simplemente no llegar.

Hoy en día no existe una solución real de automatización de este proceso, ya que las soluciones ofrecidas hoy en día mejoran la accesibilidad para poder cambiar el estado de la luz (encender o apagar), pero sigue siendo el usuario el responsable de este proceso. Además, estas soluciones atañen un gran costo de implementación ya que son precios que no son accesibles para todos.

1.1.4. Propuesta de solución

Para solucionar el problema anteriormente descrito, se creó un interruptor que a través de un microcontrolador arduino se maneja el encendido o apagado de la luz.

La solución tiene 3 formas de control:

- Control automático
- Control manual
- Control remoto

El control automático busca que el interruptor sea capaz de tomar la decisión de encender la luz a través de sensores que tendrá incorporados. Los umbrales o límites de estos sensores podrán ser programados por el usuario a través de una aplicación móvil, de manera que pueda programar la automatización de la luz dependiendo de las condiciones del entorno.

El control manual es el control que se lleva a cabo hoy en día, el cual a través de los pulsadores se puede cambiar el estado de una luz independiente de las condiciones del entorno. Esto permite al usuario cambiar de modo automático a manual sin la necesidad de la aplicación móvil.

El control remoto de la luz lo permite la aplicación móvil a través de Internet, pudiendo cambiar el estado sin la necesidad de estar en el hogar, pero solo cuando esté

establecido el modo manual, ya que esta característica busca solucionar la distancia para controlar la luz.

La aplicación móvil permite programar el comportamiento de la luz y la configuración para que el interruptor pueda acceder a Internet. Puede determinar nombres específicos para los interruptores y contemplara el uso de grupos a los cuales los interruptores podrán pertenecer. De esta manera al agregar un nuevo interruptor solo basta agregar el interruptor a un grupo para que funcione de manera adecuada en pocos pasos de configuración.

1.2. Hipótesis

Este proyecto se sustentó en las siguientes hipótesis:

- Es posible crear un interruptor que detecte movimiento.
- Es posible crear un interruptor que determina distancia.
- Es posible crear un interruptor que determina la cantidad de luz en el ambiente.
- Es posible crear un interruptor que este conectado a Internet.
- Es posible conectar un dispositivo móvil al prototipo de interruptor a través de Internet.
- El costo de tener interruptores inteligentes es menor al de las ampolletas inteligentes.

1.3. Objetivos

Al inicio de este proyecto se plantearon los siguientes objetivos.

Objetivo general

- Crear un interruptor al cual se le puedan programar reglas para el encendido y apagado automático de la luz de la casa, a través de una aplicación móvil que permita el uso eficiente de la luz eléctrica del hogar.

Objetivos específicos

- Crear un interruptor a través de un microcontrolador arduino.
- El interruptor debe permitir el encendido y/o apagado de las luces.
- Las decisiones serán tomadas según las lecturas que se tengan de los sensores agregados al arduino.
- Crear aplicación iPhone que permita la administración del interruptor. La aplicación permitirá administrar reglas para el interruptor, permitir el control remoto.
- La aplicación se comunicará a través de Internet con el interruptor.

1.4. Alcances

Al inicio del proyecto se definieron los siguientes alcances que limitaron el ámbito de los objetivos propuestos:

- El interruptor solo determinará encender o apagar la luz por las reglas establecidas.
- El rango de visión de un interruptor estará determinado por el alcance de los sensores.
- El único comportamiento que se modificará de la luz es el encendido o apagado.
- El interruptor solo será un prototipo funcional.
- La comunicación entre el dispositivo y el interruptor será por Wi-Fi.
- Las pruebas serán realizadas en un ambiente simulado (habitación).

1.5. Metodología

Para la metodología de trabajo se consideró empezar con la parte física del proyecto, realizar pruebas individuales de los componentes que se pretenden integrar, logrando un producto más avanzado con cada integración. A continuación se siguió con el software que controla el interruptor. Para poder lograr esta metodología es que se establecieron los siguientes objetivos a cumplir.

Objetivo 1: “Selección de sensores necesarios"

- Investigar cuáles son los sensores que existen y que podrían ser de utilidad.
- Realizar pruebas individuales con cada sensor para ver su comportamiento.
- Selección de los sensores definitivos que formaran parte del interruptor.

Objetivo 2: “Creación del prototipo"

- Crear Modelo de prototipo funcional.
- Pruebas de comportamiento del prototipo.

Objetivo 3: “Programar comunicación"

- Establecer modelo de comunicación.
- Pruebas de comportamiento de la comunicación.

Objetivo 4: “Creación de aplicación móvil"

- Creación de las vistas para la aplicación móvil.
- Creación del backend de la aplicación.
- Pruebas de la aplicación.

2. Marco Teórico

2.1. Generación y distribución eléctrica en Chile

Según el INE [12] Chile tiene 4 distribuidoras de energía las cuales se encargan de repartir las energía generada en Chile, estas son:

- SING (Sistema Interconectado Norte Grande)
- SIC (Sistema Interconectado Central)
- Aysén
- Magallanes

Los sistemas SIC y SING se llevan el mayor porcentaje de energía generada, siendo 90 % y 6,2 % respectivamente. Desde ahora nos enfocaremos en analizar el SIC ya que es el que tiene mayor importancia en la distribución eléctrica de Chile.

Según la CNE [9] en el 2015 la matriz generativa estaba compuesta por biomasa, gas Natural, mini hidráulica de pasada, carbón, hidráulica embalse, petróleo diésel, eólica, hidráulica de pasada y solar; siendo biomasa, carbón, hidráulica de embalse, hidráulica de pasada y gas natural las que contemplan el mayor porcentaje de producción de energía con 21 %, 27 %, 24 %, 21 %, 16 % respectivamente, dejando a Chile en una alta dependencia de recursos hídricos con un 41 % del total.

La distribución del consumo de la energía en Chile se reparte de la siguiente manera:

Transporte	33,1 %
Residencial	15,0 %
Industria	24,5 %
Comercial y público	5,9 %
Minería	15,8 %
Consumo propio del sector energía	5,7 %

Cuadro 2.1: Distribución energética 2015

Si bien el consumo residencial no se lleva gran parte del consumo en Chile, es el sector que puede variar más rápidamente su consumo implementando medidas de ahorro energético.

2.2. Eficiencia energética en el hogar

Según la AChee [1] fundación privada encargada de promover, fortalecer el uso eficiente de la energía para el país, en su guía práctica de la buena energía [10], podemos encontrar una serie de recomendaciones para el ahorro de la energía eléctrica del hogar, dentro de las cuales señalan que:

> "Se pueden conseguir ahorros importantes en iluminación sectorizando el alumbrado de forma que se enciendan las luces cercanas al interruptor de luz. En zonas de paso, como escaleras o pasillos, es importante utilizar sistemas de temporización o detectores de presencia que accionen automáticamente los encendidos/apagados de la luz". (Aprendamos a ahorrar, AChee, 2016, p.13-14).

Además en la guía práctica de la buena energía señalan:

> "lugares donde se realizan encendidos y apagados frecuentes no es recomendable poner ampolletas de bajo consumo, ya que éstas ven reducida de manera importante su vida útil por el número de encendidos". (Aprendamos a ahorrar, AChee, 2016, p.24).

Cabe señalar que la luz eléctrica en el hogar representa el 27,3 % del consumo eléctrico en el hogar.

A pesar de las recomendaciones hechas por esta asociación, hoy en día no existen muchas soluciones integrales que contemplen ampolletas de bajo consumo e interruptores que puedan temporizar el encendido o apagado de la luz en el hogar.

2.3. Trabajos relacionados

Actualmente en el mercado existe dos gamas de productos que buscan dar un control más inteligente a las luces del hogar, las que se pueden agrupar en:

- Interruptores inteligentes
- Ampolletas inteligentes

Dentro de la gama de interruptores inteligentes podemos encontrar productos destacados como WeMo Light switch [8] creada por la compañía Belkin para su gama de productos *home automation* y Switchmate [14] los cuales presentan las siguientes características.

WeMo
Compatibilidad de dispositivo
Dispositivos Android con 4.0 o mayor iOs 5 o mayor Dispositivos con Wifi o datos Moviles
Requisitos del sistema
Router Wi-Fi con conexión a internet WiFi 2.4GHz 802.11n WeMo App (downloadable from App Store or Play Store)
Requerimientos Eléctricos
Cable neutro requerido Reemplaza solo con un interruptor de 1 via (no de 3) Compatible con interruptores de 1 , 2 y 3 vias

Cuadro 2.2: Características de WeMo

Switchmate
Compatibilidad de dispositivo
Aplicación compatible con ambos sistemas Android e iOS
Requisitos del sistema
Previamente instalado un interruptor Toggle o Rocket
Requerimientos Eléctricos 2 baterias AA

Cuadro 2.3: Características de Switchmate

Cabe señalar que estos productos no son comercializados en Chile, que actualmente no existe masificación en la venta de este tipo de productos en el país.

Dentro de la gama de ampolletas inteligentes podemos encontrar a muchos fabricantes con múltiples implementaciones, pero la más reconocida y/o comercializada en nuestro país viene de la mano de la marca Philips con su ampolleta Hue que tiene las siguientes características.

hue
Compatibilidad de dispositivo
Aplicación compatible con ambos sistemas Android e iOS
Requisitos del sistema
Socket de bombillo E27 0°C - 40°C
Requerimientos Eléctricos 220V

Cuadro 2.4: Características de hue

Los productos antes mencionados, solo facilitan el acceso a controlar la luz a través de una aplicación móvil, pero no automatizan el proceso de encendido y apagado, por lo tanto, el uso eficiente de la luz, sigue siendo tarea del usuario.

2.4. Hardware

En este apartado se describirán todos los componentes eléctricos, electrónicos, electromecánicos y mecánicos que se utilizaron para la creación del interruptor. Para que éste pueda funcionar de manera automática, necesita de sensores y microcontroladores que puedan determinar si encender o no la luz dependiendo de las condiciones del ambiente.

2.4.1. Arduino

Arduino [3] es una plataforma de prototipos electrónica de código abierto (open-source) basada en hardware y software flexibles y fáciles de usar. Está pensado para artistas, diseñadores, como hobby y para cualquiera interesado en crear objetos o entornos interactivos. Arduino puede sentir el entorno mediante la recepción de entradas desde una variedad de sensores y puede afectar a su alrededor mediante el control de luces, motores y otros artefactos. El microcontrolador de la placa se programa usando el Arduino *Programming Language* (basado en Wiring) y el *Arduino Development Environment* (basado en Processing). Los proyectos de Arduino pueden ser autónomos o se pueden comunicar con software en ejecución en un ordenador (por ejemplo con Flash, Processing, MaxMSP, etc.). Las placas se pueden ensamblar a mano o encargarlas pre-ensambladas; el software se puede descargar gratuitamente. Los diseños de referencia del hardware (archivos CAD) están disponibles bajo licencia open-source, por lo que eres libre de adaptarlas a tus necesidades. Arduino recibió una mención honorífica en la sección *Digital Communities del Ars Electronics Prix* en 2006.

Arduino Nano

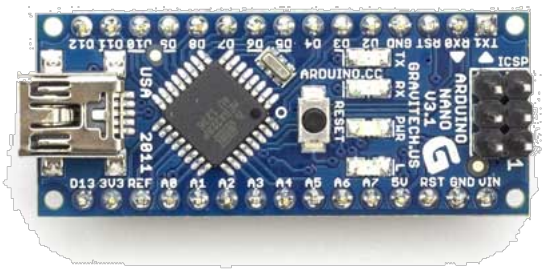

Figura 2.1: Arduino nano

El Arduino Nano [5] mostrado en la Figura 2.1 es una pequeña, completa y amistosa placa basada en el ATmega328 (Arduino Nano 3.x) o ATmega168 (Arduino Nano 2.x). Tiene más o menos la misma funcionalidad del Arduino Duemilanove, pero en un paquete diferente. El Arduino Nano carece de una sola toma de corriente continua, y funciona con un cable USB Mini-B en lugar de una normal. El Arduino Nano fue diseñado y está siendo producido por Gravitech.

Arduino Mega

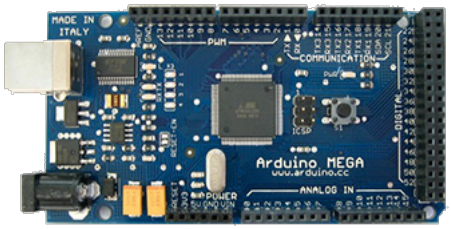

Figura 2.2: Arduino Mega

El Mega 2560 [4] mostrado en la Figura 2.2 es una placa electrónica basada en el Atmega2560. Cuenta con 54 pines digitales de entrada/salida (de los cuales 15 se pueden utilizar como salidas PWM), 16 entradas analógicas, 4 UARTs (puertos

serie de hardware), un oscilador de 16MHz, una conexión USB, un conector de alimentación, una cabecera ICSP, y un botón de reinicio. Basta con conectarlo a un ordenador con un cable USB o la corriente con un adaptador de CA a CC o una batería para empezar. El tablero de 2560 mega es compatible con la mayoría de los shield para el Uno y los anteriores juntas de Duemilanove o Diecimila.

2.4.2. Módulo Wi-Fi

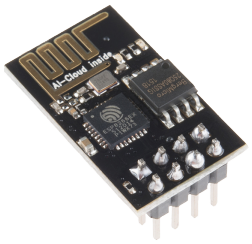

Figura 2.3: ESP-01 módulo Wi-Fi

Espressif Systems Smart Connectivity Platform (ESCP) [15] es un conjunto integrado de SOC's inalámbricas de alta integración y alto rendimiento, diseñado para plataformas móviles de espacio y energía limitada. Proporciona una capacidad sin igual para incrustar capacidades Wi-Fi dentro de otros sistemas o para funcionar como una aplicación independiente, con un costo bajo y mínimo de espacio requerido.

Dentro de la familia de SOC's creadas por Espressif para fines de este proyecto se utilizó la versión ESP-01 mostrada en la Figura 2.3 la cual tiene las capacidades necesarias para cumplir con los requisitos del proyecto.

2.4.3. Sensor de Luz

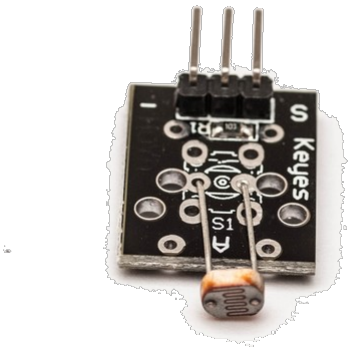

Figura 2.4: Módulo LDR keyes

Las foto resistencias [16], también conocidas como resistencias dependientes de la luz (LDR), como se puede apreciar en la Figura 2.4, son dispositivos sensibles a la luz más utilizadas para indicar la presencia o ausencia de luz. En la oscuridad, su resistencia es muy alta, pero cuando el sensor LDR se expone a la luz, la resistencia se reduce drásticamente, incluso hasta unos pocos ohmios, dependiendo de la intensidad de la luz.

2.4.4. Sensor de Distancia

Figura 2.5: Módulo Distancia HC-SR04

Como se puede apreciar en la Figura 2.5 un sensor HC-SR04 [13] es un módulo que incorpora un par de transductores de ultrasonido que se utilizan de manera conjunta para determinar la distancia del sensor con un objeto colocado enfrente de éste. Quizá la característica más destacada del HC-SR04 es que puede ser adquirido por una baja suma de dinero y esto mismo lo ha hecho muy popular. Afortunadamente el módulo HC-SR04 es bastante fácil de utilizar a pesar de su bajo precio y no demanda grán cantidad de trabajo ponerlo a funcionar, mucho menos si utilizamos una librería para sensores ultrasónicos.

2.4.5. Sensor de Movimiento

Figura 2.6: PIR (sensor de distancia)

Como se puede apreciar en la Figura 2.6 un sensor PIR [2] permite detectar el movimiento, casi siempre se utiliza para detectar si un ser humano se ha movido. Son pequeños, de bajo costo, bajo consumo de energía, fácil de usar y no se desgastan. Por esa razón, se encuentran comúnmente en los electrodomésticos y aparatos utilizados en los hogares o negocios. A ellos se refieren a menudo como PIR, "infrarrojo pasivo", "piroeléctric", o "IR" sensores de movimiento.

2.4.6. Relé

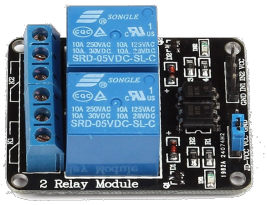

Figura 2.7: Módulo rele arduino

Como se puede apreciar en la Figura 2.7 un relé [6] es un control electrónico de interruptor que le permite activar o cerrar un circuito con tensión y/o corriente mucho más alto que el Arduino podía manejar. No hay conexión entre el circuito de baja tensión operado por Arduino y el circuito de potencia alta. El relé protege cada circuito uno del otro.

2.5. Software

En este apartado se describirán todos los componentes intangibles y lógicos que permitieron la creación del interruptor. Estos componentes son necesarios para que el hardware pueda actuar de la manera que se espera, además de representar las estructuras necesarias para el funcionamiento remoto del interruptor.

2.5.1. Software arduino

El software Arduino [11] está publicado como herramientas de código abierto, disponible para extensión por programadores experimentados. El lenguaje puede ser expandido mediante librerías C++, y la gente que quiera entender los detalles técnicos pueden hacer el salto desde Arduino a la programación en lenguaje AVR C en el cual está basado. De forma similar, se puede añadir código AVR-C directamente en los programas Arduino si se quiere.

Este software se utiliza para poder tener control sobre las placas Arduino antes mencionadas, por lo tanto, este lenguaje se utiliza de manera específica para controlar todo el hardware del interruptor

2.5.2. Swift

Swift es un nuevo lenguaje de programación para iOS, MacOS, watchOS y aplicaciones TVOS que se basa en lo mejor de C y Objective-C, sin las limitaciones de compatibilidad de C. Swift adopta patrones de programación seguras y añade características modernas para que la programación sea más fácil, más flexible y más divertido. Swift está respaldado por los frameworks Cocoa y Cocoa Touch.

Este Lenguaje de programación es utilizado para crear la aplicación móvil bajo en entorno de Mac. pudiendo ser ejecutado en una amplia gama de dispositivos como iPhone, iPod y iPad

2.5.3. PHP

PHP [17] es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo

externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

3. Desarrollo

A continuación se presenta el desarrollo completo del proyecto *Ecolight* que contempla los componentes de hardware y software del interruptor, servidor web y aplicación móvil, además de cómo se integran cada una de estas partes para dar funcionamientos a cada uno de los requisitos del proyecto.

3.1. Requisitos

Antes de continuar con el detalle de las partes que forman el proyecto *EcoLight*, definiremos los requisitos que debe cumplir el proyecto completo, esto quiere decir, las acciones que realizará el proyecto una vez unida las partes de software, hardware y servicio web que forman parte de éste.

Nombre	Interruptor crea Red Wi-fi.
Descripción	El interruptor una vez instalado debe crear una red Wi-Fi con el nombre EcoLightXXXX, siendo "XXXX", el número de serie del producto.
Involucrados	Interruptor.

Cuadro 3.1: Descripción Requisito Interruptor crea Red Wi-fi

Nombre	Obtener IP del interruptor.
Descripción	El interruptor una vez instalado y con la red Wi-Fi creada entregará una IP al momento de conectarse .
Involucrados	Interruptor Aplicación Móvil.

Cuadro 3.2: Descripción Requisito Obtener IP del interruptor.

Nombre	Encender y/o apagar luces.
Descripción	El interruptor una vez instalado debe encender y apagar las luces desde los pulsadores.
Involucrados	Interruptor.

Cuadro 3.3: Descripción Requisito Encender y/o apagar luces.

Nombre	Agregar un interruptor.
Descripción	El interruptor una vez instalado puede ser agregado en la aplicación móvil .
Involucrados	Interruptor Servicio web Aplicación Móvil.

Cuadro 3.4: Descripción requisito agregar un interruptor.

Nombre	Eliminar interruptor.
Descripción	El interruptor una vez agregado puede ser eliminado en la aplicación móvil .
Involucrados	Aplicación Móvil.

Cuadro 3.5: Descripción requisito eliminar un interruptor.

Nombre	Cambiar de Modo el interruptor.
Descripción	El interruptor una vez agregado permite cambiar los modos del interruptor, estos modos son el Modo manual, movimiento y distancia .
Involucrados	Aplicación Móvil. Servicio web. Interruptor.

Cuadro 3.6: Descripción requisito cambiar de Modo el interruptor.

Nombre	Encender luces con movimiento.
Descripción	El interruptor al cambiarlo a modo movimiento puede encender o apagar la luz habiendo movimiento en el ambiente, y apagarse cuando este movimiento deja de existir.
Involucrados	Aplicación Móvil. Servicio web. Interruptor.

Cuadro 3.7: Descripción requisito encender luces con movimiento.

Nombre	Encender luces con distancia.
Descripción	El interruptor al cambiarlo a modo distancia puede encender o apagar la luz habiendo pasado la distancia máxima establecida por el usuario, mientras esta barrera sobrepasada la luz debe mantenerse encendida.
Involucrados	Aplicación Móvil. Servicio web. Interruptor.

Cuadro 3.8: Descripción requisito encender luces con distancia.

Nombre	Agregar grupo de comportamiento.
Descripción	La aplicación permite agregar grupos con modos y parámetros para modos definidos por el usuario, estos grupos no definen estado de la luz, solo los parámetros con los cuales serán encendidas o apagadas.
Involucrados	Aplicación Móvil.

Cuadro 3.9: Descripción requisito agregar grupo de comportamiento..

Nombre	Eliminar grupo de comportamiento.
Descripción	La aplicación permite eliminar grupos de comportamientos, una vez el grupo es eliminado, todos los interruptores vuelven al grupo de comportamiento por defecto que tiene la aplicación que se llama "Ninguno".
Involucrados	Aplicación Móvil. Servicio web interruptor

Cuadro 3.10: Descripción requisito eliminar grupo de comportamiento.

Nombre	Editar parámetros del interruptor.
Descripción	La aplicación permite eliminar grupos de comportamientos, una vez el grupo es eliminado, todos los interruptores vuelven al grupo de comportamiento por defecto que tiene la aplicación que se llama "Ninguno".
Involucrados	Aplicación Móvil. Servicio web interruptor

Cuadro 3.11: Descripción requisito editar parámetros del interruptor.

Nombre	Cambio a modo manual con presionar un pulsador.
Descripción	El interruptor independiente del modo en el cual se encuentra debe cambiar a modo manual al momento de encender y/o apaga a través de los pulsadores de éste, y mantenerse así hasta que a través de la aplicación se vuelvan a cambiar los valores .
Involucrados	Aplicación Móvil. Servicio web interruptor

Cuadro 3.12: Descripción requisito cambio a modo manual con presionar un pulsador.

3.2. Hardware

Son todas las piezas electrónicas que fueron necesarias para poder encender o apagar la luz ya sea a través de la acción humana o de manera automática mediante sensores.

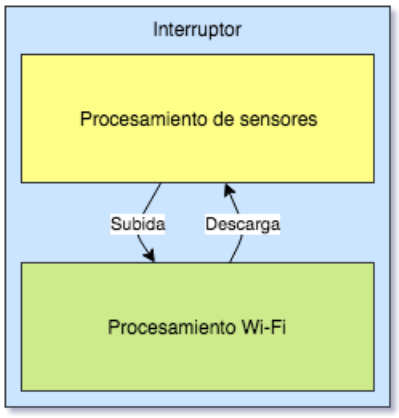

Figura 3.1: Unidades que componen el interruptor

Como se aprecia en la Figura 3.1 el interruptor está compuesto de dos unidades, la primera es la unidad de procesamiento de sensores y la segunda es la unidad de procesamiento Wi-Fi.

La unidad de procesamiento de sensores es la encargada de analizar a través de lectura de sensores la cantidad de luz que existe en el lugar, distancia a la que se encuentra un objeto, movimiento humano y pulsadores; con esta información determina si debe encender la luz.

La unidad de procesamiento de Wi-Fi es la encargada de consultar aun servidor si existe algún cambio en los parámetros del interruptor; de existir, los descarga para enviarlos a la unidad de procesamiento de sensores.

A continuación detallaré como se compone cada una de estas unidades, además, de cuáles son las acciones que realizan de manera particular.

3.2.1. Procesamiento de sensores

Para que esta unidad pueda tomar la decisión si encender o apagar la luz, necesita saber el estado del ambiente y de los pulsadores. Estos son los componentes que forman esta unidad:

- Arduino Nano
- Pulsadores
- Sensor de distancia
- Sensor de movimiento
- Sensor de luz
- Relé

Conexiones

los componentes antes nombrados se conectan de la siguiente manera.

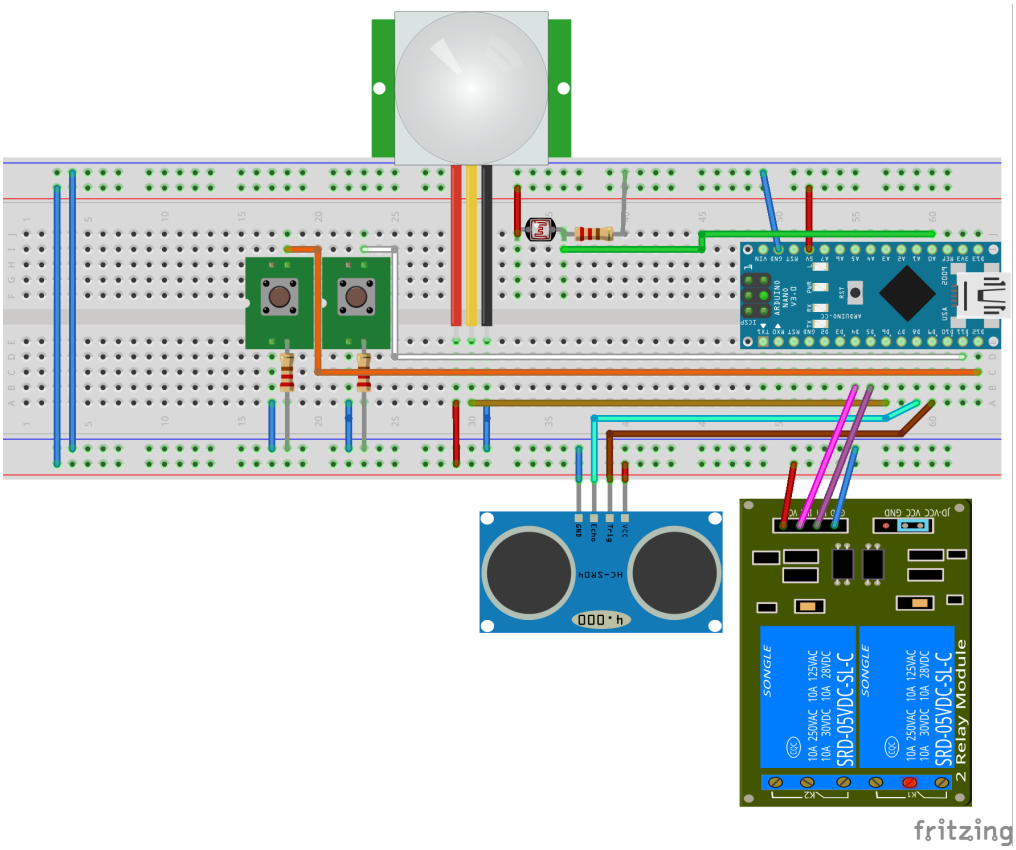

Figura 3.2: Conexiones de la unidad de procesamiento de sensores

Funcionalidades

Las piezas de hardware antes nombradas tienen como objetivo medir el entorno y a través de éstas decidir si la luz debe estar encendida. Pero si bien las piezas siempre están tomando lecturas del entorno, el interruptor solo considera un tipo de lectura a la vez, esto quiere decir, que el interruptor puede seleccionar cuál de los sensores definirá el estado de la luz. Para realizar esto, en el interruptor existen 3 modos que pueden ser seleccionados :

- Modo manual
- Modo movimiento

- Modo distancia

El interruptor puede estar solo en un modo a la vez y cada modo es encargado de tomar lectura de uno o más sensores. El modo manual, solo toma lectura de los pulsadores ya que funciona de la misma manera que funciona un interruptor convencional. El modo movimiento toma lectura del PIR (sensor de movimiento) y del LDR (sensor de luz). Finalmente el modo distancia toma lectura de los sensores HC-SR04 (sensor de distancia) y del LDR.

En modo manual solo existen dos maneras de cambiar el estado de las luces: la primera es a través de los pulsadores, al igual que los interruptores de hoy en día, y la segunda, es a través de la aplicación la cual permite cambiar el estado de las luces de manera remota.

Ambos modos automáticos toman como referencia la cantidad de luz que hay en el ambiente. Para esto se utiliza el LDR, que toma valores dependiendo de cuanta luz existe, ya que actúa como una resistencia variable que varia su resistencia dependiendo de la cantidad de luz. Los valores que puede llegar a tomar son de 0 a 1023, siendo 0 nada de luz, y 1023 mucha luz. De modo que el usuario puede establecer una cantidad mínima de luz, y si esta baja, el interruptor leerá el sensor correspondiente al modo (distancia o movimiento) y analizara si debe cambiar de estado las luces.

Programa Arduino

El código que es ejecutado en el arduino nano que pertenece a la unidad de procesamiento de sensores se encuentra en el anexo A.1. En este apartado solo destacaremos las partes importantes del código.

```
void SerialSendData(){
  Serial.println("Enviando datos");
  String data = String(id)+"&"+nombre+"&"+grupo+"&"+String(estado)+
      "&"+String(luzOn)+"&"+String(distancia)+"&"+String(luz)+"&"+
      String(tiempoencendida);
  SerialArduino.println(data);
}
```

La función SerialSendData es la encargada de mandar los datos a la unidad de procesamiento Wi-Fi. Es invocado cuando el usuario presiona algunos de los pulsa-

dores para encender y/o apagar la luz.

```
void SerialInputData(){
      char caracter;
      String data="";
      while(SerialArduino.available()){
        caracter = SerialArduino.read();
        data.concat(caracter);
        delay(10);
      }
  if(data!=""){
    //id&nombre&grupo&estado&encendido&distancia&luz&retardo;
     data.trim();
     Serial.println(data);
     nombre = valueString(1,data);
     grupo = valueString(2,data);
     estado = valueString(3,data).toInt();
     luzOn = valueString(4,data).toInt();
     distancia = valueString(5,data).toInt();
     luz = valueString(6,data).toInt();
     int tiempoencendidatmp = valueString(7,data).toInt();
     tiempoencendida = tiempoencendidatmp*1000;
     if(estado == 1){
        if(luzOn == 1){
          ampolleta = true;
          ampolleta2 = true;
        }else{
          ampolleta = false;
          ampolleta2 = false;
        }
     }
  }
}
```

El método SerialInputData es el encargado de recibir la información proveniente de la unidad de procesamiento Wi-Fi. Esta función es invocada cada vez que la unidad de procesamiento descarga información desde el servidor, una vez recibido es procesado para guardar los valores correspondientes. Además, detecta en el estado que deben quedar las luces, y las enciende o apagada dependiendo lo que venga en

la información.

3.2.2. Procesamiento Wi-Fi

Esta unidad es la encargada de Procesar la información proveniente del servidor web. Para esto necesita de dos componentes electrónicos:

- ESP-01
- Arduino Mega 2560

Conexiones

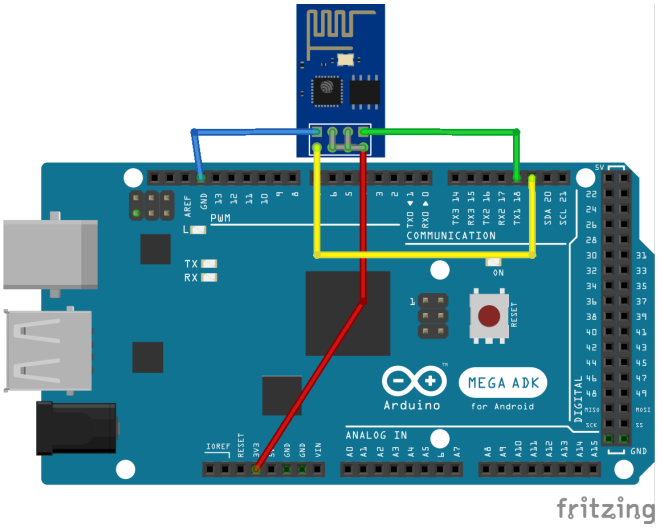

Figura 3.3: Conexiones de la unidad de procesamiento de Wi-Fi

El ESP-01 es el módulo Wi-Fi, el que permite realizar conexiones con el servidor además de múltiples funcionalidades de red que necesita el proyecto:

- Estar en modo AP(Acces Point) y cliente al mismo tiempo
- Conexión entrante por Socket
- Conectarse a una red Wi-Fi
- Realizar llamadas TCP a servidor web

Para que el módulo Wi-Fi pueda realizar cada una de estas acciones debe recibir los comandos necesarios. A continuación, en la Tabla 3.13, se muestran los principales comandos que se ejecutan en el módulo Wi-Fi para realizar las acciones antes descritas.

AT commandos		
Comando	Parámetro	Descripción
AT	Ninguno	Permite saber si el módulo esta listo
AT+CWQAP	Ninguno	Desconectarse de una red Wi-Fi
AT+CWMODE	=1(STA) =2(AP) =3(ambos)	Permite cambiar de modo STA,AP,ambos
AT+CIPMUX	=1(MULTI) =0(SIMGLE)	Permite múltiples conexiones o no
AT+CIPSERVER="modo","puerto"	MODE=1(Crear) MODE=0(Borrar) PORT=333(default)	permite crear o borrar al ESP como servidor
AT+CWSAP="ssid","pass",chl,enc	ssid=nombre pass=contraseña chl=canal enc=cifrado	Permite crear una red Wi-Fi con nombre, contraseña, canal y tipo de cifrado
AT+CIPCLOSE=ïd"	ID=numero de la conexión	Cierra conexiones establecidas identificador de la conexión
AT+CIPSTART=tipo,ip,Puerto	tipo=TCP,UDP IP=IP o URL del servidor Puerto= puerto del servicio	Permite Iniciar una conexión
AT+CIPSEND=ïd",largoData	ID=identificador largoData=largo mensaje	Permite enviar datos a una conexion ya establecida

Cuadro 3.13: Comandos AT para ESP-01

El Arduino Mega 2560 tiene dos funciones principales, la primera es enviar los comandos AT al ESP-01 para que realize acciones determinadas, y la segunda es funcionar como intermediario entre las respuestas del servidor y la unidad de procesamiento de sensores, ya que desde el servidor llega más información de la necesaria y debe ser procesada antes de ser enviada a esta unidad.

En el siguiente apartado se hablará con mas detalle de los formatos de la información aceptada para que pueda ser procesada debidamente por ambas unidades de procesamiento.

Programa

El código completo ejecutado en el arduino Mega 2560 de la unidad de procesamiento Wi-Fi se encuentra en el Anexo A.2. Destacaremos solo los sectores importantes del código en este apartado.

```
void serialEvent2(){
//id&nombre&grupo&estado&encendido&distancia&luz&retardo;
    char dato;
    String data ="";
    while(Serial2.available()){
       dato = (char)Serial2.read();
       data+=dato;
    }
    if(data.indexOf("&")>=0){
        String variables= "id=1&nombre="+valueString(1,data)+"&grupo="+
            valueString(2,data)+"&estado="+valueString(3,data)+"&
            encendido="+valueString(4,data)+"&distancia="+valueString(5,
            data)+"&luz="+valueString(6,data)+"&retardo="+valueString(7,
            data)+"";
        Serial.println(data);
        variables.trim();
        String llegada = UploadData(variables);
        if(llegada.indexOf("OK")>=0){
              Serial.println("Envio OK");
         }else{
               Serial.println("Envio ERROR");
         }
    }
}
```

El método serialEvent2 permite recibir los mensajes enviados por la unidad de procesamiento de sensores; serialEvent2 es un método entregado por la librería de arduino mega que permite almacenar el mensaje que es enviado a través del puerto serial2 de esta placa. Esto permite analizar el mensaje sin la necesidad de sincronizar las dos placas, por lo tanto, si el arduino mega se encuentra ocupado realizando la descarga de la información, apenas se encuentre disponible podrá procesar la información aunque se halla enviado momentos atrás.

```
void loop() {
    switch(activity){
      case 1:
           sendCommand("AT+CWQAP",300);
           sendCommand("AT+CWMODE=3",300);
           sendCommand("AT+CIPMUX=1",300);
           sendCommand("AT+CIPSERVER=1,80",300);
           sendCommand("AT+CWSAP=\"EcoLight1\",\"\",1,0",300);
           activity = 2;
           break;

      case 2:
           wifiConfiguration();
           break;

      case 3:
           delay(1000);
           sendCommand("AT+CIPMUX=0",300);
           sendCommand("AT+CWMODE=1",300);
           sendCommand("AT+CIPCLOSE",300);
           activity = 4;
           break;

      default:
           DowloadData();
           break;

    }
}
```

Esta porción de código es la que permite cada una de las funcionalidades de red

definida del módulo Wi-Fi. El caso 1, permite la configuración para crear una red Wi-Fi; el caso 2, permite capturar la información enviada por socket desde una aplicación móvil; caso tres, cierre de señal Wi-Fi, habilitar el modo cliente y cierre de toda conexión previamente establecida; caso 4, descarga de datos desde el servidor, También se puede apreciar, cada uno de los comandos antes definidos que son enviados al módulo para que realicen las acciones respectivas.

```
void sendCommand(String command, int retardo){
    char dato;
    String data ="";
    Serial.println(command);
    Serial1.println(command);
    delay(retardo);
    while(Serial1.available()){
       dato = Serial1.read();
       data.concat(dato);
    }
```

sendCommand es la función encargada de ejecutar los comandos antes vistos para el módulo ESP-01. Existen dos versiónes de esta función; la que vemos permite controlar el tiempo de retardo entre el envío del código y la recepción de la respuesta, y la segunda versión permite controlar el tiempo y además analizar la respuesta del módulo Wi-Fi.

3.2.3. Conexión entre unidades

Para que todo el apartado de hardware funcione como una unidad, el arduino nano y el arduino MEGA deben ir conectados. Esta conexión se realiza entre las placas arduino a través de un puerto serial. Esta comunicación se realiza para que la unidad de procesamiento de sensores pueda subir los cambios que genera un encendido o apagado manual de las luces.

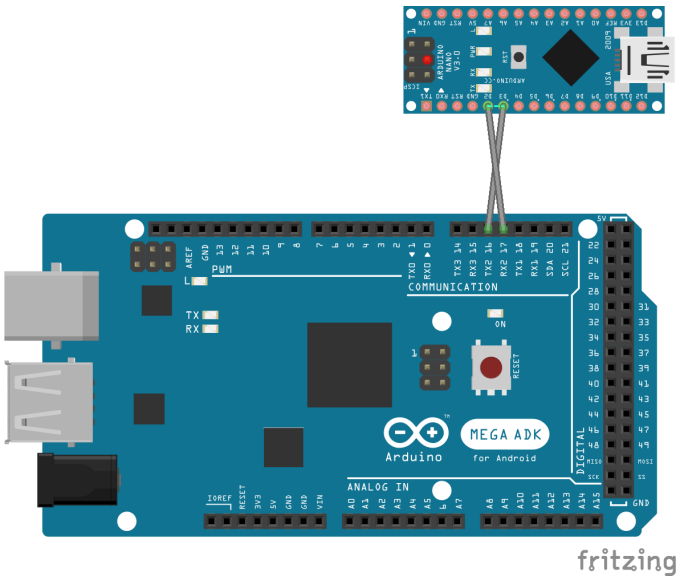

Figura 3.4: Conexion serial entre unidades

Como se muestra en la Figura 3.4, esta conexión se realiza a través de los puertos TX2 y RX2 del arduino mega y los pines 2 y 3 que funcionaran como TX y RX del arduino nano a través de softwareSerial, una librería que permite ocupar pines digitales como puerto serial.

Presupuesto

Uno de lo desafíos planteados al inicio del proyecto, es que éste pueda ser competitivo contra las otras soluciones existentes. Para lograr esto se utilizaron componentes menos costosos que permiten cumplir con los requisitos antes establecidos, ya que de volverse una solución mas costosa dejaría de ser competitiva con las soluciones ya existentes.

A continuación se detalla el precio de los componentes utilizados en este proyecto los cuales fueron obtenidos a través de EBAY.

Componente	Cantidad	Precio
Hc-sr501 Infrarrojo Pir	1	669
KeyesMódulo Sensor Ldr KY-018	1	1.654
Pulsador 10un 4 piernas para Arduino	2	669
Ultrasónico Hc-sr04 Distancia	1	669
módulo Relé Arduino	1	912
Mini Usb Nano V3.0	1	1.737
Mega 2560 R3	1	5.605
esp8266 esp-01 Wifi	1	1.359
120Pcs 11cm macho a hembra Cable	1	1521
10k Ohm 10kohm resistencias	1	669
TotalMb-102 830 protoboard	1	1345
	Total	16809

Cuadro 3.14: Precio de los componentes utilizados

3.3. Servidor Web

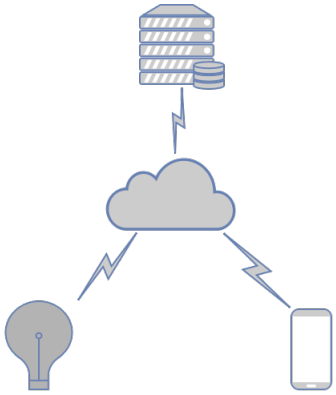

Figura 3.5: Diagrama Servidor Web Plano General

Como se muestra en la Figura 3.5 el servidor web recibe las peticiones del interruptor y del teléfono, éste es el encargado de almacenar las variables que podrían sufrir cambios tras la configuración del usuario o cambio de estado del interruptor. Las configuraciones de usuario son enviadas desde una aplicación móvil de la cual se detallará mas adelante, y los cambios de estados ocurren cuando el usuario interactúa

de manera física con el interruptor a través del pulsador.

De esta manera, almacenando y modificando las variables del interruptor en un servidor web se puede establecer una comunicación entre una aplicación móvil y el interruptor.

Para almacenar las variables y poder realizar cambios en ellas se necesitan 2 estructuras dentro de este servidor web: una base de datos que almacene estos valores y un servicio web que permita a los dispositivos modificar y recibir la información que se encuentra en la base de datos. Estos componentes se pasaran a detallar a continuación.

3.3.1. Base de datos

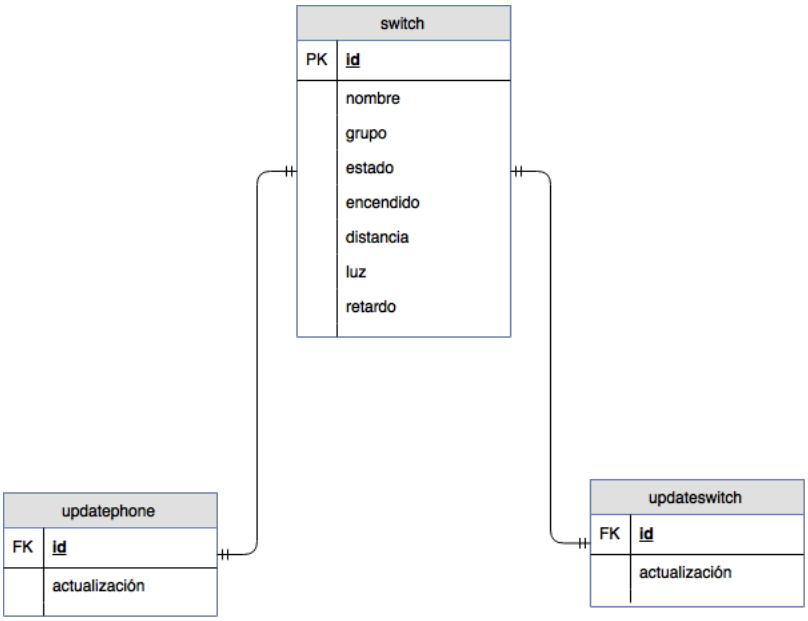

Figura 3.6: Diagrama de base de datos

Como se muestra en la imagen existen tres tablas dentro de la base de datos :

- switch
- updatephone
- updateswitch

La tabla switch es la encargada de almacenar todos los valores que pueden ser configurables por el usuario, y que pueden sufrir algún cambio en el interruptor además de un identificador único que represente a cada interruptor.

Las tablas updatephone y updateswitch son las encargadas de almacenar si el teléfono o el interruptor debe actualizar su información porque ha ocurrido un cambio.

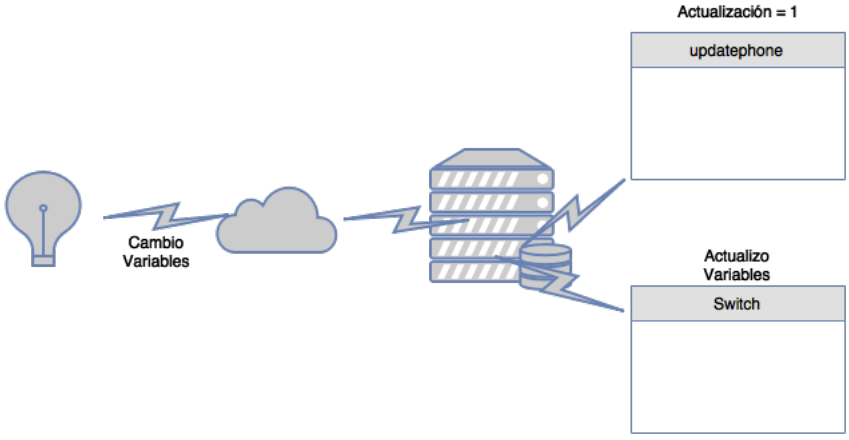

Figura 3.7: Cambios de variable desde interruptor

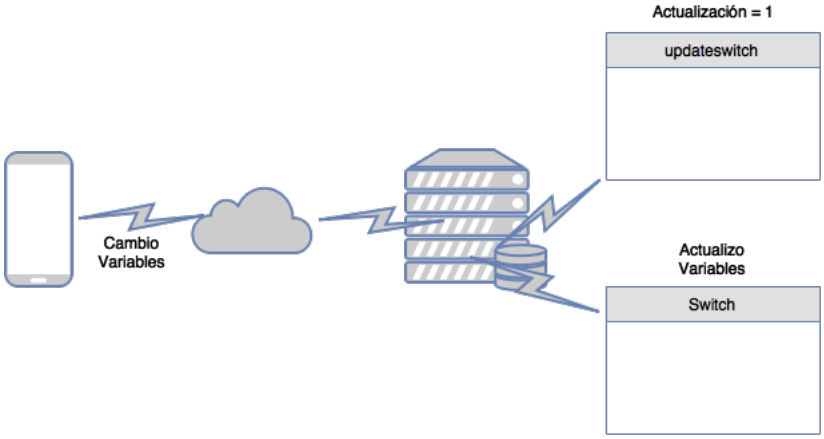

Figura 3.8: Cambios de variable desde aplicación móvil

Como se muestra en las Figuras 3.7 y 3.8, cuando el interruptor o la aplicación realiza un cambio en la información, ésta altera al tabla contraria de actualización, esto quiere decir que si el interruptor realiza cambios en la información, editará la tabla updatephone para que la aplicación pueda actualizarse y lo mismo ocurre para la aplicación, solo que con la tabla updateswitch.

Cuando el usuario elimina un interruptor de la aplicación, éste se elimina en la tabla switch y por foreign key se elimina en cascada para las demás tablas que contengan ese id.

3.3.2. Servicios PHP

Los servicios PHP son los intermediarios entre los dispositivos y la base de datos, éstos definen tres servicios:

- Download
- Upload
- Delete

Estos tres servicios están separados por dispositivo, esto quiere decir que existe servicio Download y Upload para el interruptor, y por otra parte, se encuentra el servicio Download, Upload y Delete para la aplicación móvil.

Download es el servicio de descargar los atributos del interruptor sí estos fueron cambiados, Upload es el servicio encargado de subir información a la base de datos ante algún cambio y Delete es el servicio especial de la aplicación que elimina el registro de la tabla.

Formato Respuesta

Para la Respuesta de los servicios web se estableció el siguiente formato:

Respuesta Servicio PHP	
Respuesta	Significado
&ERROR200&	Error al conectarse a la base de datos
&ERROR201&	Error al seleccionar la base de datos
&ERROR20X&	Error al ejecutar la consulta numero "X"del servicio
&OK&	El servicio Upload o Delete se ejecutaron con éxito
&NO&	El servicio Download o Upload no se ejecutaron con éxito
id&nombre&grupo &estado &encendido &distancia&luz&retardo	Formato de la informacion del interruptor , esta se entrega cuando hay actualización

Cuadro 3.15: Formato Respuesta servicio web

Programa

El código de los servicios PHP del servidor es el siguiente: Servicio Delete

```
<?php
$idSwitch = $_GET['id'];
$conn = mysql_connect("localhost","sdiaz","sdiaz2008");
if(!$conn){
        die('&ERROR200&');
}
$con_result = mysql_select_db("sdiaz",$conn);
if(!$con_result){
        die('&ERROR201&');
}
$sql1 = "DELETE FROM switch WHERE id = $idSwitch";
$result1 = mysql_query($sql1);
if(!$result1){
        die('&ERROR202&');
}else{
        echo '&OK&';
}
mysql_close($conn);

?>
```

Servicio Download

```
<?php
$idSwitch = $_GET['id'];
$conn = mysql_connect("localhost","sdiaz","sdiaz2008");
if(!$conn){
        die('&ERROR200&');
}
$con_result = mysql_select_db("sdiaz",$conn);
if(!$con_result){
        die('&ERROR201&');
}
$sql1 = "SELECT * FROM updatephone WHERE id = $idSwitch";
$result1 = mysql_query($sql1);
if(!$result1){
        die('&ERROR202&');
}
$fila = mysql_fetch_row($result1);
    if($fila[1] == 1){
        $sql2 = "UPDATE updatephone  SET actualizacion = 0 where id =
            $idSwitch";
        mysql_query($sql2);
        if(!$result2){
        die('&ERROR203&');
    }
        $sql3 = "SELECT * FROM switch WHERE id = $idSwitch";
        $result3 = mysql_query($sql3);
        if(!$result3){
                die('&ERROR204&');
        }
        $row = mysql_fetch_row($result3);
        echo $row[0]."&".$row[1]."&".$row[2]."&".$row[3]."&".$row[4]."&
            ".$row[5]."&".$row[6]."&".$row[7];
}else{
        echo '&NO&';
}
mysql_close($conn);

?>
```

Servicio Upload

```
<?php
    $idSwitch = $_GET['id'];
    $nombreSwitch = $_GET["nombre"];
    $grupoSwitch = $_GET["grupo"];
    $estadoSwitch = $_GET['estado'];
    $luzSwitch = $_GET['luz'];
    $distanciaSwitch = $_GET['distancia'];
    $retardoSwitch = $_GET['retardo'];
    $luzOnSwitch = $_GET['encendido'];
    $conn = mysql_connect("localhost","sdiaz","sdiaz2008");
    if(!$conn){
        die('&ERROR200&');
    }
    $con_result = mysql_select_db("sdiaz",$conn);
    if(!$con_result){
        die('&ERROR201&');
    }
    $sql1 = "SELECT * FROM updatephone WHERE id = $idSwitch";
    $result1 = mysql_query($sql1);
    if(!$result1){
        //die(mysql_error());
        die('&ERROR202&');
    }
    $fila = mysql_fetch_row($result1);
    if($fila[1] == 0 || $fila[1] == 1){
        $sql2 = "UPDATE updateswitch  SET actualizacion = 1 WHERE id =
            $idSwitch";
        $result2 = mysql_query($sql2);
        if(!$result2){
                //die(mysql_error());
                die('&ERROR203&');
        }
        $sql4 = "UPDATE switch SET nombre='$nombreSwitch',grupo='
            $grupoSwitch', estado=$estadoSwitch, encendido=$luzOnSwitch,
            distancia=$distanciaSwitch,luz=$luzSwitch,retardo=
            $retardoSwitch WHERE id=$idSwitch";
        $result4 = mysql_query($sql4);
        if(!$result4){
                die('&ERROR204&');
```

```
        }else{
                echo '&OK&';
        }
    }else{
        echo '&NO&';
    }
    mysql_close($conn);

?>
```

3.4. Aplicación

La plataforma elegida para la aplicación móvil es iOS, ya que con la implementación de un código, la aplicación podrá ser ejecutada en múltiples tipos de dispositivos móviles de la marca Apple. Para dispositivos con pantallas más grandes debe reconsiderar las vistas de la aplicación.

3.4.1. Funcionalidades

La tarea de la aplicación es entregar una interfaz entre el usuario y el interruptor, de manera que el usuario pueda administrar un interruptor desde su dispositivo móvil, sin la necesidad de estar cercano a él.

A continuación se listan las principales funciones que debe realizar la aplicación.

Funcionalidades Aplicación	
Funcionalidad	Descripción
Mostrar estado de luz	Mostrar estado de la luz en modo manual
Listar interruptores	Mostrar todos los interruptores inscritos
Mostrar detalles de interruptor	Mostrar los valores registrados en el interruptor
Agregar interruptor	Agregar un interruptor a la aplicación
Eliminar interruptor	Eliminar de la aplicación un interruptor
Editar interruptor	Editar los valores del interruptor
Listar grupos	Listar los grupos creados
Mostrar detalles de grupos	Mostrar los valores inscritos en un grupo
Agregar grupos	Agregar un grupo a la aplicación
Eliminar grupos	Eliminar un grupo existente

Cuadro 3.16: Funcionalidades de Aplicación móvil

3.4.2. Diseño

El diseño esta pensado para cumplir cada una de las funcionalidades anteriormente descrita.

Vista mostrar estado de luz

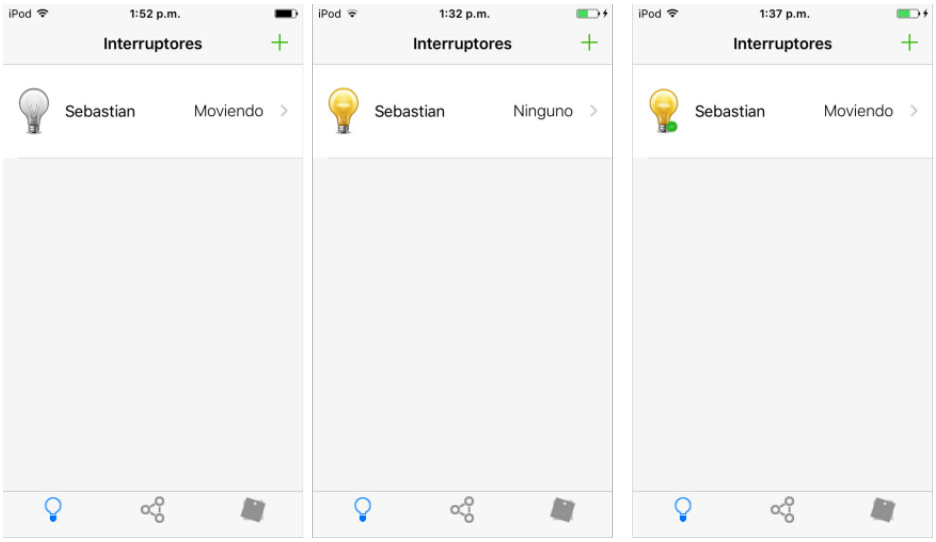

Figura 3.9: Vista estado de la luz aplicación

Como se muestra en la Figura 3.9, el estado de la luz se muestra en el icono de ampolleta que puede mostrar 3 estados encendido, apagado y automático, este icono es el que se muestra al lado izquierdo del nombre.

El estado real de la luz se ve reflejado en el modo manual. En modo automático aparece un símbolo verde que advierte que el interruptor esta en modo automático y no será reflejado el estado actual de la luz, esto por que en el modo manual es en el único que se vuelve relevante esa información, ya que es el único modo que puede tomar acciones en el interruptor si la luz esta encendida o apagada.

Vista listar interruptores y grupo

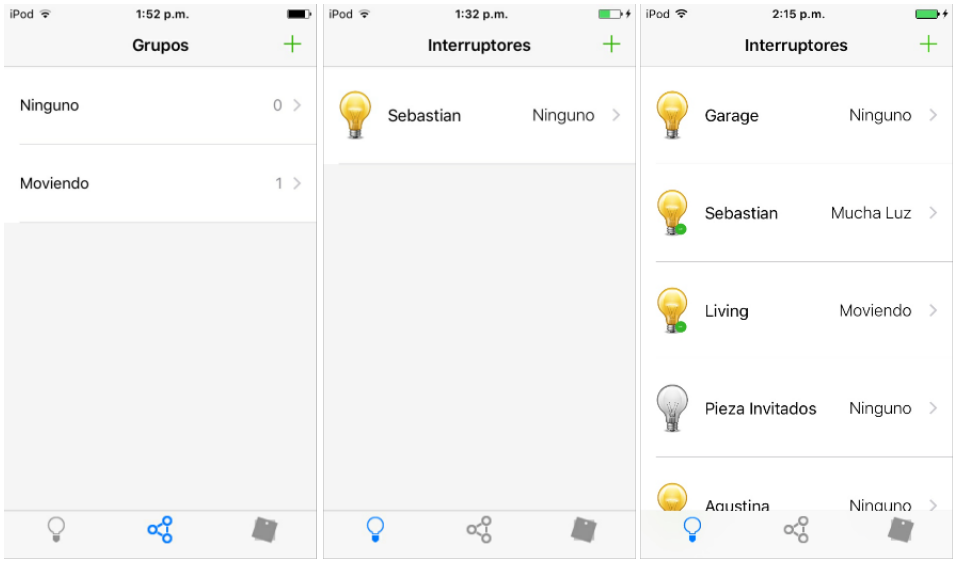

Figura 3.10: Vista listado de interruptores

Como se muestra en la figura 3.10 los interruptores registrados se muestran en formato lista, la que se crea a partir de los interruptores ya registrados, y van apareciendo o se van a medida que se agregan interruptores o se eliminan. Estas celdas muestran el detalle de la luz dependiendo del estado en que se encuentra el interruptor: el nombre y el grupo en el cual esta registrado el interruptor.

Este mismo formato es el utilizado para mostrar los grupos agregados a la aplicación, solo que en estas celdas se muestran dos campos, el nombre y la cantidad de interruptores registrados en el grupo.

Este formato de lista resulta cómodo para dispositivos donde la pantalla no es muy amplia, pero para dispositivos como ipad, es necesario reconsiderar el uso de listas, ya que la información podría ser distribuida de una manera diferente para aprovechar de mejor modo el tamaño de la pantalla.

Vista mostrar detalles de interruptor y grupo

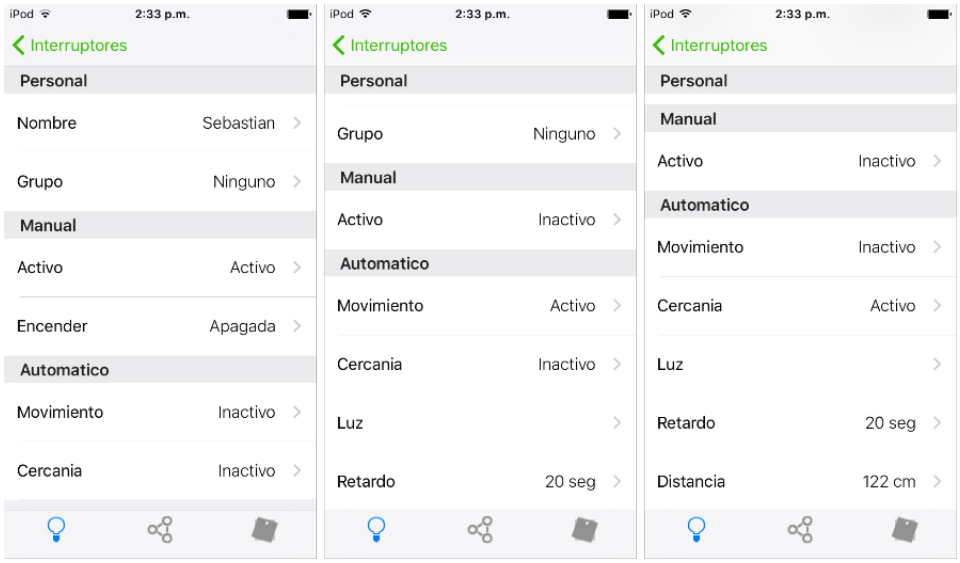

Figura 3.11: Vista detalle de un interruptor

Los interruptores muestran distinta información dependiendo del estado en el cual se encuentra, como se muestra en la Figura 3.11. Al estado normal se le permita la acción de encender o apagar, mientras en los estados automáticos se agregan los campos luz y retardo para el modo movimiento y el campo distancia para el modo distancia.

Este mismo formato es el aplicado para poder mostrar los detalles de un grupo, lo únicos campos que no aparecen son el campo grupo y la acción encender, el campo grupo no pertenece al modelo grupos y la acción encender no aparece porque los grupos definen lógica y/o comportamiento, no acciones; por lo tanto queda excluido de esta vista.

Vista agregar interruptor

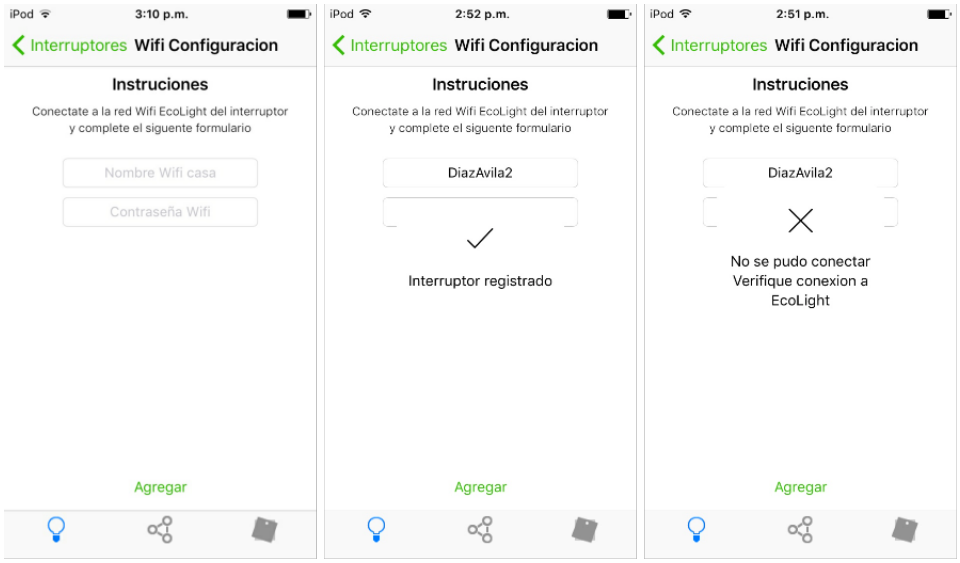

Figura 3.12: Vista agregar un interruptor

La vista de agregar interruptor permite ingresar los campos necesarios para el correcto funcionamiento del interruptor; estos parámetros son: SSID y contraseña de la red Wi-Fi a la que se conectará el interruptor. Además se tiene un botón agregar para que los parámetros sean enviados desde el dispositivo móvil al interruptor.

Esta vista puede entregar 2 mensajes al usuario, el primero cuando el interruptor se registró exitosamente y el segundo es cuando no se conectó con el interruptor por estar conectado a otra red Wi-Fi y no a la red Ecolight. Este mensaje también puede ser entregado cuando la conexión entre el iPhone y el interruptor caiga en timeout.

Vista eliminar interruptor y grupo

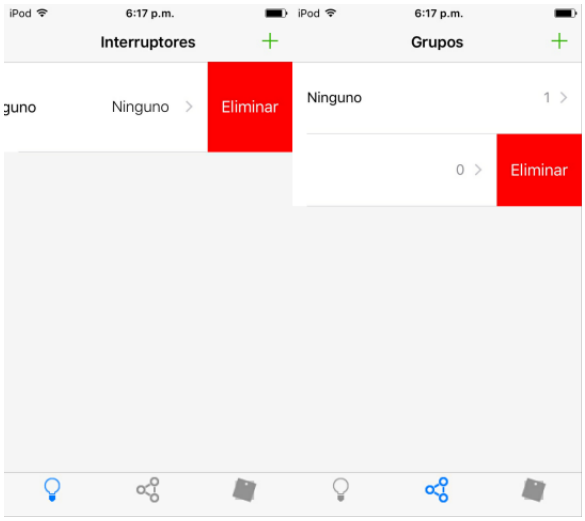

Figura 3.13: Vista eliminar un interruptor

La eliminación de un interruptor se hace con un *swipe* deslizando la celda hacia la izquierda lo que permite eliminar el interruptor de la aplicación.

Esta misma acción es la que se aplica para la funcionalidad de eliminar grupo, solo que se hace desde la vista lista de grupos.

Vista editar interruptor

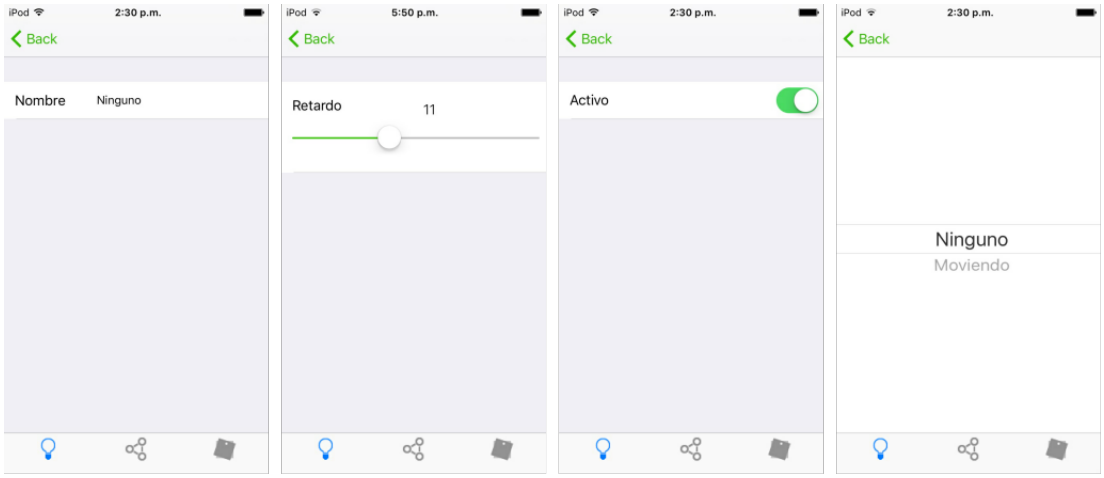

Figura 3.14: Vista editar un interruptor

Como se muestra en la Figura 3.14, la edición de los atributos de un interruptor se realiza seleccionando unos de los campos en la vista detalles de interruptor, ademas de un grupo. Esta acción puede entregar 4 posibles vistas para edición de texto: variable bajo un rango, variable booleana y selección.

Una vez ingresado el valor deseado, al volver a la pantalla vista de detalles de interruptor y al terminar toda la edición, y volver a la vista de listado de interruptores los campos serán guardados y enviados a la base de datos para que el interruptor pueda descargar los cambios.

Vista crear grupo

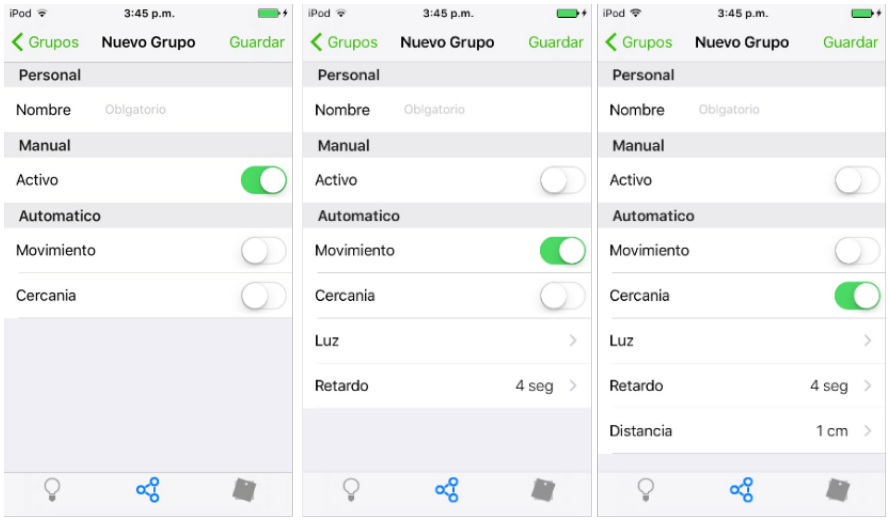

Figura 3.15: Vista Crear un grupo

Como se muestra en la Figura 3.15, la vista crear grupo respeta el formato lista anteriormente descrito, pero permite la edición del nombre y de la acciones de manera inmediata, sin una vista intermedio. En cuanto activa el modo que pertenecerá al grupo aparecen los campos que necesita ingresar.

4. Implementación

Al ser el interruptor un objeto físico, este requiere ser instalado y ser configurado para que la aplicación pueda cambiar sus modos y valores.

Instalación

Para instalar el prototipo se debe hacer una instalación muy simple. Un interruptor puede controlar desde 1 hasta 3 ampolletas, que contemplarán la siguiente instalación eléctrica.

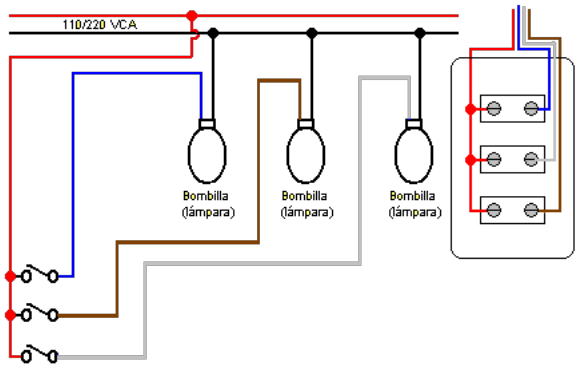

Figura 4.1: Instalación eléctrica de 3 ampolletas

Como podemos ver en la Figura 4.1, existe un cable común para todas las ampolletas que se quieran controlar, en este caso seria el rojo, mientras que los cables azul, café y gris controlan cada una de las ampolletas.

Cabe señalar que el esquema de la Figura 4.2, es una demostración de cómo el interruptor en su versión final debería conectarse a la red eléctrica de la casa, ya que para el prototipo creado, solo contempla el control de 2 ampolletas y estos cables

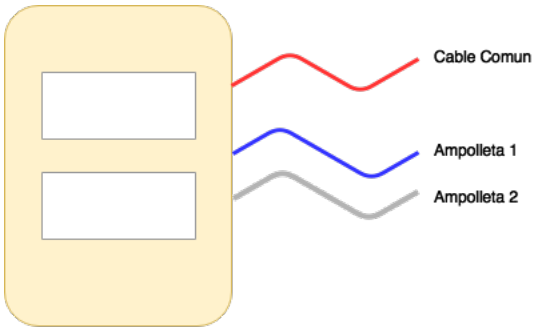

Figura 4.2: Conexión cables interruptor

deben ser conectados al relé descrito en la unidad de procesamiento de sensores, como se muestra en la siguiente Figura 4.3.

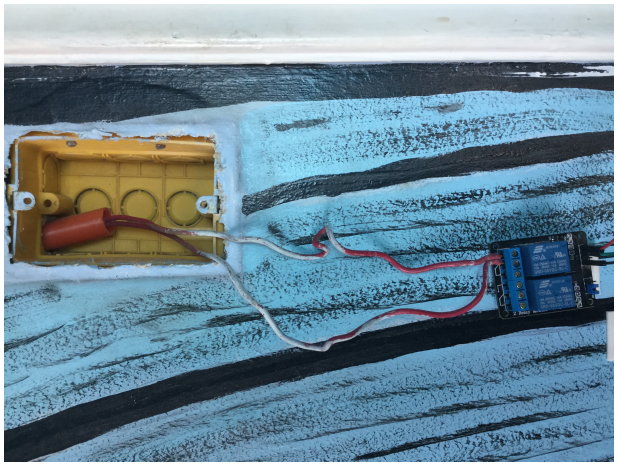

Figura 4.3: Conexión cables relé

Configuración

Cuando el interruptor está instalado este de manera inmediata en modo manual. Esto permitirá probar al usuario que el interruptor funciona independiente de las configuraciones que puedan ser ingresadas.

Figura 4.4: Red Wi-Fi creada por interruptor

Una vez instalado, creará una red Wi-Fi con el nombre EcoLightXXXX y sin contraseña. Esto permitirá al dispositivo conectarse a la red y tener una comunicación directa a través de Socket con el interruptor por medio de una IP estática que éste generá : **192.168.4.1**.

Cuando los parámetros para agregar un nuevo interruptor son agregados, son enviados en el siguiente formato **ssid=ssidIngresada&pass=passIngresada**. El dispositivo hace un parser de la información y se conecta a la red Wi-Fi.

Una vez teniendo conexión a internet, se deja esperando la respuesta al dispositivo e inserta información estática en la base de datos. Con este paso nos aseguramos que el interruptor se agregó de forma correcta, permitiendo responder el mensaje **OK&IDinterruptor** al dispositivo. Este procesa la información y registra un interruptor con el ID entregado en el mensaje.

Con el interruptor registrado en la base de datos y finalizada la respuesta al teléfono, el interruptor verificará cada 3 segundos si debe descargar la información que se encuentra en el servidor. De haber cambios, son descargados en el formato descrito en las respuestas posibles del servidor y se guardan.

Con los pasos antes descritos el interruptor se da por registrado y podemos proceder a cambiar los parámetros permitidos.

5. Pruebas

Las pruebas que fueron realizadas en el proyecto corresponden a ejecuciones de variados escenarios de los requisitos antes descritos en este documento, que contemplan los escenarios a los cuales se puso a prueba el proyecto.

Nombre prueba	Interruptor crea red Wi-fi caso normal.
Nombre requisito	Interruptor crea red Wi-fi.
Involucrados	Interruptor.
Escenario	Caso normal.
Pre-requisitos	Interruptor conectado a fuente energía.
Resultado esperado	Se crea red Wi-Fi con nombre Ecolight1.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.1: Prueba interruptor crea Red Wi-fi

Nombre prueba	Obtener IP del interruptor caso normal.
Nombre requisito	Obtener IP del interruptor.
Involucrados	Interruptor. Dispositivo Móvil.
Escenario	Caso normal.
Pre-requisitos	Interruptor conectado a fuente energía. Red Wi-Fi creada por el interruptor.
Resultado esperado	El dispositivo obtiene IP del interruptor.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna .

Cuadro 5.2: Prueba obtener IP del interruptor.

Nombre prueba	Obtener IP del interruptor con otro previamente conectado.
Nombre requisito	Obtener IP del interruptor.
Involucrados	Interruptor dispositivo Móvil.
Escenario	Otro dispositivo ya conectado.
Pre-requisitos	Interruptor conectado a fuente energía. Red Wi-Fi creada por el interruptor. Un dispositivo ya conectado a la red Wi-Fi.
Resultado esperado	El dispositivo obtiene IP del interruptor.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.3: Prueba obtener IP del interruptor con otro previamente conectado.

Nombre prueba	Encender y/o apagar luces caso normal.
Nombre requisito	Encender y/o apagar luces.
Involucrados	Interruptor. Servicio web. Aplicación móvil.
Escenario	Dispositivo configurado.
Pre-requisitos	Dispositivo agregado a la aplicación. El dispositivo esta en modo manual.
Resultado esperado	Las luces cambian de estado al presionar los botones. La aplicación refleja el cambio de las luces.
Resultado obtenido	Resultado correcto.
Observaciones	Al Cambiar las luces de forma muy rápida, son varios los cambios que intentar subirse pero solo es considerado el último.

Cuadro 5.4: Prueba encender y/o apagar luces caso normal.

Nombre prueba	Encender y/o apagar luces modo automático.
Nombre requisito	Encender y/o apagar luces.
Involucrados	Interruptor. Servicio web. Aplicación móvil.
Escenario	Dispositivo Configurado.
Pre-requisitos	Dispositivo agregado a la aplicación. El dispositivo esta en modo automático.
Resultado esperado	Las luces cambian de estado al presionar los botones. Las luces cambian de modo al presionar los botones. La aplicación refleja el cambio de las luces.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.5: Prueba encender y/o apagar luces modo automático.

Nombre prueba	Encender y/o apagar luces sin configuración.
Nombre requisito	Encender y/o apagar luces.
Involucrados	Interruptor.
Escenario	Dispositivo no configurado.
Pre-requisitos	El dispositivo está conectado a fuente energía.
Resultado esperado	Las luces cambian de estado al presionar los botones.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.6: Prueba encender y/o apagar luces sin configuración.

Nombre prueba	Agregar un interruptor caso normal.
Nombre requisito	Agregar un interruptor.
Involucrados	Interruptor. Servicio web. Aplicación móvil.
Escenario	Caso normal.
Pre-requisitos	El dispositivo está conectado a fuente energía. El dispositivo IP de la Red Wi-Fi. ingresa datos correctamente.
Resultado esperado	Mensaje de interruptor registrado.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.7: Prueba agregar un interruptor caso normal.

Nombre prueba	Agregar un interruptor caso datos mal ingresados.
Nombre requisito	Agregar un interruptor.
Involucrados	Interruptor. Servicio web. Aplicación móvil.
Escenario	Datos mal ingresados por el usuario.
Pre-requisitos	El dispositivo está conectado a fuente energía. El dispositivo IP de la Red Wi-Fi. Ingresa datos correctamente.
Resultado esperado	Mensaje de interruptor no registrado reintente.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.8: Prueba agregar un interruptor caso datos mal ingresados.

Nombre prueba	Agregar un interruptor conectado a otra red.
Nombre requisito	Agregar un interruptor.
Involucrados	Interruptor. Servicio web. Aplicación móvil.
Escenario	Dispositivo conectado a a ninguno y/o a otra red.
Pre-requisitos	El dispositivo está conectado a fuente energía. El conectado a Red Wi-Fi. Ingresa datos correctamente.
Resultado esperado	Mensaje de interruptor no registrado reintente.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.9: Prueba agregar un interruptor conectado a otra red.

Nombre prueba	Eliminar interruptor caso normal.
Nombre requisito	Eliminar interruptor.
Involucrados	Aplicación Móvil.
Escenario	Caso normal.
Pre-requisitos	El interruptor registrado en la aplicación.
Resultado esperado	Interruptor no aparece en la lista.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.10: Prueba eliminar interruptor caso normal.

Nombre prueba	Cambiar de Modo el interruptor caso normal.
Nombre requisito	Cambiar de Modo el interruptor.
Involucrados	Aplicación Móvil. Servicio web. Interruptor.
Escenario	Caso normal.
Pre-requisitos	El interruptor registrado en la aplicación.
Resultado esperado	La aplicación refleja los cambios . El interruptor aplica los cambios .
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.11: Prueba cambiar de Modo el interruptor caso normal.

Nombre prueba	Cambiar de Modo el interruptor sin internet.
Nombre requisito	Cambiar de Modo el interruptor.
Involucrados	Aplicación Móvil. Servicio web. Interruptor.
Escenario	Caso sin internet.
Pre-requisitos	El interruptor registrado en la aplicación.
Resultado esperado	El interruptor no cambia su configuración .
Resultado obtenido	Resultado correcto.
Observaciones	La aplicación no arroja ningún mensaje de que la configuración se no se guardó por falta de internet.

Cuadro 5.12: Prueba cambiar de Modo el interruptor sin internet.

Nombre prueba	Encender luces con movimiento caso normal.
Nombre requisito	Encender luces con movimiento.
Involucrados	Aplicación Móvil. Servicio web. Interruptor.
Escenario	Caso normal.
Pre-requisitos	El interruptor registrado en la aplicación. El interruptor en modo movimiento activado.
Resultado esperado	El interruptor enciende luz al movimiento .
Resultado obtenido	Resultado correcto.
Observaciones	Tiene una demora de alrededor de 1 segundo de detectar el movimiento.

Cuadro 5.13: Prueba encender luces con movimiento caso normal.

Nombre prueba	Encender luces con distancia caso normal.
Nombre requisito	Encender luces con distancia.
Involucrados	Aplicación Móvil. Servicio web. Interruptor.
Escenario	Caso normal.
Pre-requisitos	El interruptor registrado en la aplicación. El interruptor en modo distancia activado.
Resultado esperado	El interruptor enciende luz al pasar el limite máximo .
Resultado obtenido	Resultado correcto.
Observaciones	La distancia tiende a ser muy relativa, ya que el cuerpo humano no es regular y estos sensores son buenos con objetos de superficie regular.

Cuadro 5.14: Prueba encender luces con distancia caso normal.

Nombre prueba	Agregar grupo de comportamiento caso normal.
Nombre requisito	Agregar grupo de comportamiento.
Involucrados	Aplicación Móvil.
Escenario	Caso normal.
Pre-requisitos	Ninguno.
Resultado esperado	El grupo agregado aparece en la lista de grupos .
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.15: Prueba agregar grupo de comportamiento caso normal.

Nombre prueba	Eliminar grupo de comportamiento caso normal.
Nombre requisito	Eliminar grupo de comportamiento.
Involucrados	Aplicación Móvil.
Escenario	Caso normal.
Pre-requisitos	Debe existir un grupo agregado a la aplicación.
Resultado esperado	El grupo no aparece en la lista de grupos. Los interruptores en el grupo vuelven al estado normal.
Resultado obtenido	Resultado correcto.
Observaciones	Ninguna.

Cuadro 5.16: Prueba eliminar grupo de comportamiento caso normal.

6. Conclusión

Se puede concluir que el desarrollo de este dispositivo es una solución más integral al problema de uso eficiente de la luz eléctrica en el hogar, ya que permite, tal como lo fue por la ACHEE, tener un dispositivo de bajo costo que puede ser adquirido masivamente y pueda controlar el encendido de la luz solo cuando sea necesario.

Las hipótesis planteadas al inicio, fueron cumplidas, ya que el interruptor fue capaz de realizar cada uno de los puntos. El único punto que puede quedar pendiente a discusión es el costo del proyecto con respecto a instalar ampolletas inteligentes, ya que aproximadamente tener tres luces encendidas por esta tecnología, en Chile sale $149.000 de la mano de las ampolletas Hue de Philips. El prototipo de interruptor salio aproximadamente $17.000 pero solo contempla el prototipo, a esto hay que agregar toda la transformación de estos módulos de aprendizaje a un circuito integrado y la creación del interruptor físicamente. Por lo tanto no puedo asegurar que el interruptor producido en masa y comercializado podría llegar a ser más barato que las ampolletas inteligentes.

El dispositivo solo fue probado en una habitación de un hogar, dando las pruebas resultados exitosos, en cada uno de los modos, siendo el modo distancia el que presentó algunos detalles que fueron anotados en las pruebas.

Existe hardware que fue utilizado en este proyecto que fácilmente podría ser reemplazado o incluso excluido de una versión definitiva del dispositivo. Ejemplo de esto son los siguientes:

El arduino mega 2560 podría ser fácilmente reemplazado por uno de menos capacidades. No se realizó de esta manera porque, era necesario depurar de alguna manera los programas para saber que era lo que estaba ocurriendo y facilitar la programación.

El sensor de luz entrega una manera de medir la cantidad de luz que existente en el ambiente, pero en la realidad, no existe una medida cotidiana que permita medir esta cantidad de luz. Este sensor podría ser omitido y utilizar horas del día para establecer el modo automático del sensor, esto permitiría una mayor comprensión de la configuración al usuario.

Tal como se ha discutido, este proyecto está enfocado en el uso eficiente de la luz eléctrica. Según estudios realizados por la empresa bticino, solo con la instalación de un módulo de movimiento se puede alcanzar un ahorro del 55 %, pero para comprobar que el interruptor trae un ahorro de energía, primero se debe comprobar cúanta es la energía que esta consumiendo para comprobar si el sistema corriendo en conjunto con ampolletas led consume menos que el sistema completo.

7. Trabajo Futuro

Este prototipo de interruptor inteligente es una buena idea, y sienta una base sólida para poder automatizar un interruptor a través de una aplicación. No obstante existen puntos en los que se puede mejorar. El primer desafío es poder integrar todo en un circuito que pueda ser compacto permitiendo utilizar un espacio proporcional a los actuales interruptores. Esto permitiría volver mas comercial el producto ya que no confrontaría el actual estándar que existe de interruptor.

El segundo desafió que quedo pendiente en el desarrollo de este proyecto es poder entregar más información de las horas que se encuentra encendida una ampolleta, esto permitiría al usuario tener un control de cuales podrían ser cambiadas por unas de mayor ahorro. Además, se podría integrar la capacidad de almacenar cuentas de usuario para relacionar interruptores con ellas, permitiendo recuperar información solo accediendo con la cuenta del usuario respectivo, permitiendo ampliar el software de control a una página web por ejemplo.

Con respecto a la estructura montada, puede ser optimizada en algunas de las etapas. A nivel de base de datos podría ser fácilmente comprimida a dos tablas, una con la información del interruptor y otra que maneje todas las actualizaciones.

Un punto importante en este proyecto es el nivel de consistencia que debe existir del estado del interruptor con respecto a lo entregado por la aplicación, esto actualmente se trabaja solo actualizando la información de los dispositivos, ya sea del interruptor o de la aplicación. Este modelo funciona bien si las actualizaciones son consecutivas y no concurrentes. Esto podría mejorarse tomando la decisión de qué información tiene más importancia para mantenerla en el servidor.

Bibliografía

[1] Achee. Qué es la agencia. http://www.acee.cl/nosotros/que-es-la-agencia/.

[2] Lady Ada. Tutorial sensor ultrasónico hc-sr04 y arduino. https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/overview.

[3] Arduino. Que es arduino. http://arduino.cl/que-es-arduino/.

[4] Arduino. Arduino mega 2560. https://www.arduino.cc/en/Main/ArduinoBoardMega2560.

[5] Arduino. Arduino nano. https://www.arduino.cc/en/Main/ArduinoBoardNano.

[6] Arduino. Tinkerkit relay module. https://store.arduino.cc/product/T010010.

[7] Quiroz & Asociados. Estudio de previsión de demanda 2015-2035 (2050). http://www.cdecsic.cl/wp-content/uploads/2015/06/Informe-Preliminar-Estudio-de-Previsión-de-Demanda-2015-2035-2050.pdf.

[8] Inc Belkin International. Wemo light switch. http://www.belkin.com/us/F7C030-Belkin/p/P-F7C030;jsessionid=A740A3C8EE8E04B33CAB9F20D2F7D9FA/.

[9] CNE. Anuario estadistico de energia. http://www.cne.cl/wp-content/uploads/2016/07/AnuarioCNE2015_vFinal-Castellano.pdf.

[10] Ministerio de energía. Aprendamos a ahorrar. guía práctica de la buena energía. http://www.minenergia.cl/ganamostodos/docweb/GUia_practica_AChee.pdf.

[11] Rafael Enríquez Herrador. Guía de usuario de arduino. http://www.jcarazo.com/tmp/Arduino_user_manual_es.pdf.

[12] INE. Enfoque estadÍstico - energía. http://www.ine.cl/canales/sala_prensa/archivo_documentos/enfoques/2008/septiembre/energia_pag.pdf.

[13] Jesus Ruben. Tutorial sensor ultrasónico hc-sr04 y arduino. http://www.geekfactory.mx/tutoriales/tutoriales-arduino/sensor-ultrasonico-hc-sr04-y-arduino/.

[14] switchmate. switchmate smart home simplefied. https://www.myswitchmate.com.

[15] Espressif Systems IOT Team. Esp8266ex datasheet. http://download.arduino.org/products/UNOWIFI/0A-ESP8266-Datasheet-EN-v4.3.pd.

[16] tkkrlab Team. Arduino ky-018 photo resistor module. https://tkkrlab.nl/wiki/Arduino_KY-018_Photo_resistor_module.

[17] Wikipedia. Php. https://es.wikipedia.org/wiki/PHP.

ANEXOS

A. Código fuente

A continuación se muestra el código fuente escrito en arduino para la sección hardware del interruptor, esto corresponde al código que ejecuta la unidad de procesamiento de sensores y la unidad de procesamiento Wi-Fi

A.1. Código unidad procesamiento de sensores

```
#include <Button.h>
#include <SoftwareSerial.h>

//********Serial Comunicación***********
const int TX = 2;
const int RX = 3;
SoftwareSerial SerialArduino(RX, TX);
//********Parametros***********
String id = "1";
String nombre = "Ninguno";
String grupo = "Ninguno";
int estado = 1;
int luzOn = 0;
int distancia = 100;
int luz = 1;
int tiempoencendida = 1000; //10 seg
//********sensores***********
//Movimiento
int pirPin = 6;

//distancia
const int echo = 8;
```

```
const int trig = 9;

//ampolletas
const int pinAmpolleta = 4;
const int pinAmpolleta2 = 5;


//luz
const int LDR =A0;

//botones
Button button = Button(12);
Button button2 = Button(11);

//tiempo transcurrido para sensores
unsigned long tiemposensor1=0;
unsigned long tiemposensor2=0;
//********estado y botones luz***********
bool ampolleta = false;
bool ampolleta2 = false;

unsigned long time;

void onPress(Button& b){
    grupo= "Ninguno";
    estado=1;
    if(ampolleta){
      ampolleta=false;
    }else{
      ampolleta=true;
    }
    if(!ampolleta && !ampolleta2){
       luzOn = false;
    }else{
       luzOn = true;
    }
    SerialSendData();
}

void onPress2(Button& b){
    grupo= "Ninguno";
```

```
      estado=1;
      if(ampolleta2){
        ampolleta2=false;
      }else{
        ampolleta2=true;
      }
      if(!ampolleta && !ampolleta2){
         luzOn = false;
      }else{
         luzOn = true;
      }
      SerialSendData();

  }

  void setup(){
    Serial.begin(9600);
    SerialArduino.begin(9600);
    pinMode(pirPin, INPUT);

    pinMode(pinAmpolleta, OUTPUT);
    pinMode(pinAmpolleta2, OUTPUT);

    pinMode(trig, OUTPUT); /*activación del pin 9 como salida: para
        el pulso ultrasónico*/
    pinMode(echo, INPUT); /*activación del pin 8 como entrada: tiempo
         del rebote del ultrasonido*/
    // Assign callback function
    button.pressHandler(onPress);
    button2.pressHandler(onPress2);
  }

  void loop(){
    unsigned long currentMillis = millis();
    switch(estado){
      case 1:
        imprimirInterruptor();
        Serial.println(analogRead(LDR));
        break;

      case 2:
```

```
        //Movimiento
        imprimirInterruptor();
        if(analogRead(LDR) <= luz){
         if(digitalRead(pirPin) == HIGH){
           ampolleta=true;
           ampolleta2=true;
         }else{
              if ((unsigned long)(currentMillis - tiemposensor1) >
                 tiempoencendida) {
                  ampolleta=false;
                  ampolleta2=false;
                  tiemposensor1 = currentMillis;
               }
         }
        }
        break;

      default:
         //distancia
         imprimirInterruptor();
        if(analogRead(LDR) <= luz){
          if(calcularDistancia() <= distancia){
             ampolleta=true;
             ampolleta2=true;
          }else{
               if ((unsigned long)(currentMillis - tiemposensor2) >
                  tiempoencendida) {
                   ampolleta=false;
                   ampolleta2=false;
                   tiemposensor2 = currentMillis;
                }
          }
        }
         break;
    }
    encenderLuz();
    SerialInputData();
    // update the buttons' internalz
    button.process();
    button2.process();
  }
```

```
void encenderLuz(){
  if(ampolleta){
    digitalWrite(pinAmpolleta,HIGH);
  }else{
    digitalWrite(pinAmpolleta,LOW);
  }
  if(ampolleta2){
    digitalWrite(pinAmpolleta2,HIGH);
  }else{
    digitalWrite(pinAmpolleta2,LOW);
  }

}

void imprimirInterruptor(){
  Serial.println("Estado:"+String(estado)+" Grupo:"+grupo);
}


void SerialInputData(){
  char caracter;
  String data="";
  while(SerialArduino.available()){
    caracter = SerialArduino.read();
    data.concat(caracter);
    delay(10);
  }
  if(data!=""){
    //1&esadmaskndpasndpasnd&Ninguno&1&1&1&1&1
    //id&nombre&grupo&estado&encendido&distancia&luz&retardo;
     data.trim();
     Serial.println(data);
      //preguntar si viene con el formato para asegurar

     nombre = valueString(1,data);
     grupo = valueString(2,data);
     estado = valueString(3,data).toInt();
     luzOn = valueString(4,data).toInt();
     distancia = valueString(5,data).toInt();
     luz = valueString(6,data).toInt();
```

```
      int tiempoencendidatmp = valueString(7,data).toInt();
      tiempoencendida = tiempoencendidatmp*1000;
      if(estado == 1){
         if(luzOn == 1){
           ampolleta = true;
           ampolleta2 = true;
         }else{
           ampolleta = false;
           ampolleta2 = false;
         }
      }
  }
}

void SerialSendData(){
  Serial.println("Enviando datos");
  String data = String(id)+"&"+nombre+"&"+grupo+"&"+String(estado)+
      "&"+String(luzOn)+"&"+String(distancia)+"&"+String(luz)+"&"+
      String(tiempoencendida);
  SerialArduino.println(data);
}


String valueString(int posicion,String data){
  for(int i=0;i< posicion;i++){
      data = data.substring(data.indexOf("&")+1);
    }
   return data.substring(0,data.indexOf("&"));
}


long calcularDistancia(){
  long distancia2;
  long tiempo;

  digitalWrite(trig,LOW); /* Por cuestión de estabilización del
      sensor*/
  delayMicroseconds(5);

  digitalWrite(trig, HIGH); /* envío del pulso ultrasónico*/
  delayMicroseconds(10);
```

```
  tiempo=pulseIn(echo, HIGH); /* Función para medir la longitud del
       pulso entrante. Mide el tiempo que transcurrido entre el
      envío
  del pulso ultrasónico y cuando el sensor recibe el rebote, es
      decir: desde que el pin 8 empieza a recibir el rebote, HIGH,
      hasta que
  deja de hacerlo, LOW, la longitud del pulso entrante*/

  distancia2= int(0.017*tiempo); /*fórmula para calcular la
      distancia obteniendo un valor entero*/
  /*Monitorización en centímetros por el monitor serial*/
  return distancia2;

}
```

A.2. Código unidad de procesamiento Wi-Fi

```
/* id del interruptor que se almacena aquí para no tener que
    pedirlo al interruptor, al momento de insertar los datos */
int id =1;


/* define la actividad en la que se encuentra, 1 = configuración,
    2= preparado para recivir datos de Wi-Fi, 3= configuracion para
    descargar datos , 4= descarga de datos del servidor */
int activity = 3;

void setup() {
  Serial.begin(9600);
  Serial1.begin(9600);
  Serial2.begin(9600);
  delay(1000); // Let the module self-initialize

}

void loop() {
    switch(activity){
      case 1:
          sendCommand("AT+CWQAP",300);
```

```
            sendCommand("AT+CWMODE=3",300);
            sendCommand("AT+CIPMUX=1",300);
            sendCommand("AT+CIPSERVER=1,80",300);
            sendCommand("AT+CWSAP=\"EcoLight1\",\"\",1,0",300);
            activity = 2;
            break;

        case 2:
            wifiConfiguration();
            break;

        case 3:
            delay(1000);
            sendCommand("AT+CIPSERVER=0",300);
            sendCommand("AT+CIPMUX=0",300);
            sendCommand("AT+CWMODE=1",300);
            sendCommand("AT+CIPCLOSE",300);
            activity = 4;
            break;

        default:
            DowloadData();
            break;

    }
}


void sendCommand(String command, int retardo){
    char dato;
    String data ="";
    Serial.println(command);
    Serial1.println(command);
    delay(retardo);
    while(Serial1.available()){
       dato = Serial1.read();
       data.concat(dato);

    }

}
```

```
String sendCommandData(String command, int retardo){
    char dato;
    String data ="";
    bool enviado = false;
    Serial.println(command);
    Serial1.println(command);
    delay(retardo);
    while(!enviado){
       while(Serial1.available()){
          dato = (char)Serial1.read();
          data.concat(dato);
       }
       if( data.indexOf("OK") >=0 || data.indexOf("FAIL") >=0 ||
           data.indexOf("ERROR") >=0 || data.indexOf("IP") >=0 ||
           data.indexOf(">") >=0){
          enviado = true;
       }
    }
    return data;
}

String sendURL(String URL, int retardo){
    char dato;
    String data ="";
    bool enviado = false;
    Serial.println(URL);
    Serial1.println(URL);
    while(!enviado){
       while(Serial1.available()){
          dato = (char)Serial1.read();
          data.concat(dato);
       }
       if( data.indexOf("CLOSED") >=0 ){
          enviado = true;
       }
    }
    return data;
}
```

```
void wifiConfiguration(){
    char dato;
    String data ="";
    while(Serial1.available()){
       dato = (char)Serial1.read();
       data+=dato;
       delay(30);
    }
    if( data.indexOf("IPD")>=0 && data.indexOf("ssid")>=0 && data.
        indexOf("pass")>=0){
       String idPhone = data.substring(data.indexOf("IPD")+4,data.
           lastIndexOf(","));
       String ssid = data.substring(data.indexOf("ssid")+5,data.
           indexOf("&"));
       String pass = data.substring(data.indexOf("pass")+5);
       String RespuestaPhone = "ERROR";
       imprimir("idconect ",idPhone);
       data = sendCommandData("AT+CWJAP=\""+ssid+"\",\""+pass+"\""
           ,5000);
       if(data.indexOf("IP") >=0){
          String respuesta = UploadDataID("id=1&nombre=Ninguno&
              grupo=Ninguno&estado=1&luz=1&distancia=1&retardo=1&
              encendido=1");
          if(respuesta.indexOf("OK")>=0){
            RespuestaPhone = "OK&1";
            activity = 3;
          }
       }
        data = sendCommandData("AT+CIPSEND="+idPhone+","+String(
            RespuestaPhone.length()),300);
        if(data.indexOf(">")>=0){
          data = sendCommandData(RespuestaPhone,300);
          imprimir("repsuesta del telefono:",data);
        }
        sendCommand("AT+CIPCLOSE=5",300);
    }
}
```

```
void imprimir(String mensaje,String valor){
  Serial.println(mensaje+valor);
}


String UploadDataID(String variables){
  String data = "";
  data = sendCommandData("AT+CIPSTART=1,\"TCP\",\"sdiaz.nosze.co
      \",80",1000);
  if(data.indexOf("OK")>= 0){
    String URL = "GET http://sdiaz.nosze.co/prueba1/switch/Upload.
        php?"+variables;
    data = sendCommandData("AT+CIPSEND=1,"+String(URL.length()+2)
        ,300);
    if(data.indexOf(">")){
      data = sendCommandData(URL,300);
      if(data.indexOf("OK") >= 0){
        return "OK";
      }else{
        return "ERROR";
      }
    }else{
      return "ERROR";
    }
  }
}


String UploadData(String variables){
  String data = "";
  data = sendCommandData("AT+CIPSTART=\"TCP\",\"sdiaz.nosze.co\",80
      ",1000);
  if(data.indexOf("OK")>= 0){
    String URL = "GET http://sdiaz.nosze.co/prueba1/switch/Upload.
        php?"+variables;
    data = sendCommandData("AT+CIPSEND="+String(URL.length()+2)
        ,300);
    if(data.indexOf(">")){
      data = sendCommandData(URL,300);
      if(data.indexOf("OK") >= 0){
```

```
          return "OK";
        }else{
          return "ERROR";
        }
      }else{
        return "ERROR";
      }
    }
}




void DowloadData(){
  String data = "";
  data = sendCommandData("AT+CIPSTART=\"TCP\",\"sdiaz.nosze.co\",80
      ",700);

  if(data.indexOf("OK")>= 0){
    String URL = "GET http://sdiaz.nosze.co/prueba1/switch/Download
        .php?id=1";
    data = sendCommandData("AT+CIPSEND="+String(URL.length()+2)
        ,300);

    if(data.indexOf(">")){
       data = sendURL(URL,600);
       if(data.indexOf("CLOSED")>=0){
          String respuesta = data.substring(data.indexOf(":")+1,
              data.indexOf("CLOSED"));
          Serial.println(respuesta);
          if(respuesta.indexOf("&")>=0){
              Serial2.println(respuesta);
          }
       }
    }
  }
  sendCommand("AT+CIPCLOSE",300);
}

void serialEvent2(){
    //1&esadmaskndpasndpasnd&Ninguno&1&1&1&1&1
```

```
        //id&nombre&grupo&estado&encendido&distancia&luz&retardo;
        char dato;
        String data ="";
        while(Serial2.available()){
            dato = (char)Serial2.read();
            data+=dato;
        }
        if(data.indexOf("&")>=0){
             String variables= "id=1&nombre="+valueString(1,data)+"&
                 grupo="+valueString(2,data)+"&estado="+valueString(3,
                 data)+"&encendido="+valueString(4,data)+"&distancia="+
                 valueString(5,data)+"&luz="+valueString(6,data)+"&
                 retardo="+valueString(7,data)+"";
             Serial.println(data);
             variables.trim();
             String llegada = UploadData(variables);
             if(llegada.indexOf("OK")>=0){
                   Serial.println("Envio OK");
              }else{
                    Serial.println("Envio ERROR");
              }
        }
    }


    String valueString(int posicion,String data){
      for(int i=0;i< posicion;i++){
          data = data.substring(data.indexOf("&")+1);
        }
       return data.substring(0,data.indexOf("&"));
    }
```