



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA ELECTRÓNICA

DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN ANDROID
PARA EL CONTROL INALÁMBRICO DE SEGURIDAD,
ILUMINACIÓN Y AUDIO MEDIANTE UNA CENTRAL BASADA EN
ARDUINO PARA IPREX

Trabajo de titulación presentado para optar al grado académico de:
INGENIERO EN ELECTRÓNICA, TELECOMUNICACIONES Y
REDES

AUTORES: ERICA PAULINA PADILLA UVIDIA
JHON HENRY LOAYZA MERIZALDE
TUTOR: ING. WILLIAM CALVOPÍÑA

Riobamba-Ecuador

2016

© 2016, Érica Padilla & Jhon Loayza.

Se autoriza la reproducción total o parcial, con fines académicos, por cualquier medio o procedimiento, incluyendo la cita bibliográfica del documento, siempre y cuando se reconozca el Derecho de Autor.

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA ELECTRÓNICA

El tribunal del trabajo de titulación certifica que: El trabajo de investigación: “DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN ANDROID PARA EL CONTROL INALÁMBRICO DE SEGURIDAD, ILUMINACIÓN Y AUDIO MEDIANTE UNA CENTRAL BASADA EN ARDUINO PARA IPREX, de responsabilidad de los señores Érica Paulina Padilla Uvidia y Jhon Henry Loayza Merizalde, ha sido minuciosamente revisado por los Miembros del Tribunal del trabajo de titulación, quedando autorizada su presentación.

NOMBRE	FIRMA	FECHA
Dr. Miguel Tasambay Ph. D DECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ING. Franklin Moreno DIRECTOR DE LA ESCUELA DE ELECTRÓNICA TELECOMUNICACIONES Y REDES
Ing. William Calvopiña DIRECTOR DE TRABAJO DE TITULACIÓN
Ing. Edwin Altamirano MIEMBRO DEL TRIBUNAL

Nosotros, Érica Paulina Padilla Uvidia y Jhon Henry Loayza Merizalde somos responsables de las ideas, doctrinas y resultados expuestos en este Trabajo de titulación el patrimonio intelectual de la misma pertenece a la Escuela Superior Politécnica de Chimborazo

Érica Paulina Padilla Uvidia

Jhon Henry Loayza Merizalde

DEDICATORIA

Este trabajo lo dedicamos con muchos amor y cariño a nuestros padres que nos dieron la vida, el apoyo y han estado con nosotros en todo momento.

A nuestros hermanos que han sido fuente de nuestra inspiración y fuerza para seguir adelante, alcanzando éxitos y ser ejemplo para ellos.

A nuestros sobrinos por ser aquella muestra de inocencia y amor constante en nuestra existencia.

Érica y Jhon

AGRADECIMIENTOS

Agradecemos primeramente a Dios por permitirnos realizar este trabajo con el cual culminamos una etapa más de nuestra vida y cumplimos nuestro sueño más anhelado.

A nuestras familias por darnos el apoyo incondicional y nunca desampararnos. En especiales a nuestros padres que han sido el motor fundamental de nuestro desarrollo.

A nuestros hermanos y sobrinos que son el motivo de superación diaria, con el deseo de llegar a ser un ejemplo para ellos y poder brindarles un mejor futuro.

Érica y Jhon

TABLA DE CONTENIDOS

	Páginas
PORTADA.....	i
COPYRIGHT.....	ii
FIRMA DE RESPONSABILIDAD.....	iii
RESPONSABILIDAD DE AUTOR.....	iv
DEDICATORIA	v
AGRADECIMIENTOS	vi
ÍNDICE DE TABLAS	xi
ÍNDICE DE FIGURAS.....	xii
ÍNDICE DE ANEXOS.....	xvi
RESUMEN	xvii
SUMMARY.....	xviii
INTRODUCCIÓN	1
CAPÍTULO I	
1. MARCO TEÓRICO	6
1.1 Introducción.....	6
1.2 Dispositivos Móviles	6
1.2.1 Smartphone.....	8
1.3 Plataforma Android.....	8
1.3.1 Historia	9
1.3.2 Características.....	11
1.3.3 Arquitectura	12
1.3.3.1 Kernel Linux.....	13
1.3.3.2 Biblioteca.....	13
1.3.3.3 Entorno ejecutable	14
1.3.3.4 Marco de aplicaciones.....	15
1.3.3.5 Aplicación.....	16

1.4	Lenguajes de programación para el diseño de aplicaciones Android	16
1.4.1	Introducción.....	16
1.4.2	Basic4Android	17
1.4.3	App Inventor.....	17
1.4.4	LiveCode	18
1.4.5	Eclipse	18
1.4.5.1	Características.....	19
1.5	Sistema de Control.....	19
1.5.1	Introducción.....	19
1.5.2	Sistema Domótico.....	20
1.5.3	Componentes Domóticos	21
1.5.4	Tipos de Arquitectura	23
1.5.4.1	Centralizada	23
1.5.4.2	Descentralizada:.....	24
1.5.4.3	Distribuida	24
1.5.5	Estándares de Comunicación	25
1.5.5.1	X10	25
1.5.5.2	LonWorks	27
1.5.5.3	KNX/EIB	27
1.5.5.4	ZigBee	28
1.5.6	Comunicaciones Inalámbricas	28
1.5.6.1	Bluetooth	29
1.5.6.2	WiFi.....	31
1.5.6.3	ZigBee	32
1.5.7	Arduino.....	36
1.5.7.1	Características generales	38
1.5.7.2	Características del hardware	38
1.5.7.3	Características del software	40
1.5.8	Componentes hardware para Arduino.....	41
1.5.8.1	Shield.....	41
1.5.8.2	Tarjeta Arduino Bluetooth	44
1.5.8.3	Sensores.....	45

CAPÍTULO II

2.	Diseño e implementación del sistema	50
2.1	Desarrollo e implementación de la App.	50
2.1.1	Instalación del JDK.....	50
2.1.2	Android SDK.....	51
2.1.3	Instalación de Eclipse	51
2.1.4	Estructura de App.	52
2.1.4.1	Control de Acceso de Usuario.....	52
2.2	Diseño e implementación de hardware	60
2.2.1	Topología de Red.....	60
2.2.2	Sistema de conexión local.....	61
2.2.2.1	Tarjetas bluetooth.....	62
2.2.3	Sistema de conexión externa.....	63
2.2.3.1	Shield ethernet	64
2.2.4	Requerimientos y elección del controlador arduino	65
2.2.5	Módulos de Comunicación Inalámbrica.....	67
2.2.5.1	Módulos Xbee.....	68
2.2.6	Diseño de etapas de la placa central.....	72
2.2.6.1	Etapas de alimentación	72
2.2.7	Actuadores y controladores.....	74
2.2.7.1	Sensor de movimiento o Pir	74
2.2.7.2	Sensor magnético.....	75
2.2.7.3	Servo-Motor.....	76
2.2.7.4	Visualización o lcd.....	77
2.2.7.5	Etapas de iluminación.....	78
2.2.7.6	Etapas de audio.....	80
2.3	Configuración de equipos para acceso remoto	82
2.3.1	Creación de cuenta No-Ip	84

CAPÍTULO III

3.	Resultados.....	94
----	-----------------	----

3.1	Funcionamiento de App.....	94
3.2	Funcionamiento de la central	104
3.3	COSTOS.....	105
3.4	Comparación con otras empresas.....	106
	CONSIDERACIONES	107
	CONCLUSIONES	108
	RECOMENDACIONES	109
	BIBLIOGRAFÍA	
	ANEXOS	

ÍNDICE DE TABLAS

Tabla 1-1	Instrucciones binarias más usadas en X10.....	26
Tabla 2-1	Comparativa de Tecnologías de Comunicación Inalámbrica.....	29
Tabla 3-1	Características de estándares WiFi.....	32
Tabla 4-1	Modelos de Arduino.....	37
Tabla 1-2	Permisos de control.....	53
Tabla 2-2	Selección de Arduino.....	66
Tabla 1-3	Costos del sistema.....	105
Tabla 2-3	Comparación de servicios.....	106

ÍNDICE DE FIGURAS

Figura 1-1.	Modelos de Smartphone	7
Figura 2-1.	Evolución del Teléfono Móvil	8
Figura 3-1.	Compatibilidad de las tecnologías con Android.....	9
Figura 4-1.	Arquitectura Android.....	12
Figura 5-1.	Logo de lenguaje Basic4Android.....	17
Figura 6-1.	Logo de lenguaje App Inventor	17
Figura 7-1.	Logo de lenguaje LIVECODE.....	18
Figura 8-1.	Logo de lenguaje Eclipse	18
Figura 9-1.	Sistema domótico	20
Figura 10-1.	Componentes Domóticos	22
Figura 11-1.	Arquitectura Centralizada	23
Figura 12-1.	Arquitectura Descentralizada.....	24
Figura 13-1.	Arquitectura Distribuida	24
Figura 14-1.	Arquitectura Mixta	25
Figura 15-1.	Transmisión de pulsos en X10.....	26
Figura 16-1.	Estándares de tecnologías inalámbricas	28
Figura 17-1.	Conexiones de red Bluetooth	30
Figura 18-1.	Estructura de Trama Bluetooth	31
Figura 19-1.	Topología de Conexión WiFi.....	31
Figura 20-1.	Estructura de trama ZigBee	33
Figura 21-1.	Topología de conexión ZigBee	34
Figura 22-1.	Topología punto a punto ZigBee	34
Figura 23-1.	Topología de árbol ZigBee	35
Figura 24-1.	Estructura hardware de Arduino	39
Figura 25-1.	Software de Programación Arduino.....	40
Figura 26-1.	Diagrama de Flujo del Shield Ethernet	42
Figura 27-1.	Estructura física de Shield WiFi	43
Figura 28-1.	Shield Motor Inalámbrico	44
Figura 29-1.	Tarjetas Bluetooth HC05	45
Figura 30-1.	Shield PIR.....	46

Figura 31-1.	Sensor de Presencia de Gas	47
Figura 32-1.	Diseño del activador de luz.....	47
Figura 33-1.	Graduador de temperatura	48
Figura 34-1.	Sensor de fuego	48
Figura 35-1.	Sensor de obstáculos.....	49
Figura 1-2.	Descarga de JDK	50
Figura 2-2.	Instalación del SDK.....	51
Figura 3-2.	Descarga de Eclipse Luna.....	52
Figura 4-2.	Programación Usuario y Claves.....	54
Figura 5-2.	Prioridades de usuario.....	55
Figura 6-2.	Conversión de mensaje a bytes	55
Figura 7-2.	Conversión de bytes a mensaje	56
Figura 8-2.	Asignación de MAC del Bluetooth.....	56
Figura 9-2.	Niveles de Usuarios	57
Figura 10-2.	Mensaje de conexión.	57
Figura 11-2.	Ingreso del dominio para acceso externo	58
Figura 12-2.	Acceso a la cámara 1	58
Figura 13-2.	Acceso a la cámara 2	59
Figura 14-2.	Control de la pantalla de proyección.....	60
Figura 15-2.	Infraestructura de IPREX.....	61
Figura 16-2.	Conexión Local	61
Figura 17-2.	Simulación Bluetooth	62
Figura 18-2.	Comunicación Arduino-Bluetooth.....	63
Figura 19-2.	Conexión Externa	64
Figura 20-2.	Shield Ethernet	64
Figura 21-2.	Reconocimiento Shield Ethernet.....	65
Figura 22-2.	Simulación Arduino en Proteus	67
Figura 23-2.	Simulación Xbee en Proteus	68
Figura 24-2.	Código de activación Xbee	69
Figura 25-2.	Simulación Xbee receptor.....	70
Figura 26-2.	Código para Arduino Pro mini.....	71
Figura 27-2.	Código de encendido y apagado	71
Figura 28-2.	Simulación Etapa de Alimentación.....	72

Figura 29-2.	Simulación de adaptación de batería.....	73
Figura 30-2.	LM2596.....	73
Figura 31-2.	Sensor detector de movimiento.....	74
Figura 32-2.	Código Sensor Pir.....	75
Figura 33-2.	Activación de sensor magnético.....	76
Figura 34-2.	Visualización de estado Servo-Motor.....	76
Figura 35-2.	LCD.....	77
Figura 36-2.	Código de visualización de lcd.....	77
Figura 37-2.	Simulación Luz Led.....	78
Figura 38-2.	Encendido de Leds.....	78
Figura 39-2.	Topología física de conmutador manual.....	79
Figura 40-2.	Simulación de luminarias.....	79
Figura 41-2.	Código de activación en Arduino.....	80
Figura 42-2.	Simulación control de audio.....	81
Figura 43-2.	Código de control de audio en Arduino.....	81
Figura 44-2.	Ingreso a router CNT.....	83
Figura 45-2.	Página principal Router Tp-Link.....	83
Figura 46-2.	Verificación del DDNS gratuito.....	84
Figura 47-2.	Página de creación de cuenta.....	85
Figura 48-2.	Registro de información.....	85
Figura 49-4.	Solución de confirmación.....	86
Figura 50-2.	Información de cuenta.....	86
Figura 51-2.	Añadir Host.....	87
Figura 52-2.	Actualizaciones de cuenta.....	87
Figura 53-2.	Activación de No-IP en el router.....	88
Figura 54-2.	Configuración Cámara 2.....	89
Figura 55-2.	Reservación de dirección IP.....	89
Figura 56-2.	Verificación de dirección asignada.....	90
Figura 57-2.	Reservación para cámara y placa central.....	90
Figura 58-2.	Verificación de dirección asignada.....	91
Figura 59-2.	Configuración de DHCP router Tp-Link.....	91
Figura 60-2.	Creación de caminos virtuales en router CNT.....	92
Figura 61-2.	Creación de caminos virtuales router Tp-Link.....	92

Figura 62-2.	Activación de acceso remoto	93
Figura 1-3.	Ingreso a la Aplicación	94
Figura 2-3.	Ingreso de usuario y contraseña	95
Figura 3-3.	Mensajes de emergencia	96
Figura 4-3.	Proceso de conexión local.....	97
Figura 5-3.	Conexión local establecida	98
Figura 6-3.	Pestañas de la Aplicación	99
Figura 7-3.	Control de acceso y pantalla de proyección	100
Figura 8-3.	Acceso a las cámaras	101
Figura 9-3.	Cambio de conexión	102
Figura 10-3.	Conexión Externa	103
Figura 11-3.	Página web de conexión externa	104
Figura 12-3.	Placa Central.....	104
Figura 13-3.	Placa del receptor Xbee	105

ÍNDICE DE ANEXOS

ANEXO A Imagen 3D de la placa central

ANEXO B Pistas de la placa central

ANEXO C Diagramas de simulación de etapas de la placa central

ANEXO D Diseño de Centro Educativo IPREX

RESUMEN

Se diseñó e implementó una aplicación Android para el control inalámbrico de una central basado en Arduino para el Instituto de Preparación de Exámenes de la ciudad de Riobamba. En este trabajo se conocieron los diferentes lenguajes de programación como Eclipse para la creación de aplicaciones Android, además se conoció los diversos estándares de comunicaciones inalámbricas como Bluetooth y ZigBee usados en la domótica. La aplicación se conecta con la central Arduino mediante el protocolo inalámbrico Bluetooth para el control local, es decir me permite activar o desactivar luminarias, puerta, pantalla de proyección, audio, cámaras, sirena y alarmas, para el control externo la aplicación se conecta con la central Arduino mediante WiFi/Datos permitiéndome el control exclusivo de luminarias y sirena. La central Arduino está conectada al internet y asignada una dirección IP fija mediante conexión Ethernet, además para acceder remotamente se procedió a configurar los router servidores de internet creando caminos virtuales al manipular puertos, esto permite conectarse de manera remota desde cualquier parte del mundo. Para el diseño se usó programas de pruebas de laboratorio gracias a simuladores como Proteus, Arduino y Eclipse. Los resultados obtenidos indicaron un 100% de fiabilidad gracias a la dualidad de control, comparando con otras empresas como IT CONTROL que ofrece solo el control local. Se concluye que se obtiene un menor costo en el manejo domótico con la unión de tecnologías inalámbricas Bluetooth, ZigBee y WiFi para un mejor control sin que el administrador esté presente. Se recomienda al gerente categorizar a los usuarios del sistema en secretaria, docentes y Jefe de cocina, ya que la aplicación maneja dispositivos diferentes para cada usuario.

Palabras claves: <APLICACIÓN ANDROID>, < CONTROL INALÁMBRICO>, <CENTRAL ARDUINO>, <PROTOCOLO INALÁMBRICO (BLUETOOTH)>, < PROTOCOLO INALÁMBRICO (ZIGBEE)>, <DISPOSITIVO ETHERNET>, <ACCESO REMOTO>, <CONFIGURACIÓN DE ROUTER>, <SIMULADOR ECLIPSE>, <LENGUAJE ARDUINO>.

SUMMARY

It was designed and implemented an Android application for wireless control of a central based on Arduino for Test Preparation Institute in Riobamba city. In this paper the different programming languages such as Eclipse for creating Android applications were known. Also, various wireless communication standards such as Bluetooth and ZigBee used in home automation became known. The application connects to the Arduino PBX via the Bluetooth wireless protocol for local control, it means, this allows me to turn on or turn off lights, door, projection screen, audio, cameras, siren and alarms. To the external control the application connects to the PBX via Arduino WiFi/data allowing me the exclusive control of lights and siren. The Arduino center is connected to the internet and assigned a fixed IP address via Ethernet connection. In addition to access remotely proceeded to configure router for internet servers, creating virtual paths when handling ports. It allows to connect remotely from anywhere in the world. For the design, laboratory testing programs were used. This was achieved thanks to simulators such as Proteus, Arduino and Eclipse. The results showed 100% reliability thanks to the dual control, comparing with other companies as IT CONTROL that only offers local control. It is concluded that a lower cost home automation management with the union of Bluetooth, ZigBee and WiFi wireless technologies for better control without the administrator is present. To the manager is recommended categorizing the system users at the secretariat, teachers and Head Chef, because the application handles devices for each user.

KEYWORDS: <ANDROID APPLICATION>, <WIRELESS CONTROL>, <ARDUINO CENTRAL>, <WIRELESS PROTOCOL [BLUETOOTH]>, <WIRELESS PROTOCOL [ZIGBEE]>, <DEVICE ETHERNET>, <REMOTE ACCESS>, <ROUTER CONFIGURATION>, <SIMULATOR ECLIPSE>, <ARDUINO LANGUAGE>.

INTRODUCCIÓN

ANTECEDENTES

Con la aparición de los teléfonos móviles inteligentes, en julio del 2005, Google decide expandir su negocio adquiriendo la empresa Android Inc. Con el fin de crear un sistema operativo móvil y gratuito. Android es un sistema operativo basado en el kernel de Linux y con una interfaz de programación orientada a objetos. Este nos permite realizar aplicaciones en una variedad de lenguajes de programación como son Java, Eclipse, App Inventor, LiveCode.

En un principio los Smartphone realizaban funciones de teléfonos normales, lo que es llamadas convencionales y una que otra aplicación interactiva, con el pasar de los años, desarrolladores han ido creando nuevas aplicaciones para diversas actividades como el intercambio de información, mensajería instantánea, video llamadas, navegación de internet.

Actualmente las aplicaciones se han ido orientando hacia la integración con otros servicios como el control automático, servicio de búsqueda, reconocimiento dactilar, reconocimiento de voz, compras electrónicas, exploradores de archivos, optimizadores, control domótico, medidor de intensidad ruido, las mismas que pueden llegar a ser parte de las actividades diarias de un ser humano común.

Con la utilización de las aplicaciones Android dentro del mundo tecnológico en combinación con nuestra vida, podemos facilitar el manejo de varias actividades, como al crear una aplicación que controle inalámbricamente la seguridad de puertas en un instituto educativo incrementaríamos el nivel de protección estudiantil, manteniendo así un control domótico de las instalaciones, a nivel de audio se mejoraría la comunicación de horarios estudiantiles o modificaciones imprevistas.

JUSTIFICACIÓN TEÓRICA

La presente investigación se enfoca en la creación de una aplicación Android mediante la evaluación de los diversos protocolos de comunicaciones inalámbricas para la funcionalidad con

una central Arduino, el mismo que posee el control domótico de un instituto educativo, además cuenta con hardware y software necesario para cumplir su función.

Existen estudios preventivos como los realizados en nuestra prestigiosa institución en el 2010 por Cárdenas Caldas Ángel Santiago y Pacheco Pérez Fabián Darío sobre la implementación de un sistema domótico con tecnología ZigBee controlado por computadora bajo la plataforma Android y Windows, donde se analiza las diferentes características que presenta, el manejo de una casa más digna, sin embargo el presente proyecto propone otros aprovechamientos como la utilización de otro protocolo inalámbrico para mejorar la seguridad de ingreso al sistema, ya que está dirigida a un lugar estudiantil.

Hogares e instituciones apuestan a la domótica como apoyo tecnológico para su seguridad y manejo, pero se necesita una herramienta para facilitar su uso, una solución es una aplicación, que permita al usuario un manejo más centralizado con un entorno más amigable.

La principal herramienta para el proyecto es un Arduino Uno instalado sobre el instituto educativo que se encontrará como central de control. El cual será el que se comunique inalámbricamente con nuestra aplicación Android, esta para obtener conexión deberá emparejarse mediante una contraseña única, elevando así el nivel de seguridad convirtiéndose en una tecnología ideal que permitirá principalmente la facilidad de control estudiantil.

JUSTIFICACIÓN APLICATIVA

El diseño e implementación de la aplicación requiere del análisis de los diferentes protocolos y herramientas inalámbricas que existen actualmente, esto permitirá establecer todo lo necesario para ser configurado e instalado en el Arduino, posteriormente diseñar la estructura del sistema y la posición del Arduino como central dentro del instituto educativo, contando con todos los permisos administrativos de quienes supervisan las instalaciones.

Es importante recalcar que en la tecnología a utilizar se debe considerar la distancia de línea de vista necesaria con la central realizando así pruebas de localización estratégica, para su correcto funcionamiento.

Para que se cumpla a cabalidad las funciones requeridas, las configuraciones del Arduino deben ser específicas con cada uno de los componentes a controlar y los mismos se deben encontrar configurados de manera maestro esclavo, así teniendo cada componente un pequeño control, nuestra aplicación se comunicara con la central Arduino mediante comunicación inalámbrica Bluetooth de manera única por medio de una clave de emparejamiento.

Los componentes a controlar serán dos cámaras, bloqueo/desbloqueo de puerta, 2 parlantes, pantalla de proyector y 3 luminarias.

OBJETIVOS

OBJETIVOS GENERALES

- ✓ Diseñar e implementar una aplicación Android para el control inalámbrico de seguridad, iluminación y audio mediante una central basada en Arduino para IPREX

OBJETIVOS ESPECÍFICOS

- ✓ Estudiar las características, programación y funcionamiento de una aplicación Android junto con los diferentes protocolos de comunicación inalámbrica para determinar la arquitectura funcional del sistema.
- ✓ Diseñar la arquitectura funcional para la aplicación Android a comunicarse con la central basada en Arduino.
- ✓ Implementar el prototipo con las características específicas del Arduino e instalación de software necesario para el sistema.

- ✓ Evaluar los resultados que verifiquen las distintas funcionalidades que fueron implementadas en la aplicación.

MÉTODOS Y TÉCNICAS

La presente investigación de evaluar un Arduino ideal para control domótico en el IPREX, es un trabajo desarrollador, la misma que requiere además un estudio a través de la aplicación de métodos y técnicas de investigación científica.

Para estudiar los diferentes protocolos y/o herramientas para el funcionamiento de la comunicación inalámbrica, es imprescindible abordar la investigación descriptiva ya que se describirá el estado de enlace con el sistema, además del lugar donde se desarrollará la implementación, esto nos permitirá configurar sobre una realidad.

El presente proyecto describe un trabajo aplicativo, ya que nos permitirá realizar la primera aproximación en el diseño del sistema, de tal forma que en el momento que se desarrolle la implementación se pueda cumplir a cabalidad con los objetivos.

La configuración del hardware y software según las especificaciones técnicas y operativas para obtener un enlace inalámbrico exitoso, persiguen un estudio no experimental, se detallará e implementará la aplicación Android según las características deseadas a través de la observación y el testing de las pruebas de laboratorio.

La comprobación de la comunicación inalámbrica de nuestra aplicación con la central, determinará la factibilidad funcional del proyecto, es así que la investigación analítica de acuerdo a tablas de resultados obtenidas de la comprobación de comunicación inalámbrica nos determinará la calidad de enlace obtenido con la tecnología usada.

La investigación posee una connotación de desarrollo transversal debido a que la comunicación inalámbrica del Smartphone con la central Arduino, requiere de 4 etapas de desarrollo

comprendidas en ciclos cortos de tiempo incluye análisis, diseño, implementación y validación del análisis funcional.

CAPÍTULO I

1. MARCO TEÓRICO

1.1 Introducción

El siguiente marco teórico nos ayudara a fundamentar nuestro proyecto, para las personas que no poseen ningún conocimiento sobre el mismo, teniendo así una idea clara para comprender el propósito, las necesidades y el impacto que nuestra aplicación provocará.

1.2 Dispositivos Móviles

Para determinar si un dispositivo es considerado móvil o no, existe un elevado nivel de dificultad ya que actualmente se los considera únicamente a ciertos modelos de teléfonos móviles, pero realmente existen varios tipos de dispositivos móviles los cuales no solo están bajo el ámbito telefónico.

Los dispositivos móviles forman parte de un grupo muy extenso y heterogéneo, donde sus características comunes incluyen componentes de hardware y software que amplía y diversifica su función inicial. El grupo más frecuente sin duda son aquellos que poseen la conexión telefónica y la conexión a Internet.



Figura 1-1. Modelos de Smartphone

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Dentro de los componentes más habituales son: la cámara fotográfica y vídeo, pantalla táctil, teclado QWERTY, receptor de radio, Bluetooth, dispositivos de memoria extraíbles, localizador GPS, conexión WiFi, etc. Desde el punto de vista del software, pueden incorporar también un amplio abanico de aplicaciones tales como programas ofimáticos, reproductores de audio y vídeo, organizadores, videojuegos, navegadores web o clientes de correo, entre otros.

A los dispositivos móviles los podemos clasificar de la siguiente manera:

- Dispositivo de comunicación
- Dispositivo de computación
- Reproductor multimedia

En los dispositivos de comunicación tenemos por ejemplo lo que hoy es un Smartphone, que no viene siendo más que la evolución de un teléfono móvil convencional.



Figura 2-1. Evolución del Teléfono Móvil

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.2.1 Smartphone

Un Smartphone (cuya traducción en español sería "teléfono inteligente") cuenta con varias características que lo asemejan más a un ordenador personal que a un teléfono tradicional.

Entre dichas características, se puede encontrar una mejora en la capacidad de proceso y almacenamiento de datos, conexión a Internet mediante WiFi, pantalla táctil y diversas aplicaciones de usuario como navegador web, cliente de correo, aplicaciones ofimáticas, reproductores de vídeo y audio, etc. incluyendo la posibilidad de descargar e instalar otras nuevas.

A pesar de estas importantes mejoras con respecto a sus predecesores móviles, el reducido tamaño de los Smartphone nos conlleva a una enorme limitación de hardware que los mantienen claramente diferenciados de los ordenadores convencionales.

Estas limitaciones obligan a tener presente la capacidad real del dispositivo a la hora de desarrollar su software, ya sean aplicaciones de usuario o el propio sistema operativo.

1.3 Plataforma Android

Android es un sistema operativo creado para teléfonos móviles, no siendo este el único, ya que existen otros sistemas operativos como iOS, Windows y Symbian de los cuales Android se diferencia de los demás porque se basa en una distribución de Linux, además la plataforma de

Android se caracteriza por tener esa facilidad al momento de desarrollar una aplicación ya que su código es abierto, esta plataforma es muy utilizada por los desarrolladores de aplicaciones por su principal característica que es el poder instalar en cualquier dispositivo ya sea de cualquier tipo de marca como puede ser HTC, SONY ERICSSON, MOTOROLA, SAMSUNG donde el desarrollador no tendría que crear una aplicación para cada marca sino que solo bastaría que el dispositivo tenga instalado la plataforma Android y la aplicación se podría instalar en cualquier dispositivo de esta marca, lo que no sucede con iOS, esta plataforma hace que cada aplicación sea exclusivamente solo para los dispositivos de marca Apple.

Android nos permite programar aplicaciones en muchas variaciones de Java llamada Dalvik, ya que este nos proporciona todas las interfaces necesarias para desarrollar variedad de aplicaciones las que pueden acceder a las funciones del teléfono (como GPS, WiFi, llamadas, agenda, etc.) de una forma muy sencilla en un lenguaje de programación muy conocido como es Java. Android dispone de una interesante combinación de tecnologías como GPS, WiFi, Bluetooth entre otras, permitiéndonos realizar un amplio campo de programación para nuevas aplicaciones.



Figura 3-1. Compatibilidad de las tecnologías con Android

Fuente: <http://yosoyandroid.com>

1.3.1 Historia

Con el pase del tiempo las personas han sido testigos de los avances que ha existido en las tecnologías ya que cada uno de estas causaron un impacto donde algunas se convirtieron en una adicción como fue la Televisión, Video juegos, el Internet y ahora que son los Teléfonos Celulares, los teléfonos celulares son dispositivos electrónico los cuales permiten realizar llamadas desde

cualquier punto del mundo de forma inalámbrica utilizando señales Inalámbricas para poder obtener comunicación entre dos personas.

Estas señales han ido evolucionando las cuales hoy en día no solo nos permiten realizar una simple llamada sino que también tenemos la posibilidad de poder realizar videos llamadas, poder navegar desde nuestro dispositivo electrónico, poder realizar transacciones bancarias por medio de paquetes de datos; Esto ha hecho que nuestra vida sea más fácil y también poder ahorrar lo más preciado que tenemos que es el tiempo.

Pero para esto avances no solo se debe tener una buena señal que nos permita mejorar nuestra calidad de vida, también debemos dar crédito a los avances que ha existido en el área de los dispositivos electrónicos donde existe una gran gama de dispositivos que podamos escoger y las cuales puedan cumplir algunas de nuestras exigencias como son resolución de cámara, memoria interna, tamaño del dispositivo, etc.

A todo esto cada dispositivo debe tener instalado un SO amigable para que pueda realizar su función de mejor manera y es ahí donde los SO no se han quedado atrás ya que ellos también han ido evolucionando para poder satisfacer los requerimientos de los dispositivos electrónicos.

Existe varias plataformas que han ido desarrollando sus propios SO, dirigidos a sus dispositivos como lo son Windows, Linux, Apple, Firefox, cada uno tiene sus versiones de SO, que se pueden encontrar en algunos de los dispositivos celulares como son Symbian, Windows Phone, BlackBerry, IOS, Android, Firefox Os, Ubuntu Touch, cada uno de estos SO tiene como objetivo cumplir una expectativa diferente.

El SO más utilizado en el mercado de los dispositivos Celulares es Android con su primer modelo que fue Android 1.0 Apple Pie hasta el último que es Android 6.0 Marshmallow.

Android es un SO perteneciente a la compañía de Android Inc., la misma que se inició por Octubre del 2003 por los señores Andy Rubin, Rich Miner, Nick Sears y Chris White que en sus inicios se creó con el objetivo de crear software para teléfonos móviles, pero la empresa no tenía grandes logros hasta que Google empezó a interesarse en sus trabajos y tener esa visión en el futuro de las Telecomunicaciones Inalámbricas, pero antes de empezar a realizar negocio con Android Inc.,

Google rompió algún tipo de relación laboral con empresas que se dedicaban a lo que eran desarrollo de redes sociales y también con empresas que se encargaban del posicionamiento móvil.

El 5 de noviembre del 2007 en una convención donde se daba a conocer los avances tecnológicos en las Telecomunicaciones se mostró con lo que hoy conocemos como Android junto con otras empresas fabricantes de chip, software y empresas que se encargaban a las Telecomunicaciones.

Android fue creado con el fin de poder brindar un servicio de calidad con la ventaja de ser gratuito y por ello Android está basado en Linux, pero para los creadores esta idea no se pudo cumplir en su totalidad ya que Google tiene el dominio de casi toda la empresa.

La primera empresa en tener el So Android fue HTC en un dispositivo de gama baja denominado HTC Dream con Android 1.0 que estaba a punto de revolucionar en el ámbito de las telecomunicaciones ya que se trataba del primer Smartphone con Android, después de esto Google lanzo un emulador dirigidos a los programadores con el fin de entender y desarrollar aplicaciones, juegos para Android. (Herraiz, G. 2012. pp.3-7)

1.3.2 Características

- ✚ Android es de código abierto para que los programadores puedan aportar con el desarrollo del SO y también puedan realizar aplicaciones.
- ✚ Google permite que toda aplicación que sea subida a la plataforma Android pueda ser descargada de forma gratuita.
- ✚ Utiliza un motor de navegación open Source Webkit que también es de código abierto y esta emparejado con java script.
- ✚ Utiliza framework para que así pueda simplificar la reutilización de componentes y estos puedan ser reemplazados.

- ✚ Android tiene un conjunto de librerías las cuales están programadas bajo el lenguaje de programación java.
- ✚ Cada aplicación Android tiene la capacidad de poder correr su propio proceso gracias a la máquina virtual Dalvik que tiene integrada ya que esto permite optimizar la memoria de forma que pueda correr varias máquinas virtuales de forma eficiente.
- ✚ Utiliza SQLite para almacenamiento de datos y este ya viene incluido con SDK permitiéndole así poder comunicarse con otras bases de datos.

1.3.3 Arquitectura

La arquitectura de Android, está formada por 5 capas que permiten la generación de aplicaciones. El acceso a cada capa se realiza por intermedio de librerías, en las cuales cada capa puede utilizar los elementos de la capa inferior para realizar sus funciones. Las capas de la arquitectura Android son:

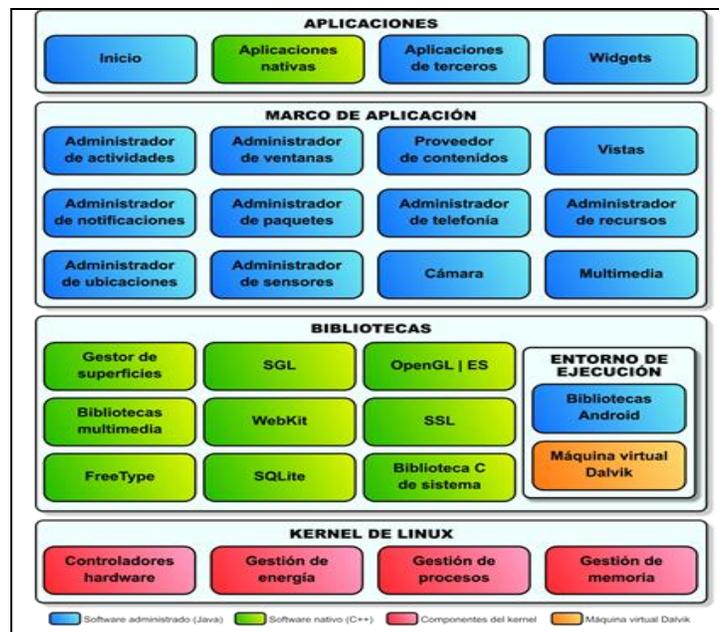


Figura 4-1. Arquitectura Android

Fuente: TUDELA, J. A. 2009.

1.3.3.1 *Kernel Linux*

Esta capa actúa como una capa de abstracción entre el hardware y el software teniendo así un kernel Linux versión 2.6 similar al que maneja Ubuntu pero con características de Android para un Smartphone, este kernel puede tener el control de etapas como son gestión de energía, gestión de procesos y gestión de memoria ya que esto nos da la opción de poder utilizar cualquier hardware del dispositivo celular sin necesidad de conocer el modelo o características del dispositivo, solo es necesario tener instalado el drivers para así poder manejar el hardware del dispositivo. (Tudela, J. A. 2009.pp 39-40.)

1.3.3.2 *Biblioteca*

En esta etapa es la encargada de la comunicación entre el hardware y la interfaz de aplicaciones, utilizando los servicios de las bibliotecas para poder realizar sus funciones, la mayoría de estas bibliotecas son escritas bajo los comandos de c/c++ y por lo tanto cualquier aplicación que sea instalado en el dispositivo celular primero debe realizar una consulta a estas librerías para poder realizar su función.

Las bibliotecas que se ejecutan son:

- ✚ Gestor de Superficie.- Este gestor se encarga de mostrar en nuestra pantalla gráficas de 3D y 2D como puede ser animaciones, transiciones, etc.
- ✚ SGL (Scalable Graphics library).- Este es el motor gráfico 2D que tiene Android y son utilizados para mostrar elementos en 2 Dimensiones, este tipo de librería es utilizados en Chrome.
- ✚ Open GL|Es.- Es un motor gráfico 3D la cual dispone de algunas versiones y va instalada dependiendo a la versión del Android y también utiliza un acelerador de hardware 3D.

- ✚ Biblioteca Multimedia.- su función es poder reproducir o grabar Imágenes, Audio y Video en formatos como JPG, GIF, PNG, MPEG4, AVC (H.264), MP3, AAC o AMR.
- ✚ Webkit.- La función que realiza esta librería es como un gestor de Navegación utilizando aplicaciones independientes o aplicaciones propias del dispositivo este mismo gestor es utilizado por Chrome y Safari.
- ✚ SSL (Secure Sockets Layers).- Está encargado de la seguridad de navegación por medio de Criptografía.
- ✚ Freetype.- Muestra fuentes de mapas de bit y vectorial.
- ✚ SQLite.- Este motor está disponible para todas las aplicaciones la cual es un gestor de Base de Datos.
- ✚ Bibliotecas C de Sistemas (LibC).- Su función principal es la ejecución de las aplicaciones.
(Tixicuro, M. 2015. pp.15-16)

1.3.3.3 Entorno ejecutable

Su función se basa en realizar una máquina virtual utilizando java lo cual esta convertido en un formato (.dex). Esta parte de la arquitectura Android no es considerado como una capa principal para su funcionamiento, solo se lo considera como un set de bibliotecas las cuales cumplen con la mayor parte de las funciones y es por esto que cuando una aplicación se ejecuta ésta corre su propio proceso sin necesidad de sobre cargar el Sistema Operativo o volver lento el dispositivo Android ya que se puede correr múltiples Aulas virtuales.

- ✚ Máquina virtual Dalvik.- Estas librerías es fundamental en este proceso para ejecutar aplicaciones ya que esta permite utilizar para memoria ejecutando varias máquinas virtuales simultáneamente ahorrando así la batería y la memoria del dispositivo.

- ✚ A partir de Android 5.0 Dalvik fue reemplazado por la máquina virtual ART ya que esta reducirá el tiempo de ejecución de las aplicaciones en un 33%.
- ✚ Núcleo de librerías.- Esta librería lo que haces es ofrecer funcionalidades elementales y de bajo nivel.

1.3.3.4 Marco de aplicaciones

Esta capa simplifica la reutilización de componentes y es la capa más importante por los desarrolladores de aplicaciones y es porque cuando una aplicación es desarrollada por Google o por terceras personas e incluso por el propio usuario utiliza el mismo conjunto de API y el mismo Frameworks.

- ✚ Activity Manager.- Administra el ciclo de vida de la aplicación
- ✚ Windows Manager.- Administra la ventana de las aplicaciones utilizando Surface Manager.
- ✚ Telephone Manager.- Permite enviar y recibir tanto llamadas como mensajes
- ✚ Content Provider.- Permite encapsular los datos los cuales sirven para compartir con otras aplicaciones y esto es gracias a API tales aplicaciones como son (Contactos, agendas, mensajes)
- ✚ View System.- Permite una gran cantidad de elementos para construir una interfaz de usuario como son: Botones, Texto, Listas.
- ✚ Notification Manager.- Esta librería como su nombre mismo lo indica realiza notificaciones al usuario cuando este lo requiera.

- ✚ Package Manager.- Permite dar a conocer sobre los paquetes que tiene instalado o los paquetes que faltan por instalar en el dispositivo.
- ✚ Location Manager.- Permite la localización del dispositivo por medio de GPS o por medio de redes de Posicionamiento.

1.3.3.5 Aplicación

Esta es la última capa que realiza la arquitectura de Android en la cual su función es ejecutar las aplicaciones que se instalan en el dispositivo celular, en ella se encuentra los siguientes elementos:

- ✚ Inicio
- ✚ Contactos
- ✚ Navegador
- ✚ Teléfono
- ✚ Otras aplicaciones

1.4 Lenguajes de programación para el diseño de aplicaciones Android

1.4.1 Introducción

En la actualidad existen muchas herramientas que facilitan a que las personas se sumerjan en el mundo del desarrollo de aplicaciones móviles, sin la necesidad de ser un programador especializado.

El lenguaje más utilizado es Java, ya que da soporte para aplicaciones de forma nativa. Pero además de este existen otros lenguajes como App Inventor y el más reciente Basic4Android.

1.4.2 *Basic4Android*



Figura 5-1. Logo de lenguaje Basic4Android

Fuente: <http://developeando.net>

Es una plataforma de programación para aplicaciones Android, donde su base de programación es Visual Basic, el mismo que está dirigido hacia las personas que recién inician en el mundo de la programación. Su pequeño detalle es que su valor va desde \$49 hasta \$249, la diferencia es el tiempo de duración para las actualizaciones, la más económica tiene una duración de 2 meses.

1.4.3 *App Inventor*



Figura 6-1. Logo de lenguaje App Inventor

Fuente: <http://androideity.com>

Esta plataforma posee un lenguaje de programación de desarrollo gráfico, donde no debes conocer ninguna línea de código, simplemente consiste en arrastrar el bloque con la acción que deseamos realizar, y que esta usa el navegador como el núcleo de la programación, esta es gratuita.

1.4.4 LiveCode



Figura 7-1. Logo de lenguaje LIVECODE

Fuente: <http://androideity.com>

Este es muy diferente a las demás plataformas, ya que usa un lenguaje de programación llamado “Programación orientada a eventos”, esto se refiere a arrastrar elementos dentro de un área de trabajo enlazándose entre sí, por lo que nos permite programar para cualquier tipo de sistemas operativos como Android, IOS, Windows, etc.

1.4.5 Eclipse



Figura 8-1. Logo de lenguaje Eclipse

Fuente: <http://www.limecreativelabs.com>

Esta plataforma está diseñada para ser extendida mediante plug-ins, ya que no utiliza un lenguaje específico, sino se basa en un IDE genérico, en el mundo de Java el más conocido es el plug-in JDT, es totalmente libre es decir gratuito. (Jiménez, J. 2014)

1.4.5.1 Características

- Dentro de Eclipse el principal concepto de trabajo se basa en la pre-configuración de ventanas y editores, relacionadas entre sí, lo que nos permiten trabajar de manera exitosa.
- El desarrollo en Eclipse está enfocado en los proyectos, que no son más que el conjunto de recursos relacionados entre sí, dentro de ellos tenemos el código fuente, documentación, entre otros.
- La existencia del IDE es para proporcionarnos asistentes y ayudas para la creación de proyectos.
- En Eclipse existe un gran depurador, de fácil manejo el mismo que tiene como objetivo mejorar el código de programación, también existe la perspectiva de depuración la misma que nos da toda la información para dicha acción.
- La colección de plug-ins, estos se encuentran disponibles en una gran cantidad, sean propios de Eclipse o no, por ello la colección es bien extensa ya que en ella están de todo tipo de pago, gratuitos, licenciados o de pago.

1.5 Sistema de Control

1.5.1 Introducción

En la actualidad todo se controla inalámbricamente desde televisores hasta autos, un gran ejemplo es la vivienda domótica, que no es más que un hogar con un sistema de control inalámbrico, el

mismo que nos ofrece mejor calidad de vida al reducir el trabajo doméstico y aumentar el nivel de seguridad.

Pero porque no pensar más allá de solo la comodidad del hogar, porque no dirigirlo hacia el ámbito estudiantil, el uso de la tecnología nos ha permitido viajar hacia el ámbito educativo para brindar mejoras.

Usaremos el término de Domótica para referirnos a la automatización y control en (encendido/apagado, abrir/cerrar) aparatos y sistemas de instalaciones como la iluminación y audio de forma centralizada y remota.

1.5.2 Sistema Domótico

Existe varias opiniones sobre que debe abarcar un sistema de control dentro del mundo de los desarrolladores de entornos domóticos, pero este sistema abordará la automatización, el control y la conectividad con la red externa de nuestro instituto, pasando a ser este una evolución de la domótica.

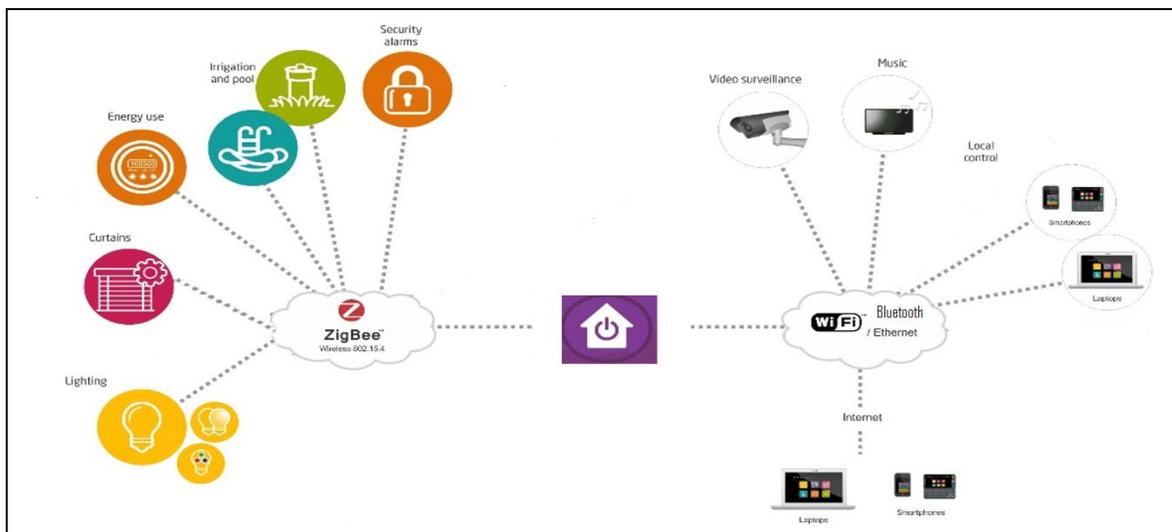


Figura 9-1. Sistema domótico

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para ello se considerará que nuestro sistema de control domótico será el conjunto de sistemas de software y hardware, donde el hardware debe incluir todos los elementos necesarios para recibir información y realizar acciones específicas en nuestro entorno.

Señalaremos los servicios que ofrece la domótica:

1. **Ahorro Energético.**- Reducir el nivel de energía sin la necesidad de sustituir los aparatos existentes por otros de menor consumo, es decir realizar una gestión eficiente de los existentes como la desconexión de equipos sin uso.
2. **Confort.**- Acciones que pueden reducir el trabajo dentro del instituto
3. **Seguridad.**- La red de seguridad debe proteger tantos los bienes patrimoniales como la seguridad personal dentro de esto se encuentran todos los tipos de alarmas, acceso de cámaras IP.
4. **Comunicaciones.**- Los sistemas de comunicación para el control pueden ser externo y/o interno, o controlados remotamente mediante el internet.
5. **Accesibilidad.**- Se hace referencia a la manera que se va a realizar el control remoto como aplicaciones o instalaciones que se desarrollen.

1.5.3 Componentes Domóticos

La amplitud de los componentes puede depender de la función que el sistema desea desarrollar, pero en general tenemos los siguientes:

- **Sensor.**- Dispositivo que está encargado en monitorizar el entorno y transmitir la información al sistema.
- **Actuador.**- Es un dispositivo capaz de recibir órdenes y ejecutarlas.

- **Controlador.-** Es el dispositivo encargado de la gestión del sistema según su programación o información que reciba.
 - **Interfaz.-** Son todos aquellos dispositivos donde se muestra la información del sistema destinado para los usuarios.
 - **Bus.-** Es el medio de transporte que comunica la información entre los dispositivos existentes.
- (Chipuxi, O, Yadira, R. 2015. p.6)

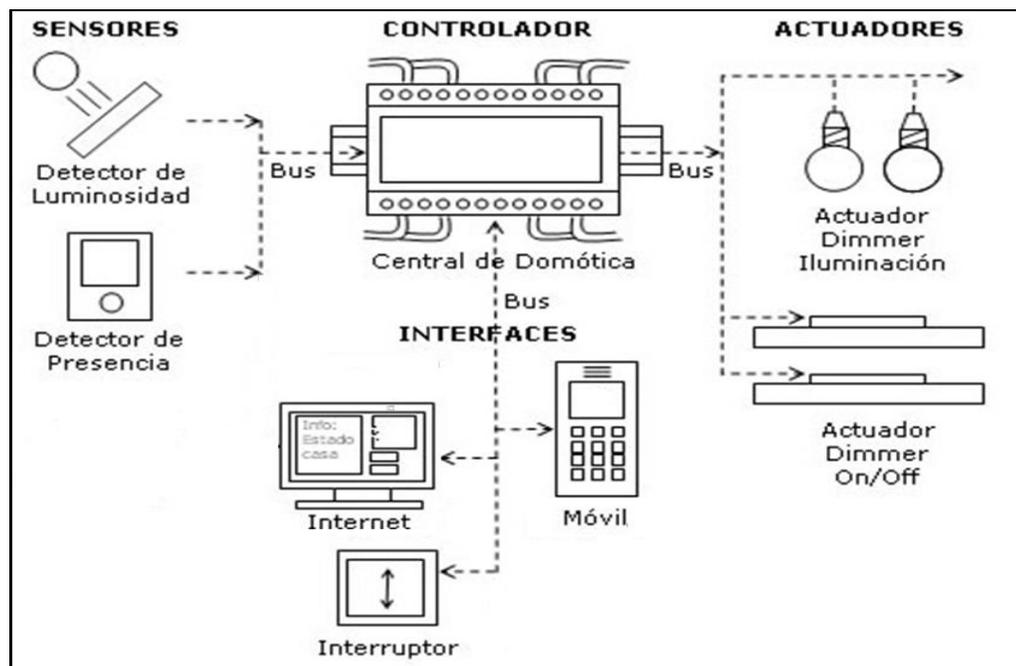


Figura 10-1. Componentes Domóticos

Realizado por: Loayza J, Padilla E. 2016

Para desarrollarse cualquiera de las arquitecturas antes mencionadas se debe conocer previamente el tipo de bus con el que va a funcionar, según la infraestructura tenemos:

- **Cableado.-** En este tipo de red los sensores y actuadores se encuentran cableados con la central, la misma que es el controlador principal de todo el sistema.

- **Inalámbrico.-** En este tipo se usan solo sensores inalámbricos que transmiten información por RF hacia la central.
- **Mixto.-** Este tipo es la combinación del cableado con el inalámbrico, dando como resultado una mayor seguridad, control y automatización de los dispositivos.

1.5.4 Tipos de Arquitectura

La arquitectura de un sistema de control domótico debe estar relacionado a la de una red donde su principal característica sea el control inteligente. Las principales son:

1.5.4.1 *Centralizada:* Es aquella donde existe un controlador centralizado, es decir envía información a todos los dispositivos que se encuentren interconectados a él, actuadores e interfaces de acuerdo un programa.

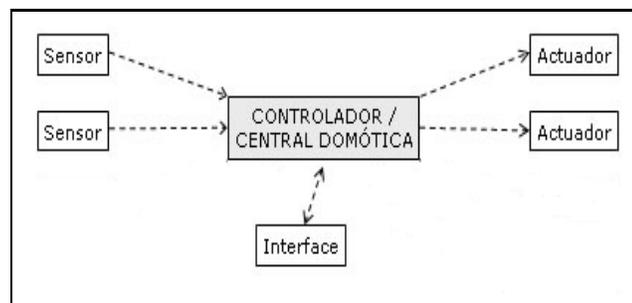


Figura 11-1. Arquitectura Centralizada

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.4.2 *Descentralizada*: Es aquella donde existen varios controladores los mismo que están interconectados para enviar la información.

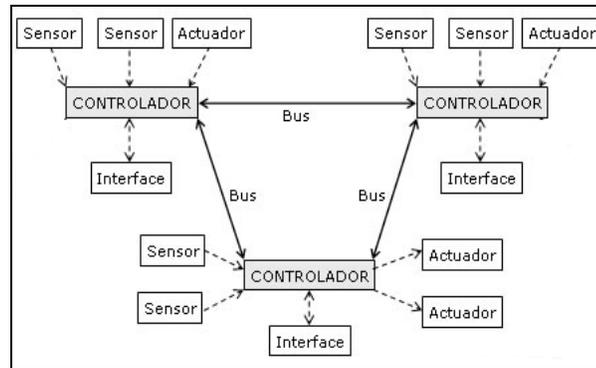


Figura 12-1. Arquitectura Descentralizada

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.4.3 *Distribuida*: Es aquella que si red es en forma de bus, es decir que cada elemento es también un controlador capaz de enviar información a todo el sistema.

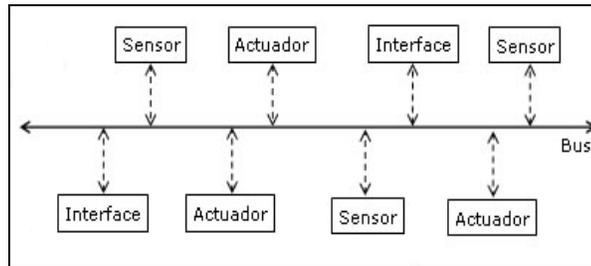


Figura 13-1. Arquitectura Distribuida

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.4.4 *Mixta*: Es aquella donde se combinan las arquitecturas de los sistemas antes mencionados también son llamadas arquitecturas híbrida. (Paullán, C; Villa, A. 2010. p.29)

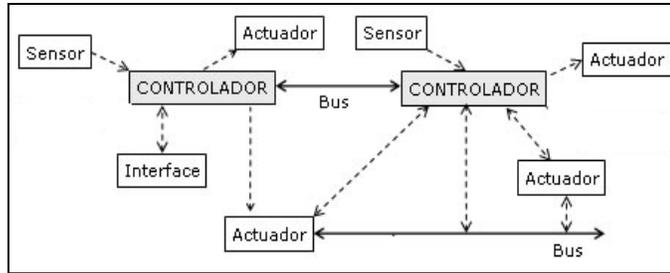


Figura 14-1. Arquitectura Mixta

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.5 Estándares de Comunicación

Una vez determinado los componentes y el medio físico de comunicación, se debe establecer el protocolo de comunicación del sistema domótico, siendo este un conjunto de normas para que productos de distintas o iguales marcas y fabricantes sean compatibles entre sí. Así tenemos:

Protocolos Proprietarios o cerrados, son protocolos específicos desarrollados por el fabricante, quien es el único que puede realizar modificaciones, además solo se usa para cierta marca.

Protocolos Estándar o Abiertos, son aquellos protocolos establecidos por varias compañías, como su nombre lo indica son abiertos, esto es, que cualquier fabricante puede desarrollar productos o mejoras que lleven este tipo de protocolos. A continuación tenemos los estándares más usados actualmente en la domótica.

1.5.5.1 X10

Este protocolo está dirigido para el control remoto de dispositivos eléctricos, es llamada la tecnología “de corrientes portadoras” debido a que utiliza la corriente eléctrica para realizar la comunicación entre los elementos domóticos en formato digital. Aquí instrucciones más utilizadas.

Tabla 1-1. Instrucciones binarias más usadas en X10

Código	Función	Descripción
0000	Apagar todo	Apaga todos los dispositivos con el código indicado en el mensaje
0001	Encender luces	Enciende todas las luces (con la posibilidad de controlar el brillo)
0110	Apagar luces	Apaga todas las luces
0010	Encender	Enciende un aparato
0011	Apagar	Apaga un aparato
0100	Opacar	Atenúa la intensidad de la luz
0101	Iluminar	Incrementa la intensidad de la luz
0111	Extender código	Código de extensión
1000	Solicitud de saludo	Solicita una respuesta del dispositivo(s) con el código de casa indicado
1001	Reconocimiento	Respuesta al comando anterior
1101	Respuesta encendido	Respuesta a la Solicitud de Estado del dispositivo está encendido
1110	Respuesta apagado	Respuesta indicando que el dispositivo está apagado
1111	Solicitud de estado	Solicitud pidiendo el estado de un dispositivo

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

X-10 realiza su transmisión por medio de ráfagas de pulsos, es decir, codifica la información en binario y la transmite de manera sincronizada al paso por cero de la corriente alterna, como podemos observar en la siguiente imagen.

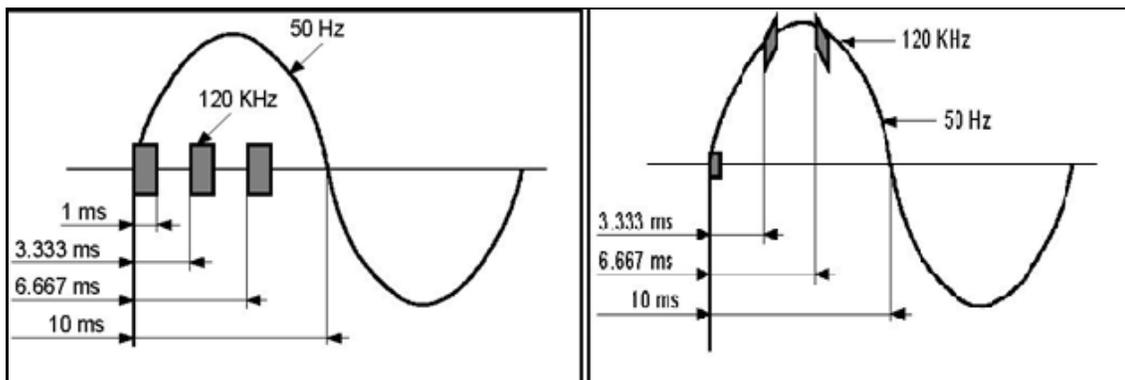


Figura 15-1. Transmisión de pulsos en X10

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Actualmente es una de las tecnologías más usadas por su bajo coste, además de su fácil instalación y configuración.

1.5.5.2 LonWorks

Esta tecnología fue creada por la compañía Echelon en 1992, la cual está basada en el protocolo LonTalk, donde posee todas las herramientas para la implementación de redes de control distribuidas y de automatización. Esta tecnología es robusta, fiable y recomendable a la automatización industrial, por lo que se encuentra homologado por:

Las normas Europeas (EN-14908), de Estados Unidos (EIA-709-1) y Chinas (G131Z20177-2006) así como por el estándar europeo de electrodomésticos (CEDEC AIS). También se ha impuesto dentro de la asociación de petroleros como estándar para el control y comunicación de la red de gasolineras (International Forecourt Standards Fortín, IFSF) y es ampliamente utilizado en el control de viviendas, edificios, industrial, de transporte ferroviario, naval, aeroespacial y la monitorización remota de contadores. (Tobar, J, Sandoval, L. 2015. p.6)

1.5.5.3 KNX/EIB

Esta tecnología nació de la unión de tres asociaciones europeas. Se basada en un Bus de datos, donde su característica principal es que utiliza su propio cableado.

La primera versión de esta tecnología trabajaba en tres modos diferentes como son:

- Modo Sistema;- Destinado para instalaciones de tipo comercial.
- Modo Fácil.- Para cualquier persona
- Modo Automático.- Conformado por la idea Plug&Play, es decir, no requería de instalación, bastaba con conectarse para funcionar.

1.5.5.4 ZigBee

Es un protocolo abierto, recogido en el IEEE 802.15.4, se caracteriza por su comunicación inalámbrica.

1.5.6 Comunicaciones Inalámbricas

Antes de iniciar en el campo de las comunicaciones inalámbricas debemos saber que es una red inalámbrica, la cual no es más que la interconexión de dispositivos con la capacidad de transmitir información entre ellos, pero sin la utilización de un medio físico. El mismo que en la actualidad se aplica con más frecuencia, debido a su fácil manejo sin la necesidad de un amplio conocimiento.

Antiguamente usaban módulos de radiofrecuencia, posteriormente aparecieron circuitos transmisores integrados que unificaron las funciones de emisor y receptor a través del uso de bandas de frecuencia.

Actualmente posee un extenso campo de aplicación, además las tecnologías inalámbricas más utilizadas son WiFi, Bluetooth y ZigBee, ya que trabajan en un rango de frecuencia no muy saturado.

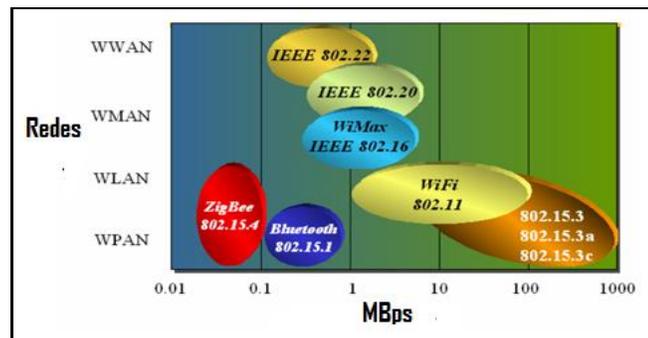


Figura 16-1. Estándares de tecnologías inalámbricas

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Estas tecnologías son totalmente diferentes, es decir fueron desarrollados para diferentes tipos de necesidades.

Tabla 2-1. Comparativa de Tecnologías de Comunicación Inalámbrica

Tecnología	Ancho de Banda	Alcance	Potencia	Aplicaciones
Bluetooth	1 - 24 Mbps	1m	1mW	Auriculares Inalámbricos, Red inalámbrica en espacios reducidos
		10 m	10mW	
		100 m	100mW	
WIFI	54-400 Mbps	Hasta 100 m.		Conexión inalámbrica a Internet
ZigBee	20 a250 Kbps	10- 100 m		Redes Inalámbricas de sensores

Fuente: Acosta, C, Soria, A. 2015.

1.5.6.1 Bluetooth

Esta tecnología se basa en la transmisión de información punto a punto entre dispositivos, la cual posibilita la transmisión de voz y datos entre los diversos dispositivos mediante un enlace por radiofrecuencia. La velocidad de transmisión varía entre 1 Mb/s y 24Mb/s. Gracias a saltos de frecuencia nos permite transmitir en Full Duplex con un máximo de 1600 saltos/seg ofreciéndonos la seguridad y robustez.

Como anteriormente mencionamos una comunicación inalámbrica va junto a una red inalámbrica, es por ello que Bluetooth nos permite la creación de pequeñas redes, donde existe un maestro de red y los demás dispositivos interconectados toman el nombre de esclavos, a esto se lo como piconet. Pero también existe la posibilidad de unir estos piconet mediante puertas de enlace y toman el nombre de Scatternet. (Acosta, C; Soria, A. 2015. pp.4-8)

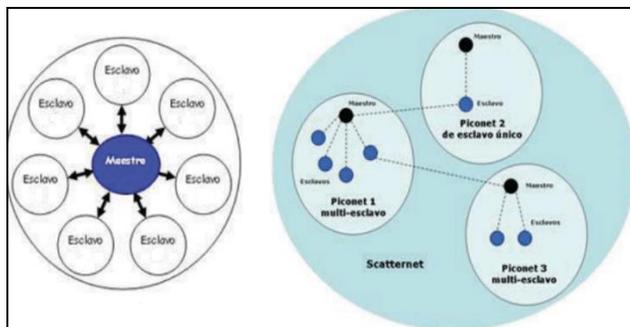


Figura 17-1. Conexiones de red Bluetooth

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Las conexiones Bluetooth se pueden referir en varios estados:

1. Standby.- Los dispositivos están en modo de reposo donde escuchan mensajes.
2. Page/inquirí.- Permite el envío de un paquete denominado page que permite realizar la conexión con otro dispositivo.
3. Active.- Permite la transmisión de datos.
4. Hold.- Realiza la conexión sin necesidad de transferir datos.
5. Sniff.- Solo los esclavos tienen este estado, escuchan a un nivel reducido.
6. Park.- Aquí el esclavo es sincronizado a la piconet.

La trama Bluetooth se envía de la siguiente manera:

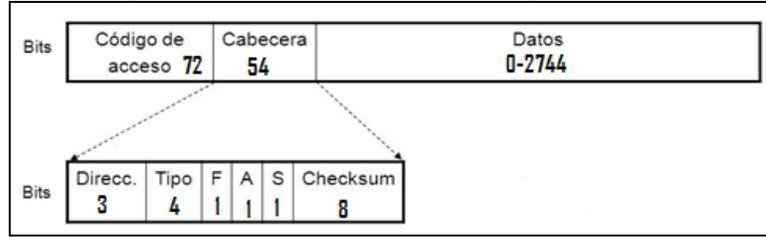


Figura 18-1. Estructura de Trama Bluetooth

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.6.2 WiFi

WiFi es una de las tecnologías más usadas hoy en día por su comunicación inalámbrica mediante ondas, también conocido por su estándar IEEE 802.11.

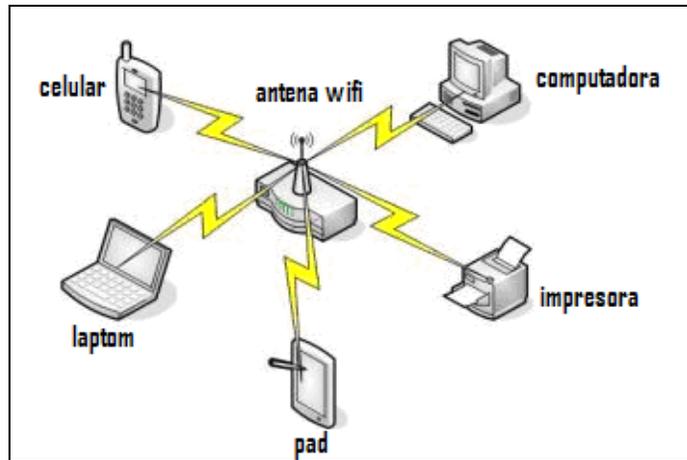


Figura 19-1. Topología de Conexión WiFi

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Inicialmente se creó para ser utilizada en redes locales inalámbricas sin embargo en la actualidad se utiliza para acceder al internet. Creada por la organización WiFi Alliance, la cual se encarga en la certificación de los equipos, los cuales deben encontrarse con todos los estándares necesarios.

En la siguiente tabla mostramos todos los estándares existentes:

Tabla 3-1. Características de estándares WiFi

Estándar	Velocidad (Teórica)	Velocidad (Práctica)	Frecuencia	Ancho de Banda	Alcance (m)	Detalles	Año
802.11	2 Mbit/s	1 Mbit/s	2,4 GHz	22 MHz	330		1997
802.11 ^a	54 Mbit/s	22 Mbit/s	5,4 GHz	20 MHz	390		1999
802.11b	11 Mbit/s	6 Mbit/s	2,4 GHz	22 MHz	460	Aceptación internacional	1999
802.11g	54 Mbit/s	22 Mbit/s	2,4 GHz	20 MHz	460	Aceptación internacional	2003
802.11n	600 Mbit/s	100 Mbit/s	2,4 GHz y 5,4 GHz	20/40 MHz	820	Mayormente utilizado, se puede configurar su ancho de banda.	2009
802.11ac	6.93 Gbit/s	100 Mbit/s	5,4 GHz	80 hasta 160 MHz		Nuevo estándar sin interferencia pero menor alcance.	2013
802.11ad	7.13 Gbit/s	Hasta 6 Gbit/s	60 GHz	2 MHz	300		2012
802.11ah			0,9 GHz		1000	Fácil distribución en áreas rurales, menor consumo de energía	2016

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

El principal defecto de la conectividad WiFi es su poca seguridad. Pero gracias a sus diversos protocolos que puede realizar una codificación más alta, garantizando su confidencialidad. Una conexión WiFi está formada por 2 puntos de acceso: emisores remotos o router, los cuales son los que reciben la señal y los dispositivos terminales o de recepción.

El mayor problema de esta tecnología es la saturación del espectro radioeléctrico debido a la gran cantidad de usuarios, lo que produce que la transmisión más lenta.

1.5.6.3 ZigBee

ZigBee es un estándar de comunicaciones inalámbricas creado por la ZigBee Alliance. El mismo que fue desarrollado con el fin de hallar soluciones que pueden ser implementadas por cualquier fabricante.

ZigBee está basado en el estándar IEEE 802.15.4 de redes inalámbricas de área personal y tiene como objetivo principal extender la vida útil de sus baterías, es decir usando un bajo nivel de envío de datos. Por ello podemos decir que es ideal para las redes domóticas, evitando el exceso de sensores y actuadores. Además que posee un bajo costo para red Wireless de información pequeña, seguro y confiable.

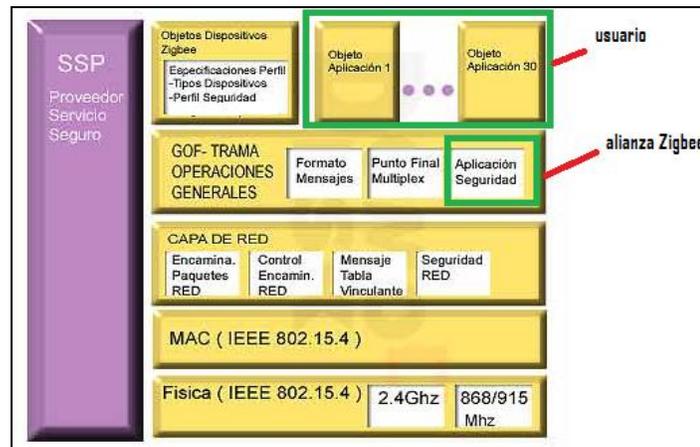


Figura 20-1. Estructura de trama ZigBee

Realizado por: Loayza J, Padilla E. 2016

Como podemos ver en la imagen usa las bandas libres ISM de 2,4 GHz (Mundial), 868 MHz (Europa) y 915 MHz (EEUU).

1.5.6.3.1 Topología

En la siguiente imagen tenemos la estructura de la topología tipo estrella determinada por las flechas blancas y la estructura tipo rejilla indicada por las flechas rojas. Donde uno de los dispositivos se convierte en coordinador de red y es responsable de inicializar y mantener los dispositivos en la red.

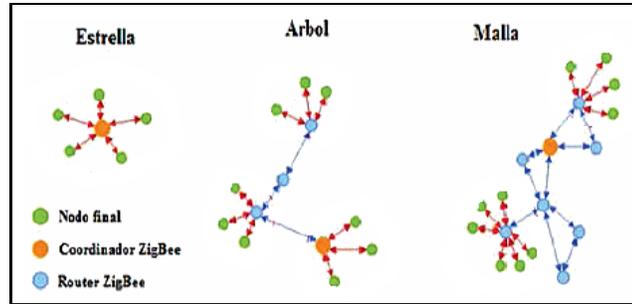


Figura 21-1. Topología de conexión ZigBee

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Punto a punto, en esta topología existe un solo FFD Coordinador. A diferencia con la topología estrella, cualquier dispositivo puede comunicarse con otro siempre y cuando estén en el mismo rango de alcance circundante.



Figura 22-1. Topología punto a punto ZigBee

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

La topología de árbol es un caso especial de topología de conexión punto a punto, en la cual muchos dispositivos son FFDs y los RFD pueden conectarse como un nodo único al final de la red. Cualquiera de los FFDs restantes puede actuar como coordinador y proveer servicios de sincronización hacia otros dispositivos o coordinadores.

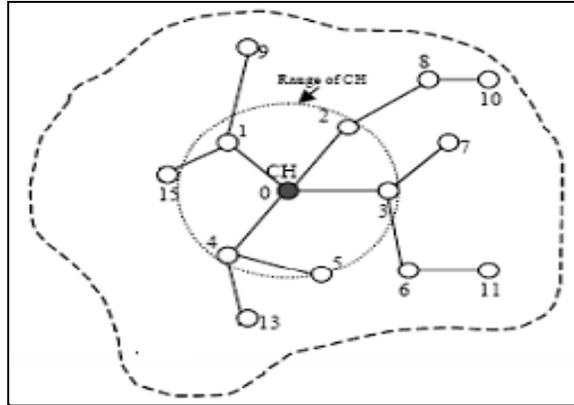


Figura 23-1. Topología de árbol ZigBee

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.6.3.2 Tipos de Dispositivos

Se definen tres tipos distintos de dispositivo ZigBee según su papel en la red:

1. **Coordinador ZigBee (ZigBee Coordinator, ZC):** Es el más importante, debe existir al menos uno por red, ya que su función principal es controlar toda la red.
2. **Router ZigBee (ZigBee Router, ZR):** Es aquel que interconecta dispositivos separados en la topología de red, además de ofrecer un nivel de aplicación para la ejecución de código de usuario.
3. **Dispositivo final (ZigBee End Device, ZED):** Es aquel que puede comunicarse con su coordinador pero no puede comunicarse con otros dispositivos, por ello se puede encontrar en un estado dormido, evitando así consumir el menor nivel de energía.

1.5.6.3.3 CARACTERÍSTICAS

- Una red ZigBee puede estar formada por hasta 255 nodos los cuales tienen la mayor parte del tiempo el ZED ZigBee dormido con objeto de consumir menos que otras tecnologías inalámbricas, por lo cual puede ser alimentado con 2 pilas AA.
- Escalabilidad de red
- Fragmentación
- Agilidad de frecuencia
- Recolección centralizada de datos.- Optimiza el flujo de información.

La mayor ventaja de esta tecnología es que reduce el tiempo de espera en la transmisión, es decir, en el envío y recepción de paquetes, por lo que lo convierte en la tecnología ideal para redes de baja tasa de transferencia de datos.

1.5.7 Arduino

Un arduino contiene un software y un hardware que está basado en una plataforma libre ya que puede ser fácil en su programación y en el uso, el cual contiene I/O donde podemos conectar en las salidas hembras, Sensores y Actuadores que pueden ser controlador de forma sencilla y fácil con el objetivo de poder realizar múltiples proyectos con fines Electrónicos, Robóticos, etc. Contiene un entorno interactivo y también cumple con la misma función de un Microcontrolador o un Pic, y es por eso que tiene sistema reprogramable las cuales se puede realizar múltiples pruebas de proyectos sin temor a poder realizar algún error, y además son dispositivos reutilizables.

Este Arduino se lo puede considerar como un Sistema Embebido ya que puede realizar múltiples funciones porque tiene integrado internamente una computadora con el objetivo de ser reprogramable para así poder realizar una función en específico.

Existe una infinidad de modelos de arduino los cuales vamos a explicar en brevedad dentro de la siguiente tabla, su memoria, tipo de procesador, velocidad de reloj y algunas descripciones y estas son propias de Arduino o conocidas como placas oficiales.

Tabla 4-1. Modelos de Arduino

MODELO DEL ARDUINO	MICRO-CONTROLADOR	FRECUENCIA DE RELOJ	MEMORIA FLASH	DESCRIPCIÓN
Pro-mini	ATmega328	8 MHz	32 kB	está diseñada para su instalación semi-permanente en objetos
UNO R3	ATmega328	16MHz	32Kb	Este tipo de Arduino puede manejar 14 pines de entrada y salida.
LEONARDO	ATmega32U4	16MHz	32Kb	Tiene entrada USB y permite emular como un ratón o un teclado.
DUE	AT91SAM3X8E	84MHz	512Kb	Este Arduino contiene 54 pines de E/S digital (de los cuales 12 pueden ser usados como salidas)
YÚN	ATmega32u4	16 MHz	32 KB	Este Arduino se puede conectar a internet ya sea por medio de cable o wifi.
ZERO	Atmel SAMD21 MCU	48Mhz	256 KB	Este arduino es la mejora del Arduino uno R3 ya que tiene potencia en el Procesamiento
MICRO	ATmega32u4	16 MHz	32 KB	Micro fue creado con el pensamiento en donde el tamaño del proyecto importa.
ESPLORA	ATmega32u4	16 MHz	32 KB	Es un derivado de Arduino Leonardo pero la diferencia es que lleva incorporado sensores listos para usar.
MEGA 2560 R3	ATmega2560	16MHz	256Kb	Consta de 54 pines lo cual lo hace a este dispositivo perfecto para proyectos grandes.
ROBOT	ATmega32u4	16 MHz	32 KB	Este es el primer Arduino Robot que maneja 2 procesadores al mismo tiempo.

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Como ya es conocimiento de todos que Existen placas que no son propias de Arduino pero que pueden realizar la misma función de un Arduino.

Su principal motivo de poder realizar copias idénticas de Arduino es por tener un mercado más competente, ya que es por esa razón que existen compañías privadas dónde sacan al mercado dispositivos muy similares a los que tiene Arduino y hasta utilizan el mismo lenguaje de programación de Arduino para poder configurar esos dispositivos.

1.5.7.1 Características generales

Todas las placas Arduino comparten las mismas características ya que en su programación puede ser flexible y fácil, pero tiene 2 puntos importantes a tratar el cual lo hace destacar a las placas Arduino como son:

- **Limitaciones.-** Arduino es una compañía que no tiene limitaciones ya que existe una gran variedad de placas las cuales van cubriendo las necesidad que valla exigiendo el cliente ya sean propias y de compañías ajenas pero que se puedan adaptar.
- **Funciones Específicas.-** las funciones especifica va a depender de la placa del Arduino que vallamos a utilizar ya que puede ser que necesitemos mayor rapidez en el procesamiento entonces utilizaremos Arduino con la exigencia que necesitemos

1.5.7.2 Características del hardware

Estas Características comparten las mismas Arquitectura del Arduino la cual en la parte central consta de un microprocesador y va a tener el mismo funcionamiento pero va a depender en el modelo de la placa o las exigencias del proyecto ya sea de placa propia o de compañías ajenas

Arduino es un hardware libre y eso quiere que no es necesario que solo las placas que son propias de Arduino pueden ser añadidas, sino que también pueden ser placas de fabricantes externos, y que sus librerías pueden ser descargadas gratis desde cualquier página de internet.

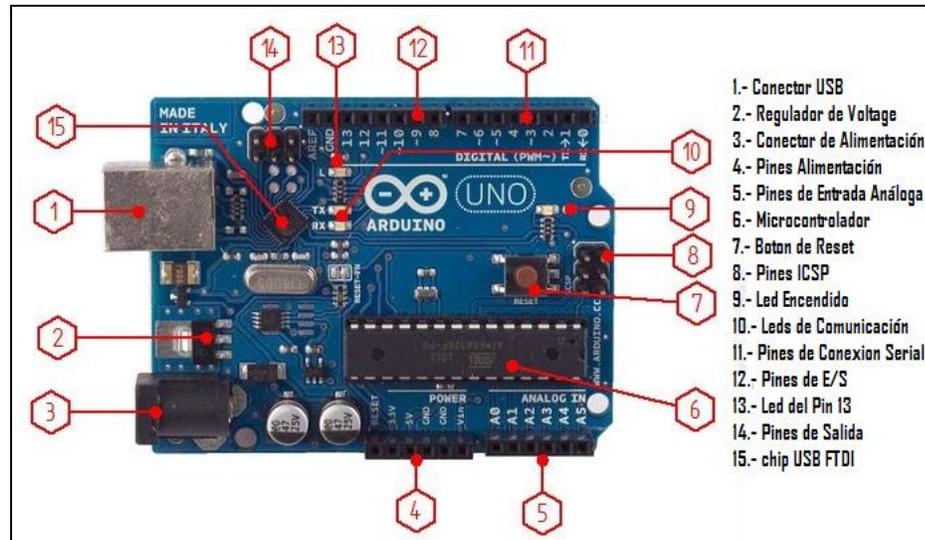


Figura 24-1. Estructura hardware de Arduino

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Se presentaran diferencias en los modelos de Arduino, las cuales dependen de las funcionalidades y elementos de cada placa como son:

- Pines de entrada
- Velocidad de reloj
- Memoria flash
- Procesadores

1.5.7.3 Características del software

Este software de Arduino es propio de la compañía y es un software libre ya que puede ser expandido por medio de librerías de C++ y que se lo puede descargar de cualquier navegador y puede instalarlo en cualquier Sistema Operativo que desee, lo interesante de este Software es su fácil su manejo o su programación y es por eso que es de fácil aprendizaje pero también a la vez puede realizar proyectos de avanzada complejidad.

El tipo de lenguaje de programación que utiliza es processing que es muy similar a C++. EL menú que dispone la empresa Arduino el cual contiene ciertos botones con funcionamientos comunes, barra de herramientas y una consola de texto.

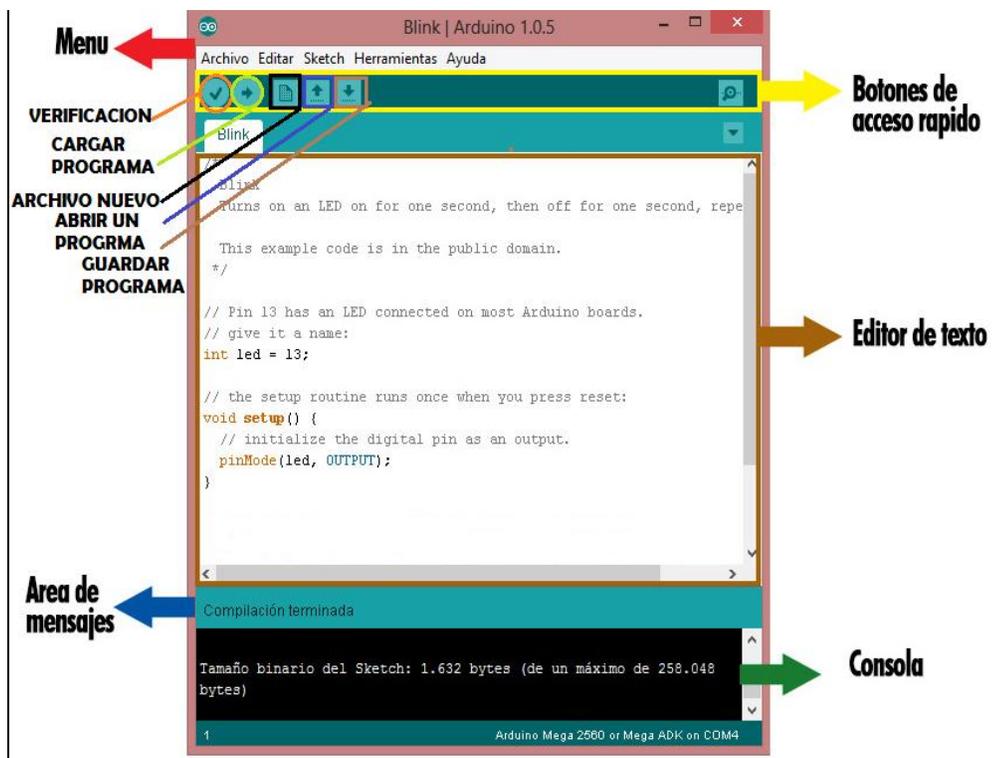


Figura 25-1. Software de Programación Arduino

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.8 Componentes hardware para Arduino

1.5.8.1 Shield

El shield es un actuador que se le puede añadir en la parte superior del Arduino el cual puede complementar la función esperada o deseada con la opción de poderse controlar desde el Arduino entre ella se encuentra una infinidad de adaptares que se podrían acoplar al Arduino la cuales daremos a conocer las más utilizadas.

1.5.8.1.1 Arduino Ethernet Shield

Con este tipo de Shield se podrá dotar una conexión LAN permitiéndole obtener internet a una placa Arduino utilizando el conector Rj-45 con el protocolo TCP/IP, su forma es muy semejante al Arduino Uno.

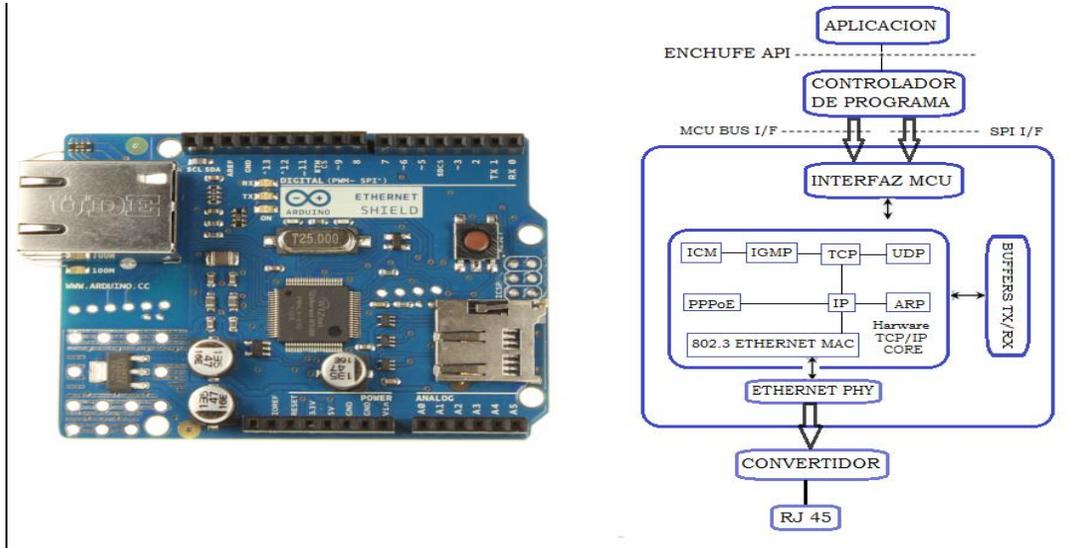


Figura 26-1. Diagrama de Flujo del Shield Ethernet

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Características:

- Microcontrolador Atmega 328
- Frecuencia 16 MHz
- 5 Voltios
- 2kb SRAM
- 1 kb de EEPROM
- 32 kb de flash
- 1 ranura de micro-sd
- 14 puertos digitales

1.5.8.1.2 Arduino Wi-Fi Shield

Este dispositivo realiza el mismo funcionamiento que el shield Ethernet pero su conexión lo hace de forma inalámbrica y así de esta forma se conectaría a una red de Acceso de internet por medio de una red WiFi utilizando el estándar 802.11 b/g.

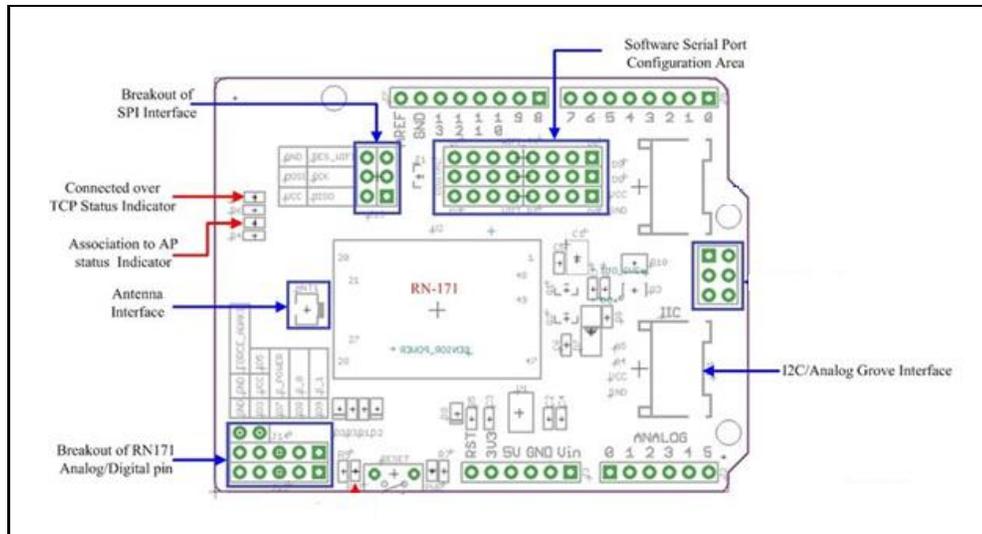


Figura 27-1. Estructura física de Shield WiFi

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Características:

- Voltaje de 3.3 a 5.4
- Potencia de transmision 0-10dbm
- Frecuencia de transmisión 2401-2480
- Canal de transmisión 0-13 canales

1.5.8.1.3 Arduino Wireless Motor Shield

Con este shield se pueden controlar diferentes componentes que contengan inductores o bobinas en su interior. Se pueden controlar por ejemplo relés, motores de corriente continua o solenoides. Con su chip L298P es posible controlar la velocidad y el sentido de giro de hasta dos motores de corriente continua. (Ruiz, P.2015.p.21)

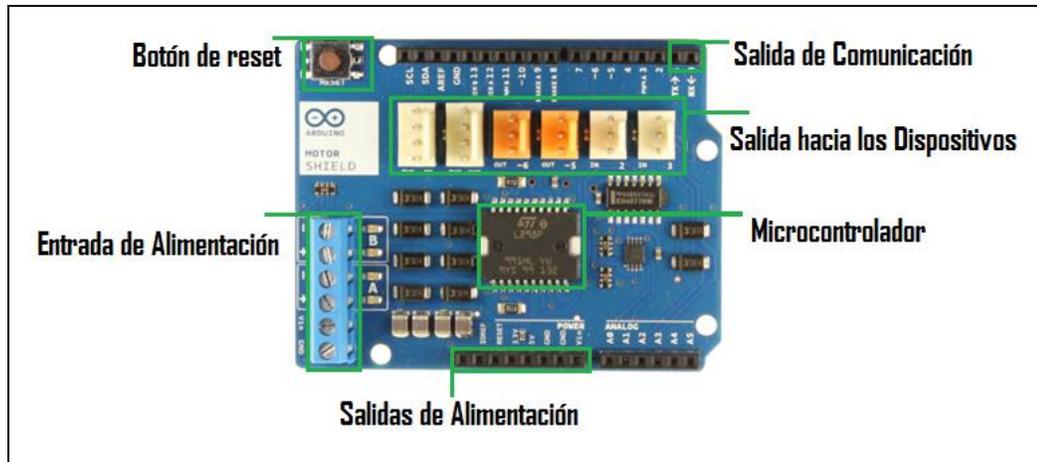


Figura 28-1. Shield Motor Inalámbrico

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.8.2 Tarjeta Arduino Bluetooth

Esta versión de Arduino incorpora un módulo de transmisión de datos de hasta 100 metros, es decir con esta placa se puede programar sin cable así como también realizar comunicaciones seriales con cualquier dispositivo Bluetooth. (Tobar, J, Sandoval, L. 2015.p.21)

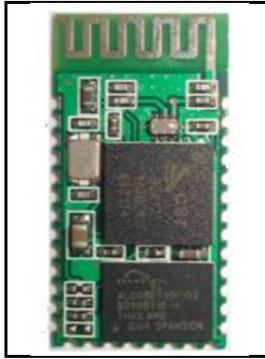


Figura 29-1. Tarjetas Bluetooth HC05

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Características:

- Microcontrolador Atmega 328
- Consta de 16 puertos
- Frecuencia de oscilación 16Mhz
- Botón de reinicio
- 6 entradas analógicas

1.5.8.3 Sensores

Todo dispositivo que tenga la capacidad de transformar en señales eléctricas a señales físicas o químicas se denomina sensores, teniendo en cuenta que estos dispositivos deben estar conectados a un cerebro electrónico como un microcontrolador. Existen dos tipos de sensores los cuales son:

- Analógico.- Son dispositivos con la capacidad de generar energía de 0-5 voltios

- Digitales.- Estos generan una señal discreta

A continuación detallaremos los diversos modelos de sensores más usados en domótica.

1.5.8.3.1 Módulo PIR

Este tipo de modulo es más conocido como un detector de movimiento basado en infrarrojo y es muy utilizado en casa, edificio, bancos y locales comerciales.

Este tipo de sensor se lo puede utilizar de varias formas como seguridad y prestación de servicios automáticos.

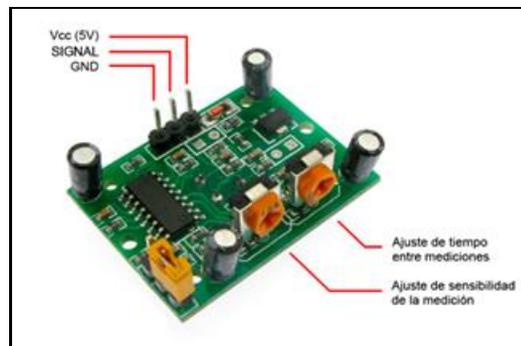


Figura 30-1. Shield PIR

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.8.3.2 Módulo de Gas

Este módulo es muy utilizado en los hogares donde el peligro es más constante, sobre todo para la familia, el dispositivo fue creado para poder detectar el gas cuando este se esté escapando del cilindro y poder darnos un aviso.

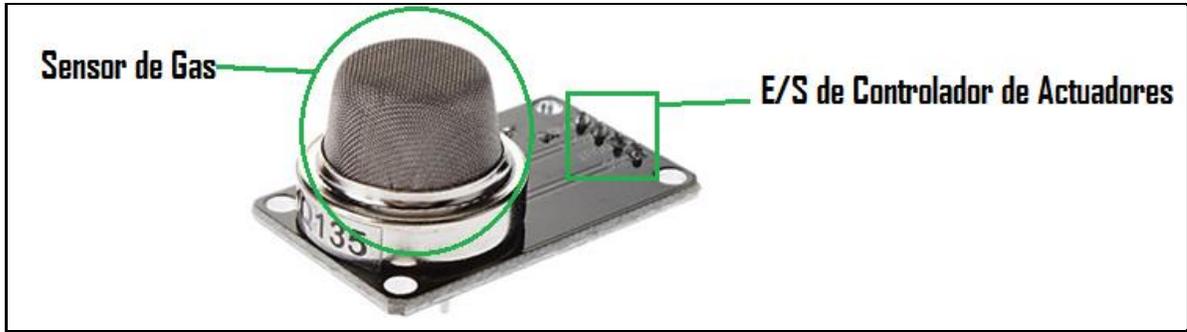


Figura 31-1. Sensor de Presencia de Gas

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.8.3.3 Módulo de Luz

Otro elemento que intervine en el hogar ya que este tipo de sensor detecta el movimiento que existe en cuarto el cual hace que se active y prenda las luces.

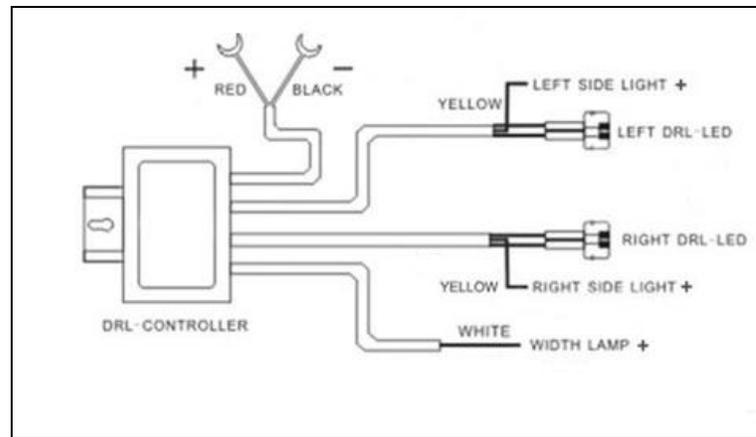


Figura 32-1. Diseño del activador de luz

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.8.3.4 Módulo de Temperatura

Estos tipos de módulos tienen la posibilidad de darnos comodidad en nuestros hogares y también ahorrarnos un poco la factura de electricidad. Su funcionamiento se basa en graduar este tipo de sensor y tener un clima templado y placentero.

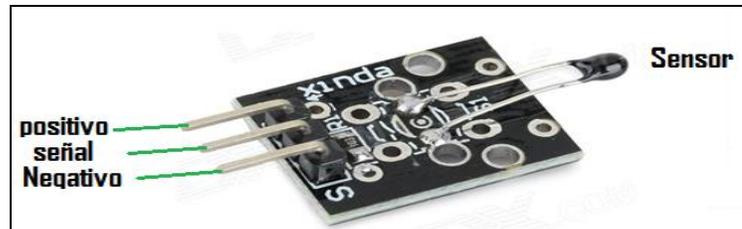


Figura 33-1. Graduador de temperatura

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

1.5.8.3.5 Módulos de llamas

Con este sensor podemos medir la radiación de infrarrojos que tiene así como fuentes de luz con un rango de longitud de onda comprendido entre los 760 y 1100 nanómetros. Por lo tanto, estos sensores son simples detectores de infrarrojo y luz. (Ruiz, P.2015.p.23)

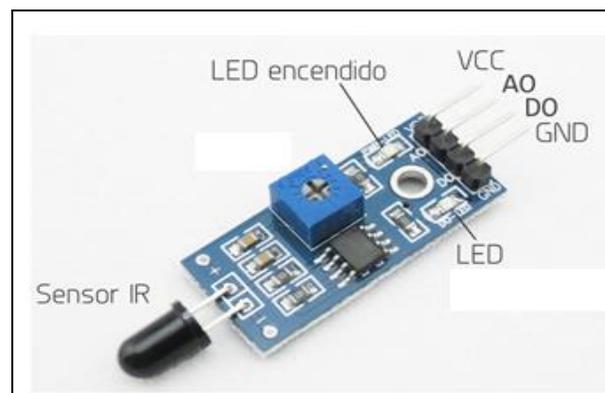


Figura 34-1. Sensor de fuego

Realizado por: Loayza J, Padilla E. 2016

1.5.8.3.6 Sensores ultra sonido HC-SR04

Este sensor es digital, el cual puede detectar distancias de 2cm a 500cm con un margen de error de 3mm, este proceso lo realiza en un rango de tiempo transcurrido cuando envía la señal ultrasónica y cuando recibe ese tipo de señal, durante este tiempo puede medir la distancia.

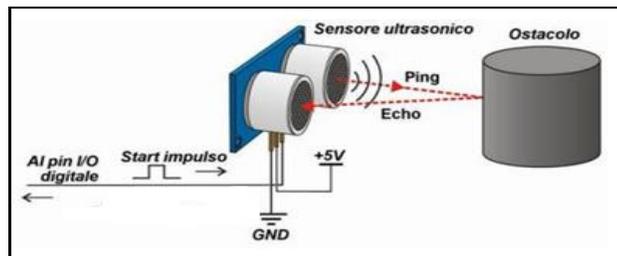


Figura 35-1. Sensor de obstáculos

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

CAPÍTULO II

2. DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

2.1 Desarrollo e implementación de la App.

2.1.1 Instalación del JDK

Para poder realizar un app dirigida Android se debe primero considerar la instalación de JDK lo cual es un software que provee herramienta que permite desarrollar programas basados en Java, el programa se lo puede encontrar en la página principal de Java.

The screenshot shows the 'Java SE Downloads' page. At the top, there are buttons for 'Java Platform (JDK) 8u91 / 8u92' and 'NetBeans with JDK 8'. Below this, the 'Java Platform, Standard Edition' section is visible, with a 'JDK DOWNLOAD' button highlighted by a red circle and a 'click' label. To the right, the 'Java SE Development Kit 8 Downloads' section is shown, featuring a table of download links for various operating systems and architectures. A red circle highlights the 'Accept License Agreement' checkbox, which is also pointed to by a 'click' label.

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.72 MB	jdk-8u91-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.65 MB	jdk-8u91-linux-arm64-vfp-hflt.tar.gz
Linux x86	154.74 MB	jdk-8u91-linux-i586.rpm
Linux x86	174.92 MB	jdk-8u91-linux-i586.tar.gz
Linux x64	152.74 MB	jdk-8u91-linux-x64.rpm
Linux x64	172.91 MB	jdk-8u91-linux-x64.tar.gz
Misc OS X	227.25 MB	jdk-8u91-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.55 MB	jdk-8u91-solaris-sparcv6.tar.Z
Solaris SPARC 64-bit	98.95 MB	jdk-8u91-solaris-sparcv9.tar.Z
Solaris x64 (SVR4 package)	149.26 MB	jdk-8u91-solaris-x64.tar.Z
Solaris x64	96.78 MB	jdk-8u91-solaris-x64.tar.gz
Windows x86	182.29 MB	jdk-8u91-windows-509.exe
Windows x64	191.4 MB	jdk-8u91-windows-x64.exe

Figura 1-2. Descarga de JDK

Fuente: LOAYZA, Jhon, PADILLA, Erica. 2016

2.1.2 Android SDK

Una vez que se instala Eclipse se debe instalar en el software librerías para que me permita desarrollar apps dirigidas a Android, estas librerías me permiten depurar, compilar aplicaciones dirigidas para el Sistema Operativo Android.

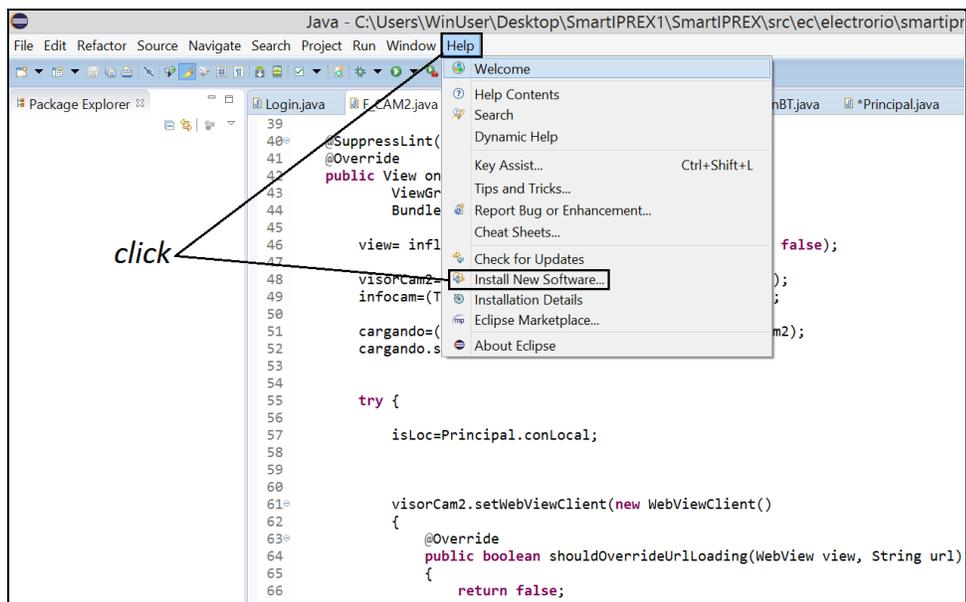


Figura 2-2. Instalación del SDK

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.1.3 Instalación de Eclipse

Para la instalación de eclipse primero de debe descargar el software el cual lo podemos encontrar en la página principal, donde encontramos una gran cantidad de software de eclipse pero el que nosotros requerimos es eclipse luna, tener en cuenta el Sistema Operativa en el que queramos instalar.

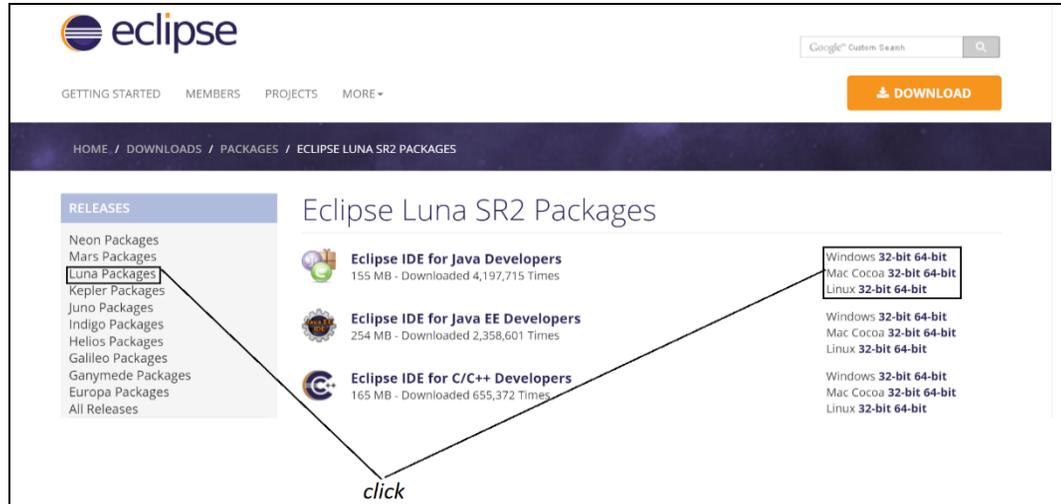


Figura 3-2. Descarga de Eclipse Luna

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.1.4 Estructura de App.

2.1.4.1 Control de Acceso de Usuario

En esta etapa de la aplicación se controlara mediante un Usuario y Contraseña para poder ingresar a la pantalla principal de la App. Para ello hemos decidido separar los usuarios de acuerdo a sus funciones.

Existen 4 tipos de usuarios con diferentes contraseñas cada una, estos Usuarios hacen referencia a la ocupación del personal que trabaja en Iprex Norte.

Los tipos de usuarios son los siguientes:

- Administración total de la aplicación
- Docentes del Iprex

- Secretaria del Iprex
- Personal de Vida & Comida

En el siguiente cuadro se mostrara los Dispositivos que se podrán controlar con cada Usuario y Contraseña.

Tabla 1-2. Permisos de control

Dispositivos Usuario	Iluminación de Letrero	sirena	Sensor	Puerta	Pantalla	Audio Comedor	Audio Aula	Luces Aula	Luces V&C	Micrófono	Cámara Aula	Cámara V&C
Docentes	X	X		X			X	X			X	X
Administradores	X	X	X	X	X	X	X	X	X	X	X	X
Cocina		X			X	X			X			X
Secretaria	X	X	X	X			X	X		X	X	X

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para poder realizar este proceso interno se debe realizar un código para que la aplicación pueda validar la información ingresada.

En la figura 39-2 se podrá visualizar el código de programación para usuarios con sus respectivas a las contraseñas.

```

1 package ec.electrorio.smartiprex;
2
3 import android.app.Activity;
17
18 public class Login extends Activity {
19
20     private static String Usuario, Contraseña;
21     EditText user,pass;
22     Button enviar,x_btn;
23     CheckBox GuardarDatos;
24
25
26     private static final String USER1="admin";
27     private static final String PASS1="administrador";
28
29     private static final String USER2="cliente";
30     private static final String PASS2="cliente";
31
32     private static final String USER3="normal";
33     private static final String PASS3="normal";
34
35     private static final String USER4="operario";
36     private static final String PASS4="operario";
37
38     //private static final String USER7="otro";
39     //private static final String PASS7="otro";
40
41     private static final String TAG="IPREX2";
42
43     int Level=0;
44     private static boolean isChecked;
45
46     public long[] pattern ={0,35};
47
48     Vibrator vib;
49

```

Figura 4-2. Programación Usuario y Claves

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

En la figura 40-2 observamos la validación de la información ingresada para poder verificar si es correcto o incorrecta y así poder administrar la aplicación desde la pantalla Principal y que dispositivos podemos controlar dependiendo del usuario que ingresemos ya que estos están con niveles de ingreso.

```

100 private void ingreso(){
101     Usuario=user.getText().toString();
102     Contraseña=pass.getText().toString();
103
104     if(!Usuario.isEmpty()&&!Contraseña.isEmpty()){
105         if (Usuario.equals(USER1)&&Contraseña.equals(PASS1)){Level=1;}
106         if (Usuario.equals(USER2)&&Contraseña.equals(PASS2)){Level=2;}
107         if (Usuario.equals(USER3)&&Contraseña.equals(PASS3)){Level=3;}
108         if (Usuario.equals(USER4)&&Contraseña.equals(PASS4)){Level=4;}
109         //if (Usuario.equals(USER7)&&Contraseña.equals(PASS7)){Level=7;}
110
111         if(Level>0){
112
113             Intent PrincipalAct = new Intent(Login.this,Principal.class);
114             PrincipalAct.putExtra("level", Level);
115             startActivity(PrincipalAct);
116             finish();
117         }else{
118             Toast toast = Toast.makeText(getApplicationContext(), "Usuario o Contraseña Incorrectos", Toast.LENGTH_SHORT);
119             toast.setGravity(Gravity.CENTER,0,0);
120             toast.show();
121         }
122     }else{
123         Toast toast = Toast.makeText(getApplicationContext(), "Usuario o Contraseña Vacio", Toast.LENGTH_SHORT);
124         toast.setGravity(Gravity.CENTER,0,0);
125         toast.show();
126     }

```

Figura 5-2. Prioridades de usuario

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para poder enviar información desde la app hacia la placa central debe viajar en bytes para que así pueda existir comunicación, en la figura 41-2 se muestra como se realizó la programación para la transformación de mensaje a bytes.

```

480 public void sendMessage(String message) {
481
482     Log.d("mensaje", message);
483     if (Servicio_BT.getState() == ConexionBT.STATE_CONNECTED) { //checa si estamos conectados a BT
484         if (message.length() > 0) { // checa si hay algo que enviar
485             byte[] send = message.getBytes();//Obtenemos bytes del mensaje
486             Servicio_BT.write(send); //Mandamos a escribir el mensaje
487         }
488     } else Toast.makeText(this, "No conectado", Toast.LENGTH_SHORT).show();
489 } //fin de sendMessage
490

```

Figura 6-2. Conversión de mensaje a bytes

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Ahora los mensajes que recibe el smartphone se debe confirmar en texto lo cual se muestra en la figura 42-2.

```

529@ @SuppressWarnings("HandlerLeak")
530@ final Handler mHandler = new Handler() {
531@     @Override
532         public void handleMessage(Message msg) {
533
534             switch (msg.what) {
535
536                 case Mensaje_Escrito:
537                     break;
538
539                 case Mensaje_Leido:
540                     byte[] readBuf = (byte[]) msg.obj;
541                     String readMessage = new String(readBuf, 0, msg.arg1);//conversion de mensajes de bytes a txt
542                     sync(readMessage);
543                     break;
544
545                 case Mensaje_Nombre_Dispositivo:
546                     mConnectedDeviceName = msg.getData().getString(DEVICE_NAME); //Guardamos nombre del dispositivo
547                     TxtConectado.setText("Conectado BT a: "+mConnectedDeviceName);
548                     BtnConectar.setText("Desconectar");
549                     conBT=true;
550                     //Principal.this.sendMessage("SYNC\r");
551                     progressDialog.dismiss();
552                     break;
553
554                 case Mensaje_TOAST://mostramos por mensaje de toast una notificacion
555
556                     break;
557
558                 case MESSAGE_Desconectado:
559                     TxtConectado.setText(noConectado);
560                     BtnConectar.setText("Conectar");
561                     conBT=false;
562                     break;

```

Figura 7-2. Conversión de bytes a mensaje

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

En la conexión del Bluetooth se debe realizar de forma manual integrando la dirección Mac del dispositivo para tener conexión con la placa central.

```

50
51 //Tipos de mensaje enviados y recibidos desde el Handler de ConexionBT
52 public static final int Mensaje_Estado_Cambiado = 1;
53 public static final int Mensaje_Leido = 2;
54 public static final int Mensaje_Escrito = 3;
55 public static final int Mensaje_Nombre_Dispositivo = 4;
56 public static final int Mensaje_TOAST = 5;
57 public static final int MESSAGE_Desconectado = 6;
58 public static final int REQUEST_ENABLE_BT = 7;
59 public static final String DEVICE_NAME = "device_name";
60 public static final String TOAST = "toast";
61
62 //public static final String address = "20:13:86:13:18:51"; //Direccion Mac del hc-06
63
64 public static final String address = "98:10:31:80:05:17";
65
66
67 //Nombre del dispositivo conectado
68 private String mConnectedDeviceName = null;
69
70 // Adaptador local Bluetooth
71 private BluetoothAdapter AdaptadorBT = null;
72
73 //Objeto del servicio de ConexionBT
74 public static ConexionBT Servicio_BT = null;
--

```

Figura 8-2. Asignación de MAC del Bluetooth

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Cuando ingresamos un usuario le asignamos un nivel de ingreso para que la App. sepa que restringir al momento de ingresar, se realizó la programación como se muestra en la figura 44-2.

```
264 private void verCamaras(){
265     boolean requisitos=false;
266
267     if(nivelUsuario>2){           //nivel de usuario
268         sin_Acceso(2);
269         requisitos=false;
270     }else{
271         if(conLocal){           //conexion local
272             if(isWifiAvailable()){
273                 requisitos=true;
274             }else{
275                 requisitos=false;
276                 sin_Acceso(4);
277             }
278         }else{                 //conexion externa
279             if(isNetworkAvailable()){
280                 requisitos=true;
281             }else{
282                 requisitos=false;
283                 sin_Acceso(3);
284             }
285         }
286     }
287 }
```

Figura 9-2. Niveles de Usuarios

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Al iniciar la aplicación se visualiza un mensaje de emergencia si queremos tener conexión local o externa para que haga esto se programa de la siguiente manera.

```
222
223 private void verAulas(){
224     boolean requisitos=false;
225
226     if(conLocal){           //conexion local
227         if(conBT){
228             requisitos=true;
229         }else{
230             requisitos=false;
231             sin_Acceso(1);
232         }
233     }else{                 //conexion externa
234         if(nivelUsuario>2){
235             sin_Acceso(2);
236             requisitos=false;
237         }else{
238             if(isNetworkAvailable()){
239                 requisitos=true;
240             }else{
241                 requisitos=false;
242                 sin_Acceso(3);
243             }
244         }
245     }
246 }
```

Figura 10-2. Mensaje de conexión.

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

La conexión externa configuramos como en la figura donde se coloca la página web de ingreso al control externo con su debido puerto abierto.

```
20 public class Aulas_Externa extends Activity {
21
22     WebView visorAulas;
23     ProgressBar cargando;
24     public long[] pattern = {0,35};
25
26     private static final String urlAulas="http://iprexnorte.fgus.net:807";    //direccion del servidor de control arduino
27
28     @SuppressWarnings("SetJavaScriptEnabled")
29     @Override
30     protected void onCreate(Bundle savedInstanceState) {
31         super.onCreate(savedInstanceState);
32         //getWindow().requestFeature(Window.FEATURE_PROGRESS);
33         getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,WindowManager.LayoutParams.FLAG_FULLSCREEN);
34         setContentView(R.layout.a_aulasex);
35         //getWindow().setFeatureInt( Window.FEATURE_PROGRESS, Window.PROGRESS_VISIBILITY_ON);
36
37         visorAulas=(WebView)findViewById(R.id.webAulas1);
38         cargando=(ProgressBar)findViewById(R.id.barAu);
39         cargando.setVisibility(View.GONE);
```

Figura 11-2. Ingreso del dominio para acceso externo

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para la conexión de las cámaras 1 en eclipse se programa de la siguiente forma como se muestra en la figura

```
3*import android.annotation.SuppressLint;[]
12
13 @SuppressWarnings("NewApi")
14 public class F_CAM2 extends Fragment {
15
16     public View view;
17
18     WebView visorCam2;
19
20     private static final String url1Cam2="https://netcam.belkin.com/app/c/login.html?partnerId=Belkin";
21     private static final String url2Cam2="https://netcam.belkin.com/app/c/login.html?partnerId=Belkin";
22
23     private static boolean isExt;
24
25     public void onCreate(Bundle savedInstanceState) {
26         super.onCreate(savedInstanceState);
27     }
28
29     @SuppressWarnings("SetJavaScriptEnabled")
30     @Override
31     public View onCreateView(LayoutInflater inflater, ViewGroup container,
32         Bundle savedInstanceState) {
33         view= inflater.inflate(R.layout.f_cam2, container, false);
34
35         visorCam2= (WebView)view.findViewById(R.id.webCam2);
36
37         try {
38             //saber si es conexion local o externa *opcional
39             if(isExt){visorCam2.loadUrl(url1Cam2);}
40             else{visorCam2.loadUrl(url2Cam2);}
41
42             visorCam2.getSettings().setBuiltInZoomControls(true);
43             visorCam2.getSettings().setDisplayZoomControls(false);
44             visorCam2.getSettings().setJavaScriptEnabled(true);
```

Figura 12-2. Acceso a la cámara 1

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

De igual forma se configuro la cámara 2 como en la figura 48-2.

```
4*import android.annotation.SuppressLint;[]
17
18 @SuppressWarnings("NewApi")
19 public class F_CAM2 extends Fragment {
20
21     public View view;
22
23     WebView visorCam2;
24     TextView infocam;
25     ProgressBar cargando;
26
27     final String user="iprexperto";
28     final String pass="iprexperto";
29
30     private static final String url1Cam2="http://192.168.0.5:8080"; //local
31     private static final String url2Cam2="http://iprexperto.ddns.net:8080"; //externo
32
33     private static boolean isLoc;
34
35     public void onCreate(Bundle savedInstanceState) {
36         super.onCreate(savedInstanceState);
37
38     }
```

Figura 13-2. Acceso a la cámara 2

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para el control de la pantalla de forma local se utilizó 3 botones de las cuales 2 son visibles y 1 es invisible así como se muestra en el siguiente código.

```

74     cargando.setVisibility(View.INVISIBLE);
75     infoestado.setText(estado3);
76     estadoPantalla1=0;
77     imgUP.setBackground(getResources().getDrawable(R.drawable.up));
78     imgDWN.setBackground(getResources().getDrawable(R.drawable.down));
79     r.sendMessage("S\r");
80 }
81
82 private void Movimientos(int mov){
83
84     if(mov==estadoPantalla1){
85         r.sendMessage("S\r");
86         infoestado.setText(estado3);
87         cargando.setVisibility(View.INVISIBLE);
88         estadoPantalla1=0;
89         imgUP.setBackground(getResources().getDrawable(R.drawable.up));
90         imgDWN.setBackground(getResources().getDrawable(R.drawable.down));
91     }else{
92         switch(mov){
93             case 1:
94                 r.sendMessage("U\r");
95                 infoestado.setText(estado2);
96                 estadoPantalla1=1;
97                 cargando.setVisibility(View.VISIBLE);
98                 imgUP.setBackground(getResources().getDrawable(R.drawable.up));
99                 imgDWN.setBackground(getResources().getDrawable(R.drawable.down));
100                break;
101             case 2:
102                 r.sendMessage("D\r");
103                 infoestado.setText(estado1);
104                 estadoPantalla1=2;
105                 cargando.setVisibility(View.VISIBLE);
106                 imgUP.setBackground(getResources().getDrawable(R.drawable.up));
107                 imgDWN.setBackground(getResources().getDrawable(R.drawable.down));
108                break;
109                 r.sendMessage("S\r");
110                 infoestado.setText(estado3);
111                 estadoPantalla1=0;
112                 cargando.setVisibility(View.INVISIBLE);
113                 imgUP.setBackground(getResources().getDrawable(R.drawable.up));
114                 imgDWN.setBackground(getResources().getDrawable(R.drawable.down));
115                break;
116            }
117         }
118     }

```

Figura 14-2. Control de la pantalla de proyección

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2 Diseño e implementación de hardware

2.2.1 Topología de Red

Descrito en el capítulo anterior y como trabajaremos tanto con tecnología ZigBee como Bluetooth escogeremos una topología que soporte las dos tecnologías, entre ellas tenemos: estrella, bus y malla.

En nuestro proyecto se trabajara con la topología tipo estrella, debido a la central está ubicada como punto centro, a la inexistente comunicación entre dispositivos finales y a su bajo costo.

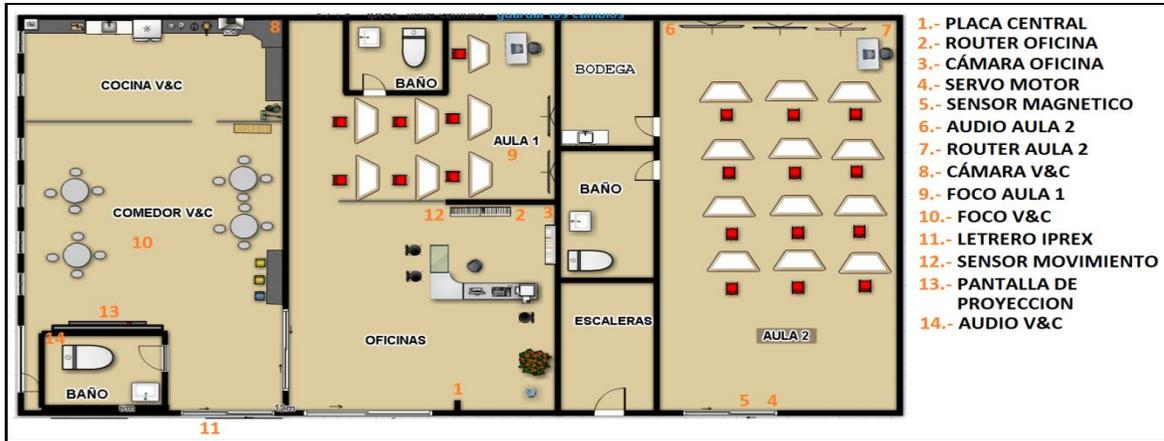


Figura 15-2. Infraestructura de IPREX

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

En la figura. Podemos observar los dispositivos a controlar y el lugar específico en el que se encuentran, los mismos que están conectados a la placa central.

2.2.2 Sistema de conexión local

Es el sistema central encargado de toda la gestión de sensores inalámbricos y actuadores, de fácil acceso, el acceso debe ser físico gracias al uso de un transceptor en nuestro caso un bluetooth.

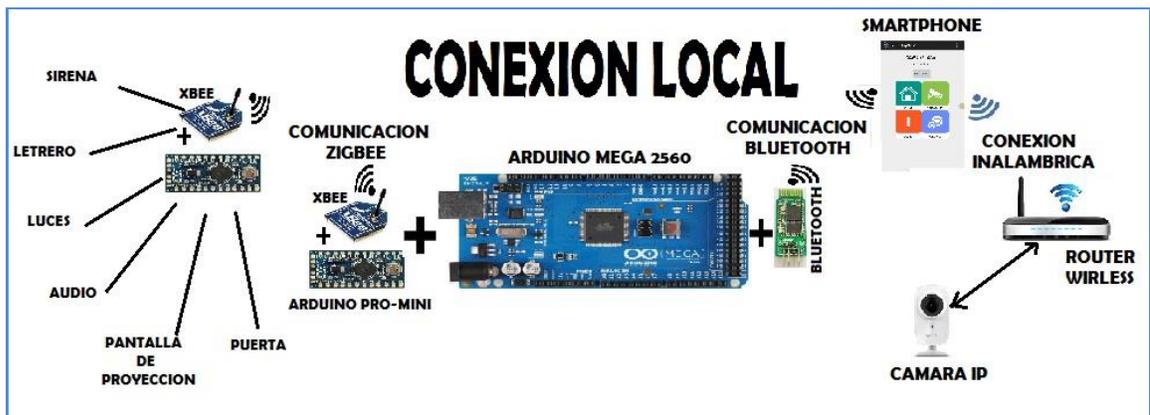


Figura 16-2. Conexión Local

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016


```

if(xByte=='R')
{
  digitalWrite(R, HIGH);
}
if(xByte=='r')
{
  digitalWrite(R, LOW);
}
if(xByte=='L')
{
  digitalWrite(L, HIGH);
}
if(xByte=='l')
{
  digitalWrite(L, LOW);
}
if(xByte=='M')
{
  digitalWrite(Mic, HIGH);
}
if(xByte=='m')
{
  digitalWrite(Mic, LOW);
}
if(xByte=='F')
{
  digitalWrite(luminaria1, HIGH);
}
if(xByte=='f')
{
  digitalWrite(luminaria1, LOW);
}
if(xByte=='W')
{
  digitalWrite(luminaria2, HIGH);
}
if(xByte=='w')
{
  digitalWrite(luminaria2, LOW);
}
if(xByte=='U')
{
  digitalWrite(up, HIGH);
  digitalWrite(paro, LOW);
  digitalWrite(down, LOW);
}
if(xByte=='D')
{
  digitalWrite(down, HIGH);
  digitalWrite(up, LOW);
  digitalWrite(paro, LOW);
}
if(xByte=='S')
{
  digitalWrite(paro, HIGH);
  delay(100);
  digitalWrite(paro, LOW);
  digitalWrite(up, LOW);
  digitalWrite(down, LOW);
}
if(xByte=='B')
{
  digitalWrite(led, HIGH);
}
if(xByte=='b')
{
  digitalWrite(led, LOW);
}

```

Figura 18-2. Comunicación Arduino-Bluetooth

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.3 Sistema de conexión externa

Además debe comunicarse con el exterior, realizando un acceso remoto mediante un shield Ethernet, para permitir la actuación directa con el sistema entero. Nuestro sistema está formado como en la figura 54-2.

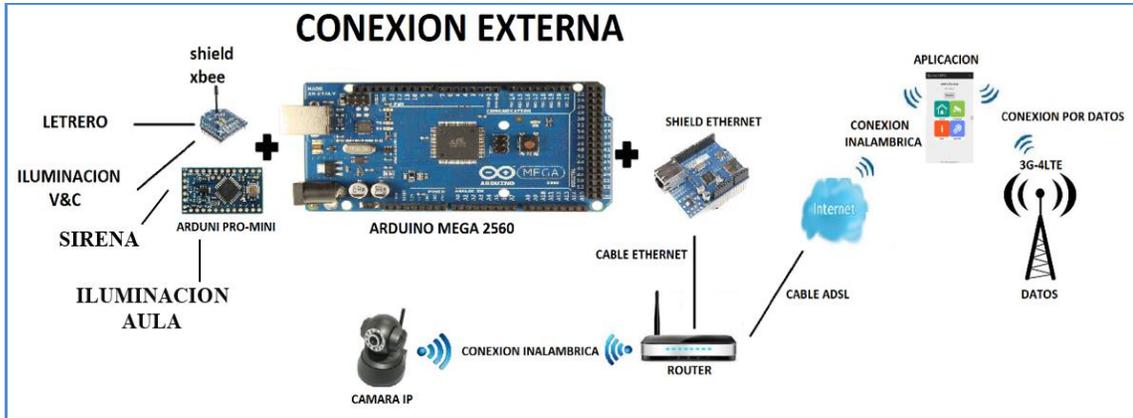


Figura 19-2. Conexión Externa

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.3.1 Shield ethernet

Para la comunicación externa utilizamos un shield ethernet como lo muestra en la figura 54-2 el cual nos permitirá poder controlar de forma externa las iluminarias y la sirena que se encuentra instala en la placa central.

En la figura se podrá observar la posición final del shield en la placa central

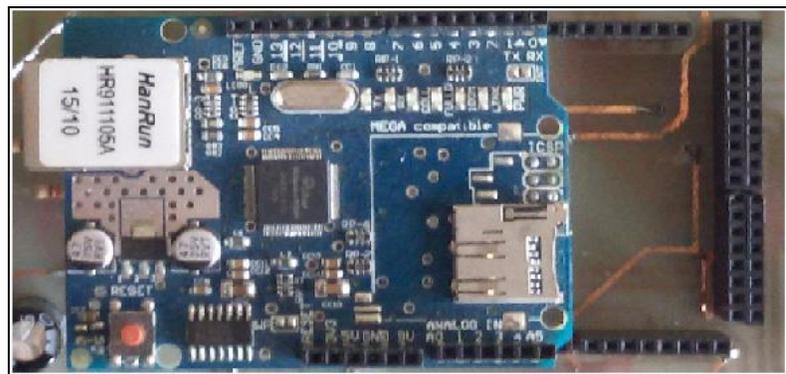


Figura 20-2. Shield Ethernet

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

El código que se colocó para que pueda realizar la comunicación entre el shield ethernet y Arduino se lo podrá observar en la figura.

```

cliente.println("HTTP/1.1 200 OK");
cliente.println("Content-Type: text/html");
cliente.println(); //Página Web en HTML
cliente.println("<html>");
cliente.println("<head>");
cliente.println("<title>CONTROL PANEL </title>");
cliente.println("</head>");
cliente.println("<body width=100% height=100%>");
cliente.println("<meta http-equiv='refresh' content='1' />");

cliente.println("<center>");
cliente.println("<br>");
cliente.println("LED : ");
cliente.println(stat6);
cliente.println("<br>");

cliente.println("<center>");
cliente.println("<br>");
cliente.println("LED : ");
cliente.println(stat7);
cliente.println("<br>");

cliente.println("<center>");
cliente.println("<br>");
cliente.println("LED : ");
cliente.println(stat8);
cliente.println("<br>");

cliente.println("<center>");
cliente.println("<br>");
cliente.println("LUMINARIA 1 : ");
cliente.println(stat1);
cliente.println("<br>");
cliente.println("<input type=submit value=ON style=width:80px;height:40px onClick=location.href='./?LE1=T' />");
cliente.println("<input type=submit value=OFF style=width:80px;height:40px onClick=location.href='./?LE1=F' />");
cliente.println("</body>");
cliente.println("</html>");
cliente.stop();
//Cierro conexión con el cliente
readString="";
.
cliente.println("<center>");
cliente.println("LUMINARIA 2 : ");
cliente.println(stat2);
cliente.println("<br>");
cliente.println("<input type=submit value=ON style=width:80px;height:40px onClick=location.href='./?LE2=T' />");
cliente.println("<input type=submit value=OFF style=width:80px;height:40px onClick=location.href='./?LE2=F' />");
cliente.println("</center>");

cliente.println("<center>");
cliente.println("Luminaria 3 : ");
cliente.println(stat3);
cliente.println("<br>");
cliente.println("<input type=submit value=ON style=width:80px;height:40px onClick=location.href='./?L1=T' />");
cliente.println("<input type=submit value=OFF style=width:80px;height:40px onClick=location.href='./?L12=F' />");
cliente.println("</center>");

cliente.println("<center>");
cliente.println("sirena 4 : ");
cliente.println(stat4);
cliente.println("<br>");
cliente.println("<input type=submit value=ON style=width:80px;height:40px onClick=location.href='./?L1=T' />");
cliente.println("<input type=submit value=OFF style=width:80px;height:40px onClick=location.href='./?L12=F' />");
cliente.println("</center>");

cliente.println("<center>");
cliente.println("Led : ");
cliente.println(stat5);
cliente.println("<br>");
cliente.println("<input type=submit value=ON style=width:80px;height:40px onClick=location.href='./?L1=T' />");
cliente.println("<input type=submit value=OFF style=width:80px;height:40px onClick=location.href='./?L12=F' />");
cliente.println("</center>");

```

Figura 21-2. Reconocimiento Shield Ethernet

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.4 *Requerimientos y elección del controlador arduino*

El objetivo por el cual se escogió Arduino mega 2560 fue por ganar la tabla de valores por colocamos algunas posibilidades donde podíamos optar.

En la siguiente tabla podremos observar de qué forma se hizo la elección:

Tabla 2-2. Selección de Arduino

	ARDUINO UNO	ARDUINO MEGA 2560	ARDUINO LEONARDO	ARDUINO PRO-MINI
PRECIO	1	0	0	1
PINES DE CONEXIÓN	0	1	0	0
MEMORIA DE PROCESO	0	1	0	0
FRECUENCIA DE RELOJ	1	1	1	0
MEMORIA FLASH	0	1	0	0
TOTAL	2	4	1	1

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

El arduino será el cerebro de nuestro Sistema, ya que en el reside todo el programa de control de los diferentes dispositivos, el mismo que tendrá un acceso local y remoto, remoto mediante la conexión ethernet al router servidor de internet.

Una vez establecidos los requerimientos podemos indagar sobre la familia Arduino ya que posee un gran número de controladores con todas las características necesarias y considerando que se manejara un número considerable de dispositivos se elige al Arduino Mega, indicado en la tabla 6-2, cuyas características se describieron en el capítulo 1.

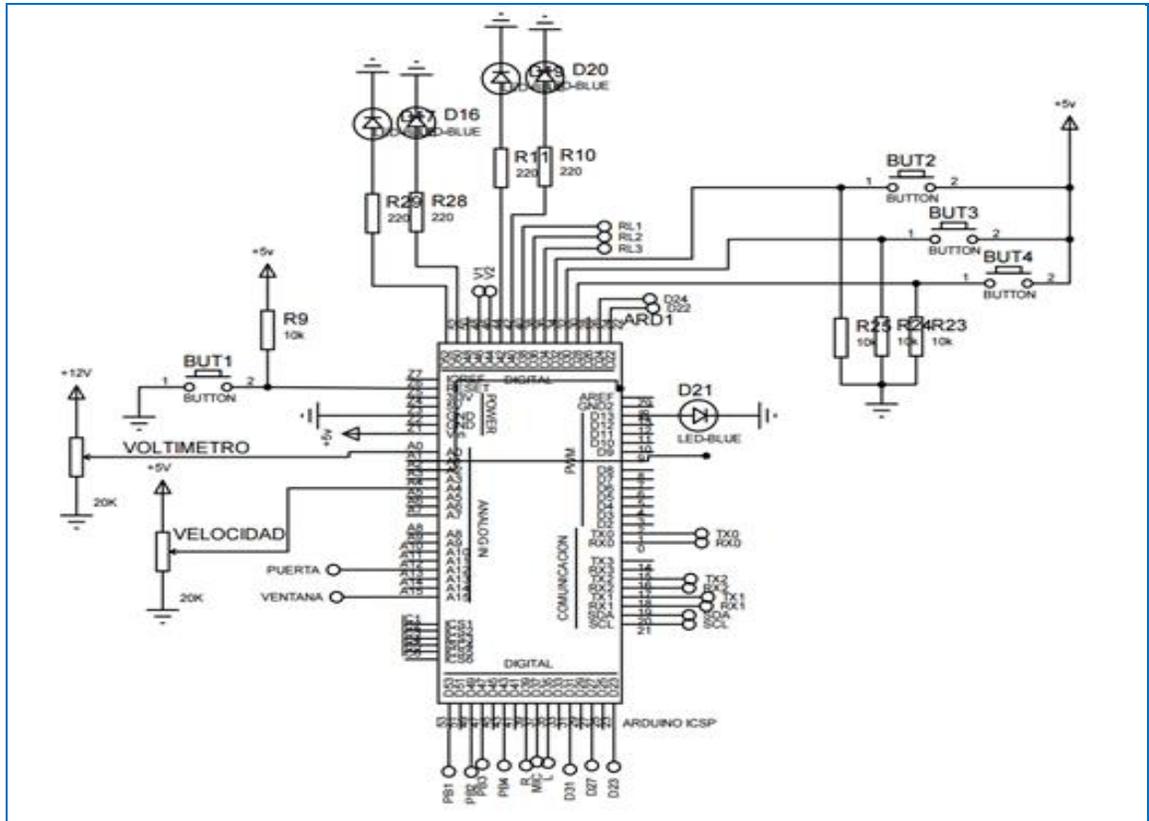


Figura 22-2. Simulación Arduino en Proteus

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.5 Módulos de Comunicación Inalámbrica

Unos de los objetivos del proyecto es el diseño inalámbrico para el manejo de dispositivos de difícil alcance, el mismo que debe cumplir con los requerimientos del estándar IEEE 802.15.4. Para ello se escogió como base el circuito integrado XBee, ya que este cumple con el estándar antes mencionado, incluso pertenece a la alianza ZigBee.

2.2.5.1 Módulos Xbee

Dentro de las características tenemos que el módulo Xbee posee un alcance en interiores de 30 m, en nuestro proyecto requerimos de 10m sin línea de vista, por lo cual hemos escogido el módulo XB4214A (Xbee s1 con antena incorporada).

En la figura se muestra nuestro módulo xbee emisor donde lo podremos visualizar tanto el diseño como en la etapa final.

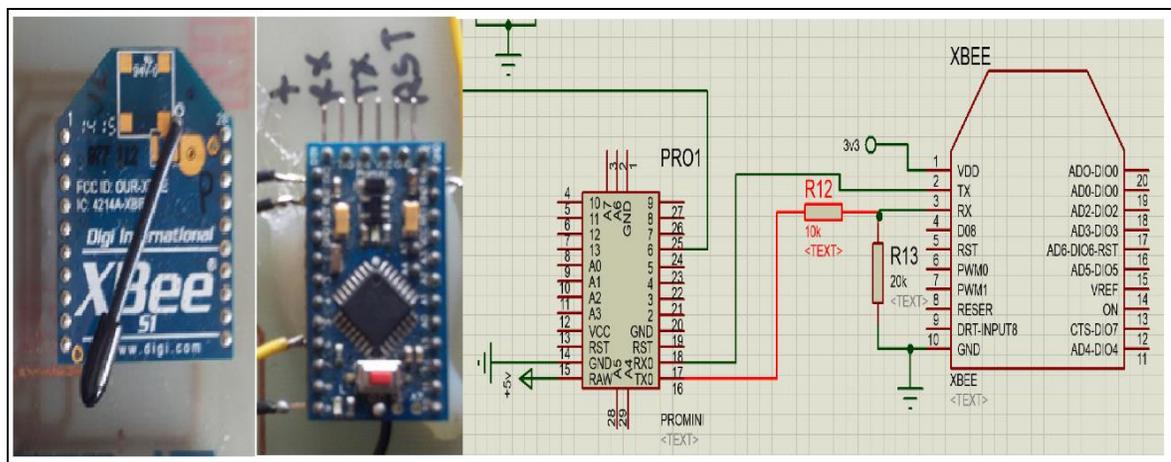


Figura 23-2. Simulación Xbee en Proteus

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

El código que permite tener esta comunicación de forma inalámbrica se podrá observar en la 59-2.

```
int p1 = 6;
int p2 = 9;

void setup() {
  Serial.begin(9600);

  pinMode(p1, INPUT);
  pinMode(p2, INPUT);
}

void loop() {
  int estadop1 = digitalRead(p1);
  int estadop2 = digitalRead(p2);

  if (estadop1==HIGH)
  {
    Serial.println("H");
    delay(1000);
  }
  if (estadop1==LOW)
  {
    Serial.println("h");
    delay(1000);
  }
}
```

Figura 24-2. Código de activación Xbee

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Como podrán observar en el diseño de la etapa del receptor colocamos un arduino pro-mini cuya función es recibir un alto o un bajo (1 ó 0) del Arduino mega 2560, este procesara esa información y la enviara por sus salidas seriales las cuales están conectadas al xbee, de esta manera el xbee procesara los datos y la enviara al receptor.

El diseño de la etapa de recepción lo podremos observar en la figura.


```

#include <Servo.h>
char xByte=0;
int rele =9;
int led=13;
int state=12;
void setup()
{
  Serial.begin(9600);
  pinMode(rele, OUTPUT);
  pinMode(led, OUTPUT);
  pinMode(state, OUTPUT);
}

void loop()
{
  digitalWrite(led, HIGH);
  if (Serial.available() > 0)
  {
    xByte=Serial.read();

    digitalWrite(state, HIGH);
    delay(50);
    digitalWrite(state, LOW);

    if(xByte=='H')
    {
      digitalWrite(rele, HIGH);
      delay(100);
    }
    if(xByte=='h')
    {
      digitalWrite(rele, LOW);
      delay(100);
    }
  }
}

```

Figura 26-2. Código para Arduino Pro mini

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Los códigos que transportan en el proceso de comunicación se podrá visualizar en la figura 62-2.

```

if(xByte=='Z')
{
  digitalWrite(p1, HIGH);

  delay(100);
}

if(xByte=='z')
{
  digitalWrite(p1, LOW);

  delay(100);
}

```

Figura 27-2. Código de encendido y apagado

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.6 Diseño de etapas de la placa central

2.2.6.1 Etapa de alimentación

En la etapa de alimentación se realizó dos formas de alimentar el circuito con corriente directa de ellas como muestra en la siguiente figura donde solo alimentamos la placa central por medio de un cargador de 12v a 3Amp.

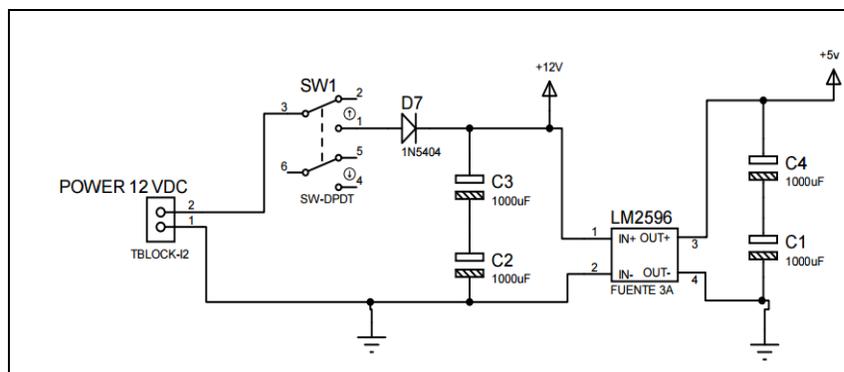


Figura 28-2. Simulación Etapa de Alimentación

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

En la siguiente figura aquí se realizó un cambio a la placa central para así poder obtener 2 formas de alimentar el circuito como una alternativa para cortes de luz y la placa se quede sin alimentación.

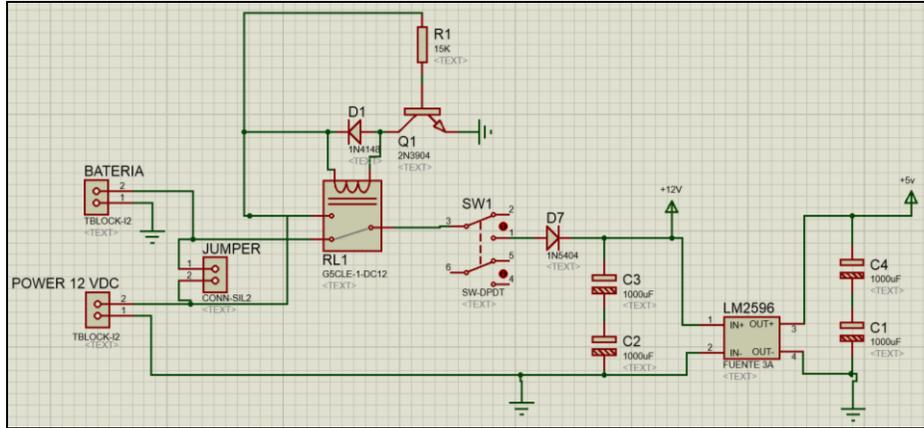


Figura 29-2. Simulación de adaptación de batería

Fuente: Realizado por: Loayza J, Padilla E. 2016

En toda la etapa de alimentación se utilizó una fuente de alimentación conmutada el LM2596 o también conocidas como fuentes switching, el objetivo de colocar este elemento en la placa central es que los reguladores comunes hacen es que si tenemos en la entrada 12v y queremos solo que salga 5v debemos de utilizar un regulador de voltaje 7805 para obtener a la salida 5v requeridos pero lo demás del voltaje se hace calor y es por eso que a esos reguladores hay que colocarles un disipador de calor por suelen recalentarse, en cambio una fuente switching hace todo lo contrario como es una fuente regulable podemos optar por controlar la salida de voltaje deseada independientemente la entrada que tengamos ya esta fuente que utilizamos soporta hasta 24v.

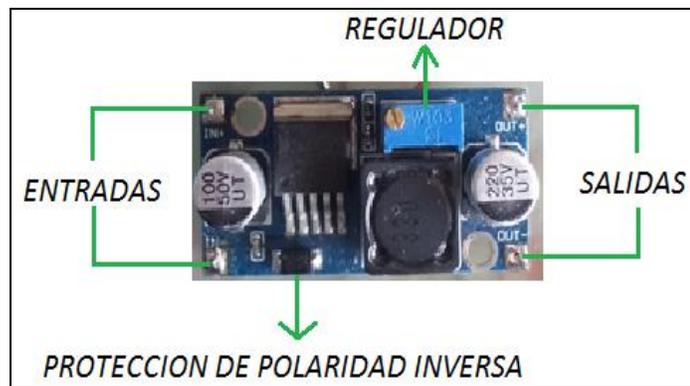


Figura 30-2. LM2596

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.7 Actuadores y controladores

2.2.7.1 Sensor de movimiento o Pir

La función que cumple el sensor PIR es controlar el acceso a intrusos, de forma que solo se activara en la noche para que de este modo pueda controlar esta necesidad.

Como se podrá observar en la figura nos daremos cuenta que se encuentra el diseño del sensor.



Figura 31-2. Sensor detector de movimiento

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

De este modo como se dijo anteriormente que solo cumplirá su función de modo nocturno, ahora para poder bloquear en el día lo desactivaremos por medio de la app como se mostrara en la figura 67-2 el código en Arduino.

```

if (pirE == HIGH) {
    digitalWrite(ledsensor, HIGH);
    stat8="MOVIMIENTO ON";
}
else {
    digitalWrite(ledsensor, LOW);
    stat8="MOVIMIENTO OFF";
}

}

if(alarma==1 && puertaE==LOW )
{
    digitalWrite(paro, HIGH);
}
if(alarma==1 && pirE==HIGH )
{
    digitalWrite(paro, HIGH);
}

if(alarma==0 && puertaE==HIGH)
{
    digitalWrite(paro, LOW);
}

if(alarma==0 && pirE==LOW)
{
    digitalWrite(paro, LOW);
}

}

```

Figura 32-2. Código Sensor Pir

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.7.2 Sensor magnético

La función que cumplirá este sensor es indicar el momento que la puerta se ha desbloqueado o se encuentra bloqueada, una segunda función que cumplirá este sensor es activar la sirena si en caso la puerta se abrió en caso contrario esta de forma desactivada lo cual lo podremos realizar es desactivando la sirena por la aplicación.

```

if (puertaE == LOW) {
    digitalWrite(ledpuerta, HIGH);
    stat7="PUERTA ABIERTA";
}
else {
    digitalWrite(ledpuerta, LOW);
    stat7="PUERTA CERRADA";
}

```

Figura 33-2. Activación de sensor magnético

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.7.3 Servo-Motor

Para el control del bloqueo y desbloqueo de la puerta utilizamos un servo-motor el cual realiza un giro de 45 grados, para poder lograr este tipo de giro se lo realizo en la programación del arduino.

```

if(xByte=='P')
{
    lcd.setCursor(0,3);
    lcd.print("Puerta   Abierta");
    myservo.write(165);
}

if(xByte=='p')
{
    lcd.setCursor(0,3);
    lcd.print("Puerta   Cerrada");
    myservo.write(90);
}

```

Figura 34-2. Visualización de estado Servo-Motor

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.7.4 Visualización o lcd

En la etapa de visualización se mostrara la activación y desactivación de los actuadores que están siendo controlador por la placa central

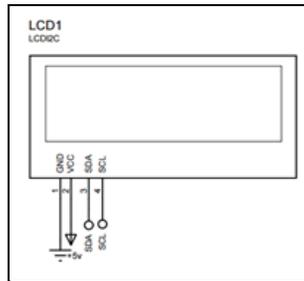


Figura 35-2. LCD

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

En la figura podremos visualizar la programación que realizamos en el Arduino.

```
lcd.backlight();
lcd.setCursor(3,0); //Start at character 4 on line 0
lcd.print("Iniciando");
delay(250);
lcd.noBacklight();
delay(250);

    lcd.backlight();
lcd.setCursor(3,0); //Start at character 4 on line 0
lcd.print("Iniciando.");
delay(250);
lcd.noBacklight();
delay(250);

    lcd.backlight();
lcd.setCursor(3,0); //Start at character 4 on line 0
lcd.print("Iniciando..");
delay(250);
lcd.noBacklight();
delay(250);

    lcd.backlight();
lcd.setCursor(1,0); //Start at character 4 on line 0
lcd.print("ANALIZANDO ESTADO");
lcd.setCursor(4,1);
    lcd.print("DEL SISTEMA");
delay(1000);
    lcd.clear();
    lcd.setCursor(1,0);
    lcd.print("BIENVENIDO A IPREX");
    lcd.setCursor(2,1);
    lcd.print("Sistema Listo");
```

Figura 36-2. Código de visualización de lcd

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.7.5 Etapa de iluminación

Existen dos etapas de iluminación, la primera de unos conectores led los cuales va a iluminar a la caja donde se va a encontrar la placa central.

El diseño que se realizó para la tira de led se podrá visualizar en la figura.

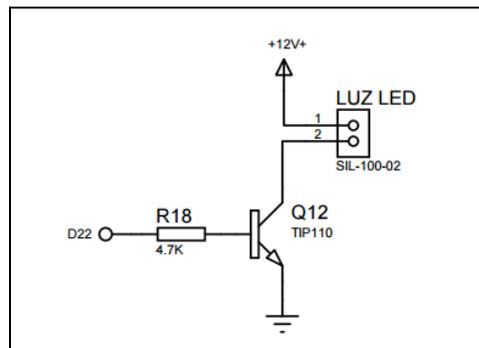


Figura 37-2. Simulación Luz Led

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

El código para que se active la tira de leds que se realizó en el Arduino se puede visualizar en la figura.

```
if(xByte=='B')
{
    digitalWrite(led, HIGH);
}

if(xByte=='b')
{
    digitalWrite(led, LOW);
}
```

Figura 38-2. Encendido de Leds

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

En la segunda etapa de iluminacion se utilizo una configuracion de conmutador ya que con esto podriamos realizar el proceso de forma manual o por la app, en la siguiente imagen se mostrara el tipo de instalacion que se realizo con un conmutador electrico y la placa central

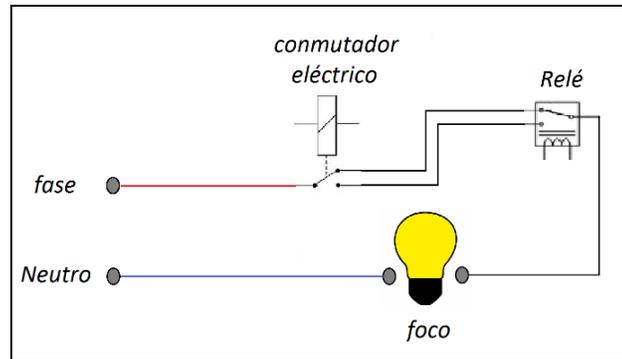


Figura 39-2. Topología física de conmutador manual

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

La figura muestra el diseño que esta hecho en la software proteus en el cual tiene la etapa electronica y sus 3 salidas el cual lo utilizaremos para la instalacion electrica con el el conmutador.

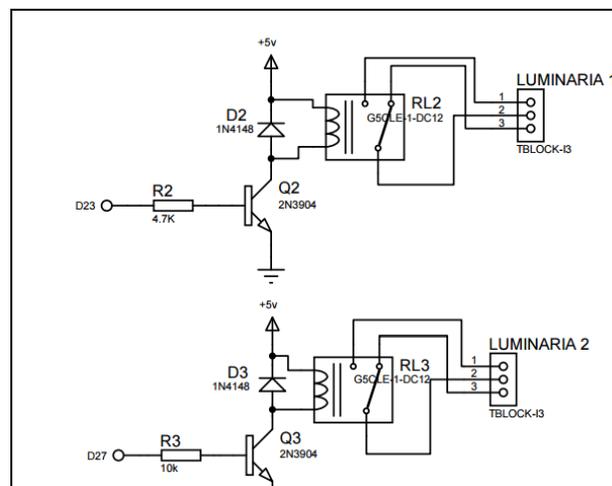


Figura 40-2. Simulación de luminarias

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para que la placa central pueda reconocer el proceso que debe realizar al momento que la app le envíe un dato primero se tuvo que realizar un programación al Arduino el cual se lo podrá ver en la figura.

```
//Encender luminaria 1
if(xByte=='F')
{
  digitalWrite(luminaria1, HIGH);
}
//Apagar luminaria 1
if(xByte=='f')
{
  digitalWrite(luminaria1, LOW);
}

//Encender luminaria 2
if(xByte=='W')
{
  digitalWrite(luminaria2, HIGH);
}
//Apagar luminaria 1
if(xByte=='w')
{
  digitalWrite(luminaria2, LOW);
}
```

Figura 41-2. Código de activación en Arduino

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.2.7.6 *Etapa de audio*

EL proceso que realiza el audio es el control de parlantes y micrófono en donde solo se podrá realizar este proceso por medio local.

Como se podrán dar cuenta en el figura se diseñó una etapa del micrófono con amplificación, el me permitirá regular el volumen del micrófono que va a salir en los parlantes.

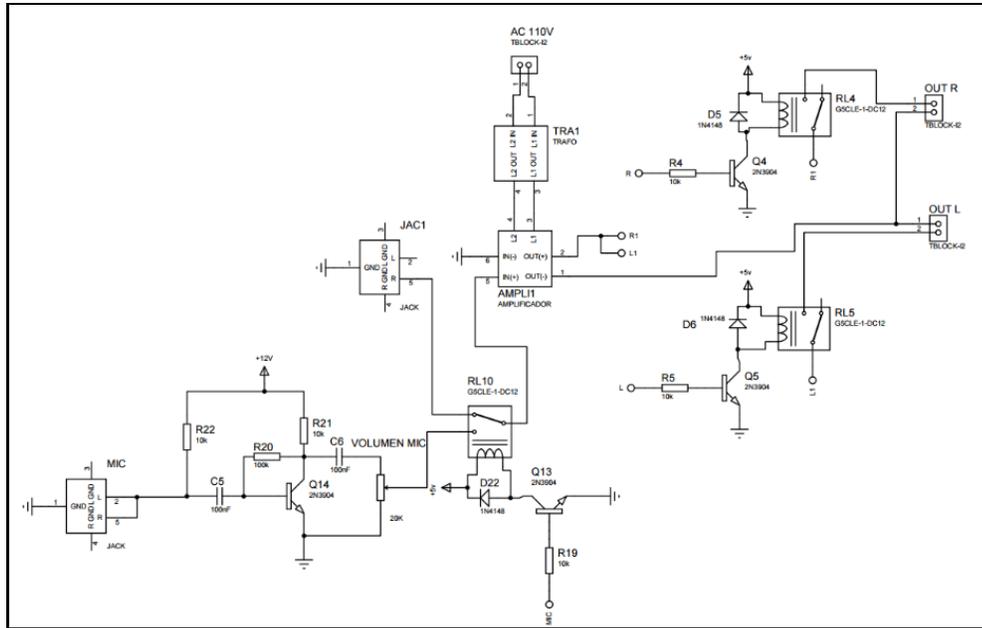


Figura 42-2. Simulación control de audio

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para que todas estas funciones pueda mi placa central manejar, se programa el Arduino Mega 2560 con siguiente código el cual se podrá visualizar en la figura 78-2.

```

if(xByte=='R')
{
digitalWrite(R, HIGH);
}
//Desactivar Audio R
if(xByte=='r')
{
digitalWrite(R, LOW);
}
//Activar Audio L
if(xByte=='L')
{
digitalWrite(L, HIGH);
}
//Desactivar Audio L
if(xByte=='l')
{
digitalWrite(L, LOW);
}

//Activar microfono
if(xByte=='M')
{
digitalWrite(Mic, HIGH);
}

//Desctivar microfono
if(xByte=='m')
{
digitalWrite(Mic, LOW);
}

```

Figura 43-2. Código de control de audio en Arduino

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

2.3 Configuración de equipos para acceso remoto

Para poder acceder externamente a todo el sistema de control se lo realizo mediante una conexión ethernet. Para ello debemos realizar algunos cambios en los equipos servidores de internet, como son los router.

Como podemos observar en la figura anterior tenemos dos router conectados entre sí, esto es debido a que el segundo router realiza la función de Zona WiFi para el local V&C.

La aplicación se conecta mediante internet con la central, pero para poder acceder remotamente se necesita conocer la dirección IP fija publica, pero como sabemos las empresas que brindan internet domestico no dan una IP fija, sino una IP dinámica, es decir una IP pública diferente cada vez que nos conectamos, una solución sería contratar una IP fija publica a cualquier empresa proveedora de internet, al uso de esta IP la denominan servicio corporativo. Sin embargo existen los servidores DDNS, que no son más que un Sistema Dinámico de Nombres de Dominio, cuya función nos permite configurar un router para asociarlo mediante un nombre de dominio a una dirección IP.

Existen varios servidores DDNS, los más conocidos son: DynDNS, No-IP, ChangeIP, etc. De estos solo No-IP es gratuito en toda su extensión, los demás son de paga o poseen restricciones, pero realmente debemos ingresar a nuestro router, ya que no todos los router soportan todos los servidores DDNS.

Una vez conocido este servicio procedemos a ingresar a nuestro Router para su debida configuración.

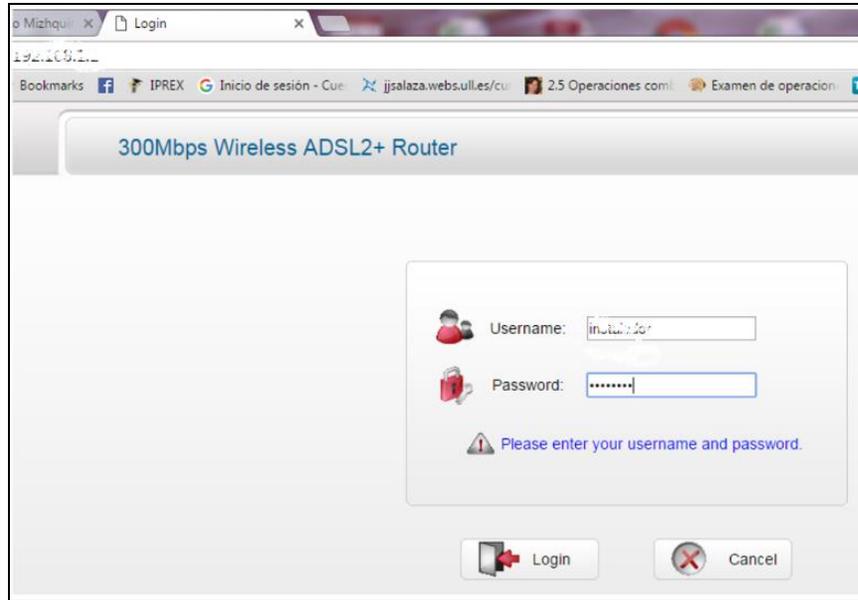


Figura 44-2. Ingreso a router CNT

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Nuestro router Huawei no soporta ninguno de los servidores DDNS gratuitos, entonces procedemos a ingresar a nuestro router Tp-Link para verificar si soporta un DDNS gratuito.

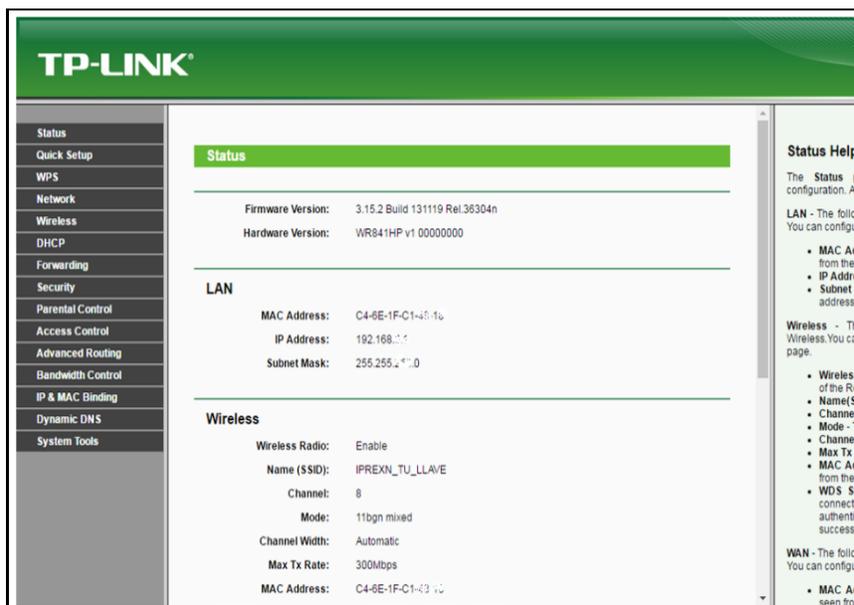


Figura 45-2. Página principal Router Tp-Link

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

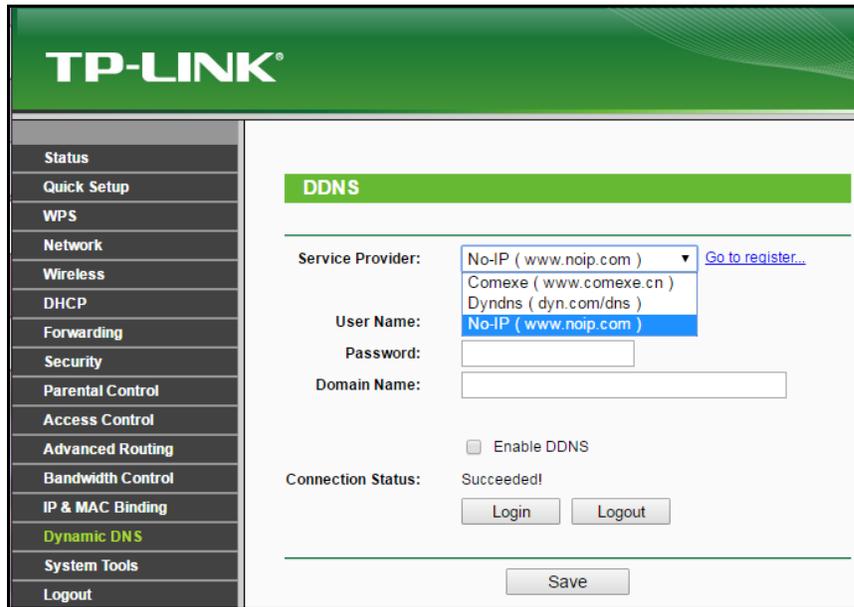


Figura 46-2. Verificación del DDNS gratuito

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Como podemos observar en la figura 81-2 nuestro router Tp-link si soporta un servidor gratuito, como es el No-IP, por lo que a continuación procedemos a crear primero una cuenta en este servidor.

2.3.1 Creación de cuenta No-IP

Observamos que está disponible el nombre de nuestro dominio como en la figura 82-2 y procedemos a registrarnos.

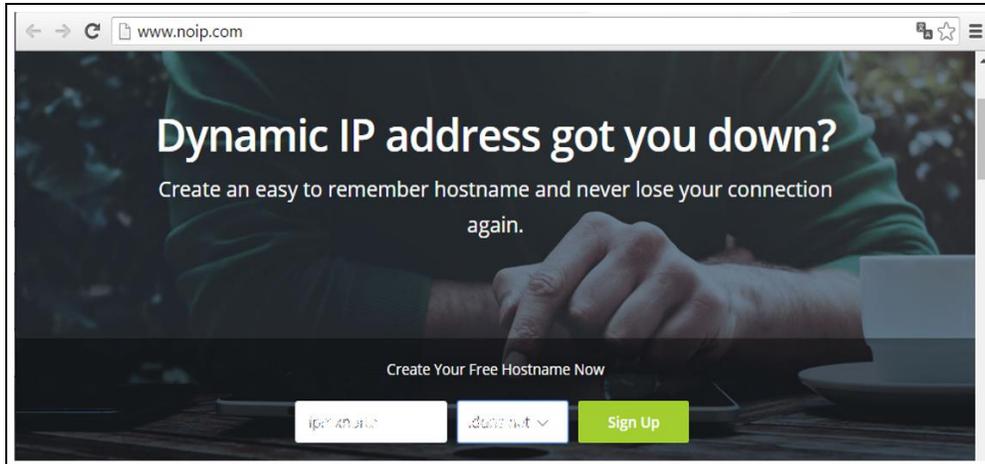


Figura 47-2. Página de creación de cuenta

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Ingresamos todos los datos solicitados y requeridos en la figura 83-2.

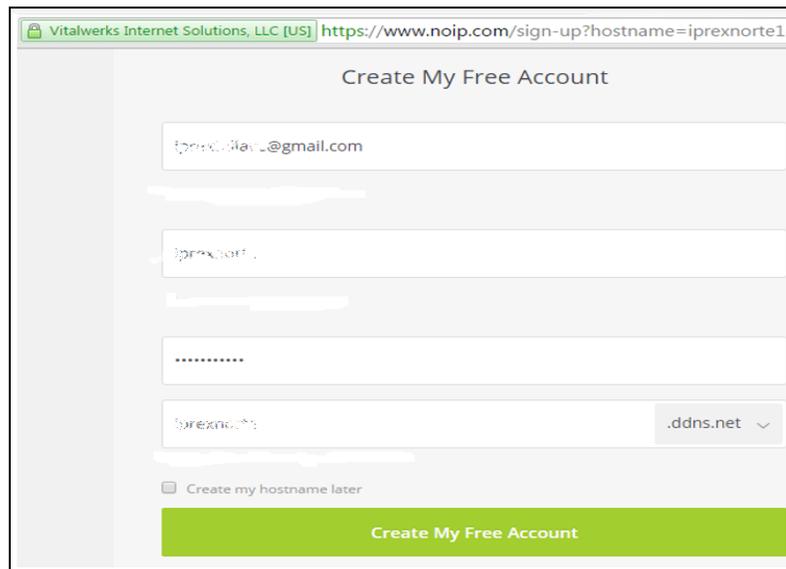


Figura 48-2. Registro de información

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para habilitar nuestra cuenta se debe confirmar desde el correo.

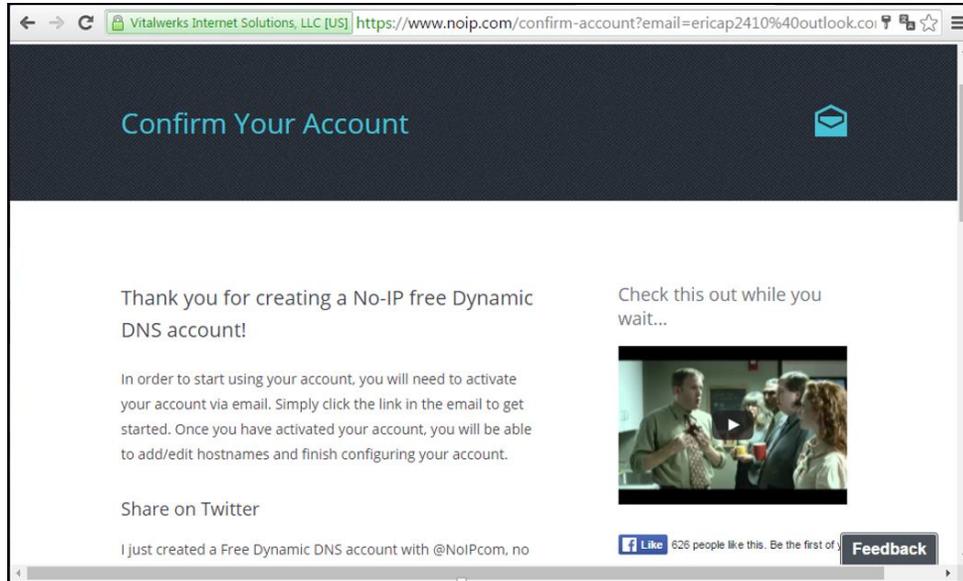


Figura 49-4. Solución de confirmación

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Una vez confirmado nuestra cuenta ya podemos ingresar. Comprobamos que nuestros datos estén bien en información de cuenta.

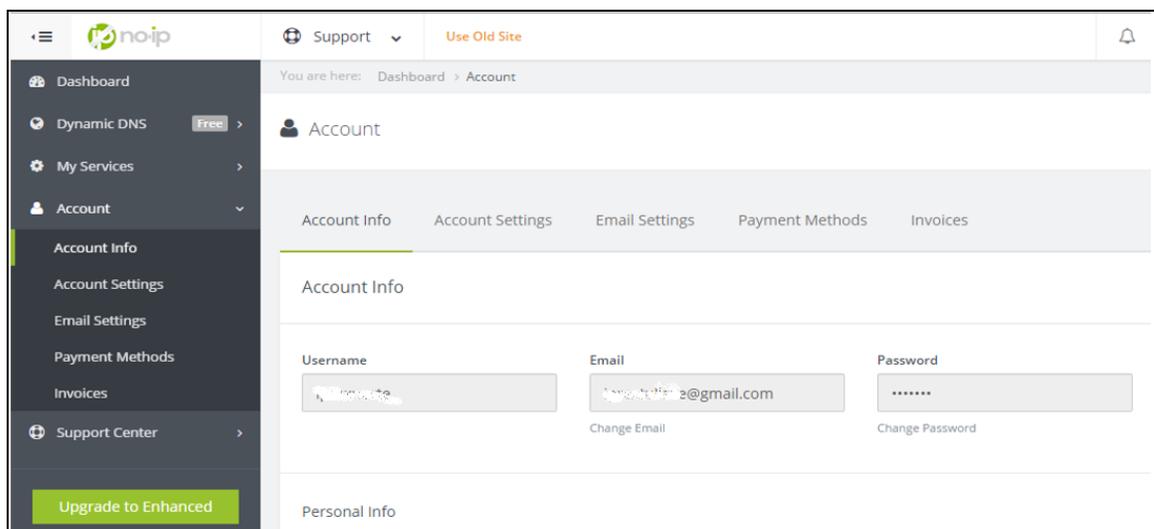


Figura 50-2. Información de cuenta

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Una vez ya con nuestra cuenta se añade el host que se va a utilizar en nuestro caso como se muestra en la figura 86-2. Donde podemos observar en la parte de la dirección se añade sola, ya que esta es detectada automáticamente por la red de internet a la que se encuentra conectada.

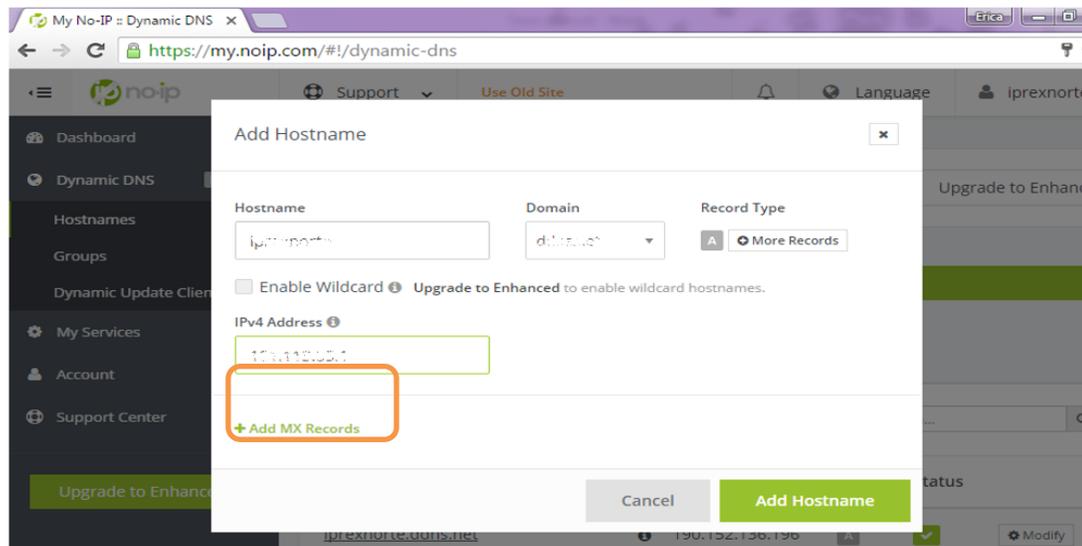


Figura 51-2. Añadir Host

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Ya añadido nuestro host y detectada nuestra dirección IP, este servidor nos va a indicar su estado y la fecha de la última actualización, como indica en la figura 87-2

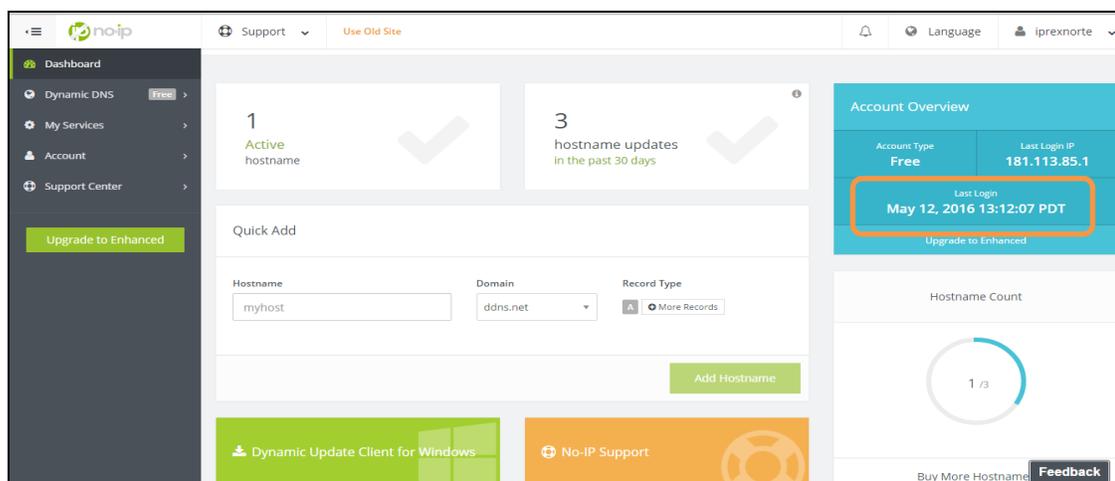


Figura 52-2. Actualizaciones de cuenta

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

En resume el proceso de funcionamiento para el No-IP es el siguiente: Creamos una cuenta, cuando el router conecta a Internet obtiene la dirección IP actual, envía la información de su IP al servidor, mediante la cuenta registrada. A partir de ese momento, el host queda asociado a la dirección IP.

Ya con nuestra cuenta lista y activada procedemos a configurar en el router como se muestra en la figura.

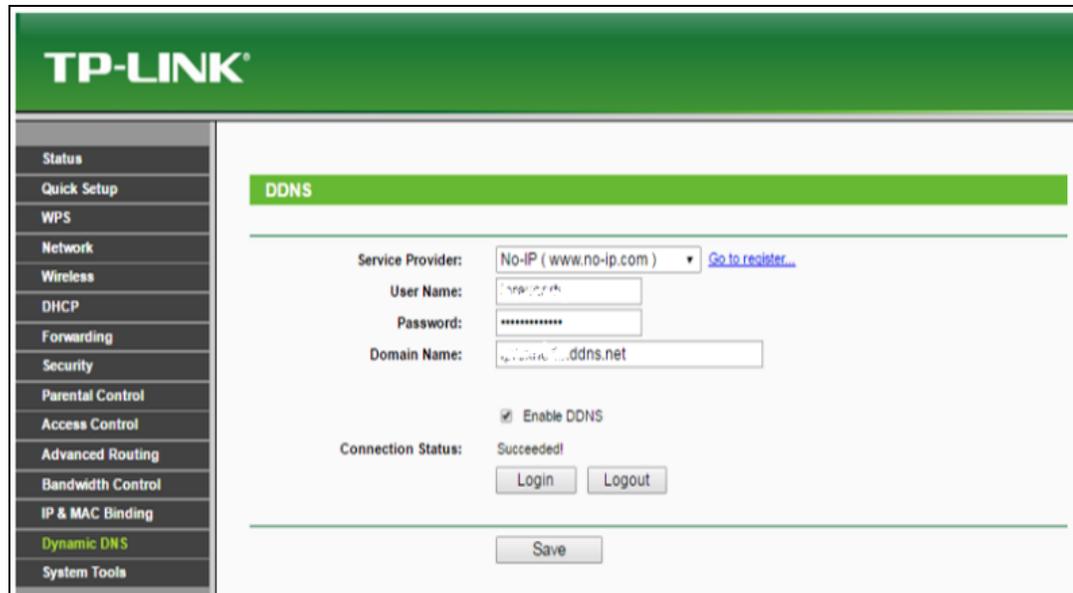


Figura 53-2. Activación de No-IP en el router

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Pues bien, como utilizamos el router Tp-Link, analizamos su forma de conexión, si bien es cierto este solo era una extensión de red, por ello vamos a utilizarlo como un puente, es decir que toda mi información ingresará al router CNT pasa por el Tp-Link y de ahí llega a mi placa y a mi cámara que se encuentra conectada inalámbricamente.

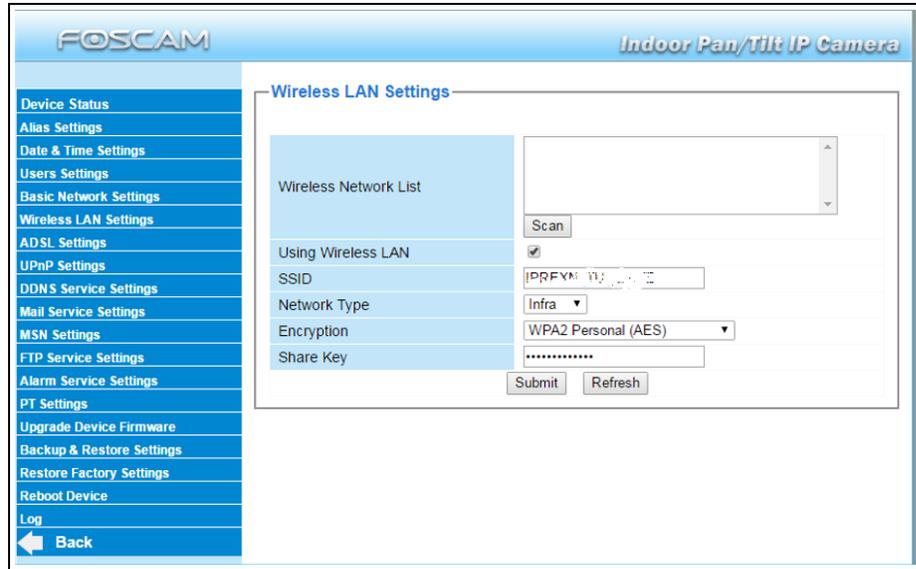


Figura 54-2. Configuración Cámara 2

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Estos dispositivos no pueden estar variando su dirección IP, entonces para ello procedemos a reservarles direcciones en el router que se encuentran conectados, en este caso el router Tp-link en el router CNT y la central con la cámara en el Tp-link.

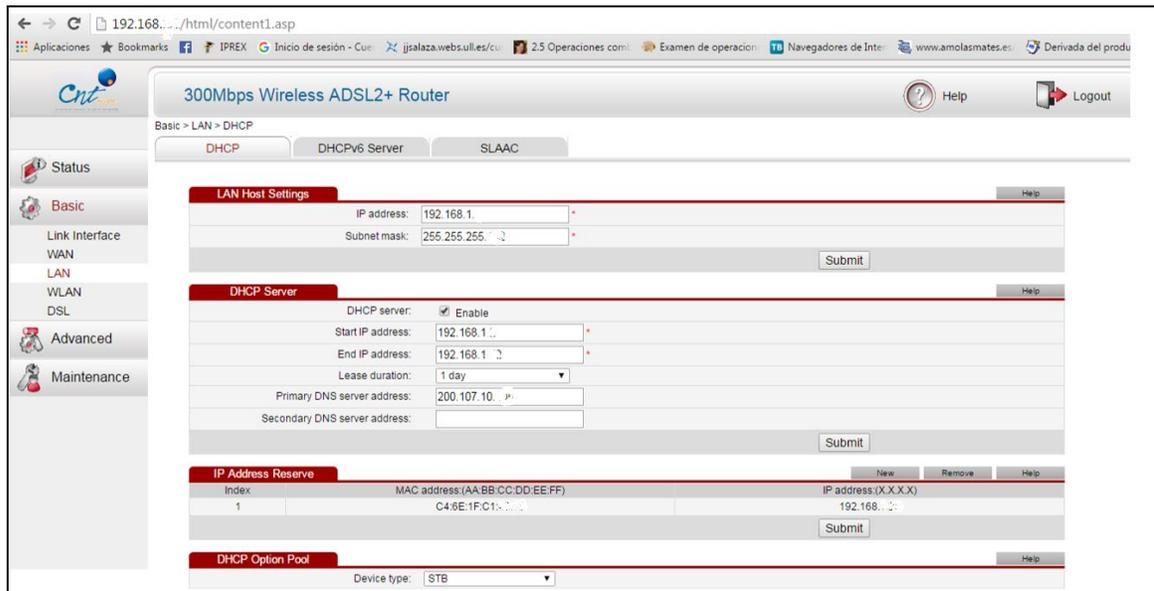


Figura 55-2. Reservación de dirección IP

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Ingresamos a verificar si mis cambios se han guardado.

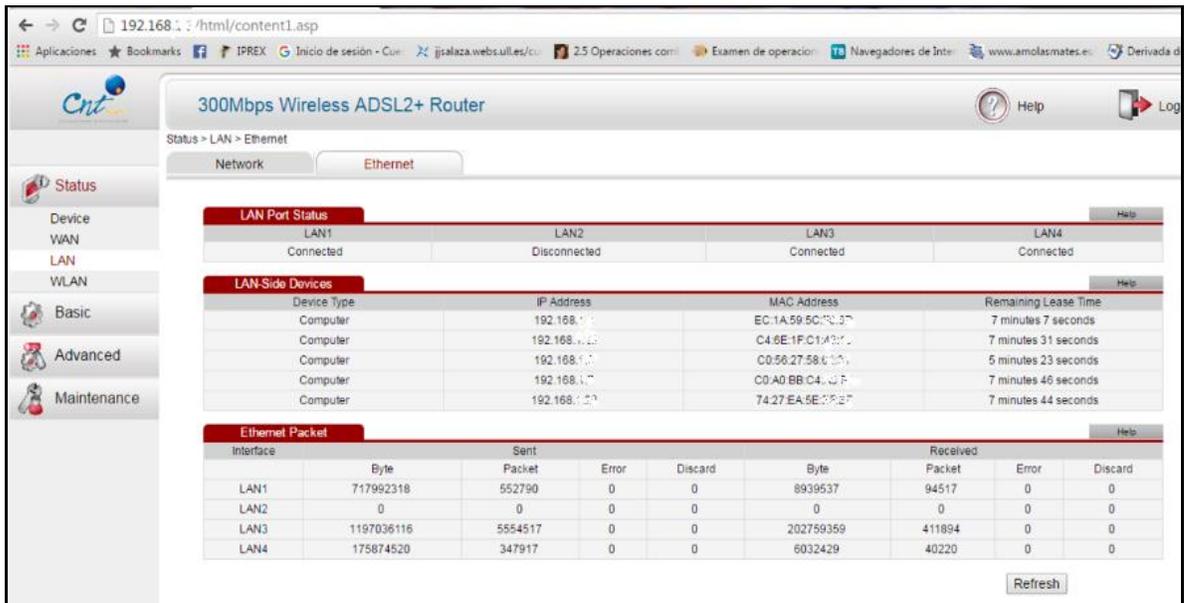


Figura 56-2. Verificación de dirección asignada

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

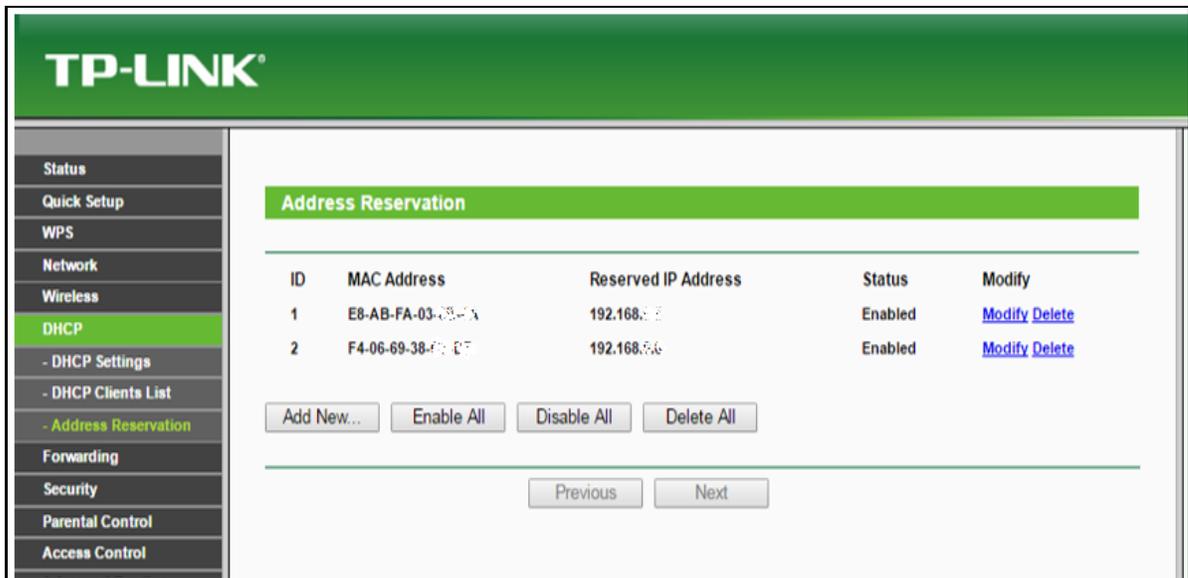


Figura 57-2. Reservación para cámara y placa central

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

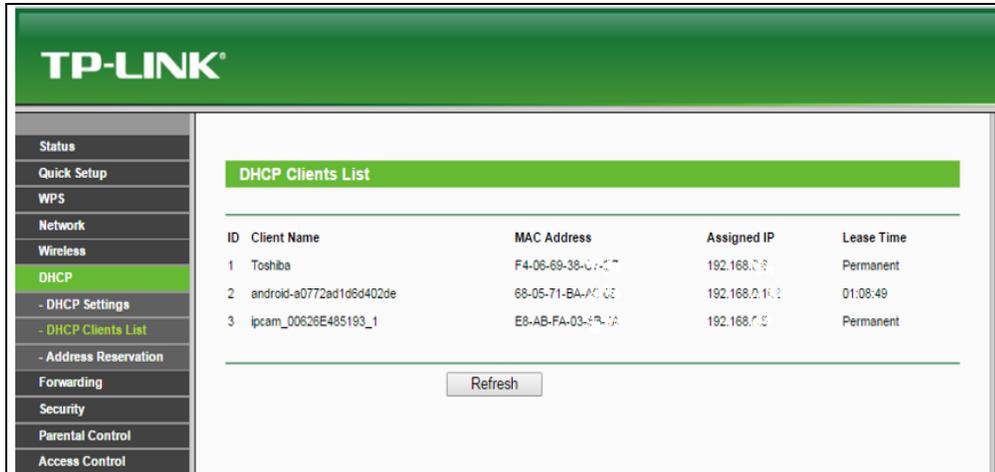


Figura 58-2. Verificación de dirección asignada

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Además verificamos que nuestro router asigne por DHCP otro rango de direcciones IP, evitando así la duplicidad de direccionamiento.

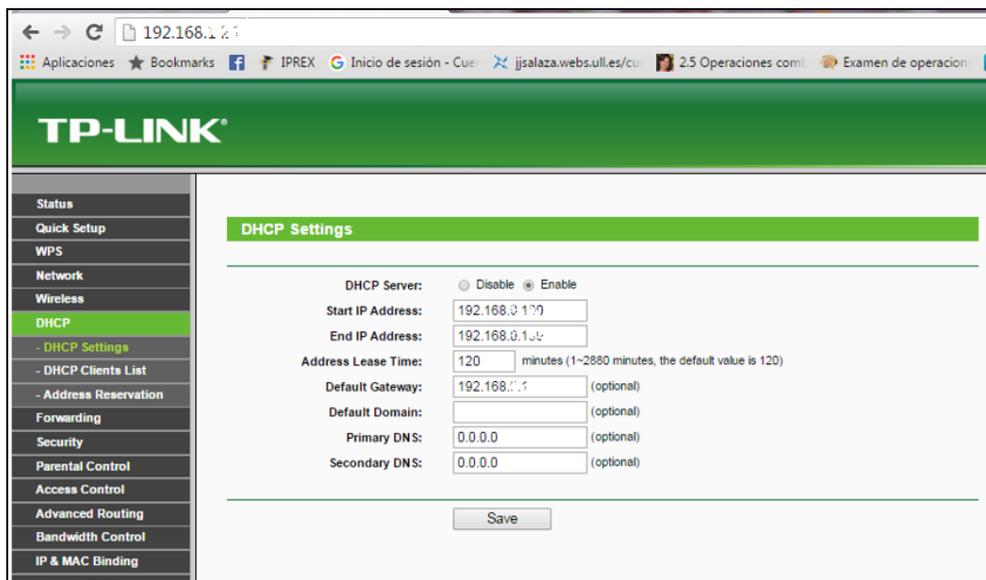


Figura 59-2. Configuración de DHCP router Tp-Link

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Si bien con todas las configuraciones realizadas deseamos ingresar mediante un dominio nosotros debemos abrir los puertos del router, es decir abrirle un camino por donde va a viajar la información.

Iniciamos por nuestro router principal, que vendría a ser el router CNT, recordando que todo debe llegar al router Tp-Link, entonces mis puertos deben abrirse direccionados hacia este router.

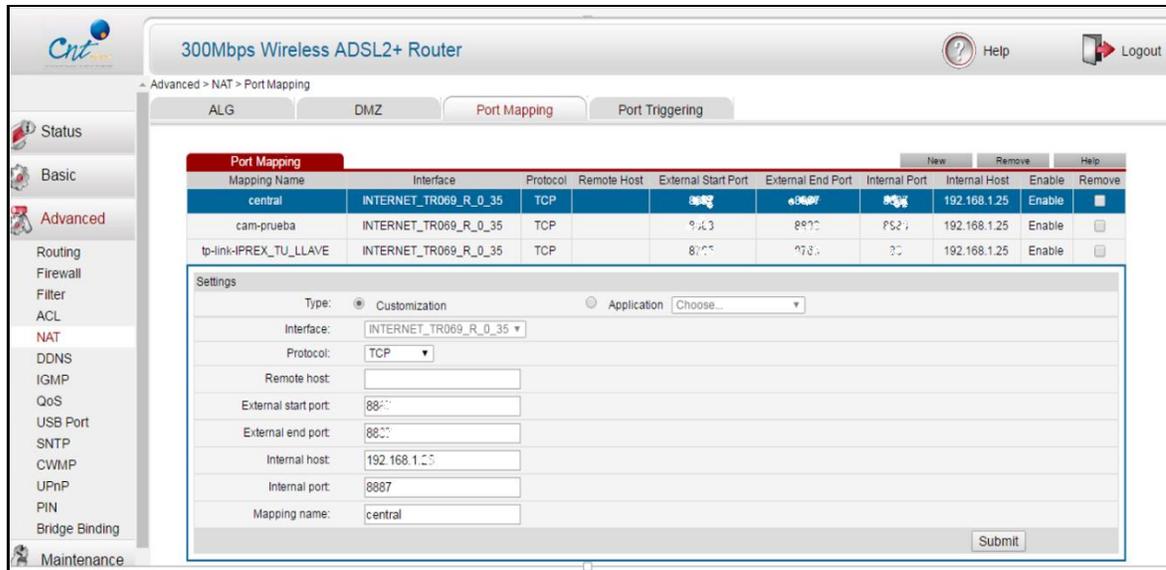


Figura 60-2. Creación de caminos virtuales en router CNT

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016



Figura 61-2. Creación de caminos virtuales router Tp-Link

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

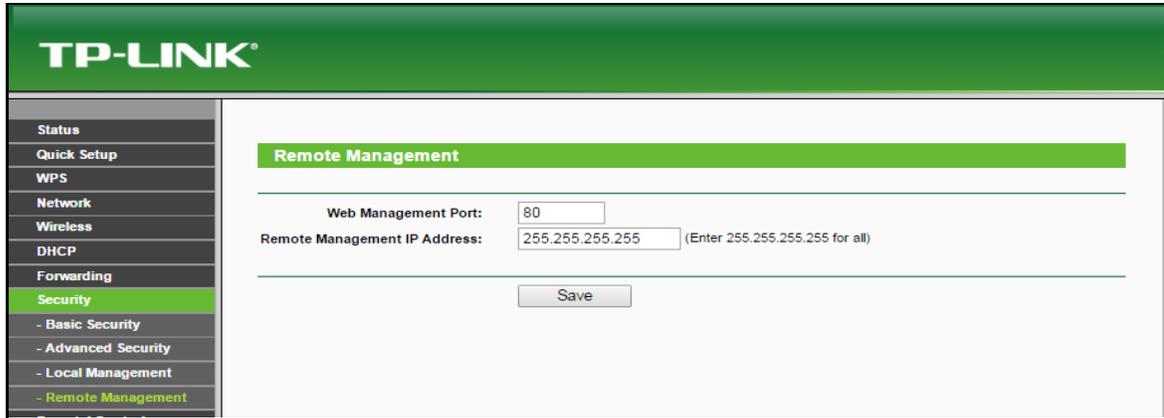


Figura 62-2. Activación de acceso remoto

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

CAPÍTULO III

3. RESULTADOS

3.1 Funcionamiento de App.

Primero para empezar a utilizar la app se debe realizar una instalación la cual se encontrara en el celular, una vez instalador empezar a buscar el acceso directo así como se puede observar en la imagen.



Figura 1-3. Ingreso a la Aplicación

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Una vez hecho click en la app la siguiente pantalla que nos muestra nuestro Smartphone es el paso número 2 donde tendremos que colocar el usuario y contraseña para poder entrar a la manipulación de la aplicación.

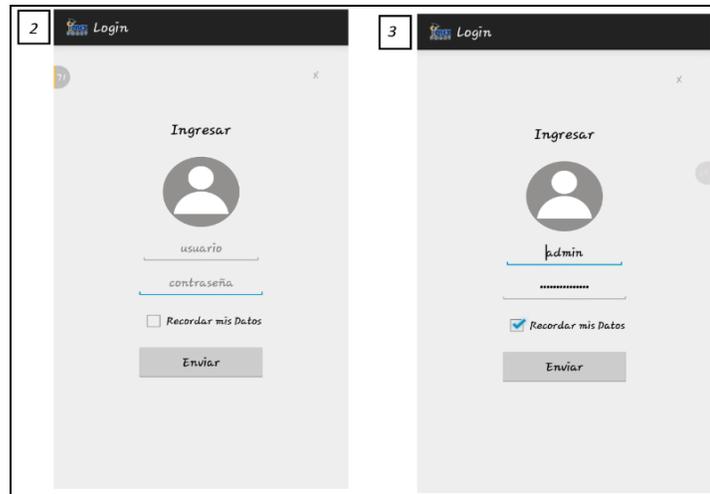


Figura 2-3. Ingreso de usuario y contraseña

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Luego de colocar el usuario y contraseña deseado iremos directamente a la pantalla principal pero antes nos saldrá un mensaje de alerta en cual no pedirá si queremos tener una conexión Local o Externa así como lo muestra la figura 100-3 el paso 4.

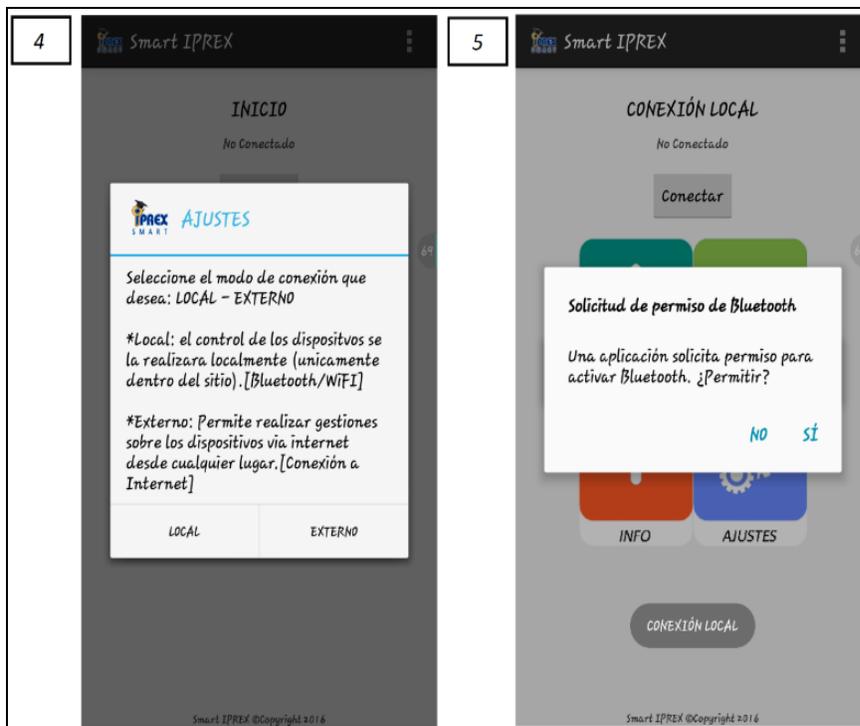


Figura 3-3. Mensajes de emergencia

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Si la opción a elegir es Local y si no tenemos activado por Bluetooth la aplica nos exigirá activarlo para poder proceder a realizar la conexión con la placa Central como lo muestra la figura 100-3 en el paso 5.

Una vez activado el bluetooth debemos dar click en la opción donde dice “Conectar” así como lo muestra el paso 6.



Figura 4-3. Proceso de conexión local

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Una vez dado click en la opción conectar la aplicación busca el dispositivo con el cual está vinculado para poder enlazar una conexión así como lo muestra la imagen 102-3 con el paso 7.

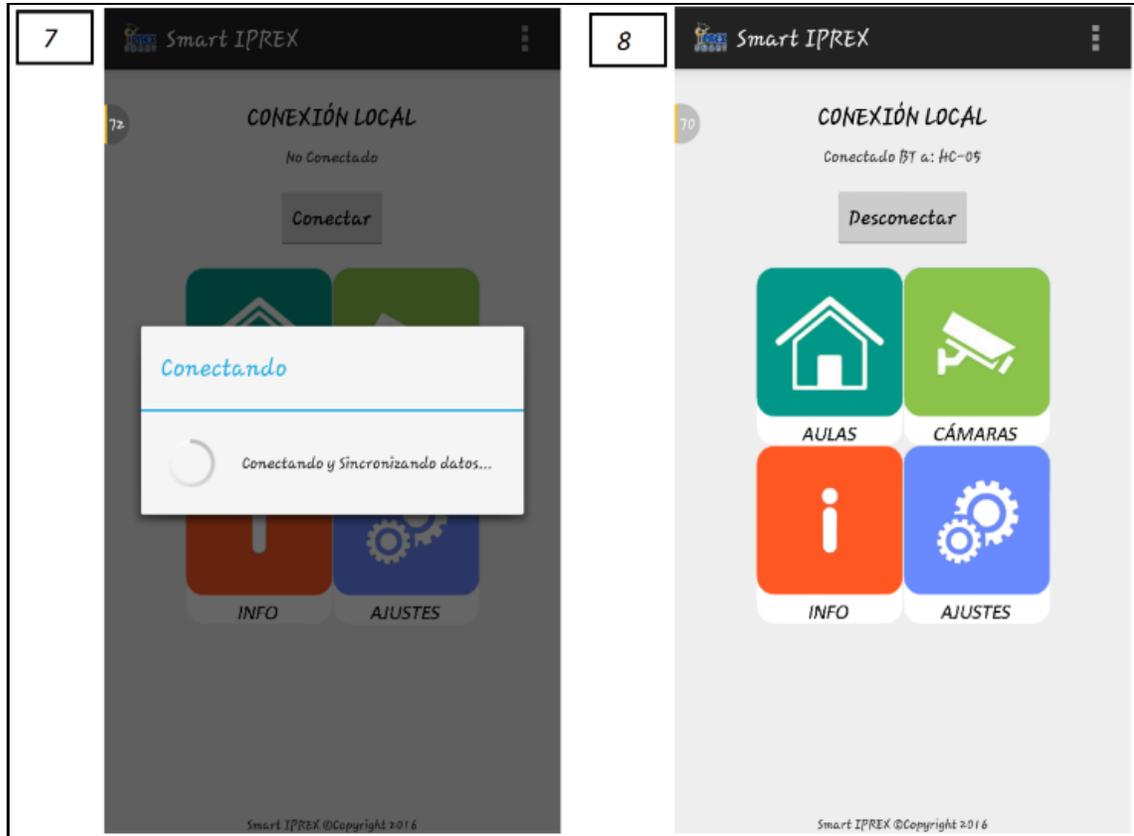


Figura 5-3. Conexión local establecida

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

En el paso 8 nos mostrara que tipo de conexión tenemos y con qué dispositivo está conectado ya que desde ahí podemos ingresar a la pantalla principal y poder escoger la opción deseada.

Cuando escogemos la opción Aula el Menú que se nos desplegara será el que se puede observar en la figura en el paso 9, donde obtendremos el control de toda la parte de iluminaria del centro educativo.

En la siguiente pestaña se encuentra el control de audio, el menú obtenido será como en la imagen 103-3 en el paso 10 donde mostrará un menú en el cual podemos elegir la utilización del micrófono o si deseamos activar o desactivar los parlantes.

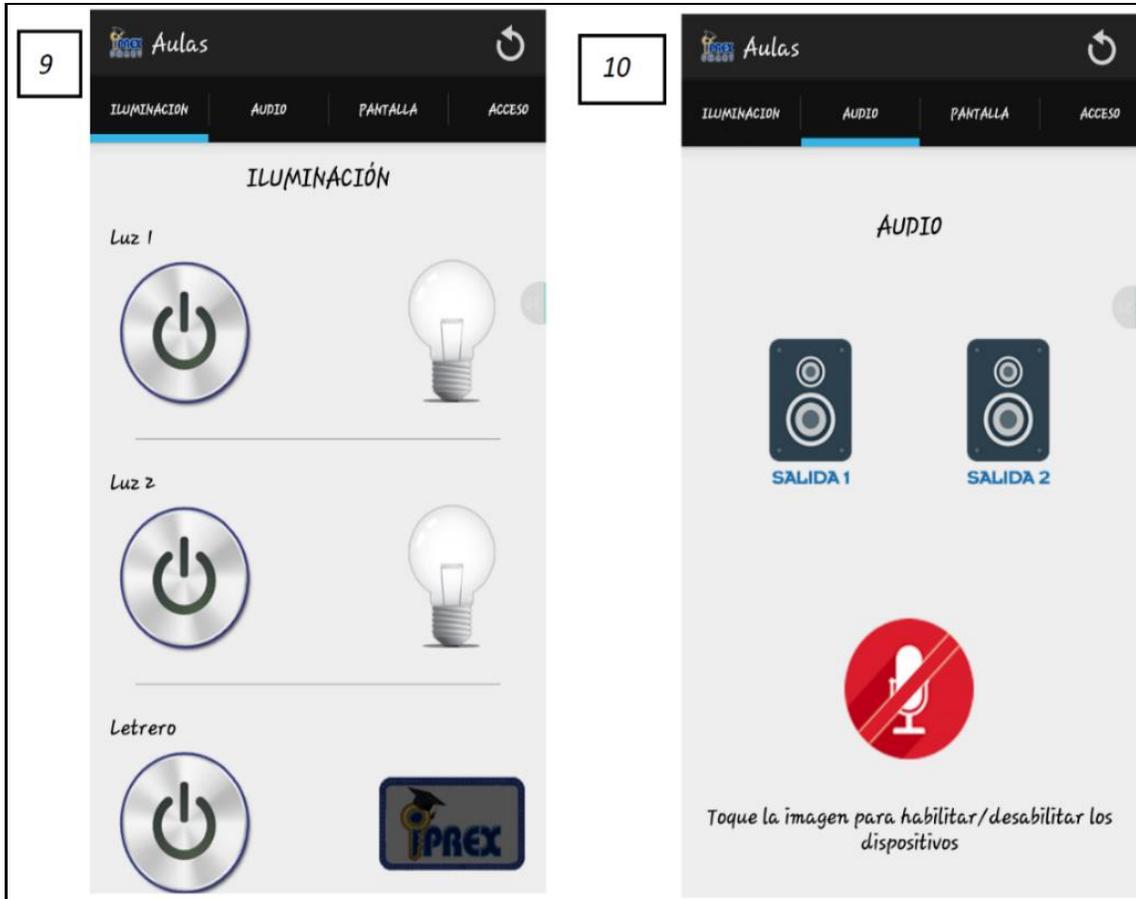


Figura 6-3. Pestañas de la Aplicación

Realizado por: Loayza J, Padilla E. 2016

Para el paso 11 este es el control de la pantalla de forma local con el cual se puede subir o bajar y obtener el control de la pantalla desde la App donde la flecha para arriba tiene 2 funcionalidades que puede ser: escoger la opción de subir, si aplastamos nuevamente puede pausar el proceso, de igual forma puede hacer la flecha para abajo.

Para le paso 12 aquí obtenemos el control de sirenas alarmas, bloqueo y desbloqueo de la puerta así como podemos observar en la figura.

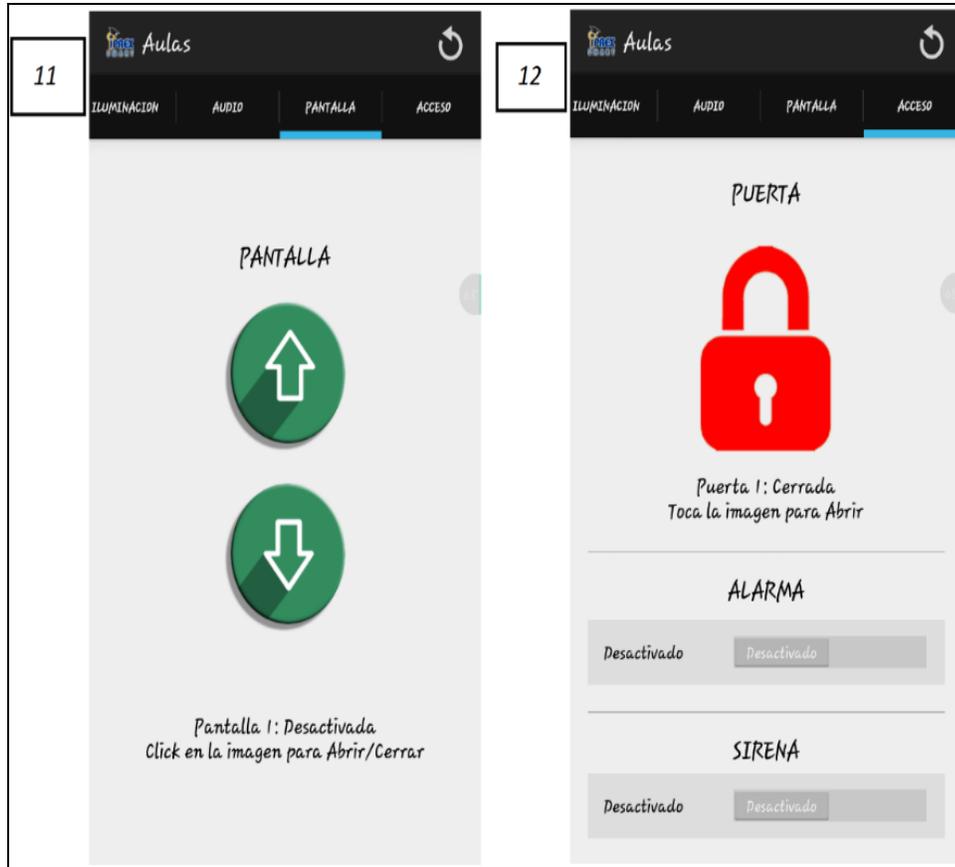


Figura 7-3. Control de acceso y pantalla de proyección

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para la opción de cámaras nos saldrán 2 ventanas cada una con la conexión de una cámara las cuales están ubicadas en la oficina del centro educativo y la segunda está ubicada en el salón de comida.

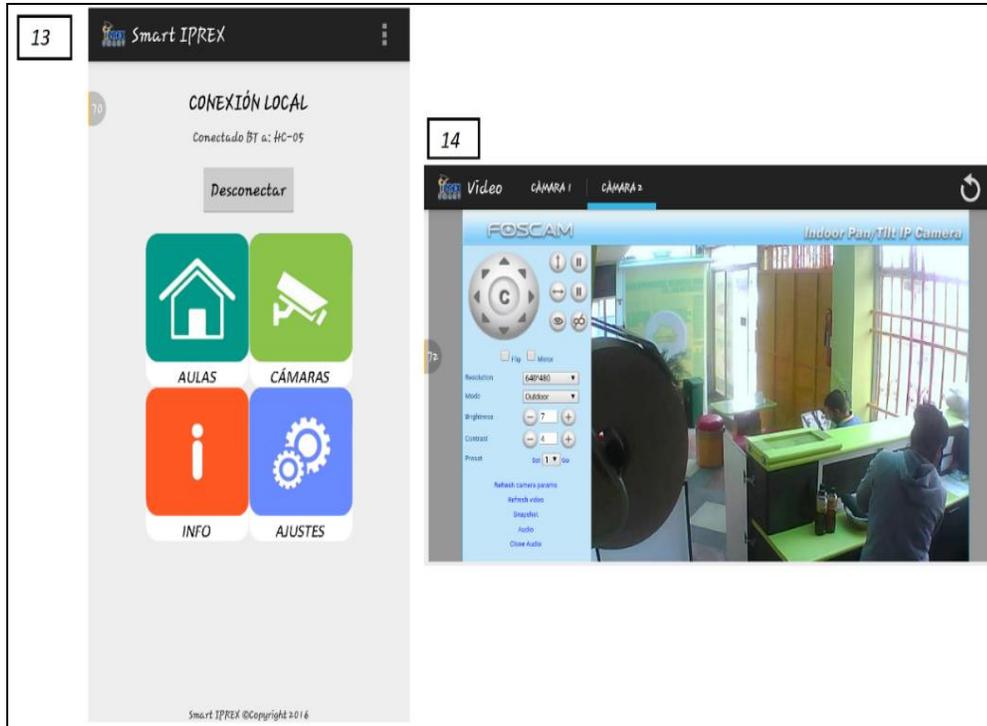


Figura 8-3. Acceso a las cámaras

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Para cambiar a conexión externa solo se debe presionar el botón ajustes como lo indica el paso 15 permitiéndonos cambiar de conexión local a conexión externa con un mensaje de emergencia confirmando que tipo de conexión desea, así como lo muestra el paso 16 en la figura 106-3.

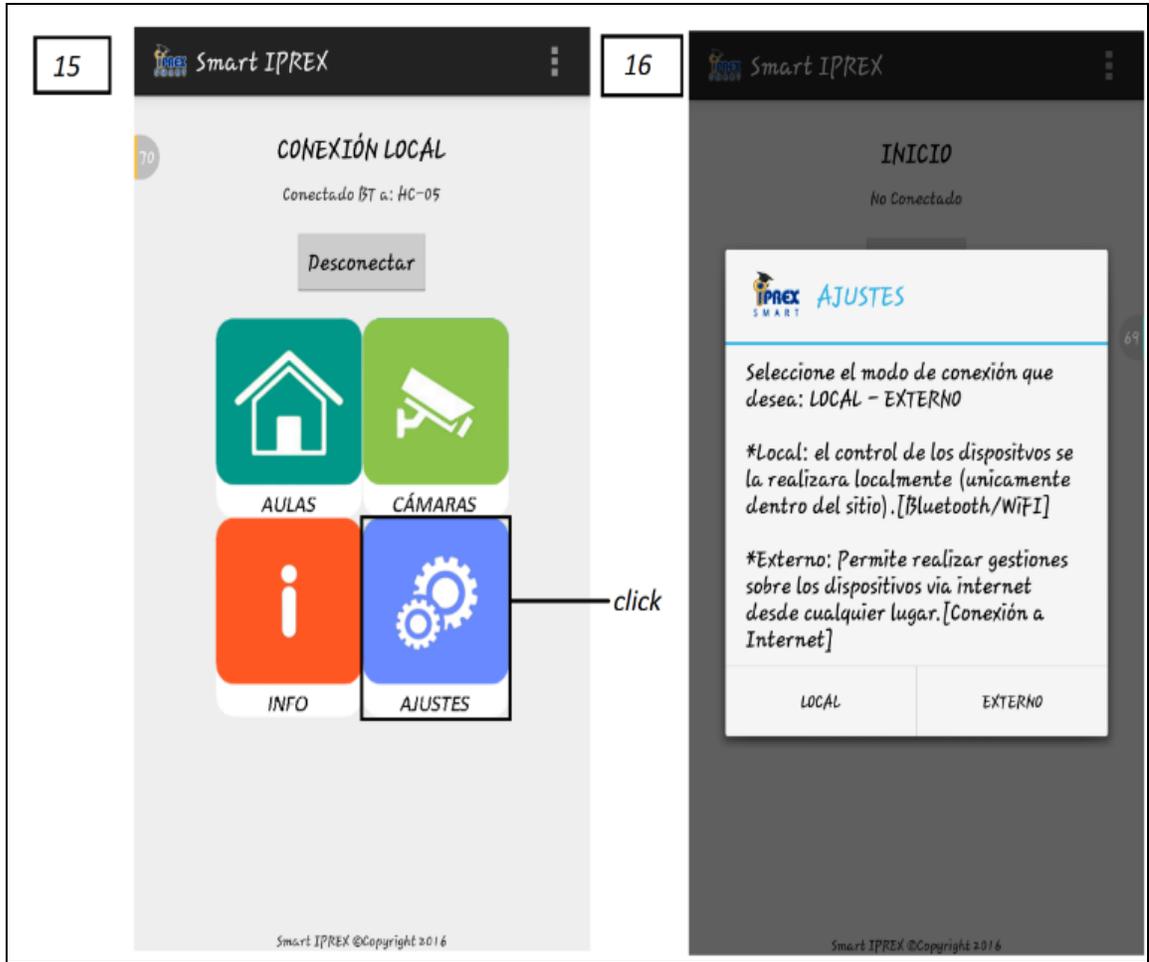


Figura 9-3. Cambio de conexión

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

La única forma de saber si hemos ingresado a en la parte de arriba que nos indica el tipo de conexión ya que el menú es el mismo, así como lo indica el paso 17.

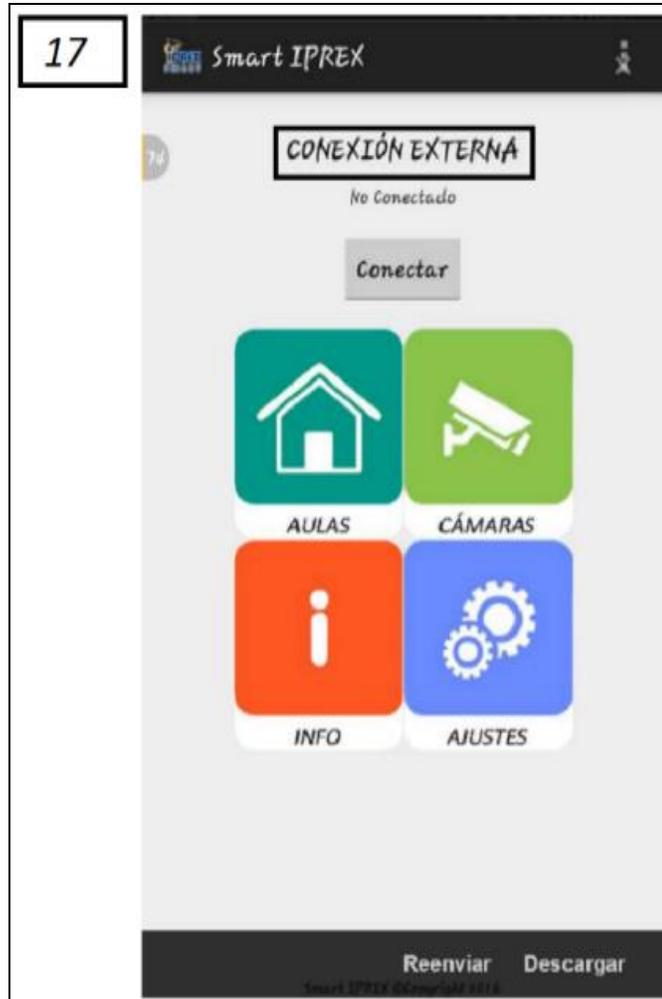


Figura 10-3. Conexión Externa

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

Ingresamos en la Opción de Aula donde el menú si cambiara en donde saldrá como se muestra en la figura del paso 18 solo para control de dispositivos de iluminaria



Figura 11-3. Página web de conexión externa

Fuente: Realizado por: Loayza J, Padilla E. 2016

3.2 Funcionamiento de la central

La central se encuentra dividida por etapas como se muestra en la figura.

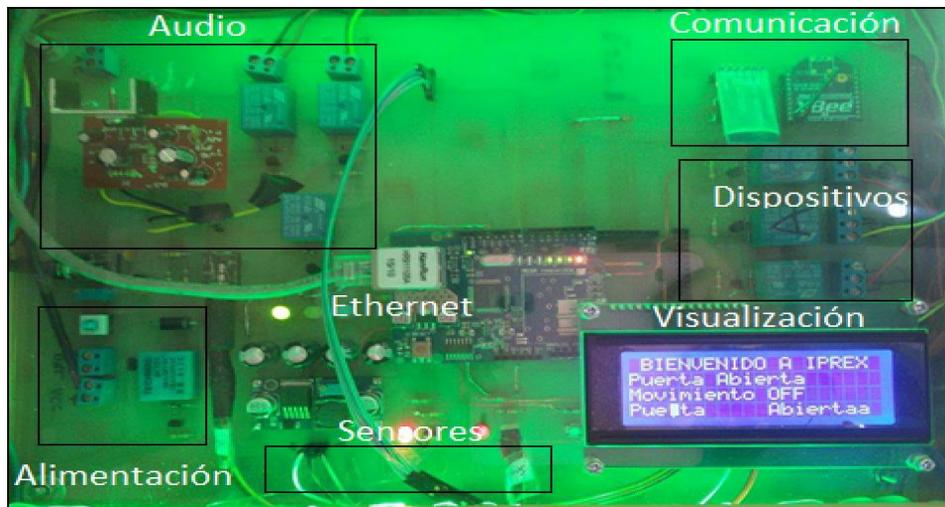


Figura 12-3. Placa Central

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

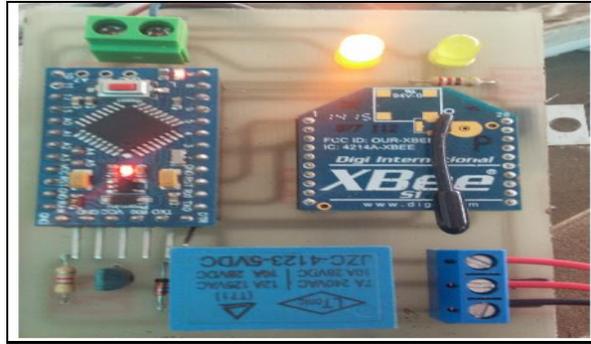


Figura 13-3. Placa del receptor Xbee

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

3.3 COSTOS

Tabla 1-3. Costos del sistema

Cantidad	Equipos a Utilizar	745,00	
1	Arduino Mega	60,00	60,00
1	Shield Ethernet	30,00	30,00
2	Placas Xbee	40,00	80,00
1	Pantalla	25,00	25,00
	Componentes electrónicos	100,00	100,00
1	Bluetooth	20,00	20,00
1	Smartphone	200,00	200,00
1	Cámara IP	100,00	100,00
1	Sirena	30,00	30,00
	Materiales de Instalación	100,00	100,00
	Software de programación	0,00	
	Otros		200,00
	TOTAL		945,00

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

3.4 Comparación con otras empresas

Tabla 2-3. Comparación de servicios

SERVICIOS	LOCAL	REMOTO	COSTO
PROPIO	X	X	1500
IT_CONTROL	X		2500
ELECTRONIC_HOME	X		4500

Realizado por: LOAYZA, Jhon, PADILLA, Erica. 2016

CONSIDERACIONES

- Antes de empezar a utilizar la app es recomendable emparejar previamente el dispositivo con el Bluetooth para mantener comunicación con la placa central.
- Al momento de empezar a realizar la manipulación de la app de forma local se debe considerar tener línea de vista o conexión con el bluetooth para poder manipular los dispositivos conectados a la placa central.
- Para poder obtener una protección de los equipos se debe colocar un ups a la placa central con el fin de proteger a la placa cuando haya caídas y subidas de voltaje.
- Para realizar modificaciones en el programa Arduino se debe tener conocimiento de cómo funciona el sistema para no dejar imposibilitado el mismo.
- Al momento de encender la placa central se debe observar que el switch esté cerrado para que el bluetooth de la placa central pueda tener comunicación con la app.
- Para poder ingresar un nuevo usuario a la app se debe realizar desde la programación de Eclipse.
- Para realizar el control del desbloqueo y bloqueo de la puerta se debe considerar que la puerta esté bien cerrada.
- La comunicación inalámbrica de un dispositivo a la central implementada en un lugar reducido puede convertirse en algo costoso ya que para cada dispositivo a controlar se requerirá de un xbee, es decir mientras más cosas queremos manipular tendrá un costo más elevado por ello en lugares reducidos lo mejor es una comunicación mixta.

CONCLUSIONES

- Para obtener acceso remoto o externo hacia un dispositivo no siempre se debe realizar un servidor web o comprar una ip pública, otra solución es realizar caminos virtuales mediante la manipulación de puertos del router servidor de internet.
- Existen diferentes maneras de realizar el control domótico pero para determinar el dispositivo más óptimo se analiza el funcionamiento destinado y la posible escalabilidad, evitando desperdicios de recursos y saturación de los mismos.
- Se desarrolló una app personalizada con usuarios predeterminados para reducir el costo del proyecto para el acceso al control domótico, para dispositivos móviles con sistema operativo Android, con una mejor confiabilidad
- Para realizar un proyecto domótico donde se vaya a manipular audio se debe optar por un dispositivo el cual me permitiera mayor velocidad de procesamiento de datos para que a medida que realice el proyecto no haya problemas con latencia de datos.
- La comunicación inalámbrica de un dispositivo a la central implementada en un lugar reducido y con una infraestructura compleja evita el cableado.

RECOMENDACIONES

- Al hallarnos junto a un local de preparación de comida, se recomienda un sistema de detección de incendios.
- Teniendo tantas cosas de valor se recomienda ampliar el manejo de cámaras a un sistema de video vigilancia.
- Para poder obtener un mejor disponibilidad de la App. Es recomendable realizarlo en un software hibrido lo cual me permitirá tener un App. para cualquier tipo de Smartphone
- Para el crecimiento del centro educativo se puede añadir o crear un sistema de alarmas de circuito cerrado.

BIBLIOGRAFÍA

1. **ARANAZ, J.** Desarrollo de aplicaciones para dispositivos móviles sobre la plataforma Android de google. [En línea] (**TESIS PREGRADO**). Universidad Carlos III de Madrid. Ingeniería en Informática. Madrid – España. 2009. pp.37-42.
[Consulta: 20 de enero del 2016.] Disponible en: http://e-archivo.uc3m.es/bitstream/handle/10016/6506/PFC_Jaime_Aranaz_Tudela_2010_116132629.pdf
2. **ACOSTA, C; SORIA, A.** Implementación de una consola domótica residencial con control local y remoto a través del internet. [En línea] (**TESIS DOCTORAL**). Escuela Politécnica Nacional. Ingeniería en Electrónica y Control. Quito – Ecuador. 2015. pp.1-8.
[Consulta: 15 de octubre del 2015.] Disponible en: <http://bibdigital.epn.edu.ec/bitstream/15000/10889/1/CD-6330.pdf>
3. **CHIPUXI, R.** Diseño e implementación de un sistema domótico remoto vía GSM para el Hotel San Miguel. [En línea] (**TESIS PREGRADO**). Universidad Politécnica Salesiana Sede Quito. Ingeniería Electrónica. Quito – Ecuador. 2015. pp.5-9.
[Consulta: 20 de noviembre del 2015.] Disponible en: <http://www.dspace.ups.edu.ec/bitstream/123456789/10180/1/UPS%20%20ST001827.pdf>
4. **GAGO, J.** VISOR DE INFRARROJOS DE 360° PARA ROBOT BASADO EN ARDUINO. [En línea] (**TESIS PREGRADO**). Universidad Pública de Navarra. E.T.S. de Ingeniería Industrial, Informática y de Telecomunicación. Navarra – España. 2014. pp.28-32.
[Consulta: 24 de mayo del 2014.] Disponible en: <http://academica-e.unavarra.es/TFGGagoBaldaJavier2014.pdf?sequence=1.pdf>
5. **HERRAIZ, G.** ANDROID. [En línea] (**TESIS PREGRADO**). Universidad Politécnica de Valencia. Ingeniería en telecomunicaciones. Valencia – España. 2012. pp.3-7.

[Consulta: 10 de febrero del 2016.] Disponible en:
<http://histinf.blogs.upv.es/files/2012/12/ANDROID-Gabriel-Herraiz-Ant%C3%B3n.pdf>

6. **JIMÉNEZ, J.** Eclipse IDE. [En línea]. Valladolid – España. 2014.
[Consulta: 23 de abril del 2016.] Disponible en: <http://www.genbetadev.com/herramientas/eclipse-ide>

7. **LEIVA, A.** Configuración del entorno de desarrollo. [En línea]. Madrid – España. 2012.
[Consulta: 10 de febrero del 2016.] Disponible en:
<http://www.limecreativelabs.com/configuracion-del-entorno-de-desarrollo/>

8. **LEÓN, J.** 5 Lenguajes para Programar en Android. [En línea]. Bogotá – Colombia. 2012.
[Consulta: 10 de febrero del 2016.] Disponible en: <http://androideity.com/2012/07/16/5-lenguajes-para-programar-en-android/>

9. **OYARCE, A.** Guía del usuario Xbee series 1. [En línea]. Santiago – Chile: Ingeniería MCI Ltda. 2010. pp.7-8-12. [Consulta: 12 de abril del 2016.] Disponible en:
<http://s3.amazonaws.com/academia.edu.documents/43436815/XBee-pdf>

10. **PAULLÁN, C; VILLA, A.** Diseño e Implementación de un Sistema Domótico Utilizando Tarjetas de Adquisición de Datos y Labview. Caso Práctico: Prototipo-Conjunto Habitacional. [En línea] (**TESIS PREGRADO**). Escuela Superior Politécnica de Chimborazo. Ingeniería en Electrónica y Computación. Riobamba – Ecuador. 2010. pp.26-27-29.
[Consulta: 02 de marzo del 2016.] Disponible en:
<http://dspace.esPOCH.edu.ec/bitstream/123456789/385/1/38T00194.pdf>

11. **RODRÍGUEZ, F.** Basic4Android: Desarrollando aplicaciones para Android al estilo Visual Basic. [En línea]. Quito – Ecuador. 2011. [Consulta: 23 de abril del 2016.] Disponible en:
<http://developeando.net/basic4android/>

12. **RUIZ, P.** Integración de un sistema de detección de focos de incendio en un robot para buques de la Armada. [En línea] (**TESIS PREGRADO**). Universidad de Vigo. Centro Universitario de la Defensa, ENM. Ingeniería Mecánica. Vigo – España. 2015. pp.21,23.
[Consulta: 25 de marzo del 2016.] Disponible en:
<http://calderon.cud.uvigo.es:8080/bitstream/11621/54/1/Memoria%20final-Ru%C3%ADz%20Villalta.pdf>

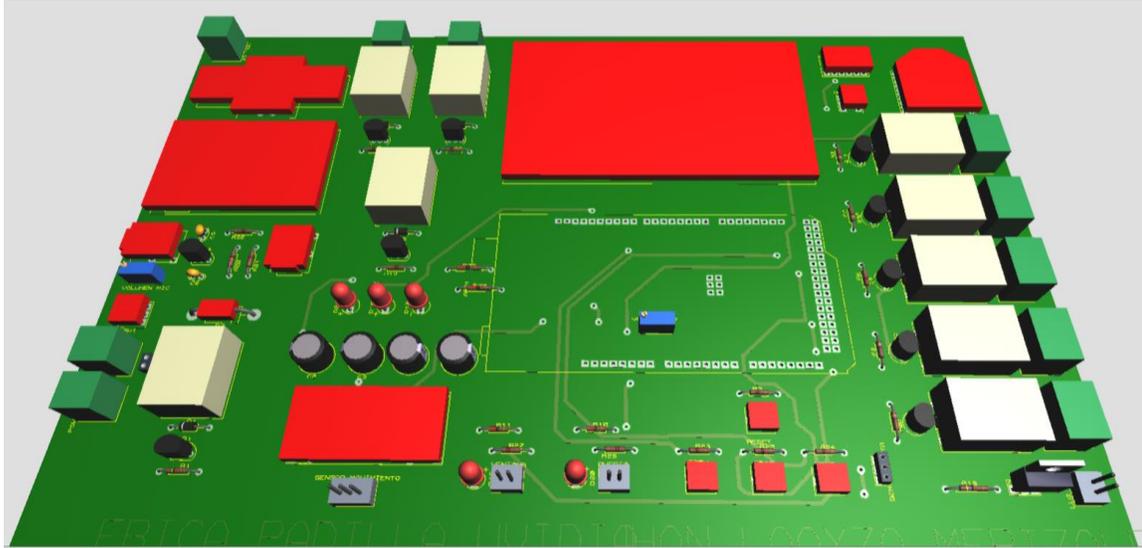
13. **TENESACA, C.** Sistema informático para la atención de un restaurante. [En línea] (**TESIS PREGRADO**). Universidad del Azuay. Ingeniería en Sistemas. Cuenca – Ecuador. 2015. p.1. [Consulta: 25 de enero del 2016.] Disponible en:
<http://dspace.uazuay.edu.ec/bitstream/datos/4541/1/11028.pdf>

14. **TIXICURO, M.** Estudio comparativo entre plataformas ANDROID e IOS para el desarrollo de aplicaciones móviles inteligentes para el centro de capacitación y asesoramiento informático CCAINFOR-Ibarra. [En línea] (**TESIS PREGRADO**). Escuela Politécnica Nacional. Ingeniería en Electrónica y Telecomunicaciones. Ambato – Ecuador. 2015. pp.15-16.
[Consulta: 18 de febrero del 2016] Disponible en:
<http://dspace.uniandes.edu.ec/bitstream/773/1/TUAEXCOMSIS004-2015.pdf>

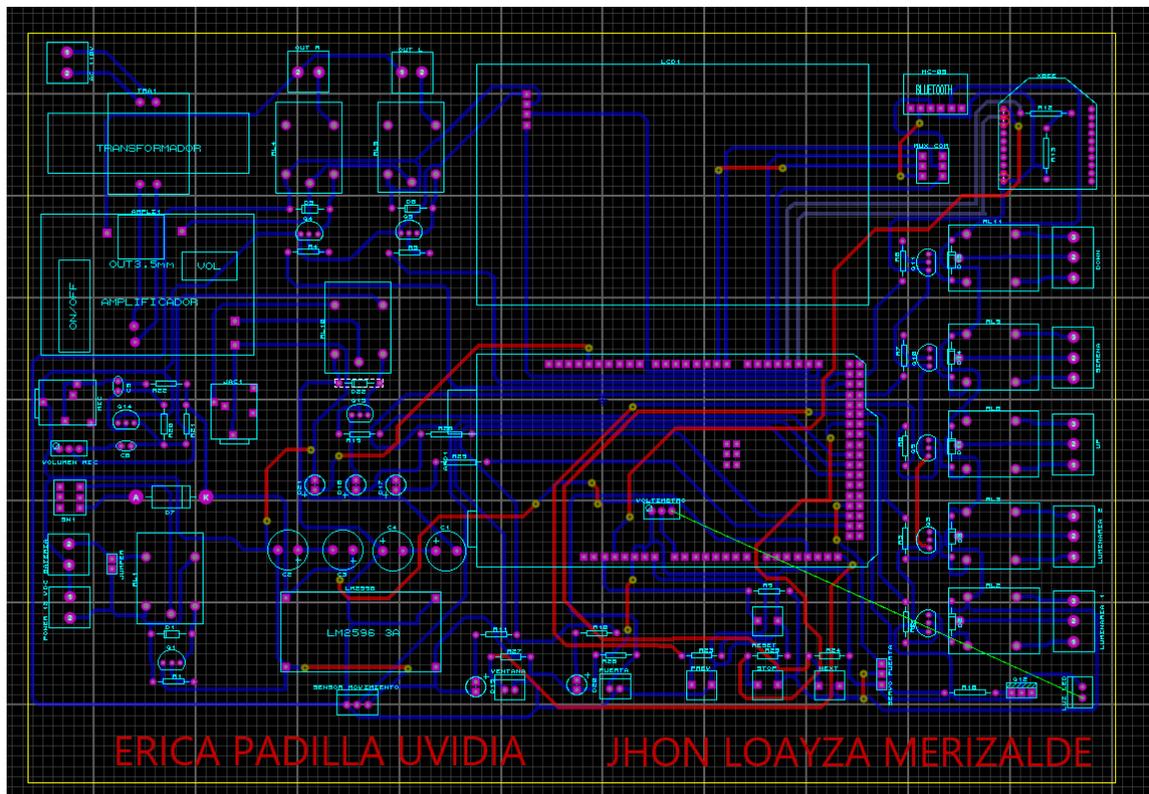
15. **TOBAR, J; SANDOVAL, L.** Implementación de un sistema domótico, monitoreado a través de una aplicación para plataformas Android, utilizando la plataforma Arduino. [En línea] (**TESIS PREGRADO**). Universidad Autónoma de los Andes. Ingeniería en Sistemas e Informática. Quito – Ecuador. 2015. pp.6-21.
[Consulta: 18 de febrero del 2016] Disponible en:
<http://bibdigital.epn.edu.ec/handle/15000/11891>

ANEXOS

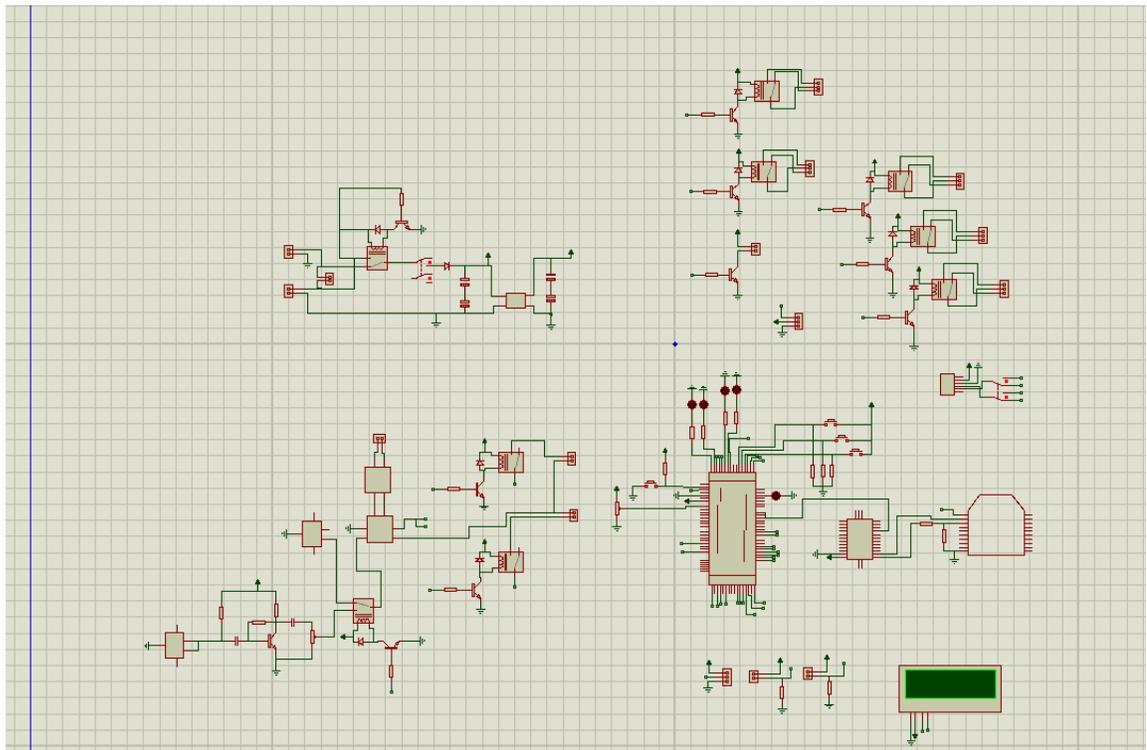
ANEXO A Imagen 3D de la placa central



ANEXO B Pistas de la placa central



ANEXO C Diagramas de simulación de etapas de la placa central



ANEXO D Diseño de Centro Educativo IPREX

