



UNIVERSIDAD NACIONAL DE LOJA

FACULTAD DE LA ENERGÍA, LAS INDUSTRIAS Y LOS RECURSOS NATURALES NO RENOVABLES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

“Diseño e implementación de un sistema integral de seguridad utilizando software libre, basado en el estándar IEEE 802.15.4 para la empresa de tecnología y servicios TECSERLED de la ciudad de Loja.”

TESIS DE GRADO PREVIA A LA
OBTENCIÓN DEL TÍTULO DE
INGENIERO EN ELECTRÓNICA Y
TELECOMUNICACIONES.

AUTOR:

Jonas Eliseo Carrillo Sisalima

DIRECTOR:

Ing. Christian Hernan Campoverde Ramírez, Mg. Sc



LOJA-ECUADOR

2019

CERTIFICACIÓN

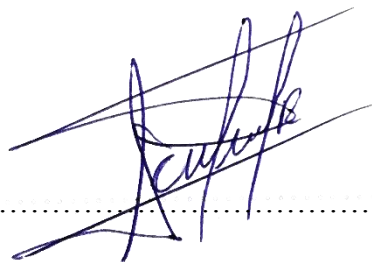
Ing. Christian Hernan Campoverde Ramírez, Mg Sc

DIRECTOR DEL TRABAJO DE TESIS

CERTIFICA:

Haber dirigido, asesorado, revisado y corregido el presente trabajo de tesis de grado, en su proceso de investigación cuyo tema versa; **“DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA INTEGRAL DE SEGURIDAD UTILIZANDO SOFTWARE LIBRE, BASADO EN EL ESTÁNDAR IEEE 802.15.4 PARA LA EMPRESA DE TECNOLOGÍA Y SERVICIOS TECSERLED DE LA CIUDAD DE LOJA”**, previa a la obtención del título de **Ingeniero en Electrónica y Telecomunicaciones**, realizado por el señor egresado: **Jonas Eliseo Carrillo Sisalima**, el mismo que cumple con la reglamentación y políticas de investigación, por lo que autorizo su presentación y posterior sustentación y defensa.

Loja, 17 de Julio de 2019



Ing. Christian Hernan Campoverde Ramírez, Mg Sc.

DIRECTOR DEL TRABAJO DE TESIS

AUTORÍA

Yo, **JONAS ELISEO CARRILLO SISALIMA**, declaro ser autor del presente trabajo de tesis y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos o acciones legales por el contenido de la misma.

Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi trabajo de tesis en el Repositorio Institucional- Biblioteca Virtual.

Firma:



Cédula: 1105362063

Fecha: 05/09/2019


CARTA DE AUTORIZACIÓN DE TESIS POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO.

Yo, **JONAS ELISEO CARRILLO SISALIMA**, declaro ser autor de la tesis titulada: **“DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA INTEGRAL DE SEGURIDAD UTILIZANDO SOFTWARE LIBRE, BASADO EN EL ESTÁNDAR IEEE 802.15.4 PARA LA EMPRESA DE TECNOLOGÍA Y SERVICIOS TECSERLED DE LA CIUDAD DE LOJA”**, como requisito para optar al grado de: **INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES**; autorizo al Sistema Bibliotecario de la Universidad Nacional de Loja para que con fines académicos, muestre al mundo la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Digital Institucional:

Los usuarios pueden consultar el contenido de este trabajo en el RDI, en las redes de información del país y del exterior, con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia de la tesis que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja, a los cinco días del mes de Septiembre del dos mil diecinueve.

Firma: 

Autor: Jonas Eliseo Carrillo Sisalima

Cédula: 1105362063

Dirección: Loja, (Av. Pío Jaramillo Alvarado y Kennedy)

Correo electrónico: jonascsl692@gmail.com – jonas.carrillo@unl.edu.ec

Teléfono:

Celular: 0982730064

DATOS COMPLEMENTARIOS

Director de tesis: Ing. Christian Hernan Campoverde Ramírez, Mg. Sc.

Tribunal de grado: Ing. John Jossimar Tucker Yépez, Mg. Sc.

Ing. Luis Eduardo Rodríguez Montoya, Mg. Sc.

Ing. Santiago Abraham Medina León, Mg. Sc.

DEDICATORIA

A Dios y a mis padres por la vida obsequiada, y de manera muy especial a mi madre y hermanos por su apoyo incondicional que han sabido brindarme para ser mejor cada día.

A mi padre, por su esfuerzo y el apoyo económico brindado en el transcurso de mi carrera universitaria.

A la memoria de mi hermano Segundo Benjamín.

A mis seres queridos

Familia, amigos y personas especiales en mi vida a quienes con mucha gratitud llevo siempre en mi corazón. Agradecimientos sinceros por acompañarme en el transcurso de este largo camino y ayudarme en este importante paso de mi vida.

Es gracias a ustedes que es posible el presente trabajo, y a quienes dedico mi tesis.

De todo corazón...

Jonas E. Carrillo Sisalima.

AGRADECIMIENTO

Expreso mis más sinceros y profundos agradecimientos a mi madre, por brindarme su apoyo incondicional y enseñarme el valor del esfuerzo y la perseverancia, al igual que lo hago con cada uno de mis hermanos, por alentarme a superar mis expectativas.

Dejo constancia de mi agradecimiento a la Facultad de la Energía, las Industrias y los Recursos Naturales no Renovables y a la Universidad, por la formación que me han dado. Mis más sinceros agradecimientos al Ing. Mg. Sc. Christian Hernan Campoverde Ramírez, director de la presente tesis, su discusión, ideas y comentarios en este proceso han sido absolutamente invaluable para mí.

Agradezco profundamente al Ing. Leonardo Camacho Ruilova, director y desarrollador de CORPORACIÓN TECSERLED por permitirme desarrollar el proyecto de titulación en los espacios de trabajo de su empresa, por sus aportes, consejos y el compromiso demostrado que hoy se refleja en la culminación exitosa del presente trabajo de titulación.

A mis compañeros de clases quienes se han convertido en mi segunda familia. Gracias por sus aportes y por estar presentes en el desarrollo de esta tesis, espero que estos lazos de amistad y de hermandad perduren en el tiempo, permitiéndonos compartir nuevas experiencias en nuestras vidas profesionales.

TABLA DE CONTENIDOS

CERTIFICACIÓN	II
AUTORÍA	III
CARTA DE AUTORIZACIÓN	IV
DEDICATORIA.....	V
AGRADECIMIENTO.....	VI
TABLA DE CONTENIDOS	VII
ÍNDICE DE FIGURAS	XI
ÍNDICE DE TABLAS	XVI
1. TÍTULO.....	1
2. RESUMEN.....	2
ABSTRACT	3
3. INTRODUCCIÓN	4
4. REVISIÓN DE LITERATURA.....	6
4.1. GENERALIDADES.....	6
4.2. SISTEMAS DE SEGURIDAD	8
4.2.1. El Hardware.....	10
4.2.1.1. Sensores	10
4.2.1.2. Actuadores	11
4.2.1.3. Controlador	11
4.2.2. El Software	11
4.3. CLASIFICACIÓN DE LOS SISTEMAS DE SEGURIDAD ELECTRÓNICA	12
4.3.1. Sistemas de seguridad locales y distribuidos	12
4.3.2. Sistemas de seguridad según su aplicación	13
4.3.3. Sistemas de seguridad según el tipo de tecnología que implementan	14
4.3.3.1. Sistemas de seguridad cableados	14
4.3.3.2. Sistemas de seguridad inalámbricos	15
4.4. SMART CITIES.....	18
4.4.1. Seguridad de las ciudades inteligentes	18
4.4.2. Panorama de riesgos de una Smart Citie	20
4.4.2.1. Hardware Inseguro.....	21
4.4.2.2. Mayor superficie de ataque	21
4.4.2.3. Consumo de ancho de banda.....	21
4.5. REDES DE SENSORES INALÁMBRICOS	22
4.5.1. Elementos de la WSN	23

4.5.1.1.	Nodo sensor o dispositivo final.....	23
4.5.1.2.	Nodo Gateway.....	23
4.5.1.3.	Estación Base.....	23
4.5.1.4.	Red Inalámbrica.....	23
4.5.2.	Características de la WSN.....	24
4.5.2.1.	Bajo costo.....	24
4.5.2.2.	Eficiencia energética	24
4.5.2.3.	Potencia computacional.....	24
4.5.2.4.	Capacidades de comunicación	24
4.5.2.5.	Seguridad y privacidad.....	24
4.5.2.6.	Topología de red dinámica.....	25
4.5.2.7.	Comunicación multi-salto	25
4.5.2.8.	Operaciones robustas	25
4.5.2.9.	Tamaño físico reducido	25
4.5.3.	Clases de redes inalámbricas de corto alcance.....	25
4.6.	LENGUAJE DE PROGRAMACIÓN PYTHON.....	26
4.6.1.	Tipado Dinámico	27
4.6.2.	Fuertemente Tipado	27
4.6.3.	Multiplataforma	27
4.6.4.	Orientado a Objetos.....	27
4.6.5.	Licencias del software	28
4.6.6.	Python frente a otros lenguajes de programación.	29
4.7.	ENTORNO RASPBERRY PI	30
4.7.1.	El Hardware de Raspberry Pi.....	30
4.7.2.	Modelos de Raspberry Pi	31
4.8.	ESTÁNDAR IEEE 802.15.4/ZIGBEE	32
4.8.1.	Tipos de tráfico.....	34
4.8.1.1.	Datos periódicos (continuos)	34
4.8.1.2.	Datos intermitentes (por eventos).....	34
4.8.1.3.	Datos periódicos con comunicación garantizada (GTS)	34
4.8.2.	Arquitectura del protocolo IEEE 802.15.4/ZigBee	34
4.8.2.1.	Capa Física	35
4.8.2.2.	Capa de control de acceso al medio	38
4.8.2.3.	Capa de red ZigBee.....	43
4.8.2.4.	Capa de aplicación.....	47
4.8.3.	Seguridad en una red ZigBee	48

4.9. MICROCONTROLADORES	49
4.9.1. Estructura general de un microcontrolador	49
4.9.1.1. Unidad central de procesamiento (CPU).....	50
4.9.1.2. Periféricos de entrada/salida	51
4.9.1.3. Memoria de programa	51
4.9.1.4. Memoria de Datos.....	52
4.9.1.5. Entradas y salidas de propósito general	52
4.9.1.6. Puertos de comunicación.....	52
4.9.1.7. Registros.....	53
4.9.2. Arquitectura de un microcontrolador.....	53
4.9.2.1. Arquitectura Von Neuman	54
4.9.2.2. Arquitectura Harvard	54
4.10. ESTRUCTURA WEB	55
4.10.1. Modelo básico de una comunicación Web	55
4.10.2. Diseño Web	57
4.10.3. Programación Web.....	58
4.10.4. Bases de datos.....	58
4.10.4.1. Bases de datos relacionales (SQL)	59
4.10.4.2. Bases de datos NoSQL.....	60
5. MATERIALES Y MÉTODOS	63
5.1. Materiales.....	63
5.2. Métodos.....	63
5.2.1. Descripción general del sistema.	64
5.2.2. Análisis de la situación actual de la empresa.....	65
5.2.2.1. Accesos y puntos específicos de la empresa a proteger	66
5.2.3. Etapa de hardware	68
5.2.3.1. Dispositivo final ZigBee	68
5.2.3.2. Central del sistema.....	76
5.2.3.3. Diseño de circuitos	79
5.2.4. Etapa de software	87
5.2.4.1. Programación del PIC 18F25K20 con las funciones del bloque del dispositivo Final ZigBee	87
5.2.4.2. Configuración de los módulos RF XBee.....	94
5.2.4.3. Configuración del sistema operativo de la central del sistema	99
5.2.4.4. Programación de la central del sistema	104
5.2.4.5. Servidor Web.....	119

6. RESULTADOS.....	124
6.1. Pruebas de recepción de señal en la topología de Red IEEE 802.15.4 adoptada ...	126
6.2. Pruebas de verificación de comunicación entre dispositivos finales y la central de alarmas.	128
6.3. Difusión de alertas a través de la aplicación Telegram.....	133
6.4. Funcionamiento del servidor Web.....	135
6.5. Limitaciones del sistema	137
7. DISCUSIÓN	139
8. CONCLUSIONES.....	143
9. RECOMENDACIONES	146
10. BIBLIOGRAFÍA	149
11. ANEXOS.....	152

ÍNDICE DE FIGURAS

Figura 1. Descripción general de los componentes de un sistema.....	9
Figura 2. Elementos de hardware que conforman el sistema de seguridad.	11
Figura 3. Diversas herramientas de software, orientadas al desarrollo de aplicaciones disponibles en la actualidad.	12
Figura 4. Esquema general de un sistema de seguridad distribuido.	13
Figura 5. Sistemas de seguridad según el medio utilizado para transmitir información.	15
Figura 6. Parámetros fundamentales sobre los cuales se basa el concepto de Smart Cities.	18
Figura 7. Estadísticas de EY LLP GISS 2015 sobre asuntos de ciberseguridad más importantes que actualmente enfrentan los negocios.	19
Figura 8. Panorama de riesgos que enfrenta el concepto de Smart City.....	20
Figura 9. Bloques de la composición general de un dispositivo dentro de una WSN.	22
Figura 10. Esquema de una red de sensores inalámbricos basada en el estándar IEEE 802.15.4, y su integración de servicios adicionales, mediante el uso de otras redes de comunicación.	23
Figura 11. Clases de redes inalámbricas de corto alcance.	26
Figura 12. Tarjetas de desarrollo Raspberry Pi. a) Raspberry Pi 3 B. b) Raspberry Pi Zero.	32
Figura 13. Diferentes estándares inalámbricos y posicionamiento de Zigbee en función del área de cobertura y la tasa de transmisión de datos.	33
Figura 14. Capas del estándar IEEE 802.15.4 y ZigBee.....	35
Figura 15. Distribución de los canales bajo el estándar IEEE 802.15.4 y la superposición existente con los canales Wi-Fi en la banda ISM de 2.4GHz.....	36
Figura 16. Estructura del paquete ZigBee.	37
Figura 17. Estructura de la supertrama.	40
Figura 18. Estructura de la trama MAC baliza.	42
Figura 19. Estructura de trama MAC de Datos.	42
Figura 20. Estructura de la trama MAC de acuse de recibo.	42
Figura 21. Estructura de trama MAC Comando.	43
Figura 22. Diversas topologías de una WSN a) Topología malla, b) topología árbol, c) Topología estrella.	45
Figura 23. Topología tipo árbol de la red ZigBee.....	46

Figura 24. Estructura general de los componentes de un microcontrolador.....	49
Figura 25. Diagrama de bloques de la arquitectura Von Neuman.....	54
Figura 26. Arquitectura Harvard de un microcontrolador.	55
Figura 27. Modelo cliente servidor en una comunicación Web.	56
Figura 28. Arquitectura de tres capas.	56
Figura 29. Diagrama de bloques del sistema integral de seguridad propuesto.....	64
Figura 30. Distribución de los espacios y áreas de trabajo de la empresa TECSERLED.	66
Figura 31. Elevación frontal de la empresa de tecnología y servicios TECSERLED. ...	67
Figura 32. Distribución de sensores que conforman el sistema de seguridad.	68
Figura 33. Composición del bloque del dispositivo Final ZigBee.	69
Figura 34. Disposición de pines del microcontrolador PIC18F25K20.....	70
Figura 35. Hardware del sensor PIR HC-SR501.	71
Figura 36. Hardware del sensor MQ-2.	72
Figura 37. Sensor magnético MC-38.....	73
Figura 38. Pulsador metálico para chasis, normalmente abierto.	73
Figura 39. Hardware del módulo RF XBee S2C.	75
Figura 40. Topología de red inalámbrica TIPO ESTRELLA implementada.	76
Figura 41. Raspberry Pi Camera Module v1.3.	78
Figura 42. Fuente de alimentación 5V 2.5A Micro USB para Raspberry pi.....	78
Figura 43. Módulo relé 4 canales 5V DC.	79
Figura 44. Conformación del elemento coordinador, utilizando un módulo XBee- Explorer.	79
Figura 45. Circuito fuente de alimentación para el dispositivo Final ZigBee.	80
Figura 46. Acondicionamiento de señales: a) sensor de movimiento, b) sensor magnético.....	81
Figura 47. Acondicionamiento de señales; a) sensor de humo, b) botón de pánico.....	83
Figura 48. Diagrama de conexiones; entre el microcontrolador con el dispositivo XBee S2C, y sensores.	84
Figura 49. Diseño del PCB del dispositivo final ZigBee y su acabado final.....	85
Figura 50. Diagrama de conexiones de la central del sistema.	86
Figura 51. Diagrama de conexión entre el puerto GPIO de la Raspberry Pi, y el módulo relé de 4 canales.....	87

Figura 52. Diseño recomendado para elaborar un programa en el compilador PROTON IDE.....	88
Figura 53. Diagrama de flujo- programación del PIC18F25K20.	91
Figura 54. Continuación del diagrama de flujo-programación del PIC18F25K20.....	92
Figura 55. Continuación del diagrama de flujo-programación del PIC18F25K20.....	92
Figura 56. Grabación del programa .hex generado, en el microcontrolador PIC18F25K20.	93
Figura 57. Configuración del dispositivo coordinador.	95
Figura 58. Campos intervenidos para la configuración del XBee S2C, como dispositivo coordinador de la red.	96
Figura 59. Dirección SH y SL del dispositivo XBee coordinador, utilizados como DH y DL respectivamente en cada dispositivo final.	98
Figura 60. Grabación de la imagen del SO en la tarjeta SD, con la ayuda del software Win32 Disk Imager.....	100
Figura 61. Conexión básica para la conexión de la Tarjeta Raspberry Pi, a través de SSH.	101
Figura 62. Obtención de la dirección IP asignada a la Raspberry Pi, en la red local por medio del software Wireless Network Watcher.	101
Figura 63. Configuración del cliente SSH para acceder a la tarjeta Raspberry Pi.....	102
Figura 64. Ingreso a la Raspbeery Pi desde el terminal PuTTty.....	102
Figura 65. Menú de configuraciones generales de la Raspberry Pi.....	104
Figura 66. Descripción general de los requerimientos, que conforman la estructura del software implementado en la central del sistema.	105
Figura 67. Creación de una base de datos del tipo Cloud Firestore en Firebase.	107
Figura 68. Creación de un Bot, generado a través de la App. de Telegram y BotFather.	108
Figura 69. Estructura general de módulos y paquetes que conforman un proyecto en Python.	109
Figura 70. Estructura del paquete Utils.	110
Figura 71. Estructura jerárquica del paquete models, que contiene el modelo de la base de datos (NoSQL) representado en código.	111
Figura 72. Estructura del paquete Services, y su relación de herencia con las clases de los módulos representados en el paquete models.	113

Figura 73. Estructura del paquete Controller y sus relaciones de herencia con las superclases mostradas.	114
Figura 74. Composición del paquete BootRaspSec.	116
Figura 75. Estructuración jerárquica de módulos y paquetes adoptada, para el sistema de seguridad propuesto.	117
Figura 76. Diagrama general del funcionamiento del sistema, y su arquitectura de adquisición, gestión, y difusión de datos.	118
Figura 77. Estructura del proyecto para el servidor web generado con Angular.	120
Figura 78. Ficheros fundamentales de un proyecto generado con Angular.	120
Figura 79. Referenciación del componente raíz por medio del fichero Index.	121
Figura 80. Componente raíz y componentes generados para el desarrollo del proyecto.	122
Figura 81. Función del fichero main.ts encargado del arranque de módulos en Angular.	123
Figura 82. Central del sistema de la WSN basada en el estándar IEEE 802.15.4.	124
Figura 83. Dispositivos finales ZigBee: ZONA 01 (Sensor Magnético), ZONA 02 (Sensor de Humo).	125
Figura 84. Dispositivos finales ZigBee: ZONA 03 (Sensor PIR), ZONA 04 (Sensor PIR).	125
Figura 85. Dispositivos finales ZigBee: ZONA 05 (Sensor PIR), ZONA 06 (Botón de Pánico).	126
Figura 86. Nivel de RSSI obtenido por cada Zona perteneciente a la WSN implementada respecto de su coordinador.	127
Figura 87. Nivel de confianza en la recepción de información, por cada zona perteneciente a la WSN.	127
Figura 88. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 01.	130
Figura 89. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 02.	130
Figura 90. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 03.	131
Figura 91. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 04.	131

Figura 92. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 05.	132
Figura 93. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 06.	132
Figura 94. Funcionalidades implementadas en el Bot de Telegram creado para la difusión de alertas.	133
Figura 95. Gestión de alarmas mediante el uso del comando “/stream Activar” y “/foto”.	135
Figura 96. Despliegue de la interfaz web desarrollada (componente dashboard).	136
Figura 97. Despliegue de la interfaz web desarrollada (componente config).	136
Figura 98. Redes WLAN operando en la banda de 2.4 GHz, en el entorno de monitoreo de la WSN implementada.	140
Figura 99. Número total de eventos falsos desencadenados, según la función de densidad de probabilidad de la distribución Poisson, aplicada a este caso en particular.	142
Figura 100. PCB del circuito correspondiente al dispositivo final ZigBee implementado.	154
Figura 101. Conexión del PICKit 3 y el microcontrolador PIC18F25K20 para realizar la programación mediante ICSP.	159
Figura 102. Agregación de una nueva red Wi-Fi a la tarjeta Raspberry Pi.	160
Figura 103. Dispositivos implementados: ZONA 01 (Sensor Magnético), ZONA 02 (Sensor de Humo).	164
Figura 104. Dispositivos implementados: ZONA 03 (Sensor PIR), ZONA 04 (Sensor PIR).	164
Figura 105. Dispositivos implementados: ZONA 05 (Sensor PIR), ZONA 06 (Botón de Pánico).	165

ÍNDICE DE TABLAS

Tabla 1. Resumen Nacional de los delitos de mayor afectación social en Ecuador periodo 2017-2018.....	7
Tabla 2. Resumen de la clasificación de las redes inalámbricas según su área de cobertura.	16
Tabla 3. Diversos lanzamientos del software Python y su compatibilidad con la licencia GPL.	28
Tabla 4. Comparación del lenguaje de programación Python con otros lenguajes de programación, mayormente utilizados en la actualidad.....	29
Tabla 5. Diferencias entre diversos modelos de Raspberry Pi.	31
Tabla 6. Características de la capa física de IEEE 802.15.4.....	36
Tabla 7. Diferencias existentes entre una base de datos del tipo SQL y las NoSQL.	62
Tabla 8. Lista de materiales.	63
Tabla 9. Resumen de accesos vulnerables en la infraestructura a monitorear.....	67
Tabla 10. Principales características comparativas de tres microcontroladores.	69
Tabla 11. Características de sensores de movimiento	70
Tabla 12. Características principales de varios sensores de gases disponibles en la actualidad.....	72
Tabla 13. Comparación de dispositivos ZigBee entre dos fabricantes.	74
Tabla 14. Descripción de los pines I/O del módulo XBee S2C.....	75
Tabla 15. . Comparación de tarjetas de desarrollo, basadas en el uso de software libre.....	77
Tabla 16. Descripción de las conexiones realizadas en la central del sistema.	86
Tabla 17. Trama enviada según el tipo de sensor activado	90
Tabla 18. Parámetros configurados en el XBee S2C nodo coordinador.	96
Tabla 19. Parámetros configurados en el XBEE S2C nodo dispositivo final.....	98
Tabla 20. Instalación de herramientas necesarias para empezar un proyecto en Angular.	119
Tabla 21. Valores obtenidos durante los 7 días de observación realizada a la comunicación existente entre dispositivos finales y la central de alarmas.	129
Tabla 22. Valores de lambda obtenidos, en base al suceso de estudio tomado para hacer la demostración.	141

1. TÍTULO

**“DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA INTEGRAL DE SEGURIDAD
UTILIZANDO SOFTWARE LIBRE, BASADO EN EL ESTÁNDAR IEEE 802.15.4
PARA LA EMPRESA DE TECNOLOGÍA Y SERVICIOS TECSERLED DE LA
CIUDAD DE LOJA.”**

2. RESUMEN

La tecnología de redes de sensores inalámbricos (WSN) ha surgido como una solución viable para muchas aplicaciones innovadoras. En este documento, se describe el diseño e implementación de un sistema de seguridad, basado en una red de sensores inalámbricos, que opera bajo el estándar IEEE 802.15.4/ZigBee, el mismo que se ha desarrollado utilizando plataformas que trabajan bajo la modalidad de *software* libre; como la tarjeta de desarrollo Raspberry Pi, los módulos RF XBee-S2C, el microcontrolador PIC18F25K20 y el entorno Angular para el desarrollo de páginas web.

La Raspberry Pi basada en el sistema operativo Linux, al ser totalmente personalizable y programable, ejecuta las funciones de central de alarmas, encargándose también de; procesar la información proveniente de la red de sensores, almacenar en la base datos Firebase y de la difusión de alertas a través de la aplicación Telegram, convirtiendo de esta forma el sistema de seguridad basado en una WSN, en un sistema integral con funcionalidades adicionales, que permiten al usuario establecer la gestión de alarmas de forma eficiente.

El desarrollo del *software* involucró el uso de varios lenguajes de programación como Python, BASIC, y Java Script, teniendo cada uno de ellos un uso específico en las etapas del diseño. El sistema se implementó en la empresa de tecnología y servicios TECSERLED de la ciudad de Loja, permitiendo validar la funcionalidad del *software* y del *hardware* respectivamente.

ABSTRACT

Wireless sensor network technology (WSN) has emerged as a viable solution for many innovative applications. In this document, it is described the design and implementation of a security system, based on a wireless sensor network, which operates under the IEEE 802.15.4/ZigBee standard, the one has been developed using platforms that work under the modality of free software; such as the Raspberry Pi development card, the XBee-S2C RF modules, the PIC18F25K20 microcontroller and the Angular environment for the development of web pages.

The Raspberry Pi based on the Linux operating system, being fully customizable and programmable, it executes the functions of alarms' central, taking over as well of processing the information which comes from the sensor network, storing in the Firebase database and the diffusion of alerts through the Telegram application, converting in this way the security system based on a WSN, in an integral system with additional functionalities, that allow the user to establish alarms' management in efficient way.

The development of the software involved the use of several programming languages such as Python, BASIC, and Java Script, each of them having a specific use in the design stages. The system was implemented in the technology and services company TECSERLED of the city of Loja, allowing to validate the software and hardware functionality respectively.

3. INTRODUCCIÓN

Con más de una década de intensa investigación y desarrollo, la tecnología de redes de sensores inalámbricos en la actualidad se encuentra como una solución accesible para diversidad de aplicaciones. La seguridad inalámbrica, es una de las tecnologías emergentes para la vigilancia inteligente de infraestructuras residenciales, comerciales, e industriales. Tecnologías inalámbricas como *Bluetooth* y *Wi-Fi* se han utilizado en este sentido, promoviendo cada vez más el uso de nuevas y mejores tecnologías en el ámbito de la seguridad a precios accesibles.

La automatización de la seguridad haciendo uso de tecnologías inalámbricas, es la última tendencia que las personas están buscando en casas residenciales, apartamentos, oficinas, negocios, etc. Sin embargo muchas tecnologías atadas a los tradicionales medios guiados se han empleado para brindar la seguridad en diversas infraestructuras, pero sufren de desventajas como complejidad en su instalación, mayores costos y limitaciones en sus servicios, al igual forma que los prestados por tecnologías inalámbricas de sistemas desarrollados en etapas iniciales, los cuales prestan sus servicios de forma aislada e independiente, sin tener una verdadera convergencia de servicios y un ahorro adecuado de los recursos de redes de comunicación.

Bajo estas premisas, teniendo en cuenta la demanda existente por este tipo de sistemas, y debido a la falta de desarrollo de proyectos en estas áreas técnicas por parte de la Universidad Nacional de Loja, que permitan aprovechar el desarrollo tecnológico para ofrecer este tipo de soluciones, surge la iniciativa de abordar esta problemática y proponer soluciones viables a través del desarrollo del presente trabajo de titulación, que consiste en el diseño de un sistema de seguridad basado en el estándar IEEE 802.15.4 haciendo uso de herramientas de software libre y su implementación en la empresa de Tecnología y Servicios TECSERLED en la ciudad de Loja.

Para el desarrollo del presente trabajo, se abarca una investigación previa de los fundamentos teóricos que involucran el uso del estándar en mención, se realiza un análisis y elección de herramientas a nivel de *hardware* y *software*, que se implementan para definir el sistema, y que finalmente serán validados mediante la comprobación de su funcionamiento, basados en los siguientes objetivos.

Objetivo General.

- Desarrollar un sistema integral de seguridad, empleando el protocolo de redes de comunicación de sensores ZigBee y el uso de herramientas de desarrollo libre, para monitoreo, control y gestión de alarmas constantes en la empresa de tecnología y servicios TECSERLED de la ciudad de Loja.

Objetivos Específicos.

- Recopilar y analizar los fundamentos teóricos, que serán la base para el diseño e implementación del sistema de seguridad.
- Identificar y establecer, las herramientas y servicios que se implementarán en el sistema de seguridad para la empresa de tecnología y servicios TECSERLED de la ciudad de Loja.
- Establecer una interfaz amigable con el usuario que integre las funcionalidades del sistema, permitiendo su configuración local y también remotamente a través de una plataforma web que contiene una base de datos para su administración.
- Integrar los componentes que conforman el sistema de seguridad, a la plataforma web desarrollada, de tal forma que permitan al usuario ejecutar acciones basadas en la domótica de forma remota, a través de medios digitales.
- Validar el funcionamiento del software libre y el hardware utilizado, en el sistema desarrollado para la empresa de tecnología y servicios TECSERLED de la ciudad de Loja.

4. REVISIÓN DE LITERATURA

4.1. GENERALIDADES

A lo largo de los últimos años, el continuo avance de la tecnología ha sido un factor determinante para el gran impacto sufrido en la mayor parte de las actividades de la sociedad. Dichos avances tecnológicos se manifiestan diariamente en la cultura, los hábitos, el trabajo y en el hogar, permitiendo que la transmisión de información, en las redes locales de comunicación genere cada vez más importancia.

En el contexto del desplazamiento de redes y que tienen que ver con el factor seguridad, en la actualidad estos sistemas son existentes y se encuentran evolucionando junto con varias aplicaciones que exigen la transmisión de información de control, en las redes privadas. Claros ejemplos son los denominados hogares y edificios inteligentes, en donde los diversos dispositivos y sensores están interconectados.

Una de las principales características que definen al ser humano, es la constante búsqueda de espacios seguros y confortables que le permitan llevar una vida más placentera, lo ha hecho desde tiempos muy remotos, cuando las cavernas eran su refugio, desde aquel entonces el ser humano empezó a modelar espacios para que lo resguarden de la intemperie, al mismo tiempo que protejan a su familia y sus bienes. La seguridad en todos sus aspectos, al ser una necesidad que ha surgido desde hace miles de años, ha ido evolucionando a la par de las formas sociales y de los desarrollos tecnológicos del momento.

En el Ecuador, país en el cual casi todos los temas de gran importancia se encuentran aún en proceso de ser resueltos, la seguridad ciudadana es uno de ellos. Siendo así, que los índices actuales de delincuencia en el país generan preocupación y malestar común entre la población. Según los últimos reportes del Ministerio del Interior correspondientes al año 2018, informan que se ha producido una disminución del 12% en los delitos de mayor afectación social, dentro de los cuales se encuentra incluido el robo a personas y entidades económicas, detalles que se muestran en la Tabla 1 [1].

Tabla 1. Resumen Nacional de los delitos de mayor afectación social en Ecuador periodo 2017-2018.

Cuadro de Resumen Nivel Nacional					
Ene-Mar					
2017-2018					
Ord.	Indicadores de violencia y delincuencia	Valor Absoluto		Variación	
		Año 2017	Año 2018	Porcentual	Absoluta
Sección de muertes violentas					
1	Homicidio intencional	279	242	-13%	-37
1.1	Homicidio/Asesinato	242	223	-8%	-19
1.2	Femicidio	37	19	-49%	-18
2	Suicidio	271	298	10%	27
Sección Robos, Hurtos y Receptación					
3	Robo a personas	7316	6448	-12%	-868
4	Robo a domicilios	3453	3224	-7%	-229
5	Robo a unidades económicas	1383	1165	-16%	-218
6	Robo a vehículos	1112	1128	1%	16
7	Robo a motos	1459	1276	-13%	-183
8	Robo de bienes, accesorios y autopartes de vehículos	2887	2444	-15%	-443
9	Robo en ejes viales o carreteras	49	50	2%	1
10	Robo a embarcaciones de espacio acuático	162	185	0%	23
11	Hurto	7495	6420	-14%	-1075
12	Receptación	1550	1183	-24%	-367
Sección Abigeato					
13	Abigeato	560	517	-8%	-43
TOTAL		27705	24282	-12%	-3423

Fuente: [1]

Si bien los índices de delitos que afectan mayormente al Ecuador son alarmantes en la actualidad, y principalmente con aquellos que tienen que ver con la sección de Robos, hurtos y recepción, en donde se ve afectado mayoritariamente el ciudadano común y el empresario o dueño de una unidad económica, refleja que la seguridad ciudadana en el país presenta graves falencias, haciendo que la situación social en donde se supone que predomine la sensación de confianza, la ausencia de riesgos y daños a la integridad física

y psicológica de las personas, sea en cierto grado totalmente lo opuesto. Para ello aparte de ser un tema netamente del estado, del cual han sido designados organismos de control y contrarresto como lo es el Ministerio del Interior, la Policía Nacional o el ECU 911, existen mecanismos o medidas que permiten disminuir estos efectos y una de ellas es la introducción en el mercado de los sistemas de seguridad, quienes se incluyen en el marco referencial y permiten de una u otra forma disminuir el número total de afectaciones.

Estos sistemas han estado presentes a lo largo del tiempo en miles de infraestructuras ecuatorianas y sus modificaciones han sido adaptadas según los requerimientos y el avance tecnológico del momento, siendo el caso que en la actualidad se habla de la nueva introducción de sistemas de seguridad que implementan comunicación inalámbrica. La comunicación inalámbrica es parte de nuestra vida diaria, diversidad de usuarios confían plenamente en el desempeño de las tecnologías inalámbricas y utilizan muy a menudo numerosos dispositivos inalámbricos en sus hogares y lugares de trabajo. La comunicación inalámbrica se encuentra presente en la mayor parte de las actividades diarias, afectando todo lo que nos rodea de diferentes formas.

La transición de sistemas de seguridad o sistemas de alarma, de cableados a inalámbricos, se ha dado rápidamente y hoy en día la aceptación de esta tecnología es muy alta para soluciones de seguridad en viviendas e instalaciones comerciales. Los entes involucrados en la instalación de sistemas de seguridad, apoyan la implementación de sistemas de alarma inalámbricos, lo que permite la adopción por gran parte de los consumidores y la aceptación de las compañías de seguro de estas tecnologías, quienes hace un tiempo atrás pensaban que el uso de esta tecnología no eran lo suficientemente confiable para la seguridad.

4.2. SISTEMAS DE SEGURIDAD

Los sistemas de seguridad hoy en la actualidad, abarcan diversos campos interdisciplinarios, muchos de ellos, tienen requisitos muy críticos. Su fracaso puede poner en peligro la vida humana y del medio ambiente, dañar gravemente la infraestructura económica, poner en peligro la privacidad personal, socavar la viabilidad de sectores empresariales completos y facilitar el delito [2].

Los requisitos de seguridad difieren mucho de un sistema a otro. Por lo general, se necesita alguna combinación de autenticación de usuario, integridad de transacción y responsabilidad, tolerancia a fallas, confidencialidad de mensajes y ocultamiento. Aun así muchos sistemas fallan debido a que sus diseñadores protegen aspectos del sistema de forma incorrecta, o los protegen, pero de forma inadecuada [2].

Lo primero que se debe aclarar es; ¿qué es un sistema?, en la práctica, esto puede denotar:

1. Un producto o componente, un protocolo, una tarjeta inteligente o el hardware de una PC, que siendo aplicada una entrada produce una salida.
2. Una colección de lo anterior más un sistema operativo, comunicaciones y otros elementos que conforman la infraestructura de una organización.
3. Todo lo anterior más una o varias aplicaciones (cuentas, nómina, diseño, etc.), personal de tecnologías de la Información (TI), usuarios internos, usuarios de administración, clientes, usuarios externos, el entorno que lo rodea, incluidos los medios, competidores, reguladores y políticos

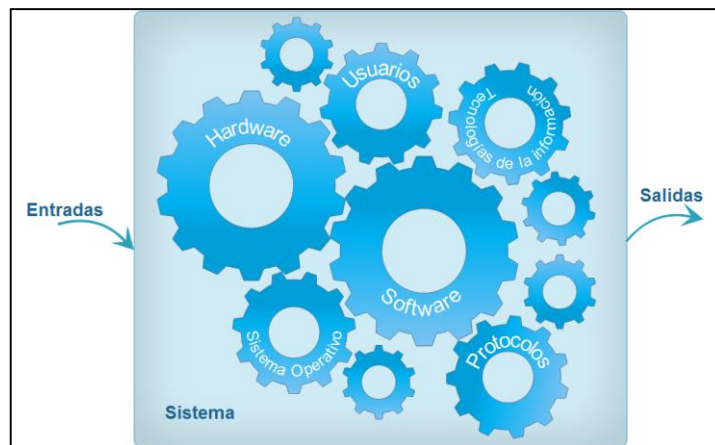


Figura 1. Descripción general de los componentes de un sistema.

Fuente: [El Autor]

En términos generales, un sistema de seguridad involucra un conjunto de elementos que se encuentran sumamente interrelacionados, siendo su objetivo principal establecer niveles de protección frente a posibles peligros, riesgos, delitos u otros sucesos que puedan afectar negativamente la integridad de: el medio ambiente, infraestructuras completas, poblaciones enteras, personas en concreto, negocios, vehículos, dispositivos, etc., en todos los aspectos.

De esta forma, una infraestructura de seguridad puede conformarse de todos estos elementos anteriormente descritos, incluyendo los dispositivos de *hardware*, componentes de *software* y el diseño de sus conexiones.

Los sistemas de seguridad electrónica o también conocidos como sistemas de alarma, se encuentran instalados en la mayoría de los edificios de oficinas y propiedades industriales, siendo también muy común en nuestro medio observar su instalación en infraestructuras residenciales. Se instalan con el único objetivo de ayudar a proteger los activos y mantener a las personas seguras [3]. Dado el caso de un evento no deseado, se debe notificar al personal adecuado para ejecutar una acción de respuesta inmediata y ayudar a los bienes o personas que puedan verse afectadas. Estos sistemas monitorean pasivamente edificios u otras áreas y se activan por eventos adversos. Los sistemas de alarma pueden tener una variada selección de sensores, cada uno de los cuales desencadena una acción por un método predefinido [4]. Su funcionamiento se centra en el uso de herramientas, procesos y los métodos necesarios para diseñar, implementar y probar nuevos sistemas totalmente acabados, adaptando los sistemas existentes a medida que su entorno evoluciona. En un sistema de seguridad electrónica se puede encontrar dos tipos de elementos principales.

4.2.1. El Hardware

Los elementos físicos que componen una infraestructura de seguridad, deben cumplir con todos los requisitos técnicos para el *software*. Se pueden dividir en tres subgrupos distintos [5]. Esta clasificación no debe entenderse como algo excluyente, dado que un elemento se puede encontrar en más de una categoría, pero esta separación permitirá comprender las funciones que se realizan en el sistema:

4.2.1.1. Sensores

Son los sentidos del sistema, al igual que los sentidos que tiene el cuerpo humano. Su función es recopilar información del entorno, transformarla en un valor análogo/digital y entregarla a un componente diseñado para controlarla [5]. Los datos obtenidos serán la entrada del equipo. Este grupo incluye cámaras, sensores de sonido, sensores de proximidad, sensores de detección de movimiento, sensores de humo, sensores infrarrojos, sensores de temperatura, entre otros.

4.2.1.2. Actuadores

Si los sensores son los que proporcionan toda la información necesaria del entorno, los actuadores serán los músculos que permitirán realizar acciones en dicho espacio. Una vez que se toma la decisión de realizar una acción, se envía una señal y se obliga a este elemento para que funcione. Ejemplos de estos son relés, altavoces, bloqueos e interruptores.

4.2.1.3. Controlador

Este es el cerebro del sistema. Un claro ejemplo de ello es un microcontrolador, un CPU, o la tarjeta de desarrollo Raspberry Pi, la cual contiene un potente procesador, que puede almacenar un programa y ejecutarlo. Recibir las señales sensoriales, procesarlas y luego activar y controlar los dispositivos actuadores y las alarmas.

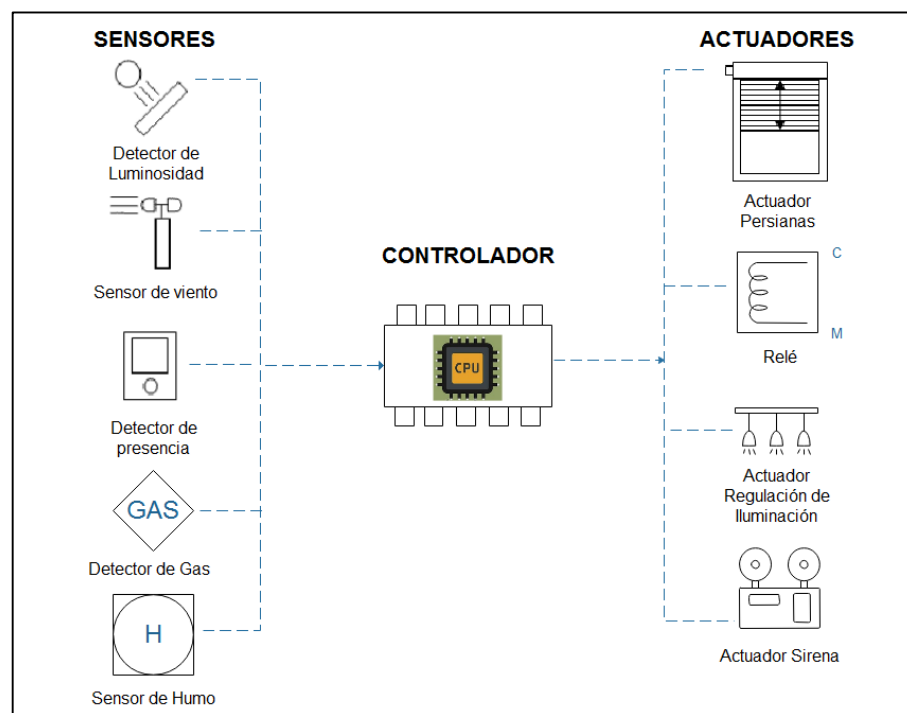


Figura 2. Elementos de hardware que conforman el sistema de seguridad.

Fuente: [El Autor]

4.2.2. El Software

Una vez que se tiene algunos datos de entrada y se quiere producir un efecto en el momento de su recepción, es necesario una herramienta que establezca las reglas que definen el comportamiento del sistema. Esto se soluciona con los recursos del *software*,

que son un conjunto de programas para su sistema. Más específicamente, se hace uso de la programación para crear programas propios y almacenarlos en un microcontrolador, microprocesador, PC, etc. Por lo tanto, se obtienen los datos que se necesitan, se lo procesa sin llevar a cabo una intervención de forma directa, y se implementa las medidas necesarias (ejecutar acciones o disparar alarmas) [5].

En este punto, no se especifica un lenguaje de programación en concreto. Sin embargo al hablar de *software*, se está refiriendo a él como un bloque para posteriormente profundizar en diferentes idiomas y aplicaciones (a lo largo del desarrollo de esta tesis, se utilizará varios lenguajes de programación, como lo es el uso de Python, Java, BASIC, etc.).



Figura 3. Diversas herramientas de software, orientadas al desarrollo de aplicaciones disponibles en la actualidad.

Fuente: [El Autor]

4.3. CLASIFICACIÓN DE LOS SISTEMAS DE SEGURIDAD ELECTRÓNICA

La clasificación de los sistemas de seguridad, parte principalmente de criterios fundamentales, como la cantidad de sitios (zonas de la infraestructura) a proteger, la aplicación del sistema y según el tipo de tecnología que implementan.

4.3.1. Sistemas de seguridad locales y distribuidos

Se consideran sistemas de seguridad electrónicos locales, aquellos que están diseñados para brindar seguridad en un lugar en específico, mientras que un sistema de seguridad distribuido es un conjunto de sistemas locales adaptados a cada sitio protegido, que además, trabajan conjuntamente con un sistema de Telecomunicaciones [6].

Los sistemas de seguridad distribuidos cubren necesidades de alto rendimiento, siendo principales usuarios de estos sistemas; los bancos, embajadas, universidades, aeropuertos, empresas petroleras, etc. Ya que este tipo de entidades mantienen infraestructuras en diferentes sitios, distribuidos incluso a nivel internacional. En algunos sistemas de alarma de este tipo, éstos trabajan en conjunto a una compañía de seguridad, encargada de ejecutar acciones de respuesta inmediata en caso de presentarse una activación de alarma.

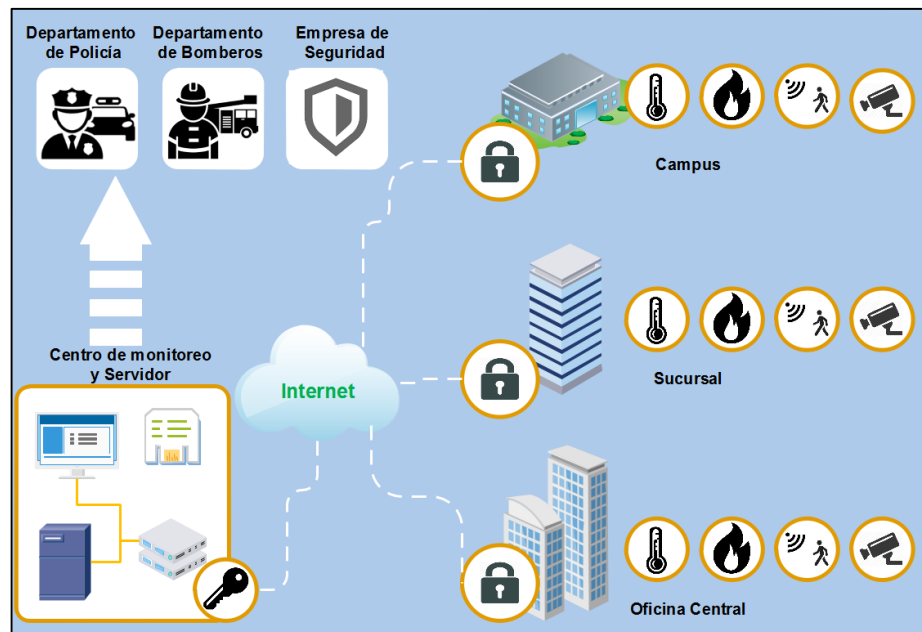


Figura 4. Esquema general de un sistema de seguridad distribuido.

Fuente: [El Autor]

4.3.2. Sistemas de seguridad según su aplicación

Según su aplicación, los sistemas de seguridad se subdividen en:

- **Alarmas de intrusión:** Encargadas de detectar movimiento, presencia y presión.
- **Alarmas técnicas:** Cuya función principal es la verificación de la presencia de incendios, humo, inundaciones, gas, fallo de suministro eléctrico, o fallo de línea telefónica.
- **Alarmas personales:** Involucran principalmente los controles de acceso, uso de tarjetas magnéticas y botones de pánico
- **Video vigilancia:** Encargadas de monitorear constantemente la infraestructura o zona de aplicación [3].

4.3.3. Sistemas de seguridad según el tipo de tecnología que implementan

En el envío de datos de los dispositivos dentro de los sistemas de seguridad, se pueden utilizar dos tecnologías si se diferencian por el medio físico por el cual se transmiten las señales, bien pueden ser cableadas o inalámbricas, la primera utiliza un medio físico (cable) entre los dispositivos, y la segunda no necesita cable para transmitir la señal entre los dispositivos.

4.3.3.1. Sistemas de seguridad cableados

Para aquellos sistemas que utilizan un medio físico tal como lo indica la Figura 5 a), la tendencia puede ser utilizar un cableado completamente dedicado para estos fines, como por ejemplo el estándar IEEE 1394, el USB en sus cuatro velocidades de transferencia de datos, ETHERNET bajo el estándar IEEE 802.3, o ya sea por el cableado existente. Es decir el cableado genérico que se utiliza para poner en accionamiento los artículos eléctricos. Hay iniciativas como la tecnología X10 que conecta todas las habitaciones de un edificio, siempre que se encuentren en la misma fase eléctrica, esta propuesta tecnológica usa las líneas eléctricas para señalización y controles. La principal desventaja de este estándar de sistema, es el desembolso económico inicial y las limitaciones existentes al enviar datos a través de este medio [5].

El tipo de sistemas cableados presenta como principal ventaja, su fiabilidad y robustez, ya que las señales que contienen la información no se encuentran expuestas a interferencias externas. Sin embargo la instalación del cableado y las canalizaciones correspondientes hacen que su montaje sea más sofisticado y costoso. Las distancias a las que pueden estar conectados los dispositivos finales o sensores con respecto a la central dependen de las características del cable utilizado en cada caso [7].

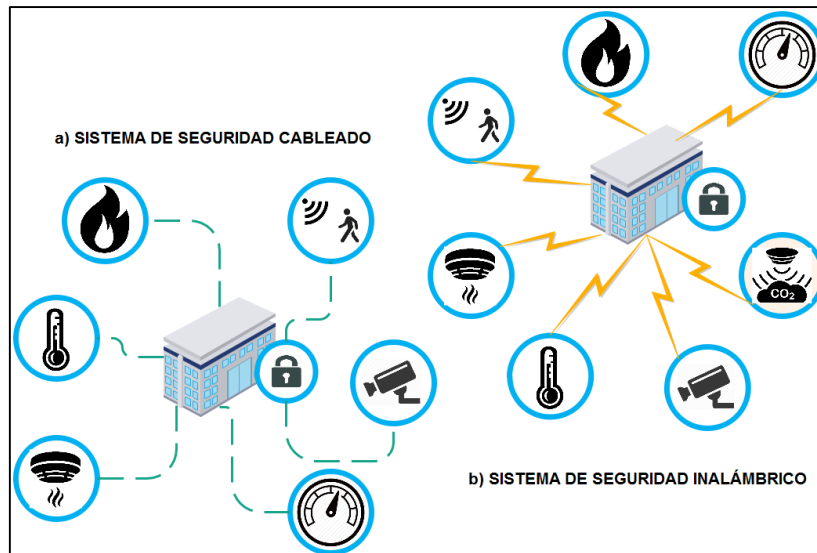


Figura 5. Sistemas de seguridad según el medio utilizado para transmitir información.

Fuente: [El Autor]

4.3.3.2. Sistemas de seguridad inalámbricos

En lo referente a sistemas de seguridad a través de medios no guiados, el mercado actual ofrece opciones inalámbricas altamente confiables, sin embargo a pesar de ser más fácil de implementar, podrían elevar el precio total del *hardware*. A parte de ello existe el factor de interferencia de radio, ya que las tecnologías inalámbricas funcionan en frecuencias muy cercanas. Más popularmente las tecnologías inalámbricas como *Wi-Fi* y *Bluetooth* usan 2.4 GHz, y considerando cuántos dispositivos que hay alrededor usando esa tecnología, que a veces se combina con el alcance relativamente bajo de potencia, a menudo impiden la comunicación [5].

Dentro de las tecnologías inalámbricas existen tres categorías organizadas por la distancia que comprenden, la primera la cual es objeto de estudio principalmente de este proyecto es la red WPAN, que se detalla a continuación:

- **Red WPAN**

(*Wireless Personal Area Network*) las cuales cubren cortas distancias, este tipo de conexiones no se utiliza para altos índices de transmisión de datos y generalmente tienen un bajo consumo de potencia. La IEEE divide grupos de trabajos encargados del desarrollo de los estándares, en estos grupos se encuentran IEEE. 802.15.1 *Bluetooth*,

IEEE 802.15.3 *Wimedia Alliance*, IEEE. 802.15.4 *ZigBee*, y otras tecnologías propietarias como Z-WAVE [3].

- **Red WLAN**

(*Wireless Local Area Network*), estas tiene una cobertura de alcance de hasta 100 metros, para que los dispositivos de esta red se puedan comunicar existen una serie de protocolos 802.11.x o WI-FI, que definen las características de una red de área local inalámbrica, permitiendo tener redes de alta velocidad que van desde los de 11 Mbps a 7 Gbps teóricos. Dependiendo del estándar opera en la banda de los 2.4 o 5 GHz [3].

- **Red WMAN**

(*Wireles Metropolitan Area Network*), quienes tienen un alcance del orden de los kilómetros, cubriendo áreas extensa como por ejemplo una ciudad completa, dotando de jerarquía de datos móviles o incluso los enlaces entre ciudades mediante retransmisiones de microondas. *Wi-Max* bajo el estándar IEEE 802.16 utiliza topología punto a multipunto para proporcionar acceso celular de banda ancha inalámbrica. En este tipo, también se encuentran las redes celulares y sus variantes que incluyen 2G (con GSM, CDMA o TDMA) 3G (con CDMA2000, EDGE o HSPA+) y 4G (LTE) [8]. Un resumen específico de todas estas tecnologías inalámbricas se muestra a continuación en la Tabla 2.

Tabla 2. Resumen de la clasificación de las redes inalámbricas según su área de cobertura.

Definición de Red	Estándar	Conocido como
Red de área personal inalámbrica (WPAN)	IEEE 802.15.1	Bluetooth
Red inalámbrica de área personal de baja velocidad (LR-WPAN)	IEEE 802.15.4	ZigBee
Red inalámbrica de área local (WLAN)	IEEE 802.11x	WiFi
Red inalámbrica de área Metropolitana (WMAN)	IEEE 802.16	WiMAX
Evolución a largo plazo (LTE)	IMT-Advance/3GPP	4G/LTE Advanced

Fuente: [El Autor]

Una ventaja principal en el uso de un sistema de seguridad basado en una de estas tecnologías inalámbricas, radica en que no precisan de ningún tipo de cableado ni canalización, pudiendo abarcar un radio de acción muy amplio, ofreciendo flexibilidad en la instalación de los componentes de manera sencilla y en muy poco tiempo. Sin embargo presenta inconvenientes relacionados al mantenimiento, ya que el

funcionamiento de los equipos se basa en el uso de pilas o baterías, por lo que es necesario revisar y sustituir periódicamente estos componentes. También esta clase de sistemas puede ser susceptible a interferencias externas producidas por emisiones de radio frecuencia o electromagnéticas, lo que las hace más vulnerables ante posibles sabotajes.

Los sistemas tradicionales y las actuales implementaciones, tienen un uso específico, en donde se enfrentan la necesidad de reducir la complejidad del modelo a un mínimo, para aumentar su fiabilidad, en comparación con el aumento de la utilidad y la riqueza de la información provista. Inicialmente, después de considerar las circunstancias tales como el precio, el vencimiento de la tecnología, y la infraestructura existente (es decir su conectividad), no es posible tener sistemas que se pueden administrar en tiempo real a bajo costo [4].

Sin embargo, hoy se cuenta con teléfonos inteligentes y demás dispositivos móviles, con la capacidad suficiente para visualizar, recibir y enviar datos recopilados por el hardware del sistema de forma inmediata. Por lo tanto es posible diseñar un sistema que sea implementado en un servidor creado por el mismo usuario, o incluso utilizar la computación en la nube para una mayor potencia de procesamiento, la cual supondría bajos niveles de mantenimiento.

En la Figura 5 a) y b) se representa un ejemplo de múltiples sensores que pueden instalarse en variedad de infraestructuras. Los cuales son encargados de generar alertas en caso de intrusiones, fugas existentes, detectar incendios, etc. En todos los casos una respuesta rápida para tales eventos puede ser crucial para salvaguardar activos y, en algunos casos, también la vida y la salud de las personas. La capacidad de respuesta en gran medida está orientada en asegurar los bienes y limitar los daños tanto como sea posible.

En casi todos los escenarios descritos anteriormente, el tiempo es crítico, esto apunta a que el sistema encargado para recibir y procesar eventos de alarma debe ser lo más eficiente posible. Las entidades involucradas deben obtener ayuda lo antes posible cuando algo les suceda a ellos o a sus edificios. Teniendo en cuenta que solo pocos minutos pueden separar un resultado efectivo de uno deficiente, es fundamental minimizar la demora en toda la secuencia de eventos, desde la alarma que se activa hasta que alguien está en el camino para ayudar. Por lo tanto un buen sistema de seguridad debe asegurar siempre una respuesta inmediata al evento desencadenado [4].

4.4. SMART CITIES

La misión de las ciudades inteligentes es una nueva y audaz iniciativa por parte de los gobiernos de países desarrollados para impulsar el crecimiento económico y mejorar la calidad de vida de las personas, al permitir el desarrollo local y aprovechar la tecnología como un medio para crear resultados inteligentes para los ciudadanos. Una ciudad inteligente es un área urbanizada donde múltiples sectores cooperan para lograr resultados sostenibles a través del análisis de la información contextual en tiempo real compartida entre la información específica del sector y los sistemas de tecnología operativa [9].

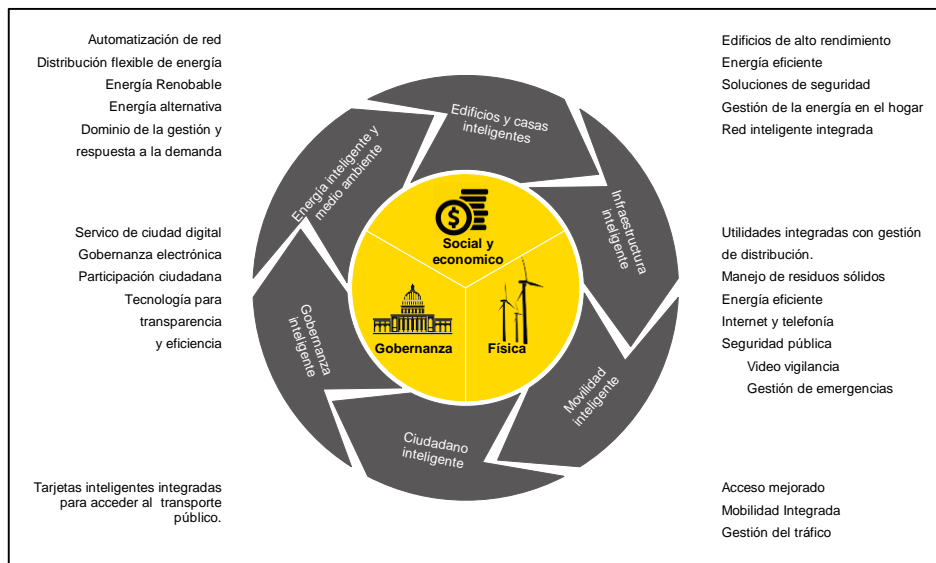


Figura 6. Parámetros fundamentales sobre los cuales se basa el concepto de Smart Cities.

Fuente: Gráfico adaptado de [9].

4.4.1. Seguridad de las ciudades inteligentes

Las ciudades inteligentes deberán tener una infraestructura física, social, institucional y económica inteligente, al tiempo que garantizan la centralidad de los ciudadanos en un entorno sostenible. Se espera que una ciudad tan inteligente genere opciones para que todos los residentes continúen con sus medios de vida e intereses de manera significativa y en completo agrado. La tecnología de vigilancia inteligente o los análisis para gestionar la multitud, el tráfico, la seguridad cibernética, la privacidad de los datos, el código de construcción para gestionar desastres naturales o causados por el hombre, etc., son factores que hacen que una ciudad sea segura y segura para que un ciudadano viva [9].

La ciberseguridad¹ efectiva es cada vez más compleja de entregar. El perímetro organizativo tradicional se está erosionando debido a la necesidad de colaboración con socios externos y empresas, las defensas de seguridad existentes están bajo una presión cada vez mayor. Los delincuentes cibernéticos están constantemente trabajando en nuevas técnicas para pasar por la seguridad de las organizaciones establecidas, accediendo a todo, desde la propiedad intelectual a la información de los clientes individuales; lo están haciendo de tal forma que puedan causar daños, interrumpir los datos confidenciales y robar la propiedad intelectual.

Cada día, los ataques provenientes de estos entes se vuelven más sofisticados y difíciles de vencer. Debido a este continuo crecimiento, no se puede decir exactamente qué tipo de amenazas surgirán el próximo año, dentro de cinco años o dentro de diez años; Solo se puede dimensionar que estas amenazas serán incluso más peligrosas que las de hoy. También se puede asegurar de que a medida que se desvanezcan las fuentes antiguas de esta amenaza, surgirán nuevas fuentes para tomar su lugar. A pesar de esta incertidumbre, de hecho, debido a ello, hay que ser claros sobre el tipo de controles de seguridad necesarios [9].

En respuesta a la Encuesta Global de Seguridad de la Información 2015 (EY LLP GISS 2015) realizada a 1,755 compañías, el 36% de los encuestados dice que es “improbable o altamente improbable” que su organización pueda detectar un ataque sofisticado. Aunque se observa una mejora considerable respecto al hallazgo de 56% en 2014, esto se debe al nivel de sofisticación, el cual aumenta continuamente.

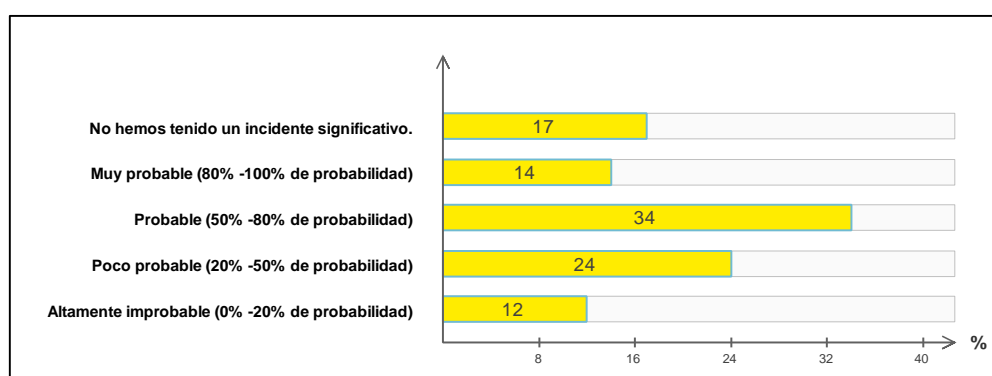


Figura 7. Estadísticas de EY LLP GISS 2015 sobre asuntos de ciberseguridad más importantes que actualmente enfrentan los negocios.

Fuente: Gráfico adaptado de [9].

¹ Práctica o mecanismo utilizado para proteger sistemas, redes y programas de ataques digitales

4.4.2. Panorama de riesgos de una Smart Citie

Es importante recordar dadas estas premisas, que la seguridad cibernética se convierte en un problema de toda la ciudad y no solamente se considera como un riesgo tecnológico. Dado que muchas oportunidades para IoT surgirán a través de la integración tecnológica y la colaboración, que continuará aumentando en complejidad, esta complejidad genera riesgos. Para gestionar eficazmente los riesgos en una ciudad inteligente, es importante definir claramente los límites de ese ecosistema.

También se debe decidir qué se está dispuesto a gestionar dentro de esos límites: si solo los riesgos que enfrentan los grupos de personas que se encuentran en la propia ciudad, o se debería tratar de influir en la mitigación de los riesgos que enfrentan las personas o datos fuera del país. Para ello se debe tener límites totalmente definidos [9].



Figura 8. Panorama de riesgos que enfrenta el concepto de Smart Citie.

Fuente: Gráfico adaptado de [8].

Las tecnologías de ciudades inteligentes capturan datos relacionados con todas las formas de privacidad y expanden drásticamente el volumen, rango y granularidad de los datos que se generan sobre personas y lugares. La privacidad puede ser amenazada y violada por una serie de prácticas que normalmente se consideran inaceptables, sin embargo, son parte de las operaciones en un sistema ecológico de ciudades inteligentes.

- **Vigilancia:** ver, rastrear, escuchar o grabar las actividades de una persona.
- **Agregación:** combinación de varios aspectos de los datos sobre una persona para

identificar una tendencia o patrón de actividades.

- **Fuga de datos:** la falta de políticas de protección de datos puede provocar fugas o el acceso incorrecto de información confidencial.
- **Uso extendido:** uso de los datos recopilados durante un período más largo que el establecido o para fines distintos al establecido sin el consentimiento del sujeto.

A parte de ello, se suman nuevos desafíos de seguridad en ciudades inteligentes como es el caso de:

4.4.2.1. Hardware Inseguro

Una de las principales preocupaciones sobre los sensores y equipos de ciudades inteligentes. Los hogares, edificios, oficinas, etc., son espacios inseguros y los dispositivos no están probados a fondo. Debido a la falta de estandarización de los dispositivos IoT, los sensores son propensos a la piratería. Personas de mal accionar pueden adulterar los sensores y alimentar datos falsos, causando fallas en la señal, y colapso del sistema, etc. [9].

4.4.2.2. Mayor superficie de ataque

Las operaciones de la ciudad inteligente utilizan un conjunto complejo y en red de infraestructura de TIC para gestionar diversos servicios. Cualquier dispositivo que esté conectado a la red es vulnerable a ser *hackeado*²; el número de puntos de entrada potenciales se multiplica en las ciudades inteligentes, y al comprometer un solo dispositivo, es posible atacar todo el sistema o la red. Esta vulnerabilidad de los sistemas se ve agravada por una serie de problemas que incluyen la seguridad y el cifrado débiles; el uso de sistemas heredados inseguros y el mantenimiento deficiente; efectos en cascada; y el error humano [9].

4.4.2.3. Consumo de ancho de banda

Miles de sensores o actuadores que intentan comunicarse con un solo servidor crearán una gran cantidad de tráfico de datos, que puede hacer que el servidor se caiga. Además, la mayoría de los sensores utilizan un enlace no cifrado para comunicarse, y por lo tanto, hay posibilidades de fallas de seguridad. El consumo de ancho de banda de miles de millones de dispositivos pondrá una tensión en el espectro de otras comunicaciones

² Termina que describe la obtención de un acceso no autorizado, a datos en un sistema o computadora

inalámbricas, que también operan en las frecuencias de megahertz como radio, televisión, servicios de emergencia, etc. [9] [10].

4.4.2.4. Riesgos de aplicaciones

Las aplicaciones han acelerado la integración de dispositivos móviles en nuestra vida diaria. Desde las aplicaciones de mapas, las redes sociales, las herramientas de productividad, los juegos, las aplicaciones han impulsado en gran medida la revolución de los teléfonos inteligentes y lo han hecho tan importante y tan extenso como lo es hoy. Mientras que las aplicaciones muestran una utilidad que aparentemente está limitada solo por la imaginación del desarrollador, también aumenta el riesgo de ser compatible con la política empresarial de dispositivos *Bring Your Own Device* (Trae tu propio dispositivo por sus siglas en Inglés BYOD) en un entorno corporativo [9].

4.5. REDES DE SENSORES INALÁMBRICOS

Las redes de sensores inalámbricos (WSN, por sus siglas en inglés) se utilizan en muchas áreas tecnológicas, que incluyen monitoreo del entorno, aplicaciones de atención médica, domótica, control de tráfico, logística, automatización industrial y gestión. Son particularmente eficientes en situaciones en las que los sistemas cableados no se pueden usar, o los sensores se encuentran en continuo movimiento. Entre todas las aplicaciones posibles, el monitoreo de los fenómenos físicos por medio de dispositivos pequeños y no invasivos representa un desafío para las nuevas generaciones de tecnologías inalámbricas [11]. Los dispositivos dentro de una WSN, son unidades autónomas que constan de un microcontrolador, una fuente de energía que por lo general siempre se trata de una batería, un radio-transceptor (RF) y un elemento sensor.

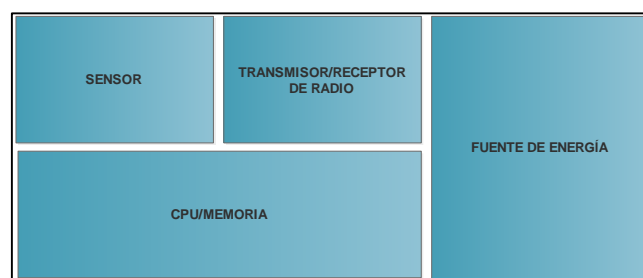


Figura 9. Bloques de la composición general de un dispositivo dentro de una WSN.

Fuente: [El Autor]

4.5.1. Elementos de la WSN

La arquitectura de red se compone de los siguientes elementos:

4.5.1.1. Nodo sensor o dispositivo final

Son unidades de detección de distinta naturaleza y tecnología que toman la información del medio a través del sensor incorporado y la convierten en señales eléctricas, para luego transmitir los datos de la respectiva conversión, hacia la estación base a través de sus puertas de enlace [11].

4.5.1.2. Nodo Gateway

Es la unidad destinada a recibir los datos de todas las unidades nodo sensor, existentes en la red, y su objetivo principal es actuar como un puente de comunicación entre dos redes diferentes [11].

4.5.1.3. Estación Base

Servidor central basado en un ordenador común o un sistema embebido que recoge todos los datos de todas las unidades de detección [11].

4.5.1.4. Red Inalámbrica

Es el establecimiento de la red basada en un estándar en específico, en el caso del presente proyecto de titulación se trata de la red basada en el estándar IEEE 802.15.4/ZigBee.

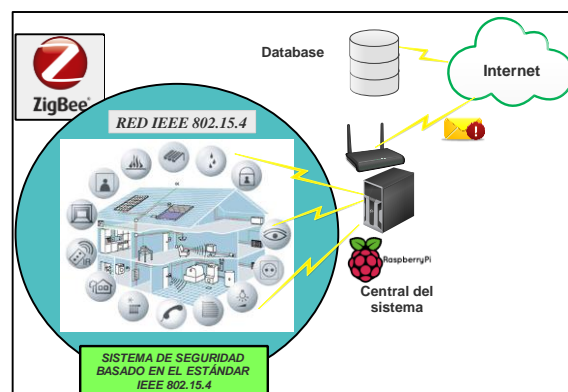


Figura 10. Esquema de una red de sensores inalámbricos basada en el estándar IEEE 802.15.4, y su integración de servicios adicionales, mediante el uso de otras redes de comunicación.

Fuente: [El Autor]

4.5.2. Características de la WSN

Las características significativas de las redes WSN se detallan a continuación:

4.5.2.1. Bajo costo

En las WSN, normalmente se implementan cientos o miles de Nodos sensores para medir cualquier entorno físico. Para reducir el costo general de toda la red, el precio unitario del nodo sensor debe mantenerse lo más bajo posible. Por lo tanto el costo es generalmente bajo en comparación a otras redes [12].

4.5.2.2. Eficiencia energética

La energía en las redes WSN se utiliza para diferentes propósitos, como computación, comunicación y almacenamiento. El nodo sensor consume más energía en comparación con cualquier otro para la comunicación. Si se quedan sin energía, a menudo se vuelven inválidos. Por lo tanto, los protocolos y el desarrollo de algoritmos deben considerar el consumo de energía en la fase de diseño [12].

4.5.2.3. Potencia computacional

Normalmente los nodos tienen capacidades computacionales limitadas, ya que se debe considerar el costo y la energía [12].

4.5.2.4. Capacidades de comunicación

En las WSN, generalmente se comunican utilizando ondas de radio a través de un canal inalámbrico. Tienen la propiedad de comunicarse en corto alcance, con ancho de banda estrecho y dinámico. El canal de comunicación puede ser bidireccional o unidireccional. Con el entorno operativo hostil, es difícil ejecutar una WSN sin problemas. Por lo tanto, el hardware y el software para la comunicación deben tener en cuenta la robustez, la seguridad y la resistencia [12].

4.5.2.5. Seguridad y privacidad

Cada nodo sensor debe tener mecanismos de seguridad suficientes para evitar el acceso no autorizado, los ataques y el daño intencionado de la información dentro del mismo [12].

4.5.2.6. Topología de red dinámica

En general WSN es una red dinámica. Teniendo en cuenta las variaciones sustanciales en la diversidad de dispositivos que la conforman, se da como resultado los frecuentes cambios en la topología de la red. Por lo tanto, los nodos WSN tienen que estar integrados con la función de reconfiguración y autoajuste [12].

4.5.2.7. Comunicación multi-salto

Una gran cantidad de nodos sensores se implementan en WSN. Por lo tanto, la forma viable de comunicarse con la estación base es tomar la ayuda de un nodo intermedio a través del enrutamiento. Si un nodo necesita comunicarse con el otro nodo o estación base que está más allá de su radio, debe hacerlo a través de la ruta de salto múltiple por nodo intermedio (hacer uso de un elemento enrutador) [12].

4.5.2.8. Operaciones robustas

Ya que los sensores se implementarán en un entorno grande y, a veces, hostil. Es necesario que los nodos deben ser tolerantes a errores y fallas. Por lo tanto, los nodos sensores necesitan la capacidad de auto-prueba, auto-calibración y autocorrección [12].

4.5.2.9. Tamaño físico reducido

Los nodos sensores son generalmente pequeños en tamaño con el rango restringido. Debido a su tamaño, su energía es limitada, lo que reduce la capacidad de comunicación [11] [12].

4.5.3. Clases de redes inalámbricas de corto alcance

Las redes inalámbricas de corto alcance se dividen en dos categorías principales: redes inalámbricas de área local (WLAN) y redes inalámbricas de área personal (WPAN). WLAN es un reemplazo o una extensión para redes de área local (LAN) cableadas como Ethernet (IEEE 802.3). El objetivo de una WLAN es maximizar el rango y la velocidad de datos [13].

Las WPAN, por el contrario, no están desarrolladas para reemplazar ninguna LAN cableada existente. Las WPAN se crean para proporcionar los medios para una comunicación inalámbrica de bajo consumo, dentro del espacio operativo personal (POS).

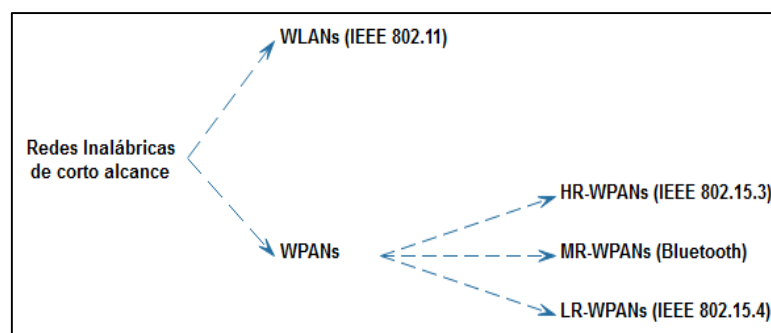


Figura 11. Clases de redes inalámbricas de corto alcance.

Fuente: [El Autor]

Las WPAN se dividen en tres clases (Figura 11): WPAN de alta velocidad (HR), WPAN de tasa media (MR) y WPAN de baja velocidad (LR) [13].

- Un ejemplo de una HR-WPAN es IEEE 802.15.3 con una velocidad de datos de 11 a 55 Mbps. Esta alta velocidad de datos ayuda en aplicaciones como la transmisión inalámbrica de video en tiempo real desde una cámara a un televisor cercano.
- Bluetooth, con una velocidad de datos de 1 a 3 Mbps, es un ejemplo de MR-WLAN y se puede usar en transmisión de voz de alta calidad en auriculares inalámbricos.
- ZigBee, con una tasa de datos máxima de 250 Kbps, está clasificado como un LR-WPAN.

4.6. LENGUAJE DE PROGRAMACIÓN PYTHON

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 en el Instituto Nacional de Investigación en Matemáticas y Ciencias de la Computación en los Países Bajos. Este lenguaje mantiene una sintaxis muy limpia que favorece un código legible. Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos [14].

Al ser un lenguaje de script, éste se ejecuta utilizando un programa intermedio llamado intérprete, en lugar de compilar el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora (lenguajes compilados). La principal ventaja de los lenguajes compilados es que su ejecución es más rápida. Sin embargo los lenguajes

interpretados son más flexibles, más portables y comprensibles a la hora de ser analizados por el ser humano [14]. Python sin embargo, tiene muchas de las características de los lenguajes compilados, por lo que se podría decir que es un lenguaje semi interpretado. En Python, como en Java y muchos otros lenguajes, el código fuente se traduce a un pseudo código intermedio llamado *bytecode*³ la primera vez que se ejecuta, generando archivos .pyc o .pyo (*bytecode* optimizado), que son los que se ejecutarán en sucesivas ocasiones.

4.6.1. Tipado Dinámico

La característica de tipado dinámico en un lenguaje de programación se refieren a que no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se determinará en tiempo de ejecución. Python usa un tipado dinámico cuando la comprobación de tipificación se realiza durante su ejecución en vez de durante la compilación [14].

4.6.2. Fuertemente Tipado

Python es un lenguaje de programación fuertemente tipado ya que no se permite tratar a una variable como si fuera de un tipo distinto al que tiene, es necesario convertir de forma explícita dicha variable al nuevo tipo previamente. En otros términos no se permiten violaciones de los tipos de datos, el valor de una variable de un tipo en concreto no se puede usar como si fuera otro tipo distinto, a menos que se realice una conversión previa [14].

4.6.3. Multiplataforma

El intérprete de Python está disponible para una variedad de plataformas como lo es UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc. Los programas realizados al no utilizar librerías específicas de cada plataforma, podrían ejecutarse sin ningún tipo de problemas en todos estos sistemas sin realizar grandes cambios [14].

4.6.4. Orientado a Objetos

La orientación a objetos es un paradigma de programación en el que los conceptos del mundo real que son relevantes al problema en concreto a tratarse, se trasladan a clases y

³ Representa a un código de programa que se ha compilado desde el código fuente, hasta el código de bajo nivel, diseñado para un intérprete de software.

objetos en un determinado programa. Este programa al ejecutarlo se comporta como una serie de interacciones entre los objetos. Python al contar con una estructura de datos de alto nivel y enfoque simple pero efectivo, también permite la programación imperativa, programación funcional y programación orientada a objetos. Su elegante sintaxis y escritura dinámica, junto con su naturaleza interpretada, lo convierten en un lenguaje ideal para la creación de scripts y el rápido desarrollo de aplicaciones en muchas áreas en la mayoría de las plataformas [15].

4.6.5. Licencias del software

En el año 2001, se formó la *Python Software Foundation* (PSF), una organización sin fines de lucro creada específicamente para poseer la propiedad intelectual relacionada con Python. Todas las versiones de Python son de código abierto. Históricamente, la mayoría, pero no todas las versiones de Python también han sido compatibles con *General Public License* (GPL⁴); La siguiente tabla resume los diferentes lanzamientos [16].

Tabla 3. Diversos lanzamientos del software Python y su compatibilidad con la licencia GPL.

Lanzamiento	Derivado de	Año	Propietario	¿GPL compatible?
0.9.0 a 1.2	n/A	1991-1995	CWI	Sí
1.3 a 1.5.2	1.2	1995-1999	CNRI	Sí
1.6	1.5.2	2000	CNRI	No
2.0	1.6	2000	BeOpen.com	No
1.6.1	1.6	2001	CNRI	No
2.1	2.0 + 1.6.1	2001	PSF	No
2.0.1	2.0 + 1.6.1	2001	PSF	Sí
2.1.1	2.1 + 2.0.1	2001	PSF	Sí
2.1.2	2.1.1	2002	PSF	Sí
2.1.3	2.1.2	2002	PSF	Sí
2.2 y superior	2.1.1	2001-Ahora	PSF	Sí

Fuente: [16]




Cabe mencionar que la compatibilidad existente con GPL no significa que se está distribuyendo Python bajo la normativa de GPL. Todas las licencias de Python, a diferencia de la GPL, permiten distribuir una versión modificada sin realizar cambios en el código fuente abierto. Las licencias compatibles con GPL permiten combinar Python con otro software que se lanza bajo la GPL.

⁴ Es el tipo de licencia asociada a algunos softwares de código abierto, en donde se detalla como el software y el código fuente que lo acompaña, se pueden copiar, modificar y distribuir libremente por el usuario.

4.6.6. Python frente a otros lenguajes de programación.

Al ser un lenguaje de programación con múltiples funcionalidades, a continuación se describe aspectos relevantes que hacen de este lenguaje de programación, un lenguaje muy popular [17].

Tabla 4. Comparación del lenguaje de programación Python con otros lenguajes de programación, mayormente utilizados en la actualidad.

Bases para la comparación	Lenguajes objeto de comparación		
	Python 	Java 	C++ 
Fácil de usar	Los códigos de Python son más cortos que Java y C++. Python sigue la estructura de códigos de programación dinámica, no solo fácil de usar sino también fácil de entender	Java no es fácil de usar en comparación con Python porque no existe un concepto de programación dinámica y los códigos son más largos que Python.	Al igual que Java, en C++ los códigos son más extensos, debido a que se trata de un lenguaje escrito estáticamente.
Código	Muy pocas líneas de código para ejecutar instrucciones.	Líneas de código más largas en comparación con Python	Líneas de código más largas en comparación con Python
Sintaxis	En Python, la declaración no necesita de un punto y coma para finalizar	Al final de la declaración, si pierde el punto y coma arroja un error	Al final de la declaración, si pierde el punto y coma arroja un error
Velocidad	Es más lento debido a que Python es un intérprete y también determina el tipo de datos en tiempo de ejecución. A pesar de ello el rendimiento del software es sumamente alto.	En términos de velocidad, Java es más rápido con relación a Python.	Debido a que C++ es un lenguaje precompilado, es decir que antes de su ejecución es necesario compilar, su velocidad de ejecución es muy rápida respecto a Python
Eficiencia	Más fácil de mantener, orientado a objetos y más simple de usar.	Menos limpio y manejable en comparación con Python.	Menos limpio y manejable en comparación con Python.
Creación rápida de prototipos	La creación rápida de prototipos es posible debido al pequeño tamaño del código	La creación rápida de prototipos no es posible debido al mayor tamaño del código	La creación rápida de prototipos no es posible debido al mayor tamaño del código
Uso en sistemas embebidos	Para sistemas embebidos, el lenguaje de programación Python se ha posicionado como una de las herramientas más efectivas para trabajar en las preferencias de los ingenieros y desarrolladores. Debido a las grandes comunidades	Existen soluciones basadas en Java para sistemas embebidos, pero su uso frente a Python no es muy popular, además de la poca información brindada para generar Investigación y	Los desarrolladores de sistemas embebidos, tradicionalmente han utilizado lenguajes de programación como C para desarrollar estos sistemas, basados en arquitectura de microprocesadores o microcontroladores. En la actualidad existen

	formadas y a la extensa información disponible, lo convierten en un lenguaje ideal para el desarrollo de prototipos en sistemas embebidos.	desarrollo en estas plataformas.	plataformas más robustas para el desarrollo de sistemas embebidos en este lenguaje, pero la información publicada para generar I+D, con respecto a Python es muy escasa.
--	--	----------------------------------	--

Fuente: [17] [18]

Python al tener un alto nivel de penetración comercial en el entorno de sistemas embebidos durante los últimos años, y al tener una alta popularidad que radica principalmente en el grado de flexibilidad para ser operado, de los elementos con los cuales se soporta, y principalmente la compatibilidad con otros sistemas comerciales, lo hacen un lenguaje de programación ideal, para el desarrollo del presente trabajo de titulación.

4.7. ENTORNO RASPBERRY PI

La Raspberry Pi es una computadora de una sola placa creada por la Fundación Raspberry Pi, una organización benéfica formada con el propósito principal de reintroducir las habilidades informáticas de bajo nivel en los niños del Reino Unido y promover la enseñanza de las ciencias de computación. El objetivo era reactivar la revolución de la microcomputadora a partir de la década de 1980, que produjo una generación entera de programadores expertos. Incluso antes de que la computadora se lanzara a fines de febrero de 2012, estaba claro que la Raspberry Pi había ganado un gran número de seguidores en todo el mundo [19].

Su nombre, surgió de la combinación del deseo de crear una computadora alternativa basada en nombres de frutas (como Apple, BlackBerry y Albaricoque) y una inclinación al concepto original de una computadora simple que puede programarse usando Python (acortado a Pi) [19].

4.7.1. El Hardware de Raspberry Pi

En el corazón de Raspberry Pi está el potente Broadcom BCM28XX “*system on a chip*”⁵ en sus diversas versiones. El BCM28XX es similar al chip que se encuentra en casi todos los teléfonos inteligentes y decodificadores del mundo que usan arquitectura ARM. La CPU BCM28XX en la Raspberry Pi funciona a 700 MHz, 900 MHz, 1.2GHz y 1.4GHz

⁵ Sistema en un chip, que combina la mayoría de los elementos de un sistema en un solo circuito o chip integrado

dependiendo el modelo utilizado. Su rendimiento en las versiones iniciales (A, B y B+) es más o menos equivalente a una computadora Pentium II de 300 MHz que estaba disponible en el año 1999 [20].

La Raspberry Pi viene con 256 MB, 512 MB o 1 GB de RAM, según el modelo adquirido, ya que en la actualidad existen algunas versiones de la tarjeta.

4.7.2. Modelos de Raspberry Pi

Raspberry Pi tiene diversas variantes: el Modelo A, el Modelo B y en la actualidad el Modelo Pi Zero. El Modelo A, es una versión de bajo costo y la primera versión oficial, que desafortunadamente omite el chip concentrador USB. La Fundación Raspberry Pi también lanzó el Modelo B y B+ de la tarjeta que tiene puertos USB adicionales y resuelve muchos de los problemas de energía que rodean a los puertos del Modelo A y B respectivamente. Con respecto al modelo Pi Zero que fue lanzado en el año 2015, éste tiene un tamaño menor que una tarjeta Raspberry normal y es un 40% más potente que el primer modelo de Raspberry. A continuación se muestra una tabla comparativa de los diversos modelos de las tarjetas de desarrollo Raspberry Pi.

Tabla 5. Diferencias entre diversos modelos de Raspberry Pi.

CARACTERÍSTICAS	Modelos Versión 1			Modelo Versión 2	Modelos Versión 3		
	A	B	B+	B	B	B+	A+
Procesador	ARM 1176JZF-S a 700 MHz			900 MHz quad-core ARM Cortex A7	1.2GHz 64-bit quad-core ARMv8		1.4GHz 64-bit quad- core ARMv8
Juego de instrucciones	RISC de 32 bits				RISC de 64 bits		
Memoria	256MB	512 MB		1 GB (Compartidos con la GPU ⁶)			512 MB
Puertos USB	1	2	4				1
Puerto Ethernet	No	Si					No
Módulo WiFi	No				Si		
Módulo Bluetooth	No				Si		
Almacenamiento	SD Card		Micro SD				
No. Pines GPIO	26		40				
Salidas de Video	Conector HDMI rev 1.3 y 1.4						
Salidas de Audio	Conector de 3.5mm, HDMI						

Fuente: [El Autor]

⁶ Unidad de Procesamientos Gráficos (GPU), es un procesador especializado para funciones de visualización.

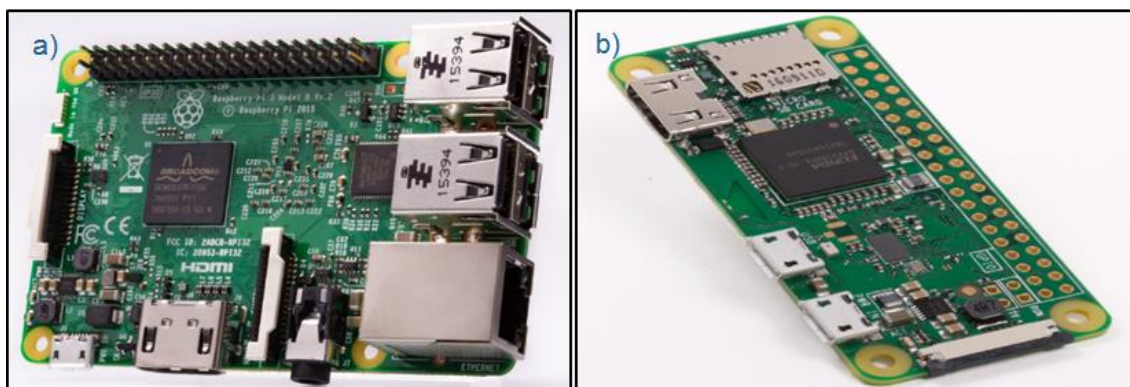


Figura 12. Tarjetas de desarrollo Raspberry Pi. a) Raspberry Pi 3 B. b) Raspberry Pi Zero.

Fuente: [42]

4.8. ESTÁNDAR IEEE 802.15.4/ZIGBEE

Los estándares de redes inalámbricas en la actualidad, están en todas partes. Frecuentemente se escucha hablar de *WiFi*, *Bluetooth* y tecnologías celulares. También es muy probable que quienes se encuentran sumamente relacionados con los avances tecnológicos, hayan escuchado hablar de ciertas tecnologías como: *Active RFID*⁷, *Wibree*⁸, *WiMAX*⁹ o USB inalámbrico. Entonces, ¿por qué *ZigBee*?

El estándar de red inalámbrica *ZigBee* se adapta a un mercado que simplemente no se llena con otras tecnologías inalámbricas, dentro del plano de las redes de sensores inalámbricos (WSN). Mientras que la mayoría de éstos estándares se esfuerzan por ir más rápido, *ZigBee* apunta a bajas tasas de datos. A medida que otros protocolos inalámbricos agregan más y más características, *ZigBee* apunta a una pequeña pila que se adapta a los microcontroladores de 8 bits. Si otras tecnologías inalámbricas buscan proporcionar cobertura de internet hasta la última milla o entregar medios de alta definición en tiempo real, *ZigBee* busca controlar dispositivos que forman parte de nuestro diario vivir (una luz de habitación, un termostato, un sensor de alarma, etc). Por otra parte si otras tecnologías inalámbricas están diseñadas para funcionar durante horas o quizás días con baterías, *ZigBee* está diseñado para funcionar durante prolongados tiempos [21].

⁷ Identificación de radio frecuencia activa (RFID) es un método de identificación automático e inalámbrico.

⁸ Wibree es una tecnología de radio de Nokia, para la conectividad local entre dispositivos electrónicos, dirigida a dispositivos que tienen una fuente de alimentación muy limitada.

⁹ Estándar de acceso inalámbrico de banda ancha, similar a *WiFi* pero con un alcance superior.

El mercado hacia el cual está orientado *ZigBee* se denomina “conexión y control de sensores inalámbricos” o, simplemente, “control inalámbrico”. De hecho, el lema de *ZigBee* es: “Control inalámbrico funcional” [21]. Para este mercado de control inalámbrico que tiene una serie de necesidades únicas, *ZigBee* es idealmente adecuado, porque sus características son:

- Altamente confiable ya que utiliza CSMA-CA (*Carrier Sense Multiple Access Collision Avoidance*) para acceso al canal de comunicación.
- Produce alto rendimiento y baja latencia para dispositivos de bajo ciclo de trabajo, un aspecto muy importante y adecuado para sensores y controles.
- Rentable
- Capaz de consumir muy poca potencia, ideal para equipos cuyo funcionamiento se basa en el uso de baterías.
- Permite el uso de ranuras de tiempo (*time slots*) para posibilitar aplicaciones de baja latencia.
- Muy seguro
- Un estándar global abierto

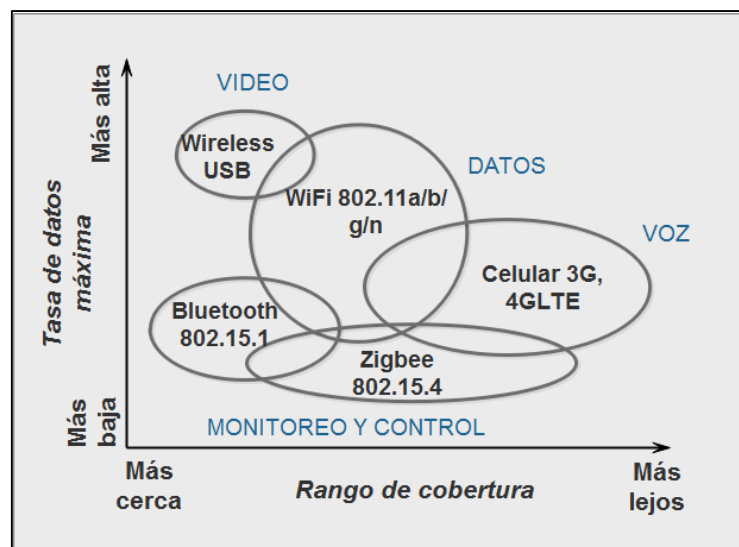


Figura 13. Diferentes estándares inalámbricos y posicionamiento de Zigbee en función del área de cobertura y la tasa de transmisión de datos.

Fuente: Gráfico adaptado de [21].

ZigBee fomenta las bajas tasas de datos. En este caso tan puntual, este protocolo tiene que ver con el monitoreo y control inalámbrico. Siendo *ZigBee* una tecnología apropiada para el control inalámbrico, logra una solución elegante que tiene sentido y se adapta muy bien

a una serie de mercados, convirtiéndose así en la elección adecuada, planteada para la solución de las necesidades requeridas en el presente proyecto de titulación.

4.8.1. Tipos de tráfico

Las aplicaciones basadas en *ZigBee* según su tráfico pueden clasificarse en uno de los siguientes tipos:

4.8.1.1. Datos periódicos (continuos)

La aplicación es quien define una tasa de datos. Se presenta cuando por ejemplo un sensor necesita transmitir la temperatura cada intervalo regular de tiempo [21].

4.8.1.2. Datos intermitentes (por eventos)

En esta situación la aplicación junto a otros estímulos externos al dispositivo definen la tasa de datos. Se manifiestan en un sistema domótico, cuando los interruptores de luces transmiten solo ante un cambio de estado. Mientras tanto están desconectados (comúnmente denominado en “modo *sleep*”) y consumiendo una cantidad de energía de batería mínima [21].

4.8.1.3. Datos periódicos con comunicación garantizada (GTS)

Existen aplicaciones de baja latencia que requieren comunicación libre de competencia por el canal. *Guaranteed time slot* (GTS) es un método de calidad de servicio que garantiza la atención por un cierto intervalo de tiempo (Δ_t) dentro de un período T llamado Supertrama [21] [22]. IEEE 802.15.4 dispone de un modo de trabajo denominado “modo baliza” que sirve como multiplexación temporal. Este aspecto fundamental se describirá con más profundidad en la sección 4.8.2.2.

4.8.2. Arquitectura del protocolo IEEE 802.15.4/ZigBee

Una de las formas comunes de establecer una red de comunicación (cableada o inalámbrica) es utilizar el concepto de capas de red. Cada capa es responsable de ciertas funciones en la red. Las capas normalmente pasan datos y comandos solo a las capas directamente arriba y debajo de ellas. Las capas del protocolo de red inalámbrica *ZigBee* se muestran en la Figura 14. Estas capas se encuentran basadas en el modelo de referencia básico de interconexión de sistemas abiertos (OSI). La división de un protocolo de red en

capas, tiene varias ventajas. Por ejemplo, si el protocolo cambia con el tiempo por alguna razón, es más fácil reemplazar o modificar la capa afectada por el cambio en lugar de reemplazar todo el protocolo. Además, al desarrollar una aplicación, las capas inferiores del protocolo son independientes de la aplicación y pueden obtenerse de un tercero, por lo que todo lo que debe hacerse es realizar cambios en la capa de aplicación del protocolo. Cada capa se conecta con las capas adyacentes a través de un SAP (*Service Access Point*). Un SAP es la ruta por la cual una capa superior requiere un servicio a una capa inferior [21].

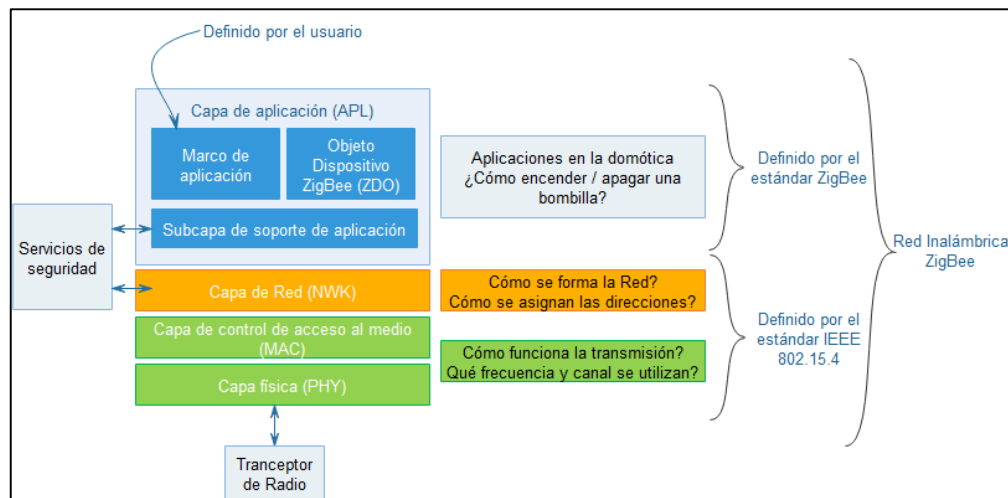


Figura 14. Capas del estándar IEEE 802.15.4 y ZigBee.

Fuente: [El Autor]

4.8.2.1. Capa Física

Como se muestra en la Figura 14, las dos capas de red inferiores están definidas por el estándar IEEE 802.15.4. Esta norma está desarrollada por el comité de normas IEEE 802 y se lanzó inicialmente en el año 2003. IEEE 802.15.4 define las especificaciones para las capas PHY y MAC de la red inalámbrica, pero no especifica ningún requisito para las capas más altas. Estas capas manejan operaciones de red de bajo nivel, las cuales se enumeran a continuación:

- Detección de la energía del receptor
- Indicador de calidad del enlace
- Evaluación del estado del canal
- Activación de funciones de direccionamiento y
- Transmisión/recepción de mensajes.

Existen tres bandas de frecuencia en la última versión de IEEE 802.15.4, que se lanzó en septiembre de 2006; 868-868.6 MHz (Banda de 868 MHz), 902-928 MHz (Banda de 915 MHz) y 2400- 2483.5 MHz (Banda de 2.4 GHz)

La banda de 868 MHz se utiliza en Europa para varias aplicaciones, incluidas las redes inalámbricas de corto alcance. Las otras dos bandas (915MHz y 2.4 GHz) son parte de las bandas de frecuencia industrial, científica y médica (ISM). La banda de frecuencia de 915MHz se usa principalmente en América del Norte, mientras que la banda de 2.4GHz se usa en todo el mundo.

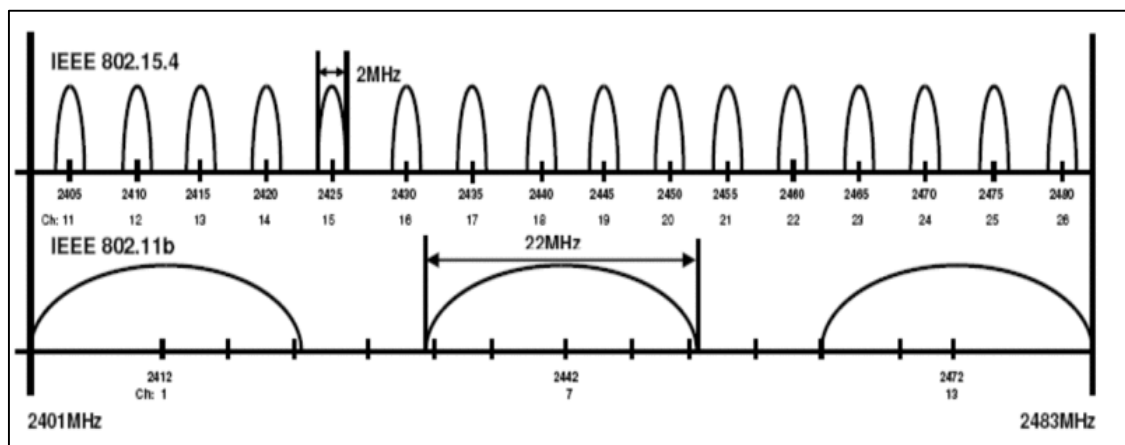


Figura 15. Distribución de los canales bajo el estándar IEEE 802.15.4 y la superposición existente con los canales Wi-Fi en la banda ISM de 2.4GHz.

Fuente: [23]

Tabla 6. Características de la capa física de IEEE 802.15.4.

Parámetros	868 MHz	915 MHz	2.4 GHz
Sensibilidad	-92 dBm	-92 dBm	-85 dBm
Nivel de entrada máximo del receptor	-20 dBm		
Rechazo a canal adyacente	0 dBm		
Rechazo a canal alternante	-30 dBm		
Potencia de salida	-3 dBm		
Número de canales	1	10	16
Espaciamiento entre canales	Canal Simple	2 MHz	5 MHz
Modulación	BPSK	BPSK	O-QPSK
Símbolos	Binarios	Binarios	16 ortogonal
Tasa de transmisión (Datos)	20 Kbps	40 Kbps	250 Kbps
Tasa de transmisión (Símbolos)	20 ksymbol/s	40 ksymbol/s	62.5 ksymbol/s
Tasa de transmisión (Chips)	300 Kchip/s	600 Kchip/s	2000 Kchip/s

Método de propagación	DSSS binario	DSSS binario	16-matrices Ortogonales
-----------------------	--------------	--------------	----------------------------

Fuente: [24]

Al ser esta capa, la más cercana al hardware; controla y se comunica directamente con el transceptor de radio. La capa PHY es responsable de activar la radio que transmite o recibe paquetes. También selecciona la frecuencia del canal y se asegura de que el canal no esté siendo utilizado actualmente por ningún otro dispositivo en otra red.

▪ Estructura general del paquete PHY

Los datos y los comandos se comunican entre varios dispositivos en forma de paquetes. La estructura general de un paquete se muestra en la Figura 16. El paquete PHY consta de tres componentes: el encabezado de sincronización (SHR), el encabezado de PHY (PHR) y la carga útil de PHY [21].

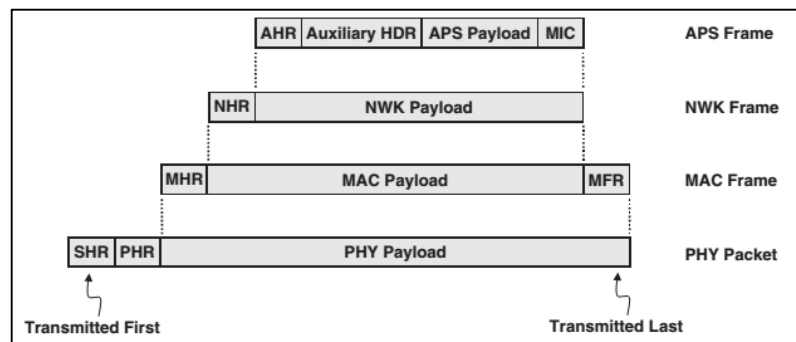


Figura 16. Estructura del paquete ZigBee.

Fuente: [13]

El SHR permite que el receptor se sincronice y se bloquee en el flujo de bits. El PHR contiene información sobre la longitud del marco, y la carga útil PHY es proporcionada por las capas superiores e incluye datos o comandos para el dispositivo receptor.

La trama MAC, que se transmite a otros dispositivos como carga útil PHY, tiene tres secciones. El encabezado MAC (MHR) contiene información como direccionamiento y seguridad. La carga útil MAC tiene un tamaño de longitud variable (incluida la longitud cero) y contiene comandos o datos. El slot final de la trama MAC (MFR) contiene una secuencia de verificación de cuadros (FCS) de 16 bits para la verificación de datos [13].

La trama NWK tiene dos partes: el encabezado NWK (NHR) y la carga útil NWK. El primero tiene direccionamiento a nivel de red e información de control. Respecto a la carga útil NWK es proporcionada por la subcapa APS. Dentro de la subcapa APS, el encabezado APS (AHR) tiene control de capa de aplicación e información de

direccionamiento. En cuanto al marco auxiliar (HDR auxiliar) contiene el mecanismo utilizado para agregar seguridad al marco y las claves de seguridad utilizadas. Estas claves de seguridad se comparten entre los dispositivos correspondientes y ayudan a desbloquear la información. La carga útil APS por su parte contiene datos o comandos, y finalmente el código de integridad del mensaje (MIC) es una función de seguridad en el marco APS que se utiliza para detectar cualquier cambio no autorizado en el contenido del mensaje [13].

La Figura 16 muestra que el primer bit transmitido, es el bit menos significativo (LSB) del SHR. El bit más significativo (MSB) del último octeto de la carga útil PHY se transmite en último lugar [13].

4.8.2.2. Capa de control de acceso al medio

Gestiona las transacciones de datos de RF entre dispositivos vecinos (punto a punto). La MAC incluye servicios como el reintento de la transmisión, la gestión de acuse de recibo, y técnicas de prevención de colisiones (CSMA-CA).

IEEE 802.15.4 implementa un método simple para permitir que múltiples dispositivos usen el mismo canal de frecuencia para su medio de comunicación. El mecanismo de acceso al canal utilizado es el acceso múltiple de detección de portadora con prevención de colisiones (CSMA-CA). En CSMA-CA, cada vez que un dispositivo desea transmitir, primero realiza una evaluación clara del canal (CCA) para garantizar que ningún otro dispositivo lo esté utilizando. Entonces el dispositivo comienza a transmitir su propia señal. La decisión de declarar claro o no un canal, se puede basar en la medición de la energía espectral en el canal de frecuencia de interés o en la detección del tipo de la señal de ocupación [13].

- **Evaluación de canal libre CCA (Clear Channel Assessment).**

Cuando un dispositivo planea transmitir una señal, primero entra en modo de recepción para detectar y estimar el nivel de energía de la señal en el canal deseado. Esta tarea es conocida como detección de energía (ED). En este modo, el receptor no intenta decodificar la señal, y solo se estima el nivel de energía de la señal. Si ya hay una señal en la banda de interés, el dispositivo en este modo de operación, no determina si se trata de una señal IEEE 802.15.4. Una forma alternativa de declarar que un canal de frecuencia está libre u ocupado es la detección de portadora (CS *Carrier Sense*). En CS, a diferencia

de la ED, se determina el tipo de señal de ocupación y, si esta señal es una señal IEEE 802.15.4, el dispositivo puede decidir considerar el canal ocupado, aun cuando la energía de la señal está por debajo de un valor definido por el usuario.

Existen 3 modos de operación del CCA [22].

- **Modo 1:** Se utiliza el nivel de energía y un umbral a partir del cual el canal está ocupado.
- **Modo 2:** Se utiliza el nivel de detección de portadora para determinar la ocupación del canal.
- **Modo 3:** Se utiliza la combinación AND u OR de los anteriores modos.
AND: La energía pasa del umbral Y la señal cumple con el estándar

OR: La energía supera el umbral O es censada una señal que cumple con el estándar.

Si el canal no está despejado, el dispositivo se apaga durante un período de tiempo aleatorio y lo intenta de nuevo. Los reintentos aleatorios se repiten hasta que el canal se aclare o el dispositivo alcance su número máximo de reintentos definido por el usuario [13].

Hay dos métodos para el acceso al canal: basado en la contención y libre de contención. En el acceso al canal basado en la contención, todos los dispositivos que desean transmitir en el mismo canal de frecuencia utilizan el mecanismo CSMA-CA, y el primero que encuentra el canal abierto comienza a transmitir. En el método libre de conflictos, el coordinador de PAN dedica un intervalo de tiempo específico a un dispositivo en particular. Esto se denomina un intervalo de tiempo garantizado (GTS). Por lo tanto, un dispositivo con un GTS asignado comenzará a transmitir durante ese GTS sin utilizar el mecanismo CSMA-CA [21].

▪ **Operación de la red PAN usando balizas**

Al utilizar un intervalo de tiempo garantizado la red PAN está operando bajo un modo denominado tramas de baliza, las cuales son tramas MAC especiales, que utilizan una estructura llamada supertrama. Las supertramas, que son opcionales en IEEE 802.15.4, están separadas por tramas de baliza. En la Figura 17 se muestran los tiempos de la supertrama.

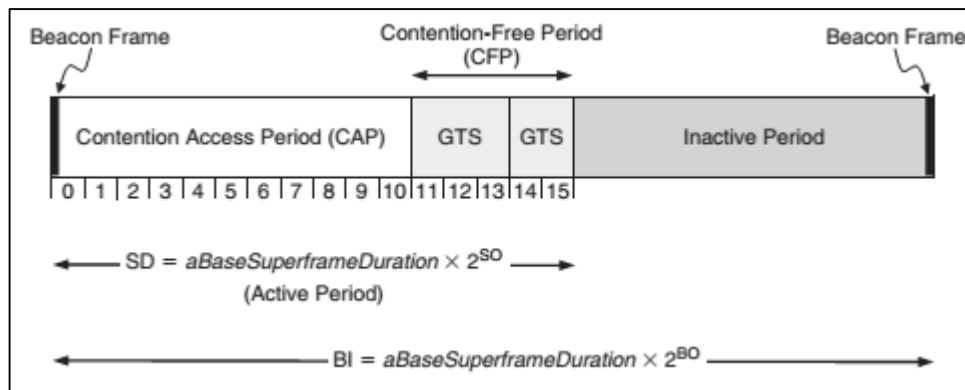


Figura 17. Estructura de la supertrama.

Fuente: [13]

Una supertrama se compone de tres tipos de períodos: Período de acceso en contienda (CAP), período libre de contiendas (CFP) y período inactivo. Los nodos que requieran transmitir durante el período CAP deben hacer uso de CSMA-CA para acceder a un canal que está disponible igualmente para todos los dispositivos. El primero que lo encuentre libre lo usará y lo tendrá disponible hasta que cese su transmisión. Si el dispositivo encuentra el canal ocupado, iniciará un período de espera aleatorio (*back off*) e intentará nuevamente usarlo [13].

Las tramas MAC de comando se deben transmitir durante el CAP. No existen garantías dentro del CAP de que el dispositivo pueda usar el canal en el momento en que lo necesita ya que está en competencia con otros dispositivos. En el período CFP, un dispositivo puede tener garantizada una ranura de tiempo (*time slot*), con lo que no necesita competir usando CSMA-CA. Esto es muy importante para aplicaciones de baja latencia. La suma de los períodos CAP y CFP constituye el período activo. Este se divide en 16 ranuras de idéntico tiempo. La baliza está en la primera ranura. Puede haber hasta 7 GTS en el CFP. Cada GTS puede durar una o más ranuras de tiempo.

La supertrama puede contener un período de inactividad. En este tiempo, los dispositivos pueden apagar los transceptores de radio para conservar la energía. Es lo que se conoce como nodo en modo “*sleep*”. El coordinador es quien define los períodos de la supertrama dando valores a las constantes que determinan el intervalo entre balizas (BI) y la duración de la supertrama (SD).

▪ CSMA-CA

Cuando un dispositivo está tratando transmitir, previamente verifica que el canal no esté en uso por otro dispositivo. Si está libre, inmediatamente comienza a transmitir. Hay transmisiones que se hacen sin verificación previa. Estas son

- Transmisión de balizas
- Transmisión durante el período CFP
- Transmisión después de haber dado ACK a un comando de pedido de datos.

El uso del CSMA-CA tiene en cuenta si se está trabajando con supertrama o no. Si es el primer caso, el tiempo activo se divide en 16 ranuras iguales, entonces el tiempo de *back off* debe ser alineado para que caiga en el CAP. Este caso se llama CSMA-CA ranurado. Cuando no se trabaja con supertrama, no se necesita sincronizar el *back off*. Este caso se denomina CSMA-CA no ranurado [22] [13].

▪ Estructura de Trama MAC

El IEEE 802.15.4 define cuatro estructuras de trama MAC [13]:

- Trama Baliza
- Trama de datos
- Trama Acuse de recibo (ACK)
- Trama de comando MAC

La trama baliza es utilizada por un coordinador para transmitir señales guía. Las señales guías se utilizan para sincronizar el reloj de todos los dispositivos dentro de la misma red. La trama de datos transmite información, las tramas de confirmación reconocen la recepción exitosa de una trama, y los comandos MAC se transmiten utilizando una trama de comando MAC. Todas las tramas se conforman de 3 partes: un encabezado (MHR), una carga útil (*payload*) y un pie (MFR). El encabezado contiene información sobre el tipo de trama, campos de direcciones y banderas de control. La carga útil tiene una longitud variable y contiene comandos, datos o nada (cero bytes). El MFR contiene una secuencia de chequeo (FCS) para verificar los datos basada en el clásico polinomio cíclico redundante (CRC). Las figuras 18, 19, 20 y 21 muestran los diferentes formatos que puede tener la trama MAC [13].

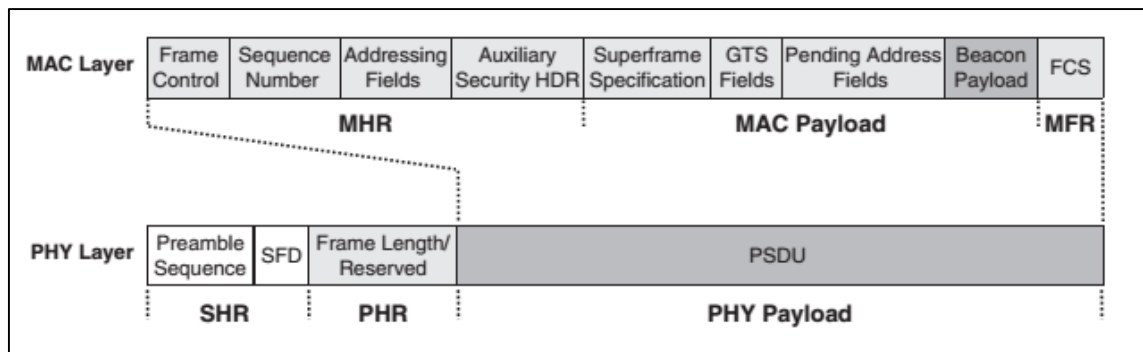


Figura 18. Estructura de la trama MAC baliza.

Fuente: [13]

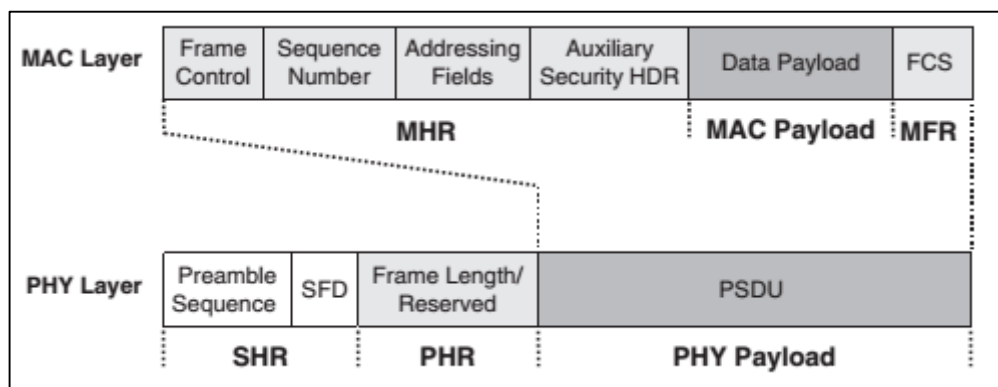


Figura 19. Estructura de trama MAC de Datos.

Fuente: [13]

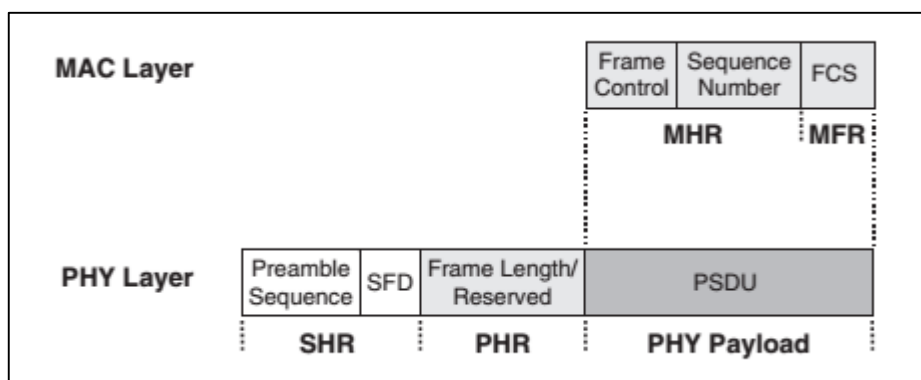


Figura 20. Estructura de la trama MAC de acuse de recibo.

Fuente: [13]

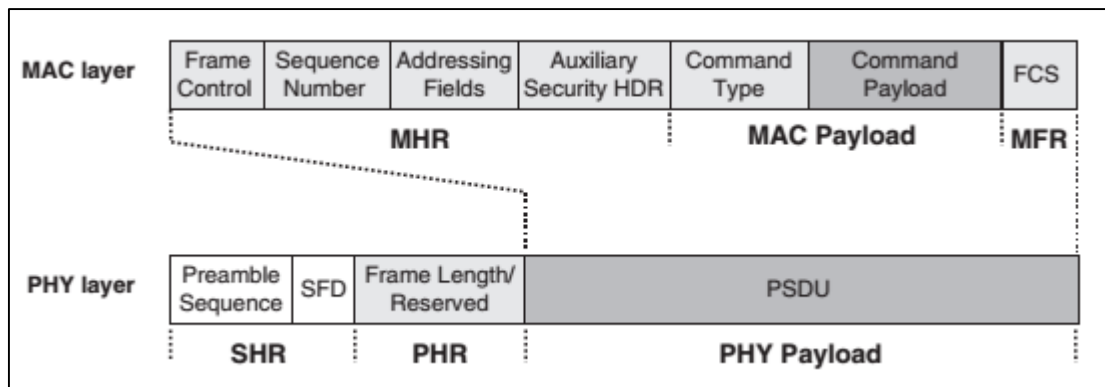


Figura 21. Estructura de trama MAC Comando.

Fuente: [13]

4.8.2.3. Capa de red ZigBee

La capa NWK se interconecta entre la MAC y la APL siendo la responsable de administrar la formación y el enrutamiento de la red. El enrutamiento es el proceso de selección de la ruta a través de la cual el mensaje se transmitirá a su dispositivo de destino. El coordinador *ZigBee* y los enrutadores son responsables de descubrir y mantener las rutas en la red. Un dispositivo final *ZigBee* no puede realizar el descubrimiento de rutas siendo el coordinador *ZigBee* o un enrutador quien realiza el descubrimiento de la ruta por el dispositivo final. La capa de red NWK de un coordinador *ZigBee* es responsable de establecer una nueva red y seleccionar la topología de la red (árbol, estrella o malla). El coordinador *ZigBee* también asigna las direcciones NWK a los dispositivos en su red [22] [13].

La capa de red del coordinador asigna direcciones de 16 bits a cada miembro de la PAN. Cada trama de red lleva un parámetro llamado radio que indica la cantidad de saltos máximos que esta puede llegar a realizar. Este parámetro se va disminuyendo en uno en cada salto. Cuando llega a cero, esa trama no será retransmitida a otro dispositivo. Existen 3 tipos de comunicación de mensajes: *broadcast*¹⁰, *multicast*¹¹ y *unicast*¹²:

- Un mensaje tipo *broadcast* tiene como destino a todo dispositivo que lo pueda recibir.
- Un mensaje *multicast* se envía solo a un grupo de dispositivos.
- Un mensaje *unicast* contiene la dirección de un único dispositivo.

¹⁰ Método de transmisión de uno a todos, envío de datos a todos los dispositivos al mismo tiempo.

¹¹ Método de transmisión de uno a varios, envío de datos a múltiples destinos simultáneamente.

¹² Envío de datos desde un único dispositivo transmisor a un único receptor.

- **Tipos de nodos ZigBee**

El estándar enumera 3 clases de nodos que se pueden encontrar dentro de una red: coordinador, ruteador y dispositivo final.

- **Coordinador.**

Por cada red PAN IEEE 802.15.4 establecida, debe existir uno y solamente un nodo coordinador dentro de ella. Dicho dispositivo actúa como nodo raíz en la topología árbol y es responsable de; Arranque de la red, configuración de los parámetros de red, admisión de nodos a la red y asignación de direcciones de red.

El coordinador requiere de un dispositivo de función completa (FFD) ya que requiere más potencia de cómputo. Es sumamente importante que la fuente de alimentación sea ininterrumpida y segura, ya que este dispositivo nunca entrará en modo “*sleep*” [22].

- **Ruteador.**

Es un nodo con funciones similares a las de un nodo coordinador, pero sin llegar a ocupar la totalidad de las funciones de dicho elemento. Su utilidad dentro de la red radica en que estos dispositivos permiten extender la cobertura de la red, al mismo tiempo que aumentan la confiabilidad con la creación de rutas adicionales de datos [22].

- **Dispositivo final.**

Estos nodos son aquellos en donde se genera la información del entorno a monitorear, y se comunican con un nodo ruteador o un nodo coordinador. Estos nodos tienen muy baja potencia de cómputo y suelen ser alimentados mediante baterías. Son dispositivos de funcionalidad reducida (RFD) según el estándar IEEE 802.15.4 [22].

- **Topologías**

ZigBee utiliza las topologías de IEEE 802.15.4 para transferencia de datos y agrega las topologías de árbol y de malla. Debido al alcance de cada nodo que se encuentra limitado, frecuentemente un paquete debe ser retransmitido varias veces por intermedio de los dispositivos ruteadores. Lo importante y destacable, es que el ruteo en cualquier topología usada se hace en la capa de red por lo tanto no es necesaria ninguna programación adicional en la capa de aplicación para que un dispositivo final se comunique con su

coordinador [13]. En la Figura 22 aparecen las topologías estrella, árbol y malla que son mayormente utilizadas dentro de una red *ZigBee*.

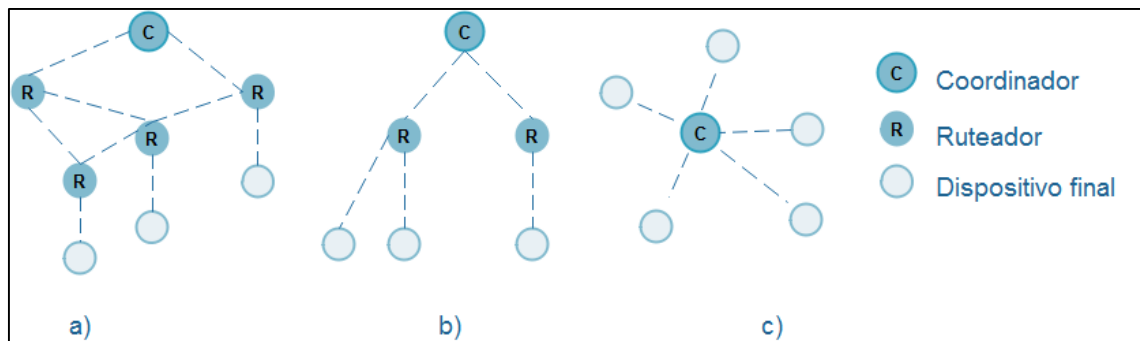


Figura 22. Diversas topologías de una WSN a) Topología malla, b) topología árbol, c) Topología estrella.

Fuente: [El Autor]

▪ Topología Estrella

Es la topología más sencilla presente en una red IEEE 802.15.4 sus principales características son:

- El elemento coordinador es el único que redirecciona paquetes
- El rango de cobertura de la red se encuentra limitado al rango de transmisión del coordinador
- El nodo coordinador tiene a su disposición uno o varios nodos hijos.
- La configuración de esta red es sumamente fácil.

▪ Topología Árbol

Una red punto a punto puede adoptar diferentes formas al definir restricciones en los dispositivos que pueden comunicarse entre sí. Si no hay restricción, la red punto a punto se conoce como una red bajo una topología de malla. Otra forma de red punto a punto *ZigBee* es una topología de árbol Figura 22 b). En este caso, un coordinador *ZigBee* (coordinador de la red PAN) establece la red inicial. Los enrutadores *ZigBee* forman las ramas y retransmiten los mensajes. Los dispositivos finales de *ZigBee* actúan como hojas del árbol y no participan en el enrutamiento de mensajes. Los enrutadores *ZigBee* permiten que la red crezca más allá de la red inicial establecida por el coordinador *ZigBee* [13].

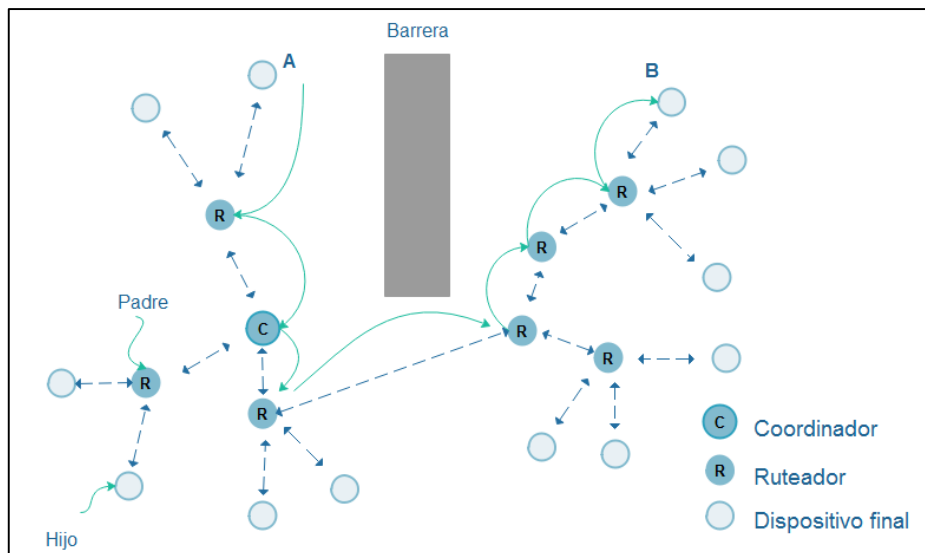


Figura 23. Topología tipo árbol de la red ZigBee.

Fuente: Gráfico adaptado de [13].

La Figura 23 ilustra un ejemplo de cómo la transmisión de un mensaje puede ayudar a ampliar el alcance de la red e incluso sortear barreras. En este caso, el dispositivo A necesita enviar un mensaje al dispositivo B, pero hay una barrera entre ellos. La topología en árbol permite retransmitir el mensaje alrededor de la barrera y alcanzar el dispositivo B. A esto se le llama *multi-hopping*¹³ porque un mensaje salta de un nodo a otro hasta que llega a su destino, al tiempo que se produce una cobertura más amplia, se produce una latencia potencialmente alta en la entrega de mensajes.

▪ Características de la topología árbol

- ✓ Los elementos router dentro de la red pueden tener nodos hijos
- ✓ Existe comunicación directa, solo entre los nodos padre-hijo.
- ✓ Existencia de ruteo jerárquico con un único camino posible entre dos nodos.

Tanto los elementos router y dispositivos finales se asocian con nodos presentes en la red. El nodo hijo es el que ingresa en la expansión de la red. Mientras que el nodo padre es el nodo que le ha dado al hijo acceso a la red.

▪ Propiedades de la relación padre-hijo

Las más destacables son:

- ✓ Solo pueden ser padres los nodos coordinadores o los nodos router.
- ✓ La jerarquía de una red *ZigBee* puede interpretarse como un árbol en donde el

¹³ Enrutamiento en múltiples saltos, es un tipo de comunicación en redes de comunicación inalámbricas en las que el área de cobertura de la red es mayor que el rango de los radios de nodos individuales.

coordinador es la raíz y los nodos finales son las hojas.

- ✓ En todo momento el nodo hijo tienen únicamente un padre
- ✓ Un hijo puede cambiar de padre [22].

Al momento de configurar la red se debe indicar los siguientes parámetros:

1. **El número máximo de hijos directos:** que se traduce en la máxima cantidad de ramas que puede tener cada nodo.
2. **Máxima profundidad de la red:** la cual hace referencia a la profundidad del árbol.
3. **Direccionamiento de nodos:** Cada nodo que ingresa a una red recibe una dirección de 16 bits. Esta dirección se usa en comunicaciones a nivel red. *ZigBee* ofrece una alternativa de asignación por defecto de direcciones a cada elemento que ingresa al árbol. La numeración depende de la configuración de hijos máximos y profundidad máxima con que se ha configurado el árbol [22].

Tal como lo indica la figura 23, lo interesante de la topología árbol dentro de la red *ZigBee*, es que cada ruteador sabe cómo encaminar cada mensaje hacia un destino determinado, comparando su propia dirección con la del destino. Lo cual elimina el problema de ruteo.

4.8.2.4. Capa de aplicación

La capa de aplicación (APL) es la capa de protocolo más alta en la red inalámbrica *ZigBee* y aloja los objetos de la aplicación; la subcapa APS (*Application Support*) y la ZDO (*ZigBee Device Object*). Los fabricantes desarrollan los objetos de la aplicación para personalizar un dispositivo que servirá para varias aplicaciones. Los objetos de la aplicación controlan y administran las capas de protocolo en un dispositivo *ZigBee*.

APS: define varios objetos de direccionamiento, incluidos perfiles, clústeres y puntos finales. ZDO: Define el rol del dispositivo dentro de la red, proporciona funciones de descubrimiento de dispositivos y servicios y capacidades avanzadas de administración de red [25].

El estándar *ZigBee* ofrece la opción de usar perfiles de aplicaciones en el desarrollo de una aplicación. Un perfil de aplicación es un conjunto de acuerdos sobre formatos de

mensajes específicos de la aplicación y acciones de procesamiento. El uso de un perfil de aplicación permite una mayor interoperabilidad entre los productos desarrollados por diferentes proveedores para una aplicación específica. Si dos proveedores utilizan el mismo perfil de aplicación para desarrollar sus productos, el producto de un proveedor podrá interactuar con productos fabricados por el otro proveedor como si ambos fueran fabricados por el mismo fabricante [13].

4.8.3. Seguridad en una red ZigBee

Dentro de una red inalámbrica, los mensajes transmitidos pueden ser recibidos por cualquier dispositivo cercano, incluido un dispositivo intruso. Existen dos preocupaciones principales de seguridad en una red inalámbrica. El primero es la confidencialidad de los datos. Ya que el dispositivo intruso puede obtener información confidencial simplemente escuchando los mensajes transmitidos. Para evitar este inconveniente cifrar los mensajes antes de la transmisión resolverá el problema de confidencialidad. Un algoritmo de cifrado modifica un mensaje utilizando una cadena de bits conocida como la clave de seguridad, y solo el destinatario podrá recuperar el mensaje original. El estándar IEEE 802.15.4 admite el uso del Estándar de cifrado avanzado (AES¹⁴) para cifrar sus mensajes salientes [13].

La segunda preocupación es que el dispositivo intruso puede modificar y reenviar uno de los mensajes anteriores, incluso si los mensajes están cifrados. Por lo tanto incluir un código de integridad del mensaje (MIC¹⁵) con cada trama saliente permitirá al destinatario saber si el mensaje ha sido cambiado en tránsito. Este proceso se conoce como autenticación de datos. Un dispositivo no autorizado no debería poder crear este MIC. Cuando recibe el mensaje el receptor calcula el MIC y si éste coincide con el que envía el transmisor, el mensaje se considera auténtico. El nivel de seguridad en el control se incrementa con el número de bits del MIC. *ZigBee* y 802.15.4 soportan MIC de 32, 64 y 128 bits.

¹⁴ Advance Encryption Standard, técnica de encriptación del National Institute of Standards and Technology (NTS)

¹⁵ Código de integridad de Mensaje generado mediante un método que conoce tanto el emisor como el receptor.

Una de las principales limitaciones en la implementación de funciones de seguridad en una red inalámbrica *ZigBee* son los recursos limitados. Los nodos son principalmente alimentados por batería y tienen una potencia de cálculo y un tamaño de memoria limitados. *ZigBee* como ya se ha mencionado, está dirigido a aplicaciones de bajo costo y es posible que el *hardware* en los nodos no sea resistente a la manipulación. Si un intruso adquiere un nodo de una red operativa que no tiene resistencia a la manipulación indebida, la clave real podría obtenerse simplemente desde la memoria del dispositivo. Un nodo resistente a la manipulación puede borrar la información confidencial, incluidas las claves de seguridad, si se detecta una manipulación indebida [13].

4.9. MICROCONTROLADORES

Un microcontrolador es un circuito integrado que en su interior posee toda la arquitectura de un computador (CPU, memorias RAM, EEPROM, y circuitos de entrada y salida), la función principal de este dispositivo es almacenar en su memoria interna, órdenes a ser ejecutadas, las cuales se definen mediante métodos de programación a través de varios lenguajes como; ENSAMBLADOR, C++, BASIC etc., destinados a estos fines.

Un microcontrolador puede ser capaz de realizar las tareas de muchos circuitos lógicos como compuertas AND, OR, NOT, NAND, conversores analógico-digital (A/D), digital-analógico (D/A), temporizadores, decodificadores, etc., simplificando todo el diseño a una placa de reducido tamaño y pocos elementos [26].

4.9.1. Estructura general de un microcontrolador

En la figura 24 se muestran los diversos componentes que definen a un microcontrolador.

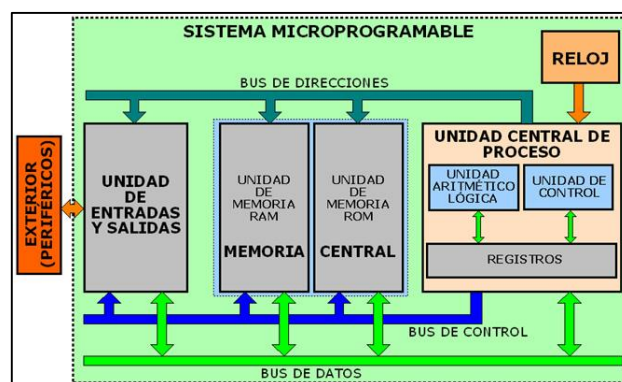


Figura 24. Estructura general de los componentes de un microcontrolador.

Fuente: [27]

4.9.1.1. Unidad central de procesamiento (CPU)

La información que ingresa y que sale desde un microcontrolador es procesada a través de la CPU, la cual a su vez se encuentra conformada varias unidades muy importantes las cuales son; Unidad Aritmética Lógica (ALU), Unidad de Control, Registros y Buses.

La ALU, es la encargada del tratamiento de la información que ingresa y sale en la CPU, a través de operaciones matemáticas y lógicas. Mientras que la unidad de control, considerada como la parte más importante dentro de la unidad de proceso, cumple con funciones específicas como: verificar la correcta ejecución de instrucciones, manipular registros, buses, ALU's, etc.

Los Registros son espacios de memoria de poca capacidad, dónde se almacenan datos procesados por instrucciones o datos procedentes de una memoria externa. El total de bits empleados en los registros de datos, determinan aspectos como la velocidad de ejecución y la capacidad de cómputo.

Los buses son elementos esenciales dentro de la comunicación interna del procesador, dado que son los únicos medios por los cuales se puede transferir información generada por los componentes internos hacia el exterior o viceversa. Según el tipo de información que transportan se pueden identificar tres tipos de buses.

- ✓ **Bus de dirección:** transporta bits o combinaciones binarias que seleccionarán la posición de la memoria o el registro de entrada/salida en el que se desea leer o escribir. Al tratarse de un bus unidireccional tal como lo muestra la Figura 24 la CPU es la única que puede utilizarlo.
- ✓ **Bus de datos:** solamente transporta información por procesar o procesada.
- ✓ **Bus de control:** es el encargado de gestionar el proceso de lectura/escritura y la operatividad de cada una de las partes que conforman el procesador. Por él circulan el conjunto de señales necesarias para la correcta coordinación de todos los elementos del sistema, tales como: órdenes de lectura o escritura, inhabilitación (desactivación) de un dispositivo, etc. [28]

4.9.1.2. Periféricos de entrada/salida

Unidad que emplea el microcontrolador para comunicarse con el exterior, permitiendo la introducción y la extracción de información al sistema. Estas unidades consisten generalmente en registros que, accionados por los buses de control y direcciones, almacenan la información suministrada por el bus de datos.

4.9.1.3. Memoria de programa

Espacio que establece el microcontrolador para almacenar de forma permanente todas las instrucciones que posea el programa de control y la información necesaria para el funcionamiento del sistema [27]. Existen algunos tipos de memoria que cumplen con los requisitos para realizar este tipo de funciones:

- **Memoria ROM:** Memoria solo de lectura, un tipo de memoria no volátil cuyo contenido es grabado únicamente en la fabricación del chip.
- **Memoria PROM:** Memoria solo de lectura programable, es un tipo de memoria que permite el almacenamiento únicamente de un programa controlador, factibles en sistemas donde el programa controlador no requiera ser modificado en un futuro.
- **Memoria EPROM:** Memoria solo de lectura programable y borrrable, este tipo de memoria permite almacenar y borrar un programa controlador las veces que sean necesarias. En este tipo de memorias la programación se realizaba mediante el programador de microcontroladores y el borrado se hacía con una fuente de luz ultravioleta.
- **Memoria EEPROM:** Memoria solo de lectura programable y eléctricamente borrrable, este tipo de memoria es borrrable a través de pulsos eléctricos.
- **Memoria FLASH:** es un tipo de memoria que ha sido producida como una mejora de las memorias EEPROM, teniendo en cuenta el factor capacidad de almacenamiento.

4.9.1.4. Memoria de Datos

Espacio establecido para el almacenamiento de datos que serán utilizados por el programa principal. Este tipo de memoria debe permitir la lectura y escritura de los datos, los cuales varían continuamente según la ejecución del programa [27].

4.9.1.5. Entradas y salidas de propósito general

Elementos del microcontrolador que permiten el ingreso de datos generados por dispositivos externos y el envío de datos ya procesados por la CPU hacia el exterior. Son conocidos como puertos bidireccionales (Entrada/Salida), dispuestos o agrupados en puertos con direcciones de 8 bits generalmente, la denotación de estos puertos se realiza con las primeras letras del alfabeto, por ejemplo: Puerto A, Puerto B, Puerto C, etc. [28].

Dado el hecho de que se requiere una gran cantidad de puertos de E/S, se ha definido que las entradas y salidas de propósito general compartan los pines con otros periféricos, así que dependiendo de la programación y de la configuración de registros estos periféricos entrarán en funcionamiento.

4.9.1.6. Puertos de comunicación

La información procedente desde y hacia el microcontrolador puede darse a través de diversos protocolos de comunicación de entre los cuales se destacan:

- **Puerto Serie:** el cual es mayormente probable de encontrar en cualquier clase de microcontrolador, ya que a través de este puerto y en conjunto con una interfaz RS-232 es posible establecer una comunicación con una PC. En la infraestructura interna del microcontrolador se puede encontrar este tipo de puertos con los nombres de UART (Transmisor-Receptor Asíncrono Universal) o USART (Transmisor-Receptor Asíncrono Síncrono Universal).
- **Puerto SPI:** *Serial Peripheral Interface*, es un tipo de puerto de comunicación basado en la comunicación serie maestro-esclavo la cual brinda una interfaz muy sencilla debido a que presenta pocos hilos de comunicación, convirtiéndolo en una excelente opción para la comunicación entre microcontroladores o periféricos externos con microcontroladores.
- **Puerto de comunicación serial síncrona I²C:** *Inter Integrated Circuit* o interconexión de circuitos integrados, solamente necesita 2 líneas para transmitir

y recibir datos; SDA para señal y SCL para la señal de reloj. Esta forma de comunicación utiliza una sincronía con un tren de pulsos que viaja en la línea SCL, de tal forma que en los flancos negativos se revisan los datos RX o TX [26]. Cada dispositivo conectado al bus tiene un código de dirección seleccionable mediante *software*, por lo que existe una relación permanente *Master/Slave*. El *Master* es el dispositivo que inicia la transferencia en el bus y genera la señal de reloj (SCL), y el *Slave* es el dispositivo direccionado, sin embargo cada dispositivo reconocido por su código (dirección), puede operar como transmisor o receptor de datos, ya que la línea (SDA) es bidireccional.

- **USB:** Universal Serial Bus, es un estándar industrial que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica entre computadores, periféricos y dispositivos electrónicos. La interfaz USB dentro de un microcontrolador brinda grandes beneficios pero funcionalmente genera una relativa complejidad.
- **ETHERNET:** dado el creciente aumento de dispositivos electrónicos conectados a la red, hoy en día se hace indispensable contar con microcontroladores que incluyan periféricos externos que puedan ser accesibles desde la red.

4.9.1.7. Registros

Del mismo modo que los registros de la CPU, los registros son un pequeño espacio de memoria a partir de los cuales se toman los datos para varias operaciones que realizan los demás circuitos del microcontrolador. Éstos almacenan temporalmente los resultados intermedios de las instrucciones ejecutadas. Se conforman de biestables colocados de tal forma que pueden almacenar la información en un momento determinado [27].

4.9.2. Arquitectura de un microcontrolador

El modelo de bloques mostrado en la Figura 24, es válido prácticamente para todos los microcontroladores del mercado. No obstante, si se profundiza más en la arquitectura, se puede dar cuenta que para cualquier sistema micro-programable basado en CPU, sea un ordenador con un microprocesador o un microcontrolador hay dos arquitecturas distintas relacionadas con el uso y distribución de la memoria, la arquitectura Von Neumann y la arquitectura Harvard [28].

4.9.2.1. Arquitectura Von Neuman

Tipo de arquitectura en la que la unidad central de proceso, se encuentra interconectada a una memoria central, dicha memoria es a su vez la unión de dos tipos de memoria; la memoria de datos y la memoria de instrucciones. La interconexión de la CPU con la memoria central se realiza por medio de un conjunto de buses dedicados los cuales son: Bus de control, bus de direcciones y uno de datos e instrucciones [28].

El principal inconveniente que posee esta arquitectura, es la diferencia en los tiempos de respuesta, ya que al contar con un único bus para datos e instrucciones, se limita los accesos a la memoria central, generando de esta forma el denominado cuello de botella, en donde los accesos a la memoria central se realizan byte por byte. Adicionalmente teniendo en cuenta la limitación de la longitud de las instrucciones por el bus de datos en este tipo de arquitectura, provoca que el microcontrolador realice varios accesos a la memoria para buscar instrucciones complejas. En la figura 25 se muestra un diagrama de bloques que representa la arquitectura Von Neuman.

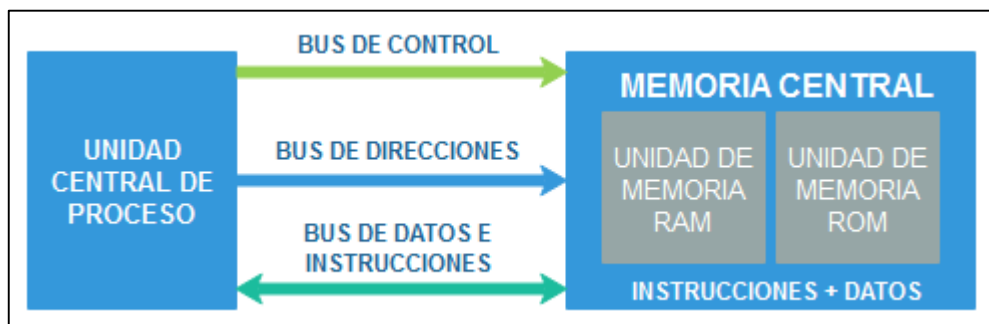


Figura 25. Diagrama de bloques de la arquitectura Von Neuman.

Fuente: Gráfico adaptado de [27].

4.9.2.2. Arquitectura Harvard

En este tipo de arquitectura la unidad central de procesos CPU se encuentra conectada a dos memorias (memoria de instrucciones y memoria de datos) a través de buses independientes. Una de las memorias contiene solamente las instrucciones del programa (Memoria de Programa) y la otra, sólo almacena datos (Memoria de Datos) [27] [28].

Al tratarse de buses totalmente independientes, permiten que la CPU pueda acceder de forma simultánea a la memoria de datos e instrucciones, siendo independientes estos

buses pueden tener distintos contenidos en la misma dirección y también distinta longitud, lo que da como resultado la optimización del uso de la memoria en general.

Las ventajas de esta arquitectura son:

- ✓ El tamaño de las instrucciones no está relacionado con el de los datos, de esta forma puede ser optimizado para que cualquier instrucción ocupe una sola posición de memoria de programa, logrando así mayor velocidad y menor longitud de programa.
- ✓ El tiempo de acceso a las instrucciones puede superponerse con el de los datos, logrando una mayor velocidad en cada operación.

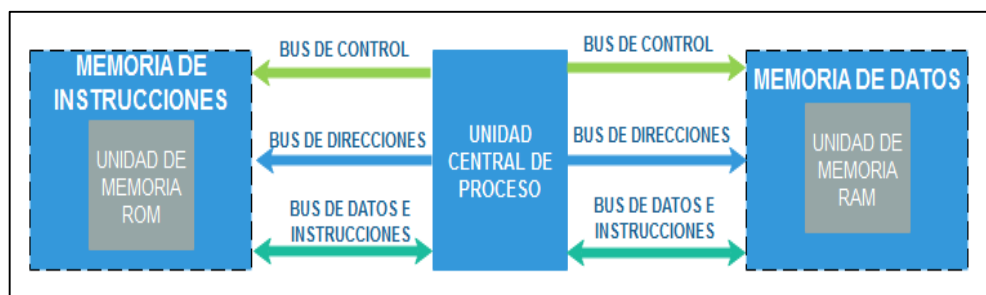


Figura 26. Arquitectura Harvard de un microcontrolador.

Fuente: Gráfico adaptado de [27]

4.10. ESTRUCTURA WEB

4.10.1. Modelo básico de una comunicación Web

Dentro de la arquitectura *Web*, el modelo de comunicación entre el cliente y el servidor se establece mediante una conexión de internet. El intercambio de información desde la capa de aplicación es decir desde un navegador hacia el servidor, se hace mediante el uso del protocolo HTTP. La comunicación a breves rasgos sigue el siguiente formato:

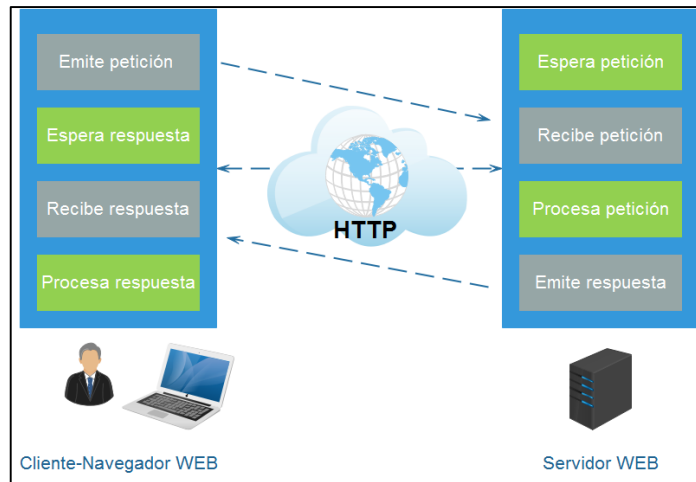


Figura 27. Modelo cliente servidor en una comunicación Web.

Fuente: [El Autor]

El usuario interactúa con una serie de aplicaciones *Web*, por medio de un navegador. La actividad del usuario, hace que se generen peticiones al servidor, donde se aloja la aplicación *Web* y se usa una base de datos que recoge toda la información relacionada con la misma. El servidor procesa la petición y devuelve la respuesta al navegador que la presenta al usuario [29].

Por lo tanto, el sistema se compone de tres elementos muy importantes:

- ✓ El navegador, que presenta la interfaz de usuario
- ✓ La aplicación, encargada de realizar todas las operaciones lógicas de la misma.
- ✓ La base de datos, que almacena la información relacionada con la aplicación.

Dicha distribución se conoce como modelo o arquitectura de tres capas, representado en la Figura 28.

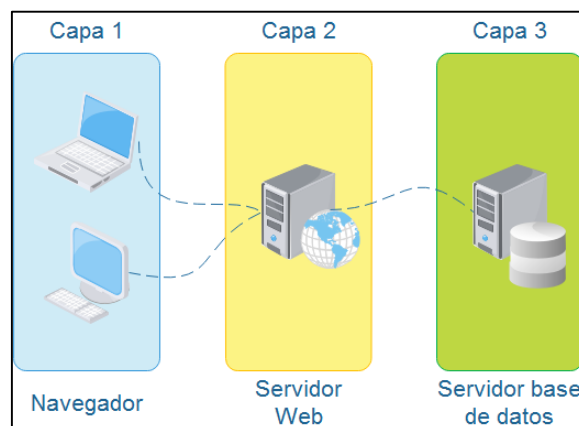


Figura 28. Arquitectura de tres capas.

Fuente: [El Autor]

Cada capa representa una funcionalidad misma que se explica a continuación:

- **Nivel de presentación (capa cliente)**

Encargada de generar la interfaz de usuario. Obtiene los datos de usuario, envía estos datos a la capa intermedia y presenta los resultados que procedan de esta [29].

- **Nivel de negocio (capa intermedia)**

Capa en donde se encuentra el verdadero núcleo de la aplicación *Web*. Contiene toda la lógica que modela los procesos de negocio y en donde se realiza todo el procesamiento de la información para atender los requerimientos del usuario. Es la encargada del procesamiento de datos del usuario y de la generación y envío de respuestas a la capa cliente [29].

- **Nivel de administración de datos**

Esta capa es la encargada de hacer constante toda la información. Suministra y aloja información requerida por el nivel de negocio [29]. Puede estar conformada por uno o más gestores de bases de datos que pueden ser del tipo de bases de datos relacionales (SQL¹⁶) o las actualmente existentes no relacionales (NoSQL¹⁷), las cuales realizan todo el almacenamiento de los datos, esta capa recibe solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

4.10.2. Diseño Web

El diseño de un sitio *Web* es muy importante. Hoy en día gran parte del diseño es manejado por el uso de *frameworks*¹⁸ que han facilitado el diseño de las páginas *Web*. Un *framework* es un marco de trabajo; asociándolo con el desarrollo de *software*, es una estructura conceptual compuesta de componentes personalizables e intercambiables, que puede servir de base para la organización y desarrollo de *software* o aplicaciones. Un *framework* puede ser considerado como como una aplicación genérica e incompleta, dentro de la cual, puede incluirse soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto [30].

¹⁶ Structured Query Language (Lenguaje de Consulta Estructurado)

¹⁷ Not only SQL (No sólo SQL) término empleado desde la llegada de la web 2.0

¹⁸ Conjunto de componentes que conforman un diseño reutilizable, que facilita y agiliza el desarrollo de sistemas Web

Hoy en día se puede encontrar un sin número de *frameworks* disponibles, los cuales junto con la combinación de HTML5¹⁹ y CSS²⁰ permite crear diseños *Web* profesionales que pueden ser personalizados rápidamente. El uso de *frameworks* ha proporcionado un nivel de diseño más eficiente

4.10.3. Programación Web

Las aplicaciones *Web* en la actualidad cuentan con un gran respaldo y sobre todo tienen mucho éxito, principalmente porque solo requieren de un navegador *Web* independientemente del sistema operativo y no es necesario la instalación de ningún *software* en los equipos terminales de los miles de usuarios que las utilizan. De esta forma, una aplicación *Web* puede ser ejecutada en múltiples plataformas. En la actualidad existen diferentes lenguajes de programación para desarrollar en la *Web*, estos han ido surgiendo debido a las tendencias y necesidades de las plataformas [29].

A medida que crecen las demandas por parte de los usuarios se han dado soluciones a través de lenguajes de programación para los sitios *Web* dinámicos; los mismos que permiten interactuar con los usuarios al mismo tiempo que se hace uso de sistemas de bases de datos. Los lenguajes de programación más utilizados son: HTML, PHP, JAVA SCRIPT, VISUAL BASIC SCRIPT, GOOGLE DART y Librerías como jQuery, Bootstrap, etc [29].

4.10.4. Bases de datos

Las bases de datos son esenciales en cualquier desarrollo *Web* ya que es el espacio en donde se almacena la información relevante y principalmente los datos, como se mostró en un inicio en el modelo de tres capas. En la actualidad existen dos tipos de base de datos, las denominadas bases de datos relacionales o SQL quienes son el modelo mayormente utilizado para diseñar bases de datos, y las no relacionales o NoSQL quienes surgen a partir de la llegada de la Web 2.0 cuando se produjo un crecimiento exponencial de datos. Momento en el cual empiezan a aparecer los primeros problemas de la gestión de toda esa información almacenada en bases de datos relacionales [29].

¹⁹ Hyper Text Markup Language en su quinta versión, usado para estructurar y presentar el contenido para sitios Web

²⁰ Hojas de Estilo en Cascada (CSS) es un lenguaje de estilo de hojas usado para describir la presentación de las páginas web

4.10.4.1. Bases de datos relacionales (SQL)

Este tipo de base de datos, es una colección de datos relacionados. Al hacer mención a los datos, se hace referencia a hechos conocidos que pueden llevarse en un registro y que tienen un significado explícito. Adicionalmente, una base de datos puede tener cualquier tamaño y complejidad, la cual puede administrarse mediante un sistema de gestión de base de datos (SGBD). Las bases de datos relacionales son aquellas que cumplen con el modelo relacional, fundamentado en el uso de relaciones. Teniendo como premisa que una relación no es más que un vínculo entre dos entidades de la base de datos. Este tipo de base de datos, busca describir el mundo real a través de tablas, al usar las relaciones de dependencia de los datos que se establecen entre sí, al igual que sus restricciones [31].

Los componentes básicos de una base de datos relacional son:

- **Tablas:** Una base de datos de este tipo puede contener una o más tablas en donde se estructuran los datos. Una tabla no puede repetirse dentro de una misma base de datos.
- **Columnas:** Subdivisiones correspondientes que conforman la tabla.
- **Registros:** Las tablas se conforman por un conjunto de registros denominados filas o tuplas.
- **Relaciones:** Son las relaciones existentes entre tablas que se establecen mediante claves primarias y claves ajenas.
- **Clave primaria:** Todo registro posee una clave primaria, la cual identifica unívocamente a un registro.
- **Clave ajena:** Son referencias colocadas en las tablas hijas, producto de las relaciones existentes entre tablas. Contienen el mismo valor que la clave primaria del registro padre.

Siendo el modelo relacional un modelo matemático de datos, éste se basa en la lógica de predicados y la teoría de conjuntos [31]. Por lo cual se puede hacer uso de los principios del álgebra relacional y el cálculo relacional para operar con los datos almacenados. Cabe mencionar que este tipo de bases de datos, presentan diversos problemas cuando se utilizan como sistema de almacenamiento, para extensas cantidades de datos manejadas

en los sistemas *Big Data*²¹, por lo que es necesario el estudio y el uso de otro tipo de base de datos que se describe a continuación.

4.10.4.2. Bases de datos NoSQL

Durante los últimos años, una gran variedad de bases de datos NoSQL han surgido, las cuales han sido creadas por compañías principalmente para cubrir sus propias necesidades, debido a los problemas generados en las bases de datos relacionales, principalmente con temas como escalabilidad, rendimiento, mantenimiento, etc.

Las bases de datos NoSQL son sistemas de almacenamiento de información que no cumplen con el esquema tradicional entidad–relación. Tampoco utilizan una estructura de datos en forma de tabla donde se van almacenando los datos sino que para el almacenamiento hacen uso de otros formatos como clave–valor, mapeo de columnas o grafos [32]. Las principales ventajas de los sistemas NoSQL se pueden destacar:

- **Escalabilidad horizontal:** que se refiere a la facilidad de añadir, eliminar o realizar operaciones con más nodos (elementos de *hardware* de sistema), sin afectar el rendimiento.
- **Habilidad de distribución:** relacionada a la escalabilidad horizontal, pero con más énfasis en el soporte, teniendo en cuenta la capacidad de replicar y distribuir los datos sobre los servidores.
- **Uso eficiente de recursos:** pues aprovechan las nuevas tecnologías, y los sistemas distribuidos en general.
- **Manejo de grandes cantidades de datos:** Debido a que utilizan una estructura distribuida mediante el uso de tablas hash (bases de datos clave-valor).
- **Consultas simples:** las consultas requieren menor cantidad de operaciones, por lo tanto se gana en simplicidad y eficiencia [32] [33].

Dependiendo de la forma en que se almacena la información, se encuentran varios tipos de bases de datos NoSQL, las más utilizadas son:

- **Bases de datos clave – valor:** que es el tipo de base de datos NoSQL más popular, y sencilla en cuanto a la funcionalidad. Aquí cada elemento se identifica por una

²¹ Término en evolución, que describe un gran volumen de datos estructurados, semiestructurados y no estructurados, que pueden ser extraídos para utilizar su información en proyectos de aprendizaje automático y aplicaciones de análisis avanzados.

llave única, permitiendo la recuperación de información de forma inmediata. Cassandra, BigTable o Hbase son ejemplos de este tipo [32].

- **Bases de datos documentales:** En este tipo de base de datos la información se almacena como un documento, utilizando para ello una estructura JSON²² o XML²³ lo que permite realizar consultas más avanzadas sobre el contenido del documento. MongoDB o CouchDB son ejemplos de este tipo.
- **Bases de datos en grafo:** Aquí la información se representa como nodos de un grafo y sus relaciones con las aristas del mismo. Una base de datos orientada a grafos, en términos generales, es cualquier sistema de información donde cada elemento tiene un puntero directo hacia sus elementos adyacentes, es decir, no sería necesario realizar consultas mediante índices [33]. Ejemplos de este tipo son Neo4j, InfoGrid o Virtuoso.
- **Base de datos orientada a objetos:** Tipo de base de datos en que la información se representa mediante objetos, al igual que se lo hace o son representados en los lenguajes de programación orientado a objetos como JAVA, C, PYTHON etc. Zope, Gemstone Db4o son ejemplos de este tipo.

Algunas de las diferencias que destacan entre los sistemas NoSQL y los sistemas SQL se resumen en la siguiente tabla.

²² Notación de Objetos de JavaScript, es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, al igual que para las máquinas lo es interpretarlo y generarlo.

²³ eXtensible Markup Language (Lenguaje de Marcas Extensible). Se trata de un metalenguaje (un tipo de lenguaje que se utiliza para decir algo acerca de otro).

Tabla 7. Diferencias existentes entre una base de datos del tipo SQL y las NoSQL.

Diferencia entre las bases de datos Relacionales y las no Relacionales	
Bases de datos SQL	Base de datos NoSQL
Hacen uso del lenguaje SQL como lenguaje de consultas	No utilizan SQL como lenguaje de consultas
Utilizan estructuras fijas como tablas, para el almacenamiento de datos	No utilizan la estructura de tablas para el almacenamiento de datos, en vez de ello hace uso de sistemas clave-valor, objetos o grafos.
Suelen estar centralizadas en una única máquina, o en ciertos casos en una estructura master-esclavo.	Se encuentran presentes en una arquitectura distribuida, en donde la información puede estar compartida en varias máquinas mediante el mecanismo de tablas Hash distribuidas.
Permite el uso de JOINS ²⁴ utilizados para combinar filas de dos o más tablas basándose en un campo común entre ellas.	No suelen permitir el uso de JOINS debido al extenso volumen de datos.

Fuente: [El Autor]

²⁴ Un JOIN se produce cuando dos o más tablas se juntan en una sentencia SQL.

5. MATERIALES Y MÉTODOS

5.1. Materiales

Tabla 8. Lista de materiales.

Descripción	Cantidad	Unidades
Sección Hardware		
Computador portátil	1	[U]
Raspberry Pi 3	1	[U]
Memoria SD 32 GB	1	[U]
Raspberry Pi Cámara	1	[U]
Cable USB 2.0 tipo A – tipo B mini de 5 pines	1	[U]
Xbee Explorer USB	1	[U]
Xbee S2C	7	[U]
Programador PICKIT 3	1	[U]
PIC18F25K20	6	[U]
Estaño	4	[m]
Pasta de soldar	1	[U]
Baquelita para PCB (tamaño A4)	0.06237	[m ²]
Sensor PIR	3	[U]
Sensor de humo MQ-2	1	[U]
Sensor magnético	1	[U]
Botón de pánico	1	[U]
Baterías de 9V	6	[U]
Adaptadores de voltaje DC (9-12V)	6	[U]
Cajas sobrepuestas Dexon	6	[U]
Componentes electrónicos varios (Revisar Anexo II)	---	---
Sección software		
Python 3.5 o superior (3.5 Usada)	1	[U]
Software XCTU V 6.3.5	1	[U]
Software PROTON IDE V 2.0.3.3 + Proton BASIC Compiler V 3.5.9.3	1	[U]
Software Angular	1	[U]
NODE JS 10.X	1	[U]
Software Putty	1	[U]
Software Pickit3 Programmer V 3.10.00	1	[U]
Software EasyEDA Online PCB design & circuit simulator	1	[U]

Fuente: [El Autor]

5.2. Métodos

En el apartado anterior se dio una descripción de los diferentes conceptos teóricos que involucran el diseño del sistema, así como también de los elementos de adquisición y transmisión de datos que pueden ser adaptados al marco referencial de la presente propuesta alternativa. Antes de proseguir en la explicación del desarrollo del prototipo se detallan los límites del diseño en base a los objetivos del presente trabajo de investigación. Principalmente, ya que el objetivo general está enfocado al diseño de un sistema de seguridad empleando redes de sensores inalámbricas bajo el estándar IEEE 802.15.4 y el uso de herramientas de desarrollo libre, se plantea la solución en función del *software* y

hardware disponible en la actualidad, en virtud de esto, el sistema que se describirá en el presente apartado contemplará los bloques funcionales del sistema que se muestra en la Figura 29.

5.2.1. Descripción general del sistema.

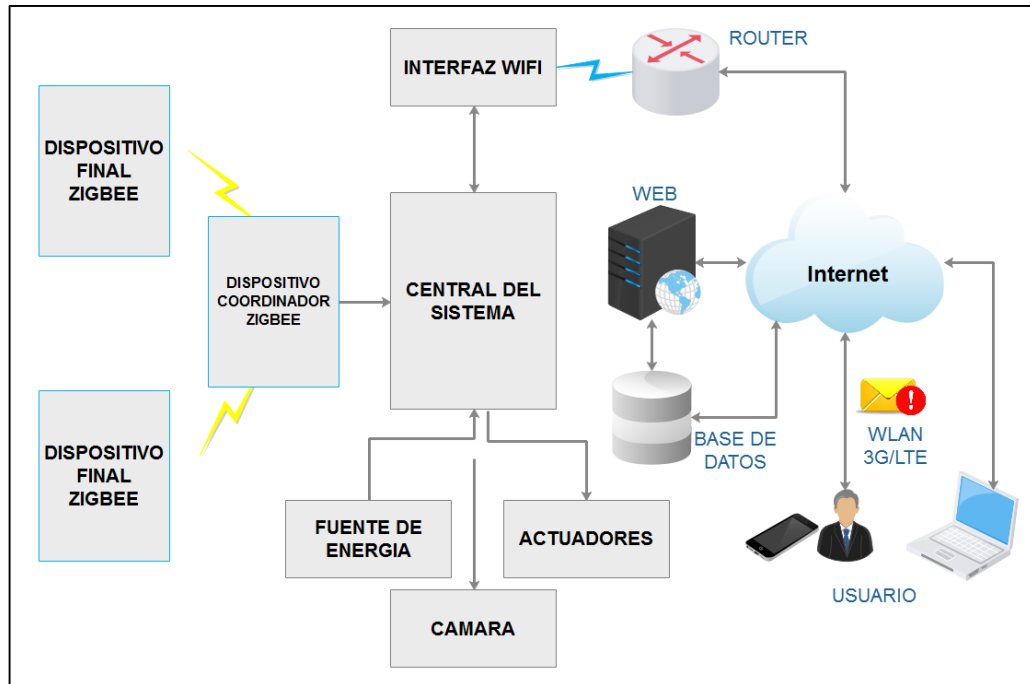


Figura 29. Diagrama de bloques del sistema integral de seguridad propuesto.

Fuente: [El Autor]

El sistema de seguridad hace uso de dispositivos RF que operan con tecnología IEEE 802.15.4/ZigBee, cada uno de los cuales se encuentra conformado con un tipo de sensor específico, el cual define una zona. Los sensores implementados, fueron previamente analizados y en base a ello, se utilizaron para definir el sistema de seguridad, dichos elementos se comunican con la central del sistema, la cual concentra la red IEEE 802.15.4 por medio de un dispositivo *ZigBee* coordinador, para el flujo constante de información proveniente desde cada dispositivo final *ZigBee*.

La central se encarga de procesar los datos proporcionados por los sensores y en base a reglas definidas previamente mediante programación; almacena la información, compara estados, y envía información necesaria por la interfaz inalámbrica hacia los diferentes servicios basados en *Internet of Things* (difusión de alertas por medios digitales,

almacenamiento en base de datos, presentación de datos en interfaz *Web*). La etapa de diseño del sistema se basa en dos ejes fundamentales:

- La etapa de *Hardware*, y
- La etapa de *Software*

Dentro de la etapa de *Hardware* se contempla la elaboración de los bloques que conforman los dispositivos finales, así como también el armado y conexión de la central del sistema con sus dispositivos periféricos. La etapa de *software* es la etapa más extensa y abarca el recorrido total de la información generada, desde el origen de la información en los sensores del dispositivo Final *ZigBee*, hasta la difusión de alertas al usuario del sistema, mediante medios digitales, así como también su representación y configuración a través del servicio *web* integrado. Previo a las etapas de *hardware* y *software* se hizo una evaluación de la situación actual de la Empresa de Tecnología y Servicios TECSERLED, la cual permitió definir el sistema de seguridad, mismo que fue implementado en la infraestructura de la empresa.

5.2.2. Análisis de la situación actual de la empresa

El continuo crecimiento del mercado en la ciudad de Loja, obligó a la empresa TECSERLED, a la expansión de su oferta a nivel de productos y servicios, motivos por los cuales generó un crecimiento de infraestructura abarcando nuevos departamentos, haciendo que sea necesaria la creación de un sistema de seguridad que incluya las funcionalidades propuestas en dichas áreas.

Para la realización del proyecto, en primer lugar se analizó los requerimientos de la empresa tomando en cuenta las necesidades y distribución de los espacios en los que se trabaja (Figura 30). A partir de este análisis y en base a conceptos técnicos se determinó el número de equipos y dispositivos necesarios para el funcionamiento óptimo del sistema. También se hizo uso de los métodos inductivo y deductivo, para determinar las mejores opciones en la ubicación de elementos que conforman el sistema de seguridad (sensores, estación central, fuentes de alimentación, etc.) dentro de la infraestructura de la empresa.

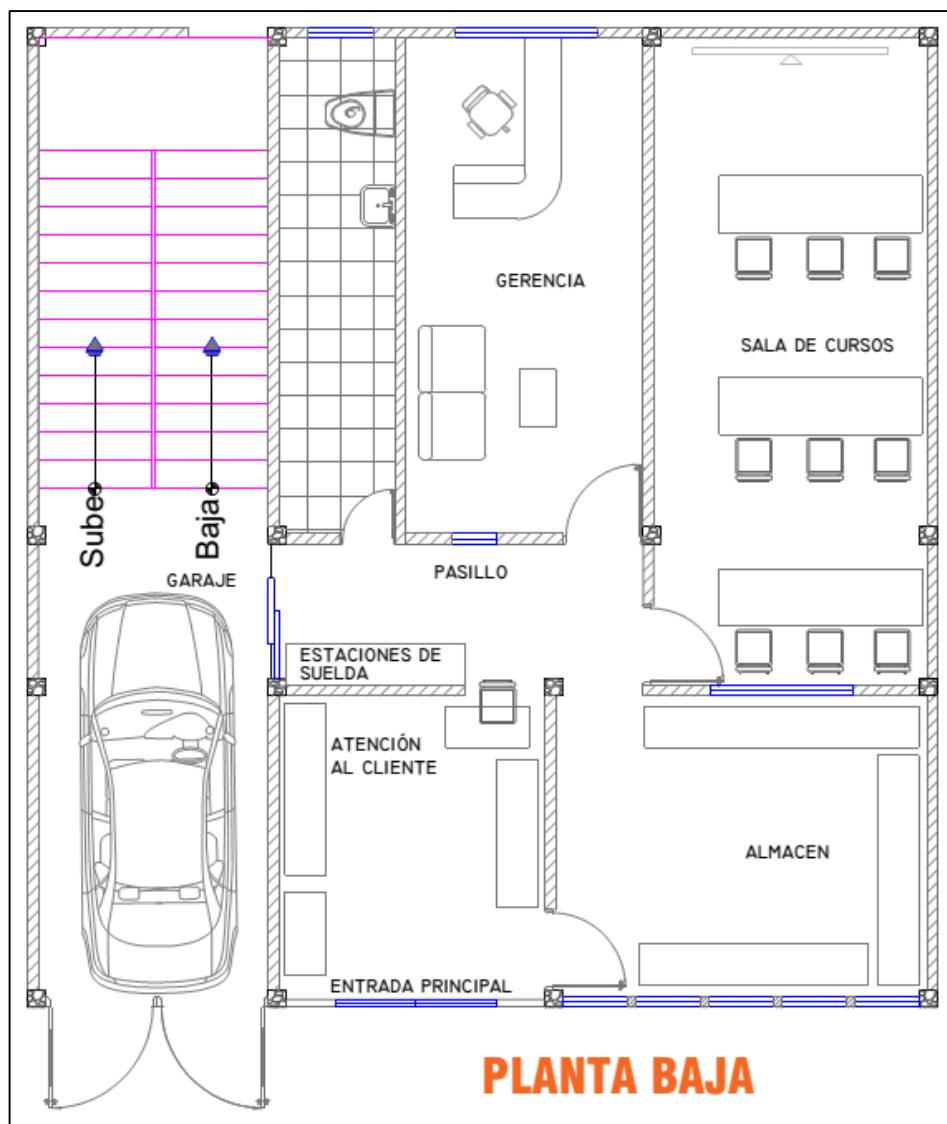


Figura 30. Distribución de los espacios y áreas de trabajo de la empresa TECSERLED.

Fuente: [El Autor]

5.2.2.1. Accesos y puntos específicos de la empresa a proteger

A continuación se detalla el número de puertas y ventanas con las que cuentan los espacios de trabajo de la empresa, los cuales fueron objeto de estudio para seleccionar de forma adecuada el tipo de sensor específico. La Figura 31 muestra la elevación frontal de la infraestructura con la que cuenta la empresa, aportando un panorama más claro de la distribución de espacios vulnerables.

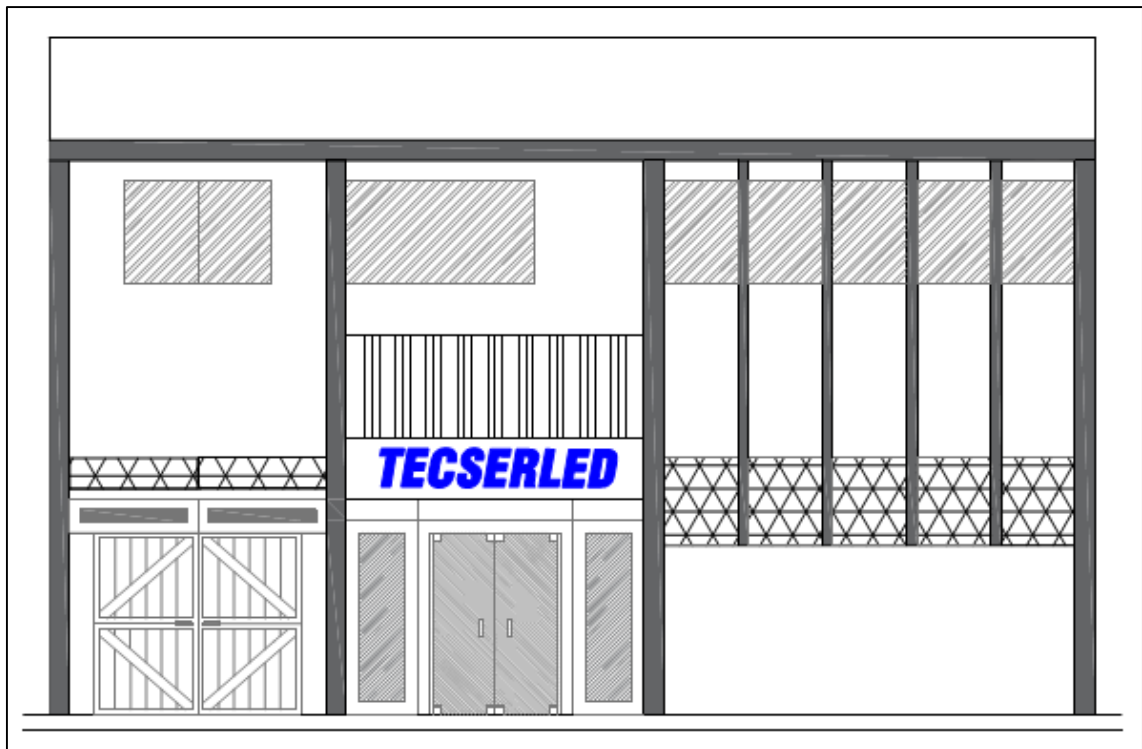


Figura 31. Elevación frontal de la empresa de tecnología y servicios TECSERLED.

Fuente: [El Autor]

Tabla 9. Resumen de accesos vulnerables en la infraestructura a monitorear.

Nivel/Planta	Fachada/Espacio	Punto vulnerable	Cantidad
Planta Baja	Frontal	Puertas	2
Planta Baja	Interior	Puertas	5
Planta Baja	Frontal	Ventanas	5
Planta Baja	Posterior	Ventanas	2

Fuente: [El Autor]

Analizados los espacios de trabajo y áreas vulnerables presentes en la infraestructura de la empresa que ocupa un 90% de la planta baja, se determinó la creación de un sistema de seguridad que abraque principalmente el despliegue de alarmas de intrusión, alarmas técnicas, personales y de video vigilancia. Dichos elementos o sensores, se disponen según lo muestra la Figura 32 y pueden ser observados a mayor detalle en el Anexo I.

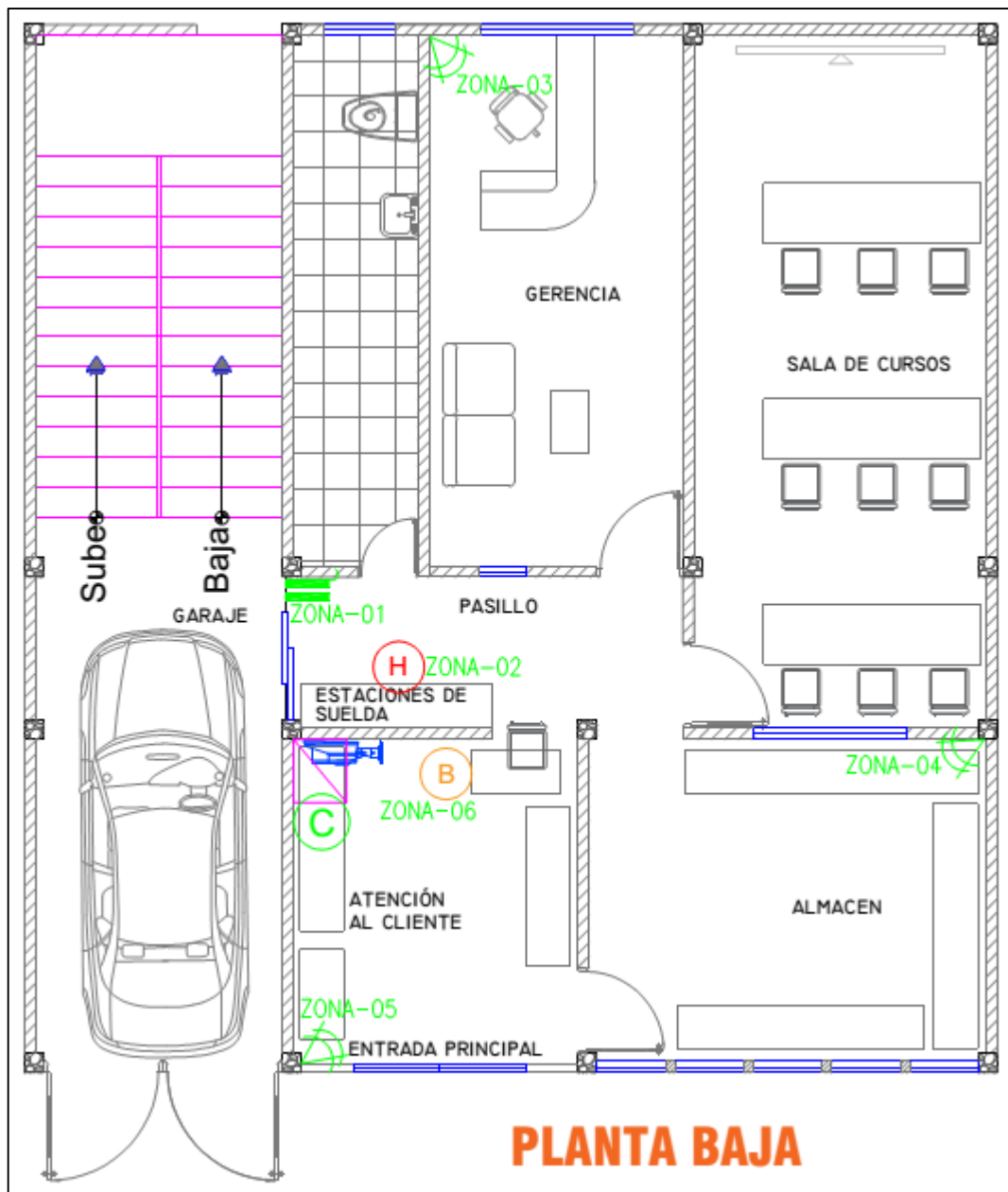


Figura 32. Distribución de sensores que conforman el sistema de seguridad.

Fuente: [El Autor]

5.2.3. Etapa de hardware

5.2.3.1. Dispositivo final ZigBee

La construcción del *hardware* comenzó con el bloque del dispositivo final *ZigBee* (Figura. 33), el cual está conformado por una fuente de alimentación, un sistema basado en el uso de microcontrolador, interfaz de comunicación serial, sensores, e indicadores.

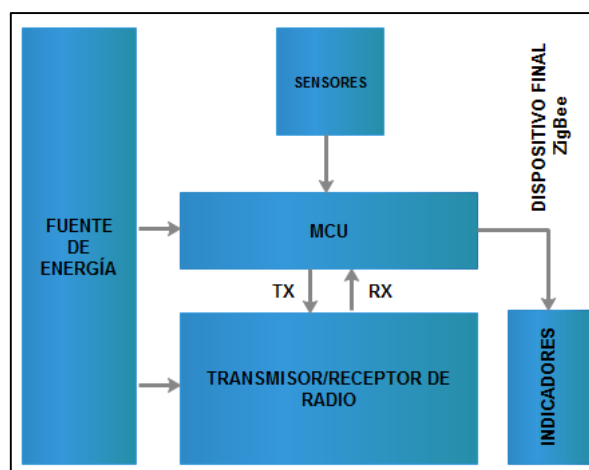


Figura 33. Composición del bloque del dispositivo Final ZigBee.

Fuente: [El Autor]

Cada bloque representado en la Figura 33. Dispone de al menos un dispositivo electrónico que fue seleccionado según sus características y su costo, teniendo en cuenta los requerimientos del sistema en general. A continuación se detallan las tablas comparativas de las principales características de los elementos implementados en este bloque funcional.

▪ Microcontrolador

La Tabla 10 resume las características de tres microcontroladores, las cuales han permitido seleccionar al microcontrolador PIC18F25K20 como el más idóneo para conformar el bloque del dispositivo final *ZigBee*, esto basado en los siguientes parámetros.

Tabla 10. Principales características comparativas de tres microcontroladores.

Microcontrolador	Voltaje de operación	Nro. De pines	Nro. De pines E/S	Costo (\$)
PIC16F877A	5V	40	35	8,00
PIC18F25K20	3.3V	28	25	7,00
ARDUINO NANO	5V	30	20	7,00

Fuente: [El Autor]

- ✓ **Voltaje de operación:** opera en un voltaje compatible con el dispositivo de RF utilizado, asegurando un consumo de potencia relativamente bajo y necesario para este tipo de aplicaciones.
- ✓ **Software libre:** hace uso de un entorno de desarrollo (IDE) totalmente liberado para explotar las funcionalidades de este dispositivo, el entorno de desarrollo es PROTON IDE y utiliza el lenguaje de programación BASIC de muy fácil

comprensión y aplicación.

- ✓ **Costo:** Debido a las prestaciones que presenta este dispositivo, a sus tamaño reducido y a su costo relativo, hacen del mismo el indicado para formar parte del bloque del dispositivo final *ZigBee*.

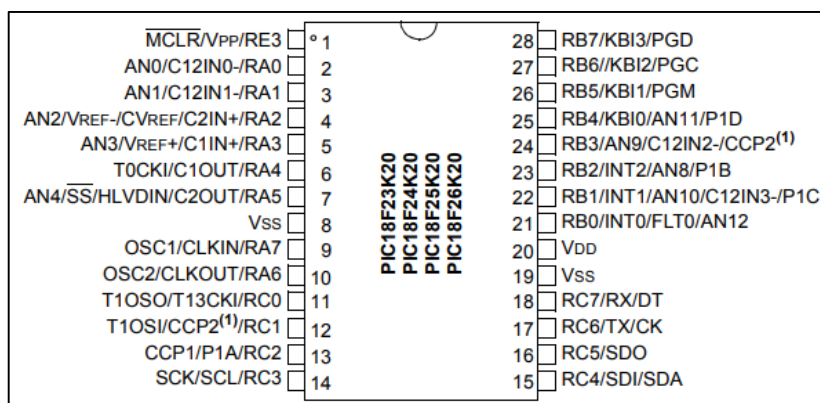


Figura 34. Disposición de pines del microcontrolador PIC18F25K20.

Fuente: [34]

▪ Sensor de movimiento

En la Tabla 11, se detallan las características principales de los sensores de movimiento comparados, de los cuales el sensor HC-SR501 resulta ser el indicado debido a las características mostradas, su costo y a su fácil obtención en las tiendas electrónicas del país. En la Figura 35 se muestra el *hardware* del dispositivo, los elementos que lo integran y la configuración de parámetros como duración del pulso de alarma y rango de sensibilidad, de fácil configuración a través de potenciómetros de ajuste.

Tabla 11. Características de sensores de movimiento

	Características	Especificaciones/Valores
Sensor PIR (#555-28027)	Voltaje de operación	3-6 VDC
	Potencia de consumo	130μA inactivo y 3mA Activo
	Rango de detección	4,5-9 metros ajustable
	Temperatura de operación	0 a 50 °C
	Angulo de detección	No proporcionado en la hoja de datos
	Voltaje de salida	ALTO: 5V – BAJO: 0V
	Costo (\$)	15 dólares
Sensor PIR (HC-SR501)	Voltaje de operación	5-20 VDC
	Potencia de consumo	< a 65mA
	Rango de detección	3-7 metros ajustable
	Temperatura de operación	-15 a 70 °C
	Angulo de detección	Menor a 110° (forma cónica)
	Voltaje de salida	ALTO: 3.3V – BAJO: 0V
	Costo (\$)	3 dólares

Fuente: [El Autor]

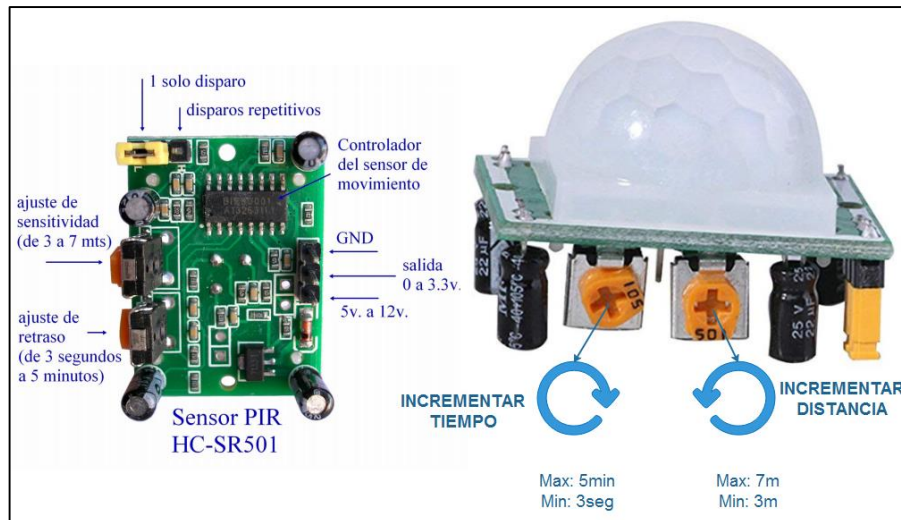


Figura 35. Hardware del sensor PIR HC-SR501.

Fuente: [35]

▪ Sensor de Humo

La Tabla 12, presenta características de sensores de humo, que son posibles implementar en un prototipo, dejando exentos a una gran variedad de sensores industriales los cuales no se encuentran en consideración dentro de la elaboración del mismo. Mencionadas ciertas características de algunos de ellos, se seleccionó el módulo sensor MQ-2 como el indicado para el sistema de seguridad debido a los siguientes factores:

- ✓ **Sensibilidad a sustancias:** Debido a que este sensor está implementado, sobre un espacio en donde se trabaja con estaciones de suelda, que generalmente liberan ciertas cantidades de humos de soldadura, cuando están en funcionamiento. El sensor es susceptible a estas partículas presentes en el humo que en definitiva, pueden ser provocadas por un incendio o producto de la combustión de los agentes de soldadura utilizados en este proceso.
- ✓ **Rango de detección (ppm):** El cual determina la cantidad de concentración de ciertas partículas que el sensor puede detectar dentro de un área determinada. El rango que ofrece el MQ-2 es ideal para realizar las pruebas en el sistema de seguridad.

Tabla 12. Características principales de varios sensores de gases disponibles en la actualidad.

Parámetros del sensor	MQ-2	MQ-4	MQ-6	MQ-135
Voltaje de operación	5V	5V	5V	5V
Rango de detección (ppm) partes por millón	300 a 10000	300 a 10000	300 a 10000	10 a 10000
Energía de calefacción	≤900mW	≤900mW	≤900mW	≤900mW
Salida analógica	0~5V	0~5V	0~5V	0~5V
Salida Digital	TTL	TTL	TTL	TTL
Sustancias detectadas	Metano, Butano, GLP, Humo	Metano, gas natural comprimido (GNP)	Butano, propano, GLP	Benceno, humo, alcohol, calidad de aire
Costo (\$)	5,00	5,50	5,00	6,00

Fuente: [El Autor]

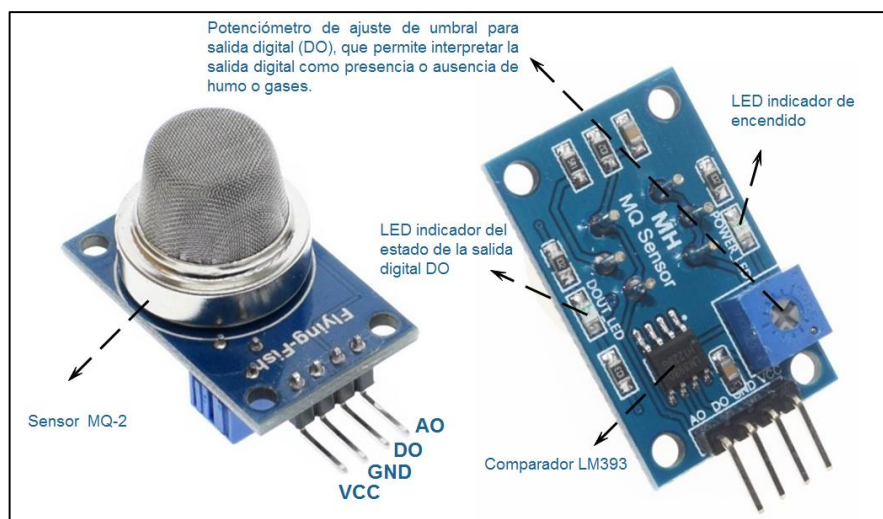


Figura 36. Hardware del sensor MQ-2.

Fuente: [El Autor]

▪ Sensor magnético

Debido a la escasa variedad de sensores magnéticos para aplicaciones electrónicas de bajos voltajes, y debido a que únicamente cumplen con la función de un interruptor, utilizando campos magnéticos para cerrar sus terminales o contactos. Se optó por utilizar un sensor magnético comercial utilizado para puertas y ventanas. El sensor utilizado es el MC-38 de fácil acceso en el mercado de componentes electrónicos ecuatoriano.

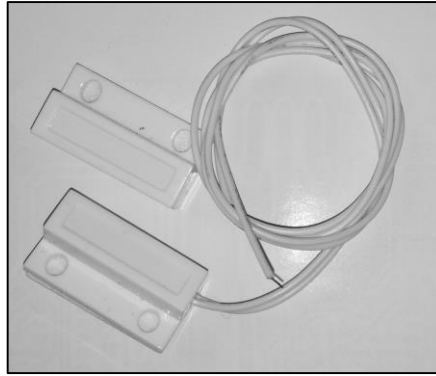


Figura 37. Sensor magnético MC-38.

Fuente: [El Autor]

▪ **Botón de pánico**

El uso de este dispositivo está limitado al funcionamiento de un pulsador normalmente abierto, en configuración activo en alto, que se detallará más adelante cuando se aborde el diseño de los circuitos que se implementó en el sistema.



Figura 38. Pulsador metálico para chasis, normalmente abierto.

Fuente: [36]

▪ **Módulo de RF ZigBee**

Para completar la construcción del dispositivo final *ZigBee* se debe hacer la elección del módulo RF a implementar en el sistema. Hoy en la actualidad existen diversos fabricantes que ofrecen soluciones tecnológicas a través de sus módulos *ZigBee*, sin embargo la disposición de sus herramientas a nivel de *hardware* como *software* se ven limitadas por su difícil acceso, costo o información necesaria para desarrollar aplicaciones, es por ello que a continuación se plantea la elección del módulo RF *ZigBee* en función de dos fabricantes seleccionados.

Tabla 13. Comparación de dispositivos ZigBee entre dos fabricantes.

CC2538DK Kit de desarrollo, IEEE 802.15.4, ZigBee, ARM Cortex M3, CC2538	Descripción: Se trata de un kit de desarrollo, que proporciona una plataforma de pruebas de rendimiento de hardware y un entorno de desarrollo de software, para el chip CC2538 basado en ARM Cortex-M3 que implementa IEEE 802.15.4 ideal para aplicaciones ZigBee de alto rendimiento.	
	Características	Especificaciones/Valores
	Voltaje de alimentación	2.0 - 3.6 V
	Frecuencia de operación	2.4 GHz
	Sensibilidad del receptor	-97 dBm
	Dispositivos USB	2.0 velocidad completa
	Velocidad de reloj	Hasta 32 MHz
	Memoria flash programable	512KB, 256KB o 128KB
	Robustez a la interferencia	ACR de 44dB
	Potencia de salida programable	Hasta 7 dBm
	Costo (\$)	500 dólares
Módulos XBee Digi (XBee-S2C)	Descripción: Los módulos XBee son soluciones integradas de fácil acceso, aplicadas a un medio inalámbrico que utilizan el protocolo de red IEEE 802.15.4 para la creación de redes punto a punto, punto a multipunto y diversas topologías, en donde se requiere transmisiones a bajas tasas de datos, baja latencia y sincronización de comunicación predecible. A continuación se detallan las características más relevantes del módulo XBee S2C	
	Voltaje de alimentación	2.1 – 3.6 V
	Frecuencia de operación	2.4 GHz
	Rango de cobertura	60 m
	Sensibilidad del receptor normal	-100 dBm
	Potencia de salida modo normal	3 dBm
	Costo (\$)	35 dólares (Unidad)

Fuente: [El Autor]

Los principales motivos para la elección de este dispositivo se mencionan a continuación:

- ✓ **Costo:** el factor principal es el costo, tal como lo refleja la tabla de comparación resulta muy económico el dispositivo XBee respecto al de su homólogo.
- ✓ **Factor I+D:** Es posible obtener grandes cantidades de información respecto a sus posibles aplicaciones, la información en la *web* es muy abundante y permite solucionar diversidad de problemas que pueden presentarse al momento de ser utilizados. Además cuentan con un IDE o entorno de desarrollo libre como es el *software* XCTU que permite explotar todas las funcionalidades del dispositivo en mención.
- ✓ **Características:** las principales características resumidas en la Tabla 13 hacen de este dispositivo ideal para el sistema de seguridad propuesto.

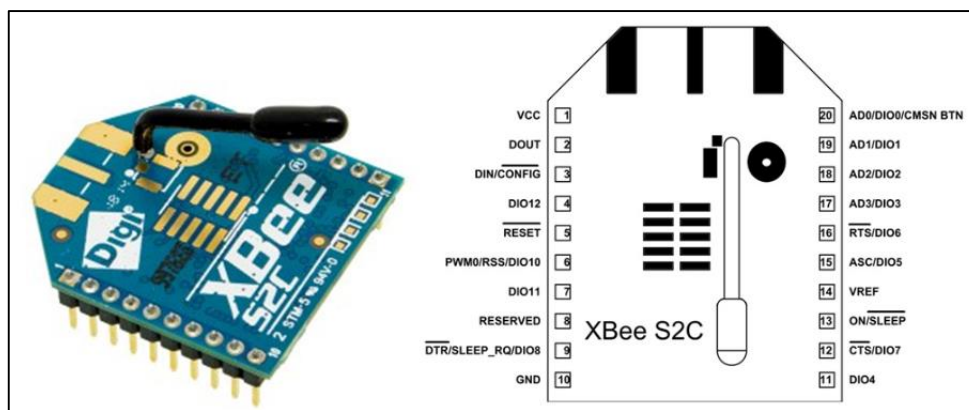


Figura 39. Hardware del módulo RF XBee S2C.

Fuente: [37]

En la Tabla 14, se muestra la disposición de pines I/O que conforman el módulo Xbee S2C.

Tabla 14. Descripción de los pines I/O del módulo XBee S2C.

Pin #	Nombre	Estado por defecto	Descripción
1	VCC	----	Fuente de alimentación
2	DOUT/DIO13	Salida	Salida de datos UART
3	DIN/CONFIG/DIO14	Entrada	Entrada de datos UART
4	DIO12/SPI_MISO	Deshabilitado	GPIO/SPI esclavo (salida)
5	RESET	Entrada	Módulo RESET
6	RSSI PWM/PWMO DIO10	Salida	Indicador de intensidad de señal RX / GPIO
7	PWM1/DIO11	Deshabilitado	GPIO
8	[RESERVADO]	---	No conectado
9	DTR/SLEEP_RQ/ DIO8	Entrada	Línea de control de suspensión de pin / GPIO
10	GND	---	Tierra
11	SPI_MOSI/DIO4	Deshabilitado	GPIO/SPI esclavo (entrada)
12	CTS/DIO7	Salida	(Línea de control) borrar para enviar control de flujo/GPIO
13	ON_SLEEP/DIO9	Salida	Indicador de estado del dispositivo / GPIO
14	VREF	---	No conectado
15	ASSOCIATE/DIO5	Salida	Indicador asociado / GPIO
16	RTS/DIO6	Entrada	(Línea de control) Solicitud para enviar control de flujo / GPIO
17	AD3/DIO3/SPI_SSSEL	Deshabilitado	Entrada analógica / GPIO / SPI selección esclavo
18	AD2 /DIO2/SPI_CLK	Deshabilitado	Entrada analógica / GPIO / SPI reloj
19	AD1/DIO1/SPI_ATTN	Deshabilitado	Entrada analógica / GPIO / servicio SPI
20	AD0/DIO0/CB	Deshabilitado	Entrada analógica / GPIO / Botón de puesta en marcha.

Fuente: [25]

- **Topología de la red implementada**

Basado en la distribución de sensores, tal como lo indica la Figura 32 y al haber definido los módulos RF a utilizar, se procedió a establecer la topología de la red, encargada de recolectar la información proveniente desde cada dispositivo final. Teniendo en cuenta las distancias existentes desde el nodo coordinador hacia cada dispositivo final (Anexo I) se decidió optar por la topología “Tipo Estrella”, pues el área de cobertura del dispositivo coordinador abarca sin mayor inconvenientes, toda la distribución de sensores en los espacios de trabajo de la empresa.

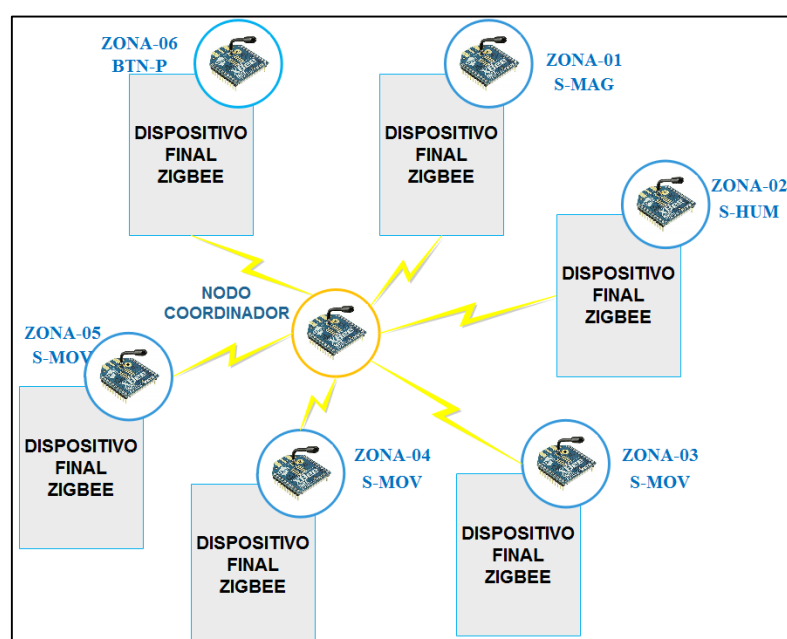


Figura 40. Topología de red inalámbrica TIPO ESTRELLA implementada.

Fuente: [El autor]

5.2.3.2. Central del sistema

Continuando con la conformación del *hardware* del sistema de seguridad, llega el momento de describir los dispositivos utilizados en la parte de la central del sistema.

- **Dispositivo central del sistema**

En la actualidad existe variedad de entornos de desarrollo disponibles para ser utilizados en variedad de aplicaciones. La elección del dispositivo central del sistema se basó en factores específicos como; costo, uso de herramientas de *software* libre, y la integración

de servicios adicionales que se llegaron a implementar en la misma. La Tabla 15, resume las características de algunos entornos de desarrollo y posteriormente se detalla la elección del dispositivo para conformar la central del sistema.

Tabla 15. . Comparación de tarjetas de desarrollo, basadas en el uso de software libre.

TARJETA PROPIEDAD	ODROID-C2	ORANGE PI	RASPBERRY PI 3
CPU	ARM CORTEX – A53	CORTEX-A7	4x ARM CORTEX – A53
ARQUITECTURA	ARMv8	QUAD-CORE	ARMv6
ARQUITECTURA	2GHz	1.2GHz	1.2 GHz
MEMORIA RAM	2GB	512MB	1GB
ALMACENAMIENTO EXTERNO	MEMORIA SD	MEMORIA SD	MEMORIA SD
PUERTOS USB	4 USB 2.0 y 1 USB OTG	2 USB 2.0 y 1 USB OTG	4 USB 2.0 y 1 USB OTG
VOLTAJE DE ALIMENTACIÓN	5V	5V	5V
PUERTOS GPIO	SI	SI	SI
INTERFAZ WIFI	NO	SI	SI
PUERTO ETHERNET	ETHERNET 10 /100/1000 Mbit/s	NO	10/100MB
LINUX	SI (Ubuntu)	SI (Ubuntu)	SI (Raspbian)
PYTHON	SI	SI	SI
PRECIO (\$)	140,00	40,00	50,00

Fuente: [El Autor]

Tomando en cuenta la información presentada en la Tabla 15, se pueden observar algunas alternativas que cumplen con los requerimientos del *hardware*, sin embargo debido a la agregación de servicios en el sistema propuesto; Orange Pi presenta desventajas frente a Odroid-C2 y Raspberry pi los cuales constan con una serie de periféricos de gran utilidad y poseen sistemas operativos que permiten desarrollar aplicaciones con Python, ideales para desarrollo de interfaces gráficas, procesos en paralelo, almacenamiento en la nube, etc. Finalmente teniendo en cuenta el factor costo del equipo es obvia la elección de la tarjeta de desarrollo Raspberry pi.

▪ Cámara

Habiendo hasta este punto elegido la central del sistema, y demás elementos se procede con la elección de este elemento importante dentro del sistema de seguridad. Existen en el mercado variedad de cámaras (Especialmente del tipo *Web Cam*), que pueden ser implementadas en la tarjeta de desarrollo Raspberry pi a través de sus puertos USB. Sin

embargo para no hacer uso de un elevado coste computacional a la tarjeta Raspberry pi, se optó por el módulo de la cámara Raspberry Pi, que se muestra en la Figura 41.



Figura 41. Raspberry Pi Camera Module v1.3.

Fuente: [38]

▪ Fuente de energía

La tarjeta de desarrollo Raspberry pi cuenta con su propia fuente de alimentación, a través de un adaptador de voltaje cuyas características son; Voltaje de entrada: 100~240V AC, Voltaje de salida DC 5V/2.5A.



Figura 42. Fuente de alimentación 5V 2.5A Micro USB para Raspberry pi.

Fuente: [39]

▪ Elementos actuadores

Los elementos actuadores son básicamente un banco de relés conectados hacia los puertos GPIO de la Raspberry Pi. Se ha optado por utilizar un módulo relé de 4 canales que se encuentra equipado con Relés, AC250 V 10 A; DC30 V 10 A y son activos a 5V, de fácil acceso en las tiendas electrónicas del país.

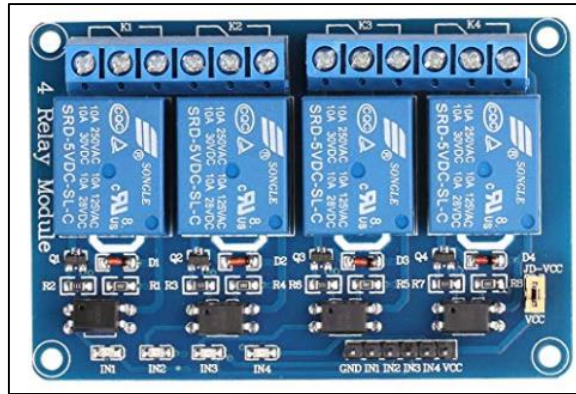


Figura 43. Módulo relé 4 canales 5V DC.

Fuente: [40]

▪ Elemento coordinador de la red WSN

Para realizar la concentración de información en el nodo coordinador, proveniente de cada dispositivo final en la WSN, se requiere de un elemento de *hardware* especial junto al dispositivo XBee-S2C seleccionado. Este elemento es el módulo XBee Explorer, que incorpora un convertor USB-Serial, traduciéndose en el puente de comunicación entre la WSN y la central del sistema.

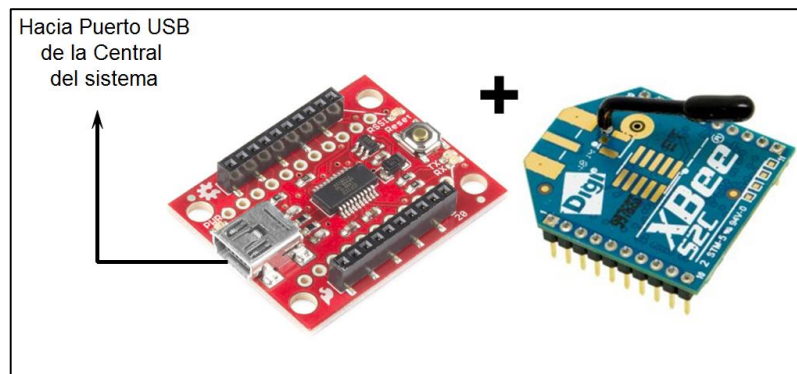


Figura 44. Conformación del elemento coordinador, utilizando un módulo XBee-Explorer.

Fuente: [El Autor]

5.2.3.3. Diseño de circuitos

Tal como se mencionó en la Figura 33, el dispositivo Final *ZigBee* se encuentra conformado por algunos bloques funcionales que hacen referencia a los circuitos electrónicos utilizados para definir este importante elemento. Se diseñó el bloque del dispositivo Final *ZigBee*, de tal forma que fuese posible implementarse todos los sensores que formarán parte del sistema de seguridad en una sola PCB, es decir, el mismo diseño

de la PCB permite implementar cualquier tipo de sensor según la ZONA a monitorear, obteniendo de esta forma un sistema modular que permite escalabilidad en la creación de la red y aceleración en los procesos de fabricación. A continuación se detallan los bloques que forman parte del dispositivo Final *ZigBee* y sus principales funciones dentro del mismo.

▪ Fuente de alimentación

Para un correcto funcionamiento, tanto del microcontrolador como del módulo RF se debe tener una fuente de alimentación de 3.3V. Dicha fuente se obtiene mediante una regulación de voltaje, que transforma el voltaje de entrada proveniente de una fuente conmutada de 12V, una batería de 12V o 9V, al habilitar el *switch* de entrada (SW). Se han utilizado los reguladores comerciales 7805 y LD 1117 que ofrecen en sus salidas los voltajes de 5V y 3.3V junto a sus condensadores de filtro (C1, C2, C3) y (C4, C5) respectivamente, tal como lo indica la Figura 45. El regulador de voltaje 7805 es utilizado para la alimentación de los sensores de Humo y movimiento, del dispositivo final ZigBee. Adicionalmente como se observa, se colocó un diodo 1N4004 que protege al circuito, ante posibles fallas humanas, como por ejemplo conectar invertidamente la polaridad de la fuente, la disposición del diodo (D1) solo permite la circulación de la corriente en un solo sentido.

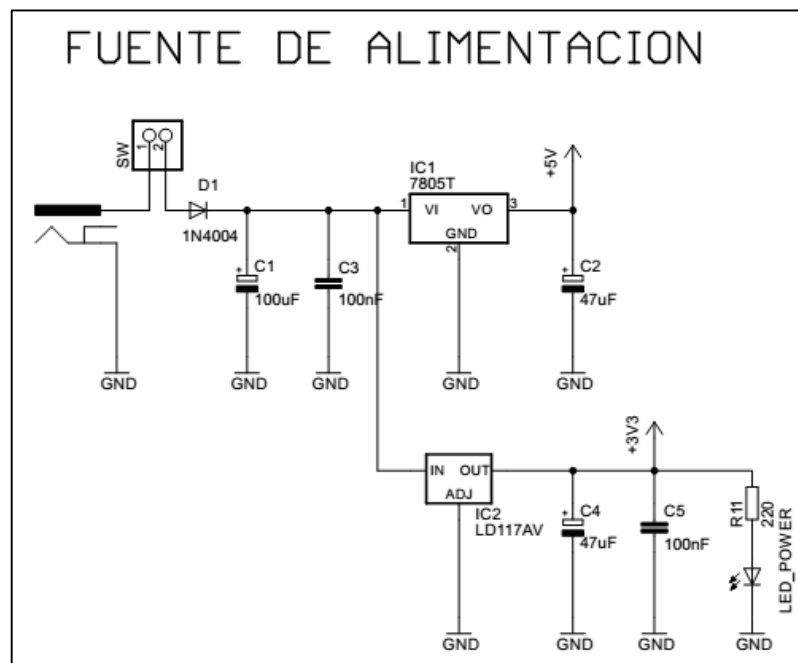


Figura 45. Circuito fuente de alimentación para el dispositivo Final ZigBee.

Fuente: [El Autor]

- **Acondicionamiento de las señales de sensores**

En la Figura 46 y 47 se muestra la conexión de los sensores que forman parte del dispositivo final *ZigBee*, y sus respectivas salidas que se conectan directamente en los pines del microcontrolador PIC18F25K20.

La Figura 46 a) detalla la conexión del sensor PIR HC-SR501 hacia el pin 22 del microcontrolador, el cual se encuentra configurado como entrada digital. Este sensor al detectar algún cuerpo que irradie calor dentro de su área de cobertura emite por medio de su salida (DO) una señal en alto de 3.3V. Teniendo en cuenta posibles fluctuaciones de voltajes superiores al nivel mencionado en su salida, se optó por colocar un diodo zener 1N4728 a la salida del sensor de movimiento con el objetivo de proteger la entrada digital del microcontrolador. Siendo el caso que exista un voltaje superior, el diodo zener entregará al pin 22 un voltaje constante de 3.3V.

En la Figura 46 b) se muestra la implementación del circuito para el sensor magnético, el cual generará dos tipos de estados lógicos; 0V (0L) indicando que los contactos se encuentran abiertos y 3.3V (1L) indicando que sus contactos se encuentran cerrados, esta salida se conecta al pin 21 del microcontrolador que se encuentra configurado como entrada digital, la configuración de la lectura de este sensor en el microcontrolador, se debe tener muy en cuenta ya que es activa en bajo.

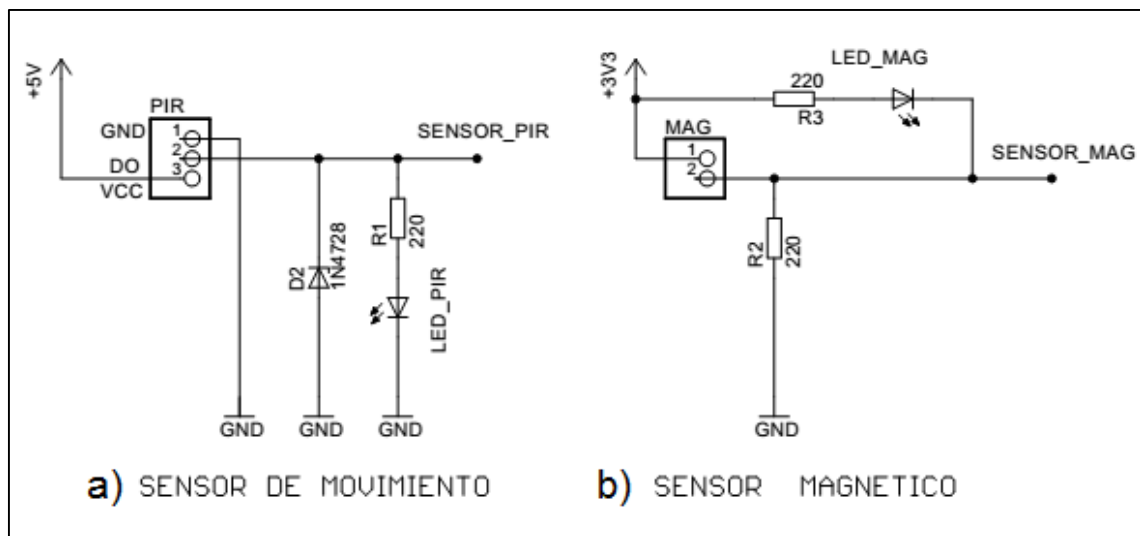


Figura 46. Acondicionamiento de señales: a) sensor de movimiento, b) sensor magnético.

Fuente: [El Autor]

En la Figura 47 a) se muestra el circuito para el módulo sensor de humo MQ-2, gracias a la salida digital (DO) presente en el módulo adoptado, la cual internamente trabaja con un comparador que con la ayuda de un potenciómetro hace posible la calibración de un umbral, permitiendo interpretar la salida digital como presencia o ausencia de gases, da como resultado una salida que envía un estado lógico en alto (5V) cuando no existe presencia de gases o humo en el entorno del sensor, y un estado lógico cero (0V) cuando detecta presencia de humo en su entorno.

Esta salida se conecta al pin 23 del microcontrolador declarado como entrada digital. Para este caso, y tal como ocurre con el sensor magnético, se debe tener muy en cuenta la lectura del sensor que es activa en bajo.

Adicional a ello, tomando en cuenta que se opera con un microcontrolador a 3.3V y teniendo presente que el rango de voltaje a la salida del sensor de humo varía de 0V a 5V, se procede a limitar su salida con la colocación de un diodo zener 1N4728, lo que provocará que las salidas lógicas dadas por el sensor de humo cambien al rango de 0V a 3.3V respectivamente, protegiendo de esta forma la integridad de la entrada digital del microcontrolador.

Finalmente en la Figura 47 b) se representa la configuración del circuito para la detección de activación del botón de pánico. Este circuito mantiene un nivel lógico bajo (0V) mientras el pulsador permanece abierto. Al presionar el pulsador, el nivel lógico en el pin I/O (en este caso el pin 24) pasa a ser alto (3.3V).

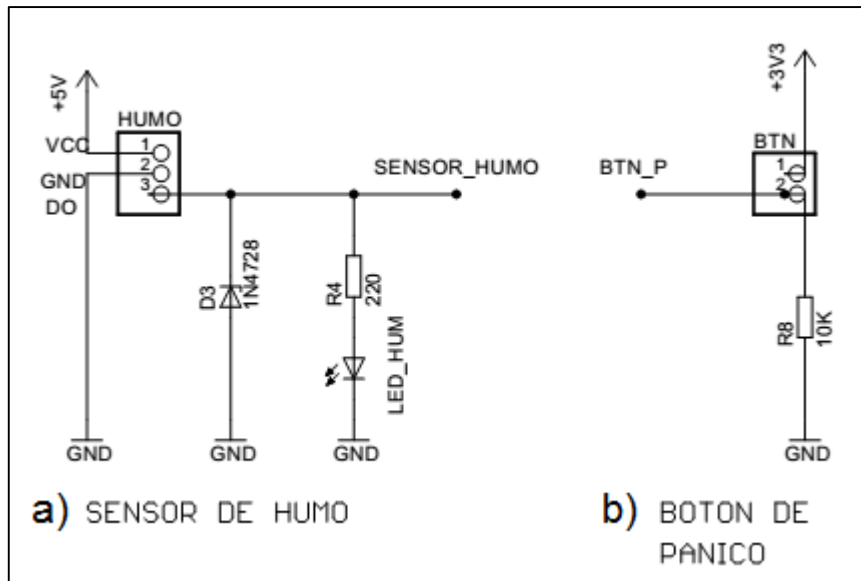


Figura 47. Acondicionamiento de señales; a) sensor de humo, b) botón de pánico.

Fuente: [El Autor]

En todos los circuitos mostrados, los bloques representan la conexión física de los sensores en la PCB.

▪ Microcontrolador y dispositivo XBee

Explicados los circuitos de fuente de alimentación y acondicionamiento de señales de sensores, se procede a la conformación del circuito microcontrolador, quien es el encargado de recoger toda la información proveniente desde cada sensor y enviarla a través de la interfaz serial hacia el módulo Xbee-S2C, también es el responsable de controlar elementos conectados a él. Todos estos procedimientos son ejecutados por el microcontrolador previa programación del mismo. El circuito de la Figura 48 muestra la conexión existente entre el microcontrolador y el dispositivo Xbee así como las conexiones de los puertos I/O llevadas a cabo para el funcionamiento del bloque Dispositivo Final.

Tal como se mencionó en un principio tanto el microcontrolador y el Xbee son alimentados con 3.3V. El circuito de RESET se conecta al pin 1 (MCLR/VPP), el cual mantiene al microcontrolador encendido hasta que el botón de RESET es presionado enviándolo a un nivel de voltaje bajo 0V (0L), provocando el reinicio del mismo. Este circuito es muy útil en las etapas de prueba iniciales, en donde se requiere verificar el correcto funcionamiento del programa.

software libre. El acabado final del PCB correspondiente al dispositivo Final *ZigBee* se muestra en la Figura 49.

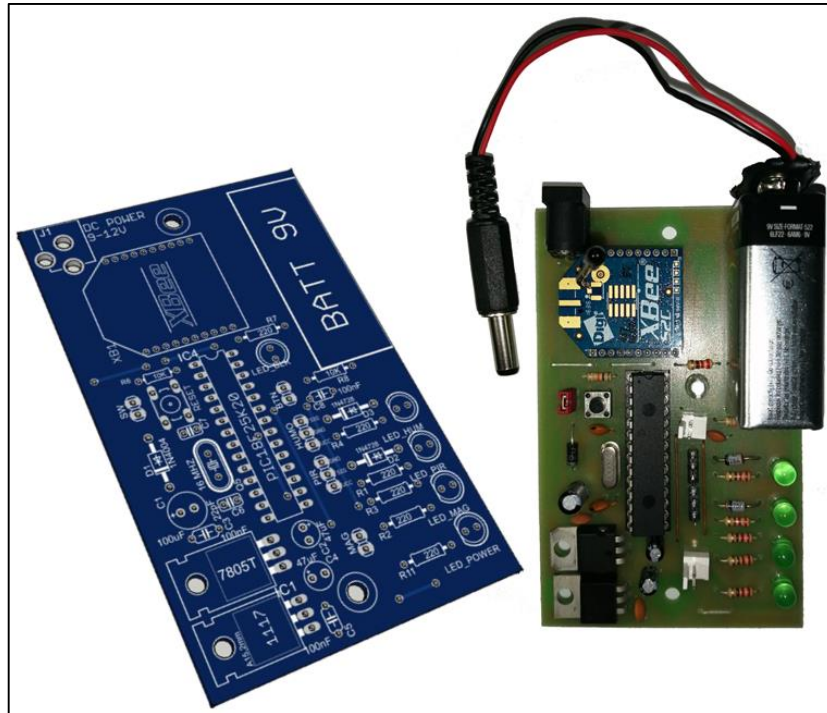


Figura 49. Diseño del PCB del dispositivo final ZigBee y su acabado final.

Fuente: [El Autor]

La implementación del *hardware* referente a la central del sistema se puede observar en las Figuras 50 y 51 respectivamente.

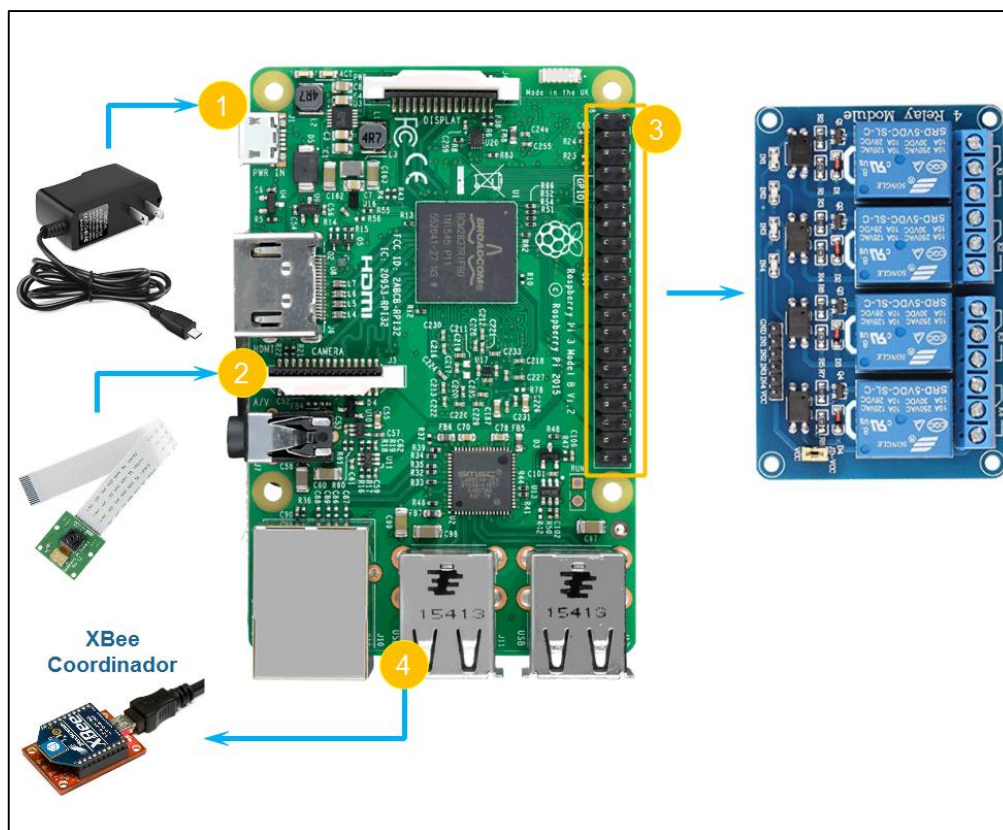


Figura 50. Diagrama de conexiones de la central del sistema.

Fuente: [El Autor]

Las conexiones de los componentes mostrados en la Figura 50 se describen a continuación:

Tabla 16. Descripción de las conexiones realizadas en la central del sistema.

Punto Nro.	Descripción
1	Conexión de la fuente de energía para la tarjeta Raspberry pi
2	Conexión de la cámara de Raspberry pi
3	Conexión de los elementos actuadores a través de los puertos GPIO de la Raspberry pi. En la Figura 51 se muestra en detalle el conexionado realizado.
4	Conexión del módulo XBee coordinador con la ayuda de una tarjeta XBee Explorer, por medio de un cable USB 2.0 tipo A - tipo B mini de 5 pines, desde un puerto USB de la tarjeta

Fuente: [El Autor]

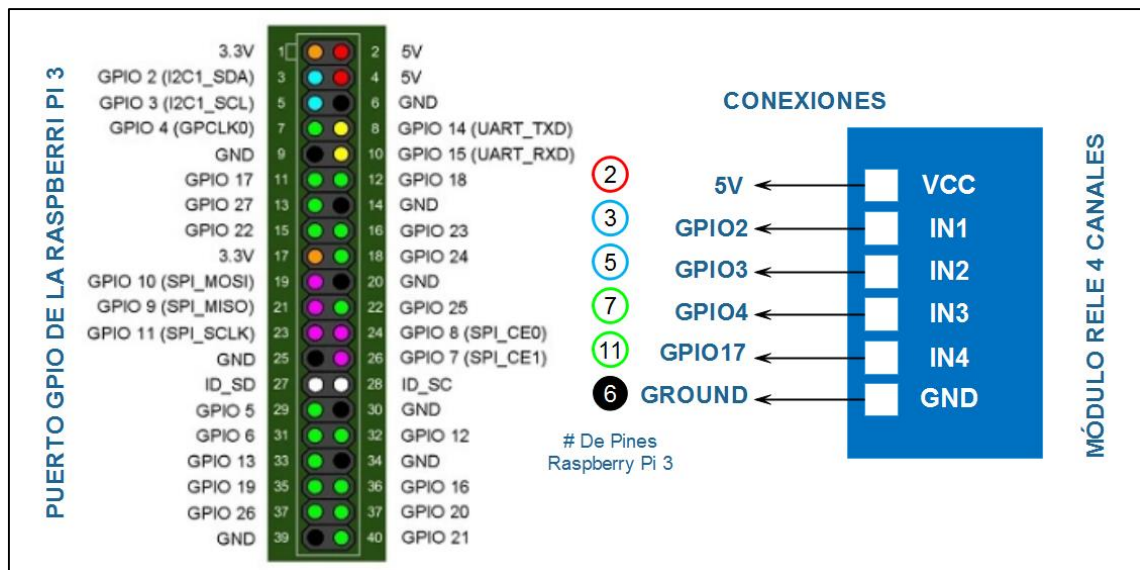


Figura 51. Diagrama de conexión entre el puerto GPIO de la Raspberry Pi, y el módulo relé de 4 canales.

Fuente: [El Autor]

5.2.4. Etapa de software

De igual forma, que como se hizo en la etapa de *hardware*, para la etapa de *software* se comenzó con el diseño del bloque del dispositivo final. Para la programación del microcontrolador PIC18F25K20 se hizo uso del entorno de desarrollo PROTON IDE, mientras que para la programación de los dispositivos finales XBee y elemento coordinador se hizo uso de la herramienta XCTU, ambos basados en el concepto de *software* libre.

5.2.4.1. Programación del PIC 18F25K20 con las funciones del bloque del dispositivo Final ZigBee

El compilador mencionado (PROTON IDE), es muy flexible y permite que la mayoría de los tipos de declaración de variables o constantes, se coloquen en cualquier lugar dentro del programa BASIC. Sin embargo, es posible que no produzca los resultados correctos o que se produzca un error de sintaxis inesperado debido a que una variable se declara después de que se supone que debe utilizarse. El diseño recomendado para un programa se muestra a continuación y es el diseño adoptado para la realización del código.

```

{
  Declares
}
{
  Includes
}
{
  Constants and Variables
}

GoTo Main           ' Jump over the subroutines (if any)

{
  Subroutines go here
}
{
  Main:
  Main Program code goes here
}

```

Figura 52. Diseño recomendado para elaborar un programa en el compilador PROTON IDE.

Fuente: [41]

A continuación se realiza una explicación sobre el funcionamiento del programa desarrollado:

1. Se declara las librerías a utilizar
2. Se declara las configuraciones de los registros para los puertos de Entrada/Salida
3. Se declaran todas las variables a utilizar en el programa y se inicializan en un valor predeterminado.
2. Antes de hacer un salto a la rutina principal se declara la sentencia que indica la habilitación de interrupción por TIMER0
4. Se escribe la sentencia que realiza el salto a la rutina principal en el programa (GoTo Main:)
5. Tal como lo muestra la Figura 52, se realiza la creación de las subrutinas a implementar
6. Se declara la subrutina “Interrupt_blk” que tiene como objeto:
 - ✓ Crear una base de tiempo a partir de la interrupción generada por el desbordamiento del TIMER0. La ecuación que define el intervalo de tiempo por cada interrupción se muestra en la Ec.01.

$$TMR0\ Overflow = (2^{16} - TMR0)(Prescaler)(4/Fosc) \text{ Ec.01}$$

Para asegurar una interrupción, por ejemplo cada 10ms, el nuevo valor que debe escribirse en TMR0 es **64286**. Resultado del despeje de TMR0 en la Ec.01.

$$10ms = (65536 - \mathbf{64286})(128)(4/64MHz)$$

- ✓ La contabilización de 1000 interrupciones genera una base de tiempo de 10 segundos, que es utilizada para enviar datos del estado actual de los sensores hacia la central de alarmas. Se utilizó esta base de tiempo por factores demostrativos, este valor puede ser modificado mediante programación y, en base a la ecuación (Ec.01) posible obtener diversidad de bases de tiempo.

7. Se declara la subrutina “envio_dato_normal” cuya función es:

- ✓ El envío de una trama de datos por puerto serie, del estado normal de los sensores (es decir que no ha ocurrido ningún evento) utilizando la base de tiempo de 10 segundos. El formato de la trama es la siguiente: “InicioDeTrama_Zona_TipoDeSensor_EstadoDelSensor_FinDeTrama”.
Ejemplo: "**I_ZON01_SMAG_EST:00_F**". Esta subrutina es utilizada para indicar la comunicación constante y existente con la central del sistema.

8. Se declara la subrutina “envio_dato_Alerta” cuya función es:

- ✓ El envío de una trama de datos por puerto serie, que indica una activación de determinado sensor. El envío de la trama de datos es similar a la del envío normal de datos, solo que en este caso se indica un cambio en el estado del sensor, y no se encuentra en función a alguna base de tiempo, el dato es enviado cuando se desencadena un evento, ejemplo: "**I_ZON01_SMAG_EST:01_F**". Nota: se envía una trama diferente en función al sensor activado (Tabla 17).

Tabla 17. Trama enviada según el tipo de sensor activado

Activación de sensor	Trama enviada
Magnético	I_ZON01_SMAG_EST:01_F
Humo	I_ZON02_SHUM_EST:01_F
Movimiento	I_ZON03_SMOV_EST:01_F I_ZON04_SMOV_EST:01_F I_ZON05_SMOV_EST:01_F
Botón de pánico	I_ZON06_BTNP_EST:01_F

Fuente: [El Autor]

9. Se declara la función principal “Main”

- ✓ Esta función contiene la declaración de un ciclo *while*, la cual permite que todas las líneas de código que se encuentran dentro de ella se repitan en un lazo infinito. De esta forma es posible leer constantemente las entradas digitales conectadas a los sensores y al hacer uso recursivo de las subrutinas declaradas en los puntos anteriores, se lleva el proceso de envío de información hacia la central del sistema por medio del módulo Xbee S2C y su interfaz serial.

Las Figuras 53, 54 y 55 muestran el diagrama de flujo que fue la base para el desarrollo del programa. El código de programación para un mejor análisis y comprensión, se encuentra en el Anexo III

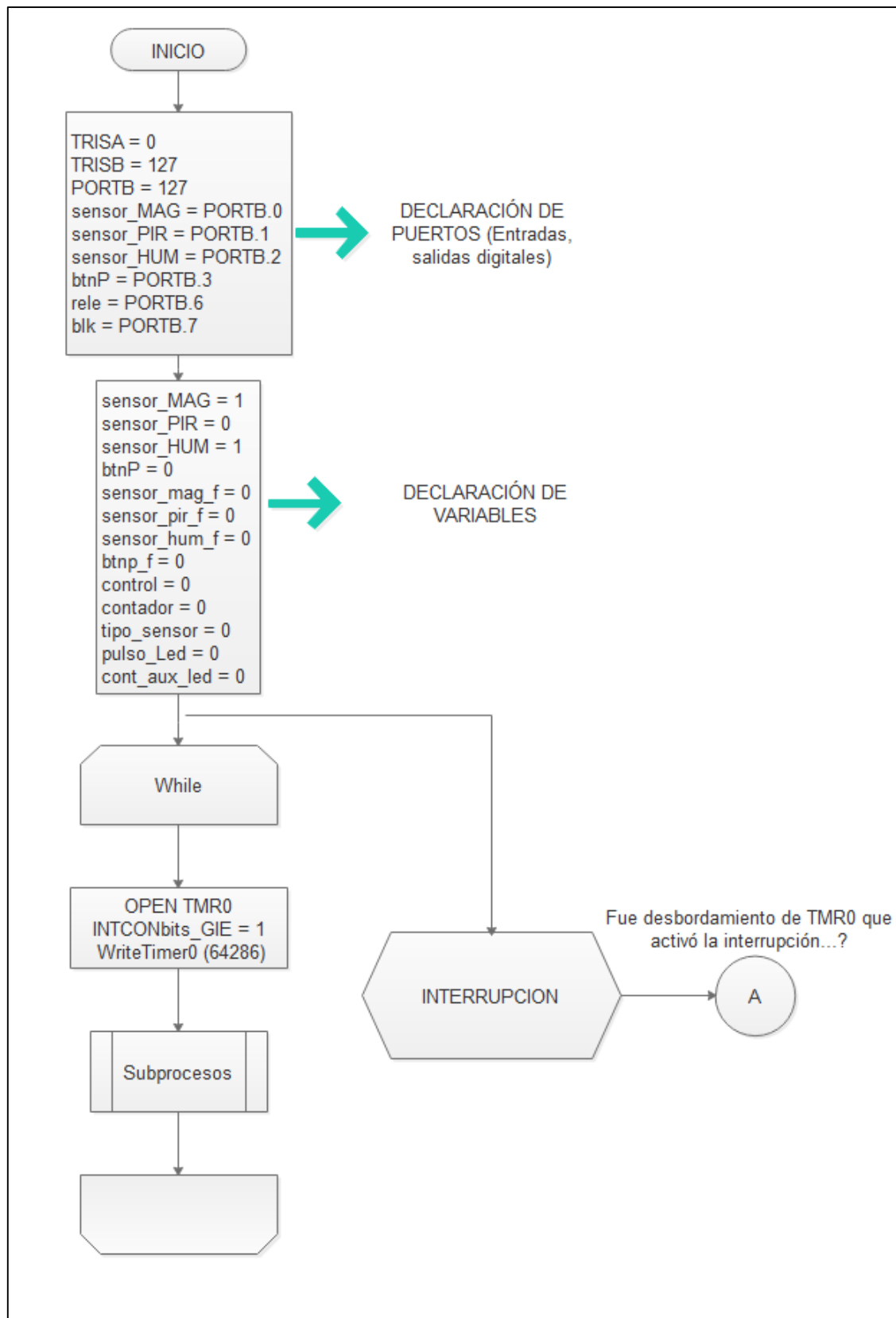


Figura 53. Diagrama de flujo- programación del PIC18F25K20.

Fuente: [El Autor]

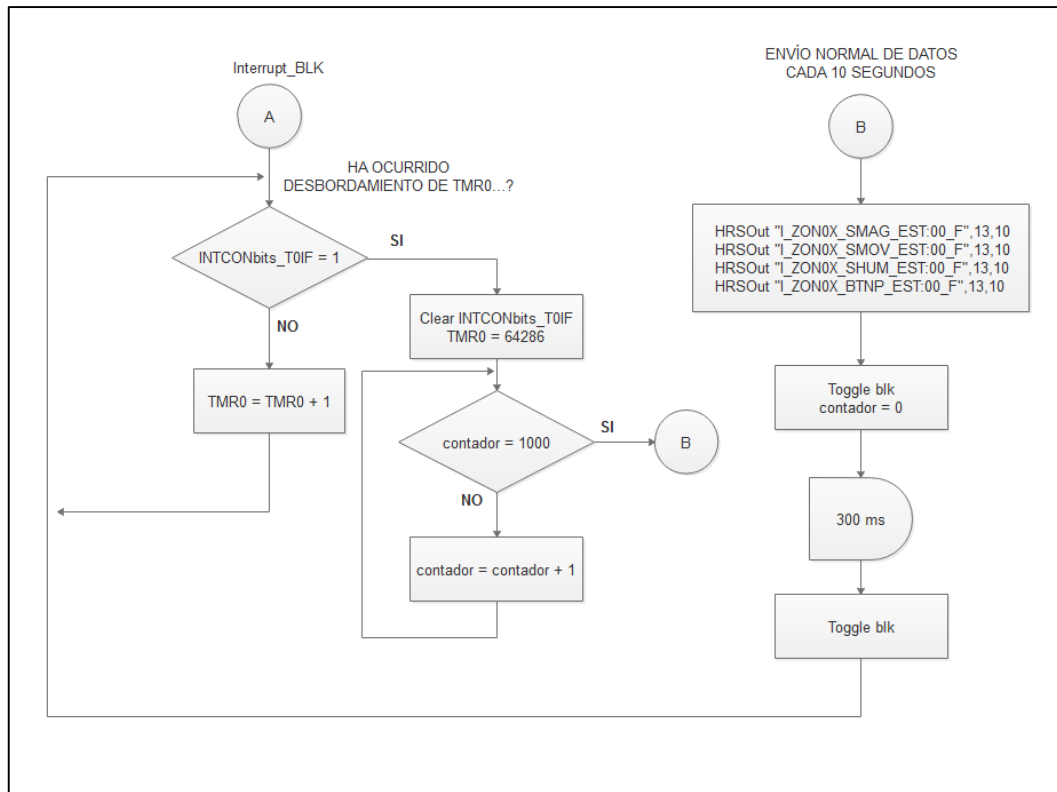


Figura 54. Continuación del diagrama de flujo-programación del PIC18F25K20.

Fuente: [El Autor]

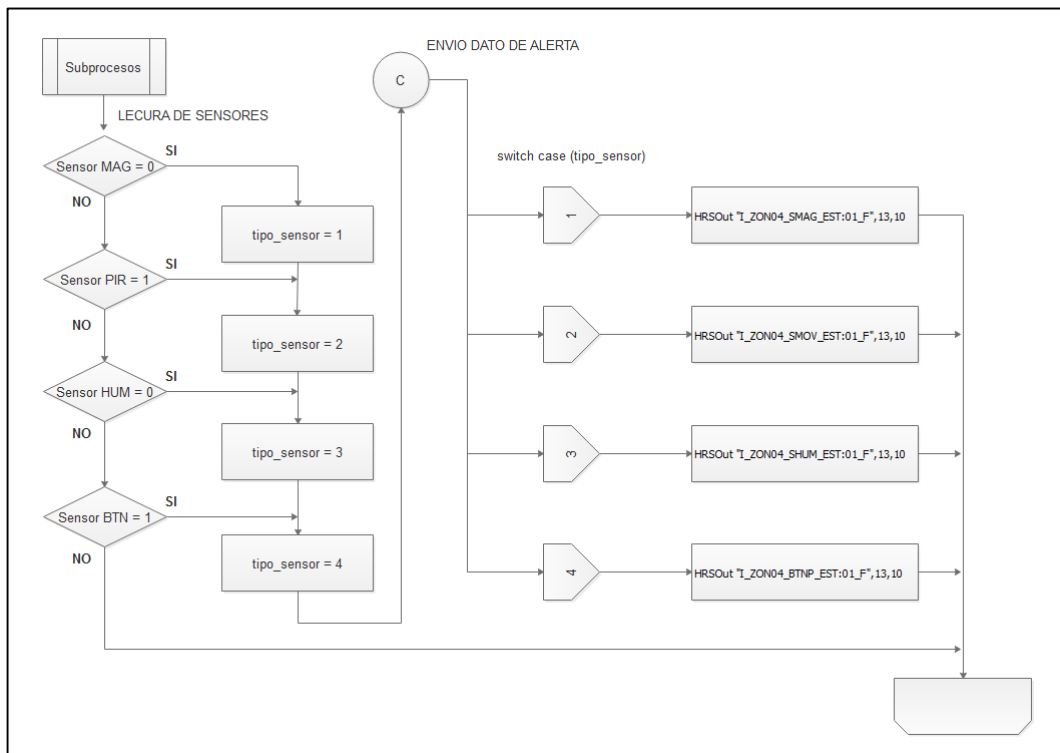


Figura 55. Continuación del diagrama de flujo-programación del PIC18F25K20.

Fuente: [El Autor]

- Grabación del programa en el microcontrolador

Para grabar el programa .hex creado en el entorno PROTON IDE, se precisa del *software* PICKit 3 Programmer y el programador de microcontroladores PICKit 3. Previa la conexión del programador a un puerto USB del computador, se realiza las conexiones respectivas mostradas en el Anexo IV, entre el programador y el microcontrolador PIC18F25K20, para la Programación Serial en Circuito (ICSP). Una vez conectado el PICKit 3 a un puerto USB del computador se procede a abrir el entorno del *software*.

- ✓ Como primer paso en el entorno, en la pestaña Device Family, se selecciona la familia del microcontrolador a programarse, en este caso se selecciona “PIC18F_K_”
- ✓ A continuación se procede a ubicar en archivo .hex generado por el compilador de PROTON IDE en el directorio guardado, accediendo a la pestaña File y al haber seleccionado “Import Hex”
- ✓ Una vez cargado el programa .hex generado, se da clic en el botón Write y el resultado debe ser el mostrado en la Figura 56.

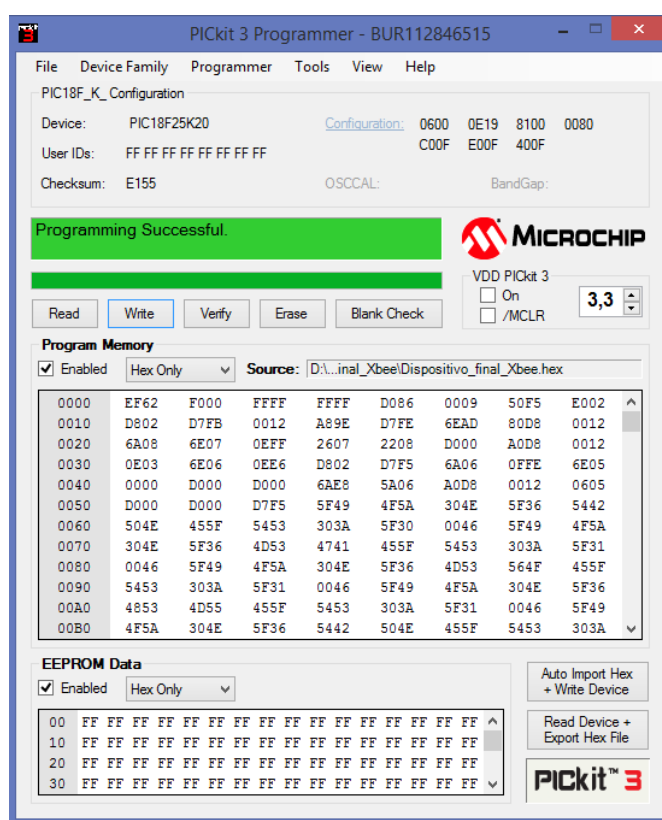


Figura 56. Grabación del programa .hex generado, en el microcontrolador PIC18F25K20.

Fuente: [El Autor]

5.2.4.2. Configuración de los módulos RF XBee

La empresa DIGI ha desarrollado el *software* XCTU que es una aplicación libre y multiplataforma, diseñada para permitir a los desarrolladores interactuar con los módulos RF XBee a través de una interfaz gráfica fácil de usar. Para obtener todas las funcionalidades de esta herramienta, se descargó el instalador del *software* XCTU desde la página DIGI <https://www.digi.com/products/iot-platform/xctu>

Una vez instalado el *software*, con la ayuda de un cable USB 2.0 tipo A - tipo B mini, un dispositivo XBee Explorer y un puerto USB de la computadora se procede a realizar la configuración de los dispositivos XBee.

Teniendo en cuenta la topología adoptada, se precisa de la configuración de dos tipos de nodos que conforman la red IEE 802.15.4, estos nodos son: el nodo coordinador y el nodo dispositivo final.

- **Configuración del nodo coordinador**

A continuación se describe la configuración del módulo XBee-S2C como nodo coordinador, mediante la programación realizada en el *software* XCTU.

- ✓ Se coloca el módulo XBee-S2C en la tarjeta Xbee Explorer USB
- ✓ Se conecta la tarjeta Xbee Explorer a un puerto USB de la PC
- ✓ Se inicia el programa XCTU
- ✓ Dentro de la interfaz del programa, se presiona el botón Discover devices, o a su vez la combinación de teclas (Ctrl+Shift+D)
- ✓ Inmediatamente se abre una ventana emergente, en donde es posible seleccionar el puerto USB en donde está conectado el dispositivo. Se selecciona el puerto y se da clic en siguiente.
- ✓ Se verifica los parámetros por default; Baud Rate: 9600, Data Bits: 8, Parity: none, Flow Control: none, Stop Bits: 1. O a su vez se los configura.
- ✓ Habiendo verificado/configurado los parámetros del puerto se da clic en finalizar y el resultado será una pantalla similar a la de la Figura 57.

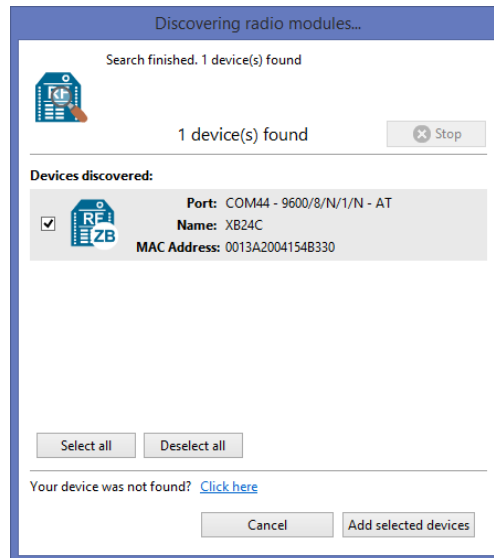


Figura 57. Configuración del dispositivo coordinador.

Fuente: [El Autor]

- ✓ Continuando con la configuración se da clic en el botón agregar dispositivos seleccionados, con lo cual el dispositivo XBee conectado es agregado al espacio de trabajo. Al hacer doble clic sobre él es posible ingresar a la interfaz de configuración, la cual dependiendo de la versión del *hardware*, desplegará su respectivo *firmware*.
- ✓ Lo siguiente, es dar clic en el botón de la función Update, y se configura el XBee con el *firmware* más reciente de ZIGBEE TH Reg (En este caso Versión 4060).
- ✓ Una vez actualizado el *firmware* del dispositivo, se despliega una serie de campos en donde es posible realizar configuraciones. Los campos en los cuales se realiza modificaciones se muestran en la Figura 58.

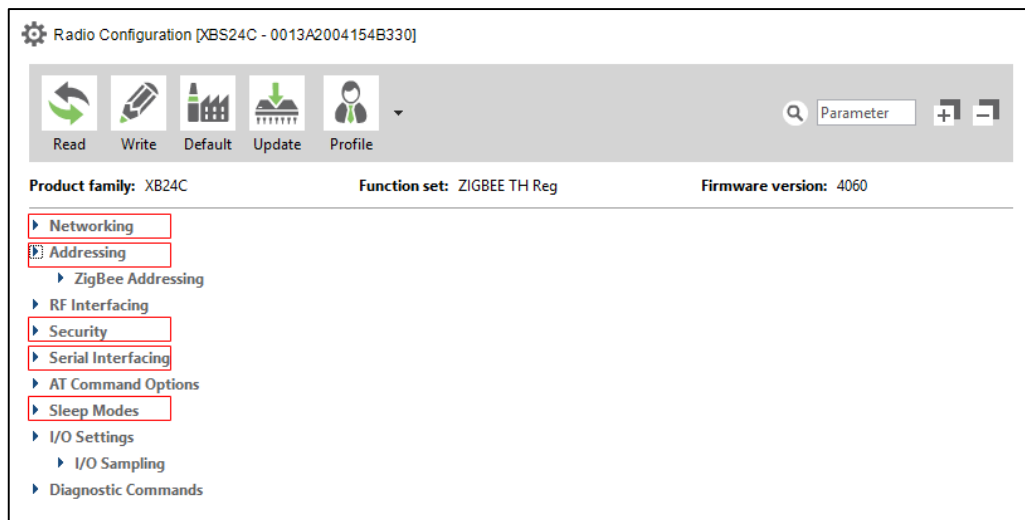


Figura 58. Campos intervenidos para la configuración del XBee S2C, como dispositivo coordinador de la red.

Fuente: [El Autor]

- ✓ En la Tabla 18 se indican la serie de parámetros configurados en cada uno de los campos a ser intervenidos. En el resto de campos los parámetros quedan establecidos por *default*.

Tabla 18. Parámetros configurados en el XBee S2C nodo coordinador.

CAMPO/CARPETA	INDICADOR	NOMBRE	VALOR ESTABLECIDO
Networking	ID	PAN ID	92
	CH	Canal de operación	18
	SC	Escaneo de canales	7FFF
	CE	Habilitación de coordinador	Enabled [1]
Addressing	DH	Dirección de destino alta	13A200
	DL	Dirección de destino baja	FFFFFFFF
	NI	Identificador del Nodo	NODO COORDINADOR
Security	EE	Habilitar encriptación	Enabled [1]
	KY	Clave de enlace de confianza	A11B22
	NK	Clave de seguridad de Red	C33D44
Serial Interfacing	AP	Modo de transmisión de la interfaz serial	Transparent mode [0]
Sleep modes	SP	Periodo de suspensión cíclico	20x10ms
	SN	Número de periodos de suspensión cíclicos	1
	SM	Modo de suspensión	No Sleep (Router) [0]

Fuente: [El Autor]

- ✓ Terminada la configuración de los diversos parámetros, se pulsa el botón “write” y se espera el proceso de grabación del *firmware* en el dispositivo.
- ✓ Terminado este proceso se procede a cerrar el XCTU, desconectando el XBee Explorer junto al dispositivo configurado.

Se debe mencionar que algunos parámetros mostrados en la Tabla 18 con sus valores respectivos se encuentran definidos por defecto en el dispositivo. Tal es el caso de:

- **CH:** el cual es un valor predeterminado al momento de hacer la configuración, y es el canal elegido por el dispositivo como canal de operación.
- **SC:** el cual establece o lee la lista de canales a escanear. En el nodo coordinador, representa la lista de campos de bits de canales para elegir antes de inicializar la red. El valor tomado por defecto es: 0x7FFF lo cual indica que el coordinador intenta encontrar un canal de uso claro. Este valor también es tomado por los dispositivos finales, los cuales escanean a través de todos los canales habilitados para encontrar un coordinador.
- **AP:** que define el funcionamiento de la transmisión de datos a través de la interfaz serial en modo transparente o API. En este caso en específico se ha utilizado la configuración del dispositivo en modo transparente, el cual permite obtener los datos igual que los de su origen, sin recargos de información adicional a la información enviada. Este modo se encuentra configurada por defecto en el dispositivo.

▪ **Configuración del nodo dispositivo final.**

Teniendo hasta este punto configurado el dispositivo coordinador de la red, se necesita la configuración de 6 dispositivos XBee como nodos finales. A continuación se describe la configuración de uno de ellos, misma que se repite para los dispositivos restantes. El procedimiento para la configuración del dispositivo final es similar al del nodo coordinador, en la Tabla 19 se resumen los parámetros configurados para este dispositivo.

Tabla 19. Parámetros configurados en el XBEE S2C nodo dispositivo final.

CAMPO/CARPETA	INDICADOR	NOMBRE	VALOR ESTABLECIDO
Networking	ID	PAN ID	92
	CH	Canal de operación	18
	SC	Escaneo de canales	7FFF
	CE	Habilitación de coordinador	Disabled [0]
Addressing	DH	Dirección de destino alta	13A200
	DL	Dirección de destino baja	4154B330
	NI	Identificador del Nodo	ZONA-0x-Sx
Security	EE	Habilitar encriptación	Enabled [1]
	KY	Clave de enlace de confianza	A11B22
Serial Interfacing	AP	Modo de transmisión de la interfaz serial	Transparent mode [0]
Sleep modes	SP	Periodo de suspensión cíclico	20x10ms
	SN	Número de periodos de suspensión cíclicos	1
	SM	Modo de suspensión	No Sleep (Router) [0]

Fuente: [El Autor]

A diferencia del nodo coordinador, el parámetro **CE** para estos dispositivos se encuentra deshabilitado, debido a que esta función únicamente la realiza el nodo coordinador de la red. En los parámetros de direccionamiento tanto **DH** como **DL** contienen la dirección MAC del elemento coordinador (**SH** y **SL** respectivamente), tal como lo muestra la Figura 59.

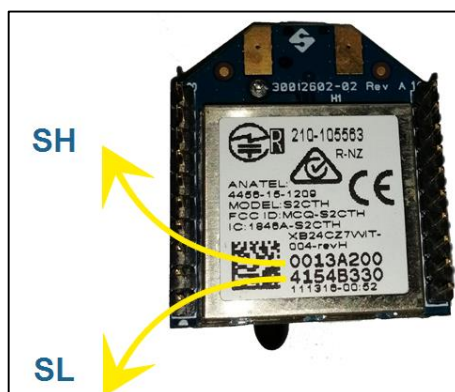


Figura 59. Dirección SH y SL del dispositivo XBee coordinador, utilizados como DH y DL respectivamente en cada dispositivo final.

Fuente: [El Autor]

Un punto muy importante que se debe mencionar son los parámetros de seguridad en el nodo dispositivo final. **NK** no se configura ya que es única y exclusiva competencia del nodo coordinador, los enrutadores y dispositivos finales con seguridad habilitada (EE = 1) adquieren la clave de red cuando se unen a una red.

Reciben la clave de red cifrada con la clave de enlace (**KY**), si comparten esta clave de enlace con el coordinador, por tal razón este parámetro si es configurado en el nodo dispositivo final.

Al tener configurados los elementos que intervienen en la red de sensores inalámbricos bajo el estándar IEEE 802.15.4. Se procede a integrar los servicios adicionales al sistema de seguridad propuesto, mismo que se detalla a continuación.

5.2.4.3. Configuración del sistema operativo de la central del sistema

El *hardware* de la tarjeta Raspberry Pi es controlado desde un sistema operativo basado en Linux Embebido. Para continuar con el desarrollo del proyecto tal como se indicó en la Tabla 15, se decidió usar el sistema operativo (SO) Raspbian. La razón se debe a que esta es la única distribución que recibe actualizaciones oficiales por parte de la empresa propietaria, incluye en su sistema toda la potencialidad del lenguaje de programación Python y además nuevamente el factor Investigación más desarrollo (I+D) se hace presente, pues en la comunidad de usuarios, es el SO que más se usa, por lo tanto existe gran cantidad de información disponible. A continuación se detalla los pasos seguidos para la implementación del sistema operativo del sistema de seguridad.

- **Instalación de Sistema Operativo**

Al ser un dispositivo embebido, la tarjeta Raspberry Pi solo se inicia desde una tarjeta micro SD. Se recomienda usar una tarjeta SD clase 10 con una capacidad disponible de mínimo 8Gb (En este caso se utilizó una micro SD de 32Gb). Teniendo lista la tarjeta Raspberry Pi y una tarjeta micro SD a mano, el siguiente paso es preparar la tarjeta SD con el sistema Operativo Raspbian.

Las imágenes de todos los sistemas operativos, se encuentran disponibles para su descarga desde la página oficial de la fundación Raspberry Pi: [***https://www.raspberrypi.org/downloads/***](https://www.raspberrypi.org/downloads/)

Una vez descargado, se descomprime el archivo y se obtiene la imagen, en este caso: 2017-09-07-raspbian-stretch.img, la cual es grabada en la tarjeta SD con la ayuda del programa Win32 Disk Imager, disponible para su descarga en el siguiente enlace <https://sourceforge.net/projects/win32diskimager/>

En el programa se localiza la ruta de la imagen del SO descargado, se selecciona la tarjeta SD y se da clic en el botón Write, hecho este proceso, se espera a que complete la grabación tal como lo muestra la Figura 60.

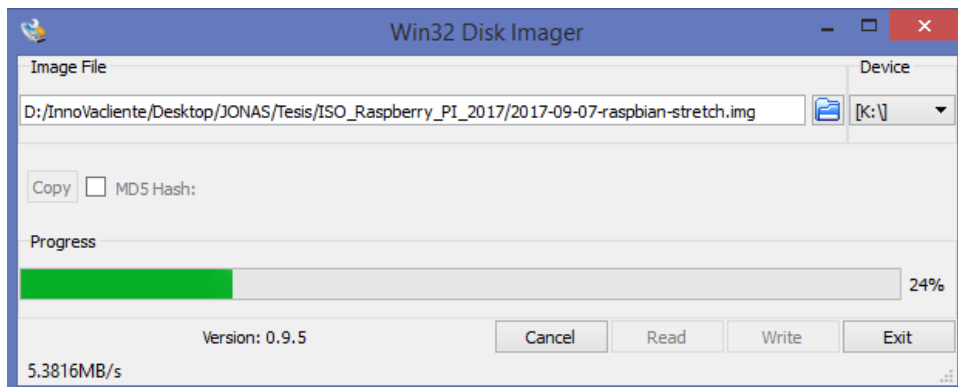


Figura 60. Grabación de la imagen del SO en la tarjeta SD, con la ayuda del software Win32 Disk Imager.

Fuente: [El Autor]

▪ **Conexión remota usando el protocolo SSH**

Para poder acceder a la Raspberry Pi existen dos formas de hacerlo:

- ✓ Conectando una pantalla a la salida HDMI, un teclado y un mouse a los puertos USB.
- ✓ Por conexión remota desde otra computadora utilizando el protocolo SSH

La primera opción requiere de periféricos adicionales que ocupan espacio y demandan de nuevos gastos, en caso de no tenerlos. Por lo que se optó trabajar con la Raspberry Pi a través de conexión remota SSH. Este protocolo hace uso del puerto 22 permitiendo acceder a un host remoto y enviar órdenes a través de un intérprete de comandos. Para ello se precisa conocer la dirección IP que se le asigna dentro de una red local a la Raspberry Pi. Inicialmente con la ayuda de un cable de red, se conecta el puerto Ethernet

de la tarjeta a un puerto del router tal como lo indica la Figura 61 y se conecta a la fuente de alimentación.



Figura 61. Conexión básica para la conexión de la Tarjeta Raspberry Pi, a través de SSH.

Fuente: [El Autor]

Una vez encendida la tarjeta Raspberry Pi, y con la ayuda de ayuda de un *software* escáner de redes, como lo es el software; Wireless Network Watcher, disponible en https://www.nirsoft.net/utis/wireless_network_watcher.html para su descarga, se procede a verificar la dirección IP asignada a la tarjeta, tal como lo muestra la Figura 62.

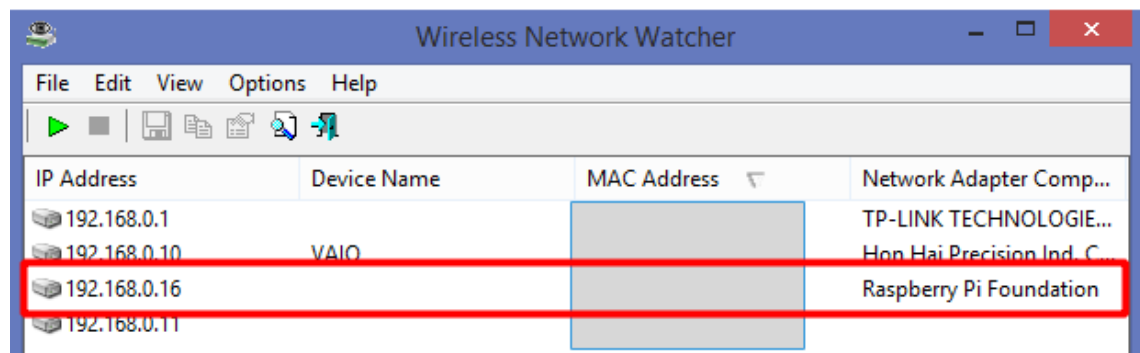


Figura 62. Obtención de la dirección IP asignada a la Raspberry Pi, en la red local por medio del software Wireless Network Watcher.

Fuente: [El Autor]

Usando un cliente SSH como Putty, el cual es un *software* multiplataforma, es posible ingresar al dispositivo tal como se indica en la Figura 63.

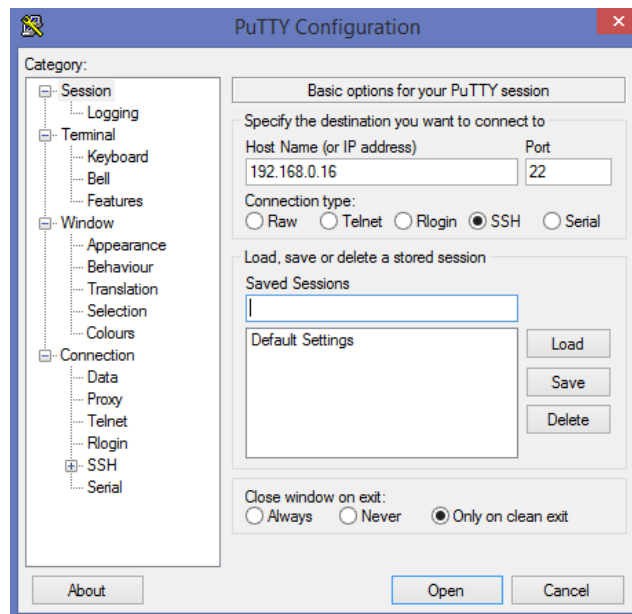


Figura 63. Configuración del cliente SSH para acceder a la tarjeta Raspberry Pi.

Fuente: [El Autor]

Una vez configurado se procede a abrir el terminal en donde pedirá el nombre de usuario por default que es “pi” y el ingreso del *password* por default que es “raspberrypi”. Al haber ingresado correctamente los datos, se debe obtener un resultado similar al de la Figura 64.

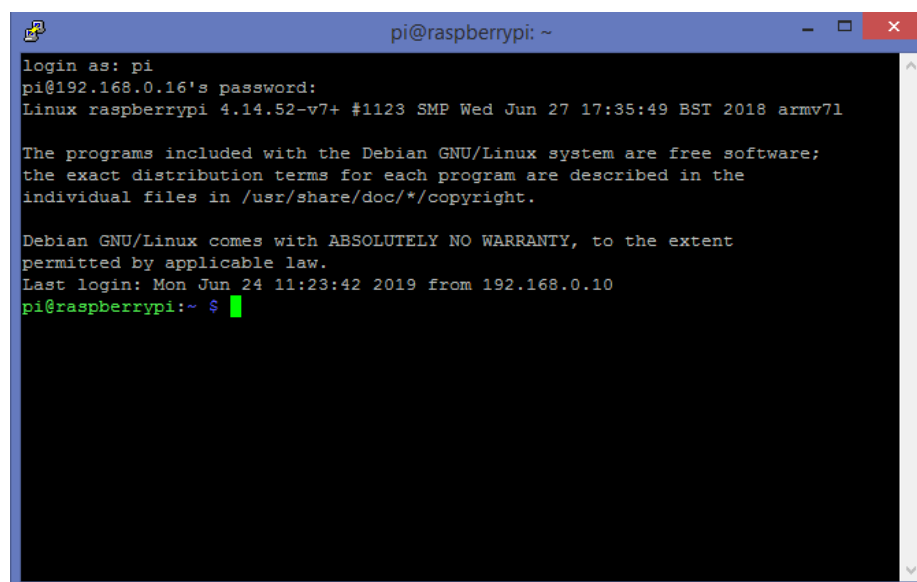


Figura 64. Ingreso a la Raspberry Pi desde el terminal PuTTY.

Fuente: [El Autor]

No es necesario ingresar como root al sistema ya que el usuario “pi” tiene los mismos privilegios de root, si se utiliza cualquier comando que requiera privilegios solo se debe

anteponer **sudo**. Para habilitar la interfaz *Wi-Fi* incorporada en el modelo Pi 3, es necesario modificar el fichero “**wpa_supplicant.conf**” ubicado en la ruta **/etc/wpa_supplicant/** conociendo el SSID de la red y la contraseña a la que se requiere conectar, tal como lo muestra el Anexo V.

Habiendo configurado correctamente el fichero, es posible en adelante, conectarse a una red *WLAN* por medio de la interfaz *Wi-Fi* existente en el modelo Pi 3, sin la necesidad de utilizar el cable de red que inicialmente se implementó. Esto evidentemente se traduce en flexibilidad y facilidad de operación del dispositivo a través del protocolo SSH.

El sistema operativo Raspbian no está totalmente listo, y se deben realizar algunas configuraciones que se detallan a continuación. Para continuar con las configuraciones se hace uso de la conexión inalámbrica *Wi-Fi* en el dispositivo, utilizando la configuración previamente realizada. Es necesario ingresar a la configuración inicial de Raspberry Pi, el comando que permite ejecutar esta acción mediante consola es: **sudo raspi-config** el cual despliega la ventana que se muestra en la Figura 65. En donde se indican los campos a ser intervenidos.

En el punto Nro. 1 se realizó el cambio de la contraseña “*raspberry*” que se encuentra por defecto.

En el punto Nro. 4 se estableció la zona horaria a la que pertenece el Ecuador, necesaria para generar información en función de la hora y fecha del sistema.

En el punto Nro. 5 se habilitó el uso del módulo cámara e interfaces de comunicación como: Puerto Serial y puertos GPIO que son utilizados en el desarrollo del programa para la central del sistema.

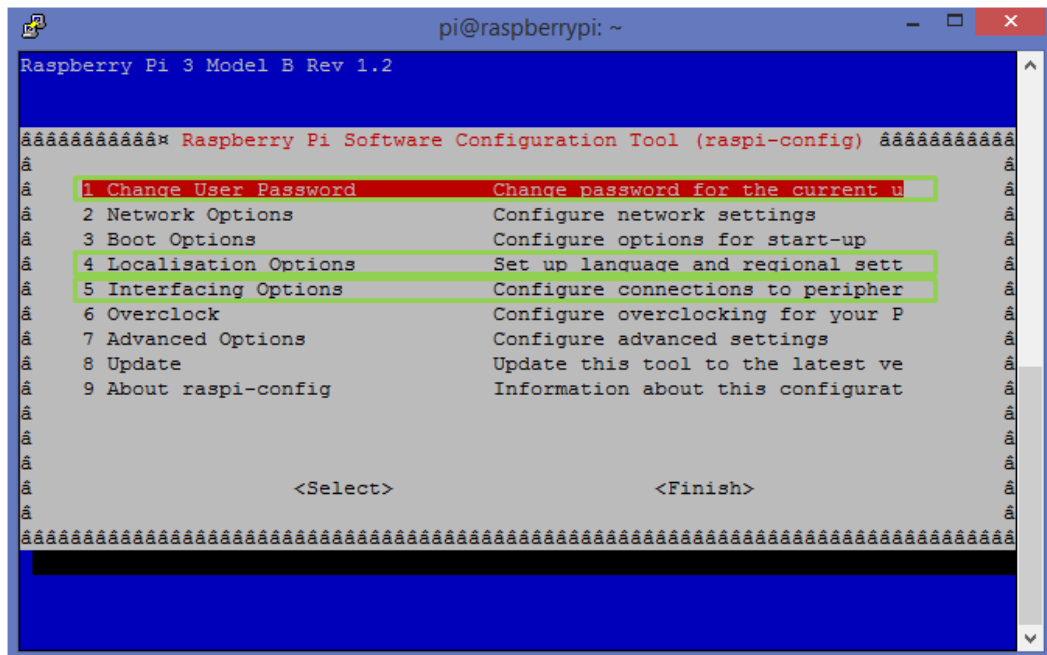


Figura 65. Menú de configuraciones generales de la Raspberry Pi.

Fuente: [El Autor]

Realizados estos cambios adicionales se procede a reiniciar la tarjeta y se actualiza la lista de paquetes del sistema a sus últimas versiones, a través de los comandos **sudo apt-get update** y **sudo apt-get upgrade**. En general, se debe hacer esto regularmente, con lo cual se mantiene actualizado el sistema operativo, ya que será el equivalente a la última imagen publicada disponible en <https://www.raspberrypi.org/downloads/>

5.2.4.4. Programación de la central del sistema

Teniendo en óptimas condiciones el sistema operativo, se procedió a elaborar los scripts que conformaran el cuerpo del programa principal, utilizados para definir la central del sistema en el lenguaje de programación Python. Estos scripts o códigos de programación se basan en el modelo adoptado para el desarrollo de *software*, denominado: “Modelo de Capas”, el cual está sumamente ligado a la construcción de aplicaciones complejas, desarrolladas mediante la programación orientada a objetos, permitiendo que las aplicaciones de *software* sean concebidas, desarrolladas y distribuidas en componentes.

Para poder hacer uso efectivo de este método, primeramente se definió los requerimientos y los alcances con los cuales se basó el *software* del sistema de seguridad propuesto. Abreviadamente se pudo tener una descripción general de estas especificaciones en el diagrama mostrado en la Figura 66.

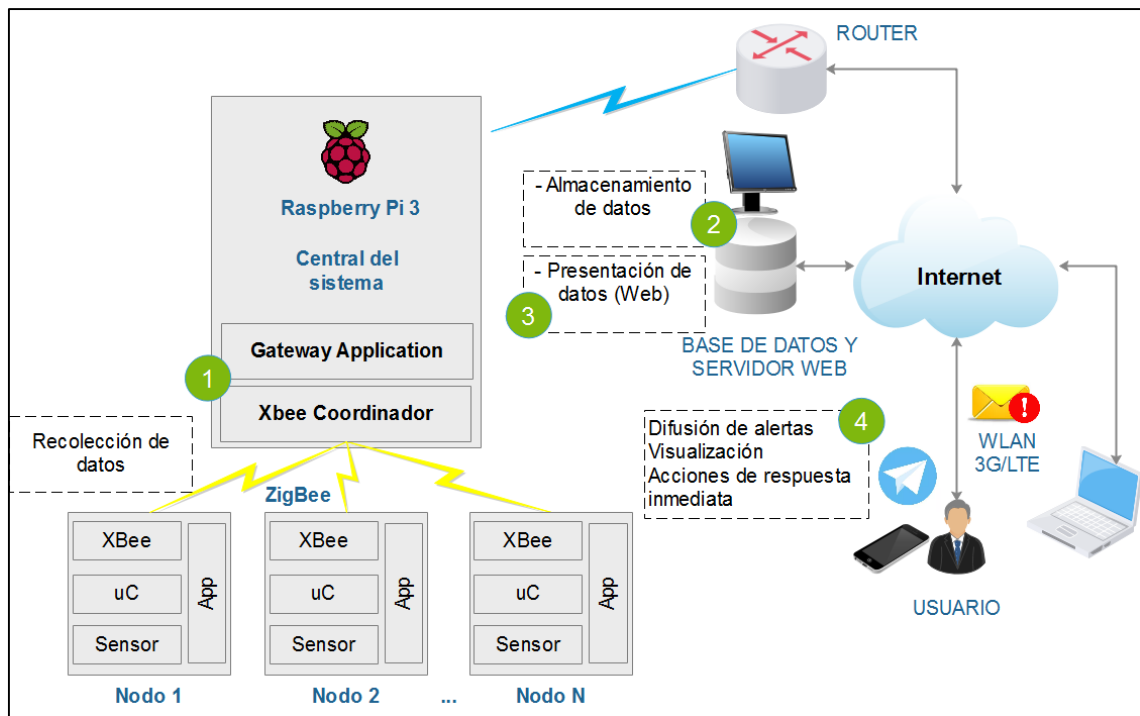


Figura 66. Descripción general de los requerimientos, que conforman la estructura del software implementado en la central del sistema.

Fuente: [El Autor]

Como se muestra en la Figura 66, existen 4 puntos elementales sobre los cuales se basa el desarrollo del *software* para la central del sistema de seguridad.

- En el primer punto se requiere el desarrollo de un algoritmo que permita recolectar la información proporcionada por la red de sensores inalámbricos basados en el estándar IEEE 802.15.4/ZigBee, al mismo tiempo que mantenga una comunicación directa para actualizar estados en la base de datos.
- El segundo punto comprende el almacenamiento de los datos generados por la red de sensores inalámbricos, para lo cual se ha implementado una base de datos del tipo no relacional o NoSQL basada en la plataforma Firebase.
- En el punto tres se requiere de la visualización de los datos almacenados en Firebase, para ello se hizo uso de la herramienta Firebase Hosting que proporciona un hosting seguro y rápido para las aplicaciones *web*, permitiendo también la sincronización con el microservicio alojado en esta plataforma, en este caso permite la sincronización con la base de datos alojada.
- Finalmente el punto cuatro requiere de una función específica, dedicada a realizar tareas de difusión de alertas y ejecutar acciones de respuesta inmediata, por medio de la aplicación Telegram, la cual fue seleccionada en base a la existencia de

múltiples librerías para muchos lenguajes de programación distintos, entre ellos Python, que permiten interactuar con la API de Bots de Telegram. Evitando de esta forma crear completamente una aplicación para realizar las difusiones de mensajes de alertas.

La adopción de estos puntos, no necesariamente indica la creación del *software* siguiendo esta secuencia, más bien refleja las necesidades y alcances básicos requeridos en el sistema. El desarrollo del *software* inicial parte desde esta perspectiva, pero la conformación del código, se fundamenta en función del crecimiento en la demanda de las diversas necesidades por cubrir, que surgen a medida que se avanza en el desarrollo de *software*.

▪ Creación de la base de datos en Firebase

Para crear una base de datos utilizando esta plataforma, es necesario tener una cuenta de Gmail, la cual permite *loguearse*²⁵ y acceder a todas las funcionalidades de esta herramienta en la dirección: [**https://firebase.google.com/**](https://firebase.google.com/)

Una vez en el entorno, al dar clic en “Ir a la Consola”, se despliega el *dashboard* (o panel de instrumentos). Firebase se compone por proyectos, que alojan bases de datos en tiempo real, *Hosting*, *Storage*, *Authentication*, etc., cuya gestión de datos se realiza a través del formato JSON.

El siguiente paso es crear un nuevo proyecto en el cual se aloja la base de datos, Firebase permite crear dos tipos de bases de datos denominados: *Cloud Firestore* (considerado como la nueva generación de *Realtime Database*, que permite entre otros aspectos realizar consultas más potentes) y *Realtime Database* que es la base de datos original de Firebase. Para el desarrollo del presente proyecto se decidió el uso de la primera opción y al alojarla en un nuevo proyecto denominado **test-9fb38**, éste se estructura tal como lo muestra la Figura 67 en donde se pueden observar los espacios de trabajos para comenzar con la gestión de los datos.

²⁵ Verbo empleado con frecuencia en el idioma español (no contemplado en el diccionario de la RAE) que hace referencia al “logging” que se utiliza, para la autenticación del ingreso a un servicio por medio de un usuario y contraseña

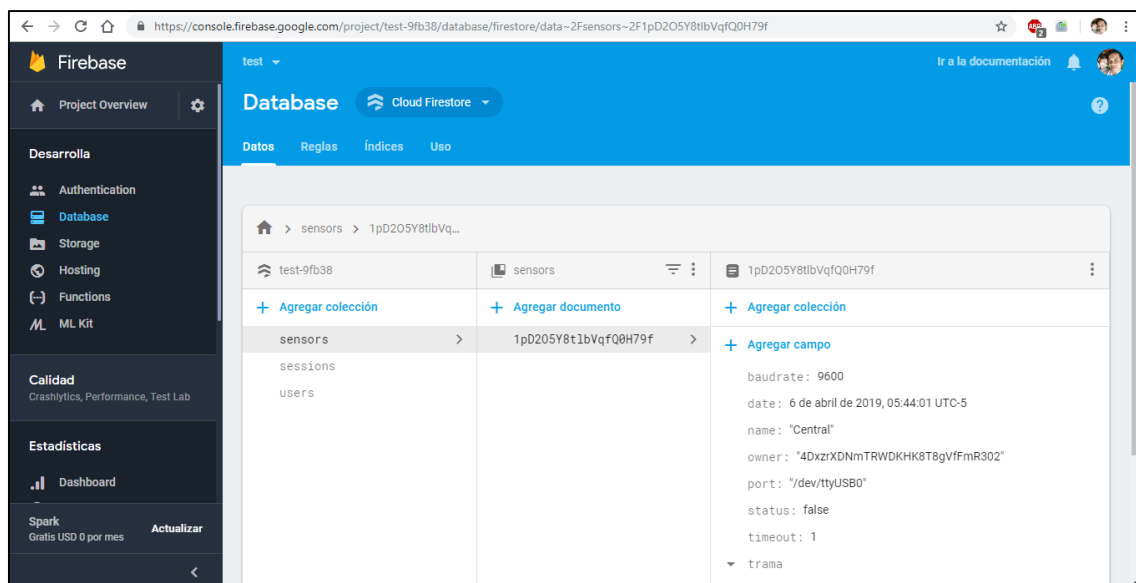


Figura 67. Creación de una base de datos del tipo Cloud Firestore en Firebase.

Fuente: [El Autor]

Para complementar las funcionalidades de Firebase en Python (Dentro de la Raspberry Pi), es necesario descargar su librería a través del comando: “**sudo pip install firebase-admin**”.

▪ Creación del Bot de Telegram

Telegram ha añadido el sistema de Bots a su entorno, cuya API²⁶ se comunica con los servidores de Telegram vía el protocolo HTTP y transmite los paquetes en formato JSON, a partir de eso, la comunidad de programadores ha generado una API para Python que es: “python-telegram-bot”, existen otras APIS en este lenguaje, pero la mencionada es la que se utilizó para este proyecto, debido a que es la más completa en funcionalidades, hoy en la actualidad.

Para la creación del bot, lo primero que hay que realizar, a través de la App de Telegram, es buscar al BotFather utilizando el buscador de Telegram, una vez ubicado se inicia una conversación con él.

Al iniciar una conversación se despliega una serie de opciones de las cuales se selecciona “/newbot”, al hacerlo pedirá que se le agregue un nombre con la terminación en “bot” en este caso se decidió llamarlo “jonascbot”. Tras comprobar que todo esté correcto,

²⁶ Application Programing Interfaces (Interfaz de programación de aplicaciones) que representa la forma en como un módulo de software se comunica o interactúa con otro.

BotFather devolverá el *token*²⁷ de conexión, que es utilizado en el código de programación en Python que forma parte de la central del sistema. La Figura 68 resume los pasos seguidos para la creación del Bot.

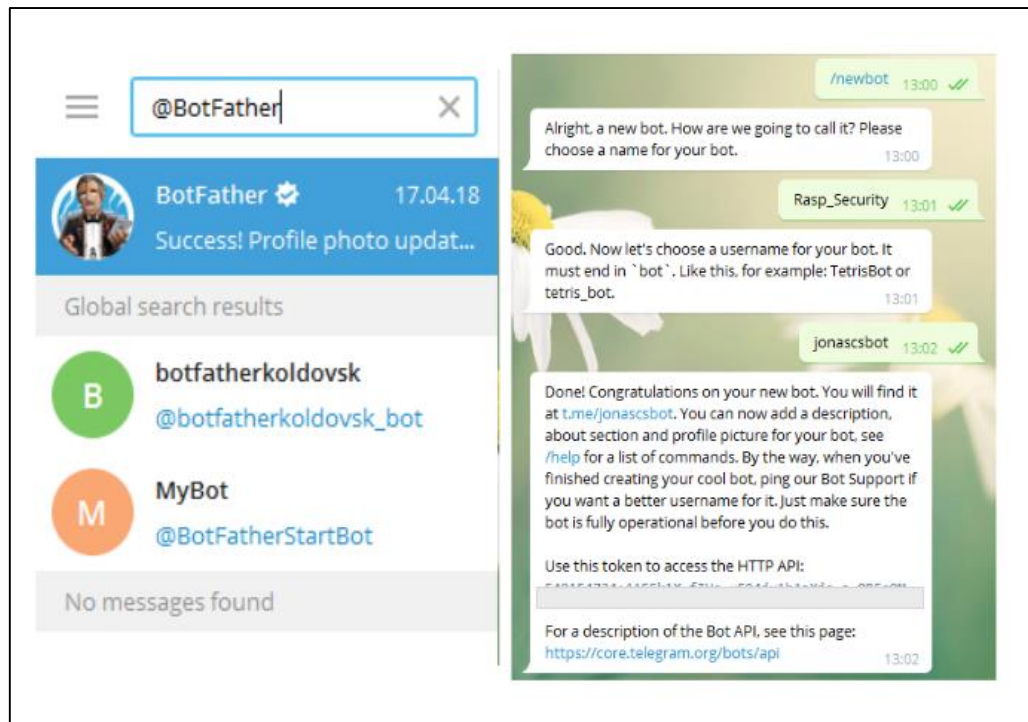


Figura 68. Creación de un Bot, generado a través de la App. de Telegram y BotFather.

Fuente: [El Autor]

Al igual que con Firebase para completar las funcionalidades del bot en Python se descarga la librería correspondiente con el comando: **sudo pip install python-telegram-bot**

Basados en las especificaciones de la Figura 66 y con el uso de las plataformas definidas, se comenzó con el desarrollo del *software* que se implementó en la tarjeta Raspberry Pi utilizando el lenguaje de programación Python V3.5. Como los requerimientos para el *software* son diversos, y se requiere el uso de un sinnúmero de librerías, la estructuración del código, se realizó a través de la jerarquización en módulos y paquetes de un proyecto Python (Figura 69).

²⁷ Cadena de caracteres que tienen un significado coherente en cierto lenguaje de programación, entregado a un usuario autorizado de un servicio computarizado, para facilitar el proceso de autenticación.

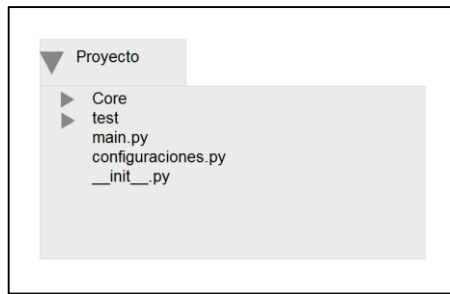


Figura 69. Estructura general de módulos y paquetes que conforman un proyecto en Python.

Fuente: [El Autor]

Para crear paquetes en Python es necesario realizar dos pasos importantes:

1. Informar a Python que un directorio es un paquete.
2. Crear en ese directorio un archivo imprescindible de nombre `__init__.py`

La descripción de paquetes utilizados en el desarrollo del *software*, se hace a continuación. Se fue creciendo en el desarrollo del *software*, desde los elementos o partes del código más esenciales y fáciles de diseñar, para luego introducirse a las estructuras de adquisición más complejas, muchas de las cuales absorben funcionalidades de las partes de código desarrolladas en un inicio.

De esta forma se tiene la creación de los siguientes paquetes que conforman el paquete Core (o núcleo del sistema):

- **Paquete Utils**

Utils es un paquete que contiene un módulo llamado `Bootutils.py` el cual sirve para ser consumido por otros módulos dentro de la estructura de adquisición de datos, principalmente por los módulos relacionados al Bot de Telegram que se describen más adelante. Tiene funcionalidades que pueden ser accedidas desde cualquier parte del código. Las dos funciones principales declaradas dentro de este módulo son: verificar la escritura de comandos “`comand_is()`” útil en la interacción con la App de Telegram, para devolver los valores respectivos en función del comando, y verificar si un comando está en la lista de comandos permitidos “`comand_is_permited()`”, los cuales trabajan directamente con el diccionario de datos “`comandos_telegram`”, presente en el módulo de configuraciones al que en este caso se lo ha denominado `configApp.py`, que se desarrolló paralelamente a la creación de este paquete, el cual contiene las configuraciones básicas del sistema de seguridad.

El módulo `configApp.py` se encuentra alojado en el directorio raíz del proyecto, y entre otros elementos, abarca un diccionario de datos para los comandos de Telegram, el *token* del Bot utilizado, las credenciales para la conexión con Firebase y mensajes pre configurados para el Bot.

La Figura 70 muestra la estructura del paquete Utils, y la relación existente con el diccionario de datos existente en el módulo configApp.py alojado en el directorio raíz del proyecto.

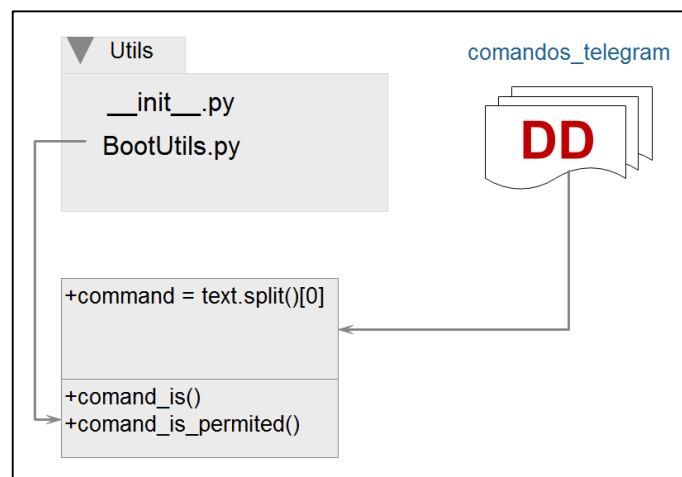


Figura 70. Estructura del paquete Utils.

Fuente: [El Autor]

- **Package models**

Contiene el modelo de la base de datos no relacional (NoSQL) representada en código, a través de los módulos construidos, para llamar a las funciones de la base de datos (Firebase). Este paquete se encuentra estructurado por los módulos `SensorModels.py`, `SessionsModels.py`, `TramaModels.py`, `UserModels.py` que se traducen en los modelos que conforman la base de datos (o de las tablas si se tratase de un tipo de base de datos SQL).

Por ejemplo `SensorModels` como tal, hereda los atributos y métodos de la superclase `Firestore_admin`. Contando de igual forma con los siguientes atributos propietarios como:

- Un nombre
- Un puerto asociado
- La fecha de activación
- El Baudrate (velocidad de transmisión con la que se reciben los datos)
- Una trama

- Un propietario “owner”
- Y un estatus que indica el estado del sensor

También cuenta con la función para obtener la representación de un objeto de su mismo modelo, en forma de diccionario “from_dict(source)” y otra función para retornar un diccionario del objeto “to_dict(self)” creado por la función anterior. En términos generales cada módulo representa la construcción de un objeto, representado en forma de diccionario, esto se debe a que el formato JSON adoptado por Firebase, requiere de la información en este tipo de formato, el cual es necesario para identificar un valor cualquiera dentro de la base de datos a través de una clave. La Figura 71 representa la estructura del paquete models. Todos los módulos contenidos en este paquete requieren del archivo configApp.py que aloja las credenciales para las operaciones con Firebase.

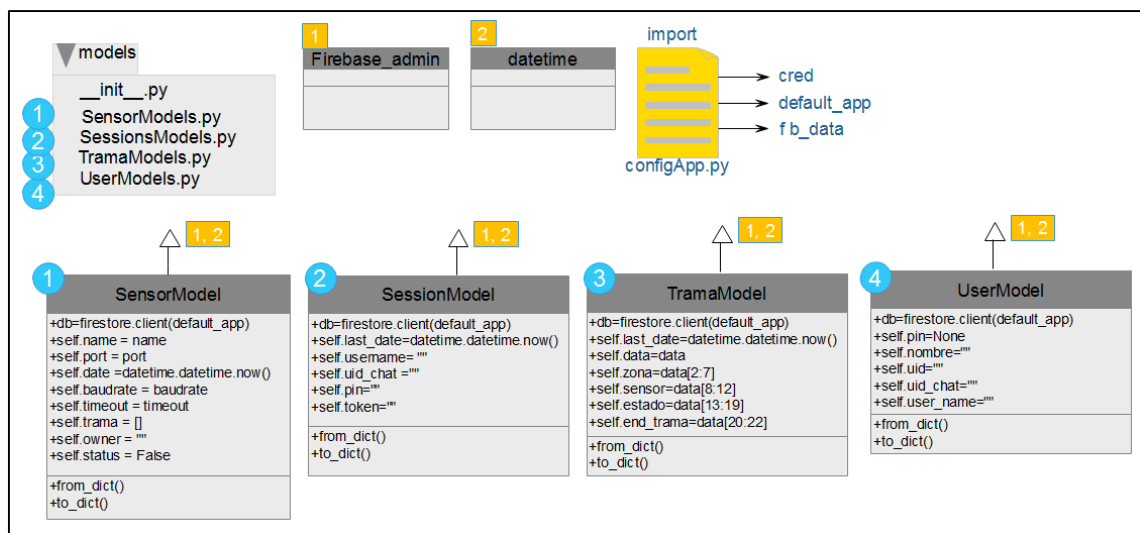


Figura 71. Estructura jerárquica del paquete models, que contiene el modelo de la base de datos (NoSQL) representado en código.

Fuente: [El Autor]

▪ Paquete services

Como tal; es el paquete que consume de models, siendo el encargado de la comunicación con Firebase, en términos descriptivos es el puente de conexión existente entre el modelo de la base de datos contenida en models y Firebase, dependiendo de los datos alojados en models, services se encarga de crear la gestión de datos con Firebase. Comunicarse con Firebase, actualizar, consultar, modificar, etc., son algunas de las tareas que realiza services.

Por ejemplo la clase `SensorService`, es un servicio para manejar la estructura de `SensorModel` y hereda todos los atributos de esa clase, permitiendo operar a nivel de servicios con las funciones “obtener sensor”, “setar_datos”, “chequear sensor” (útil para saber si existe o no), “guardar sensor”, “callback_sensor”, “escribir en el sensor”, entre otros. De esta forma cualquier funcionalidad adicional que se desee agregar y que se encuentre relacionada al sensor se lo debe hacer en esta clase.

De la misma forma `SessionService` ocupa del modelo `SessionModel`, en el cual se tiene las funciones: “actualizar sesión”, “obtener sesión”, “guardar sesión” que han sido consideradas como las más elementales. Para `UserService` quien hereda los atributos de `UserModel` tenemos las siguientes funciones “actualizar un usuario”, “guardar un usuario”, “obtener un usuario”, “obtener un usuario para un uid_chat”, “eliminar un usuario”.

Como se puede observar, al tener una clase por cada módulo representado en `models`, también se tiene una clase por cada módulo representado en `services`, Trama no es la excepción y también tiene su módulo representado en `services`, la clase `TramaService` derivada de su módulo, se incluye en `SensorService` la cual tiene como objeto recuperar la trama de datos enviada por cada sensor existente en la WSN y verificar un cambio de estado que desencadene la difusión de alertas al usuario.

La estructura del paquete `Services` se muestra en la figura 72.

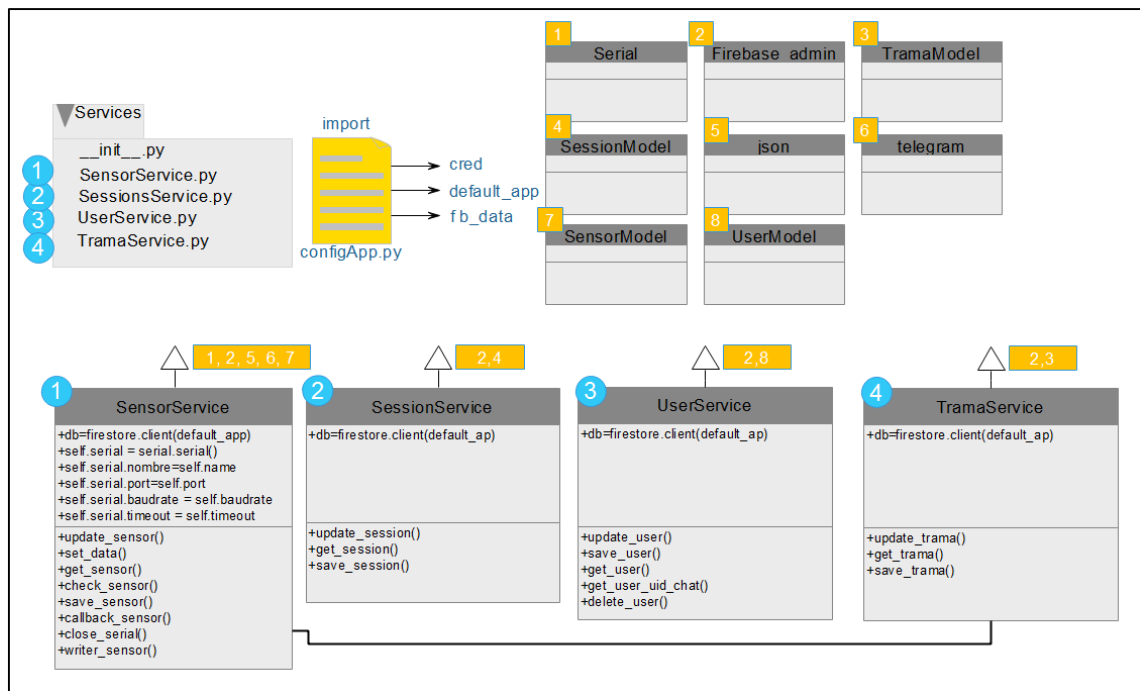


Figura 72. Estructura del paquete Services, y su relación de herencia con las clases de los módulos representados en el paquete models.

Fuente: [El Autor]

Hasta este punto se puede evidenciar una conexión bastante fuerte entre Services y models, haciendo indispensable que Srvices ocupe de models para comunicarse con Firebase.

Los puntos 1 y 2 requeridos en la Figura 66, se encuentran solventados con la implementación del código presente en estos paquetes, pues la lectura de sensores ha sido incluida en la parte de servicios que conectan directamente a Firebase, el motivo por el cual se lo implementó aquí, principalmente se debe al tema de almacenamiento en la base de datos, ya que se requería que los sensores estén constantemente enviando información hacia ella, y como Services es el encargado de hacer todo ese proceso, a la lectura de los datos se la incluyó en esta etapa.

▪ Paquete controller

En este paquete se encuentra estructurada la programación que controla el funcionamiento del módulo relé de 4 canales y la cámara conectada a la tarjeta Raspberry Pi. Los módulos existentes en este paquete son Actuador.py y camara.py.

Actuador.py a través de su clase Actuador define el comportamiento de los pines del puerto GPIO específicamente los pines GPIO 02, 03, 04, 17 los cuales son activados o desactivados al momento de presionar los botones [On/Off Luces Internas], [On/Off

Luces Externas], [On/Off Suministro de Gas], [On/Off Suministro Eléctrico] respectivamente, que son llamados mediante la función “cmd_acciones” presente en el Boot.py alojado dentro del paquete BootRaspSec la cual es accedida desde la aplicación de Telegram.

Del mismo modo la clase Camara alojada en el módulo Camara.py define las acciones a ejecutarse por este importante elemento. Las funciones principales dentro de esta clase son “tomar_foto”, “obtener ruta de almacenamiento” y “subir foto hacia base de datos”.

El módulo Boot.py también accede a las funcionalidades de la clase Camara, lo hace a través de la función “cmd_foto” la cual es accedida desde la aplicación de Telegram que da como resultado la devolución de una instantánea capturada, después de la petición realizada por el usuario. En la Figura 73 se muestra la estructura del paquete Controller.

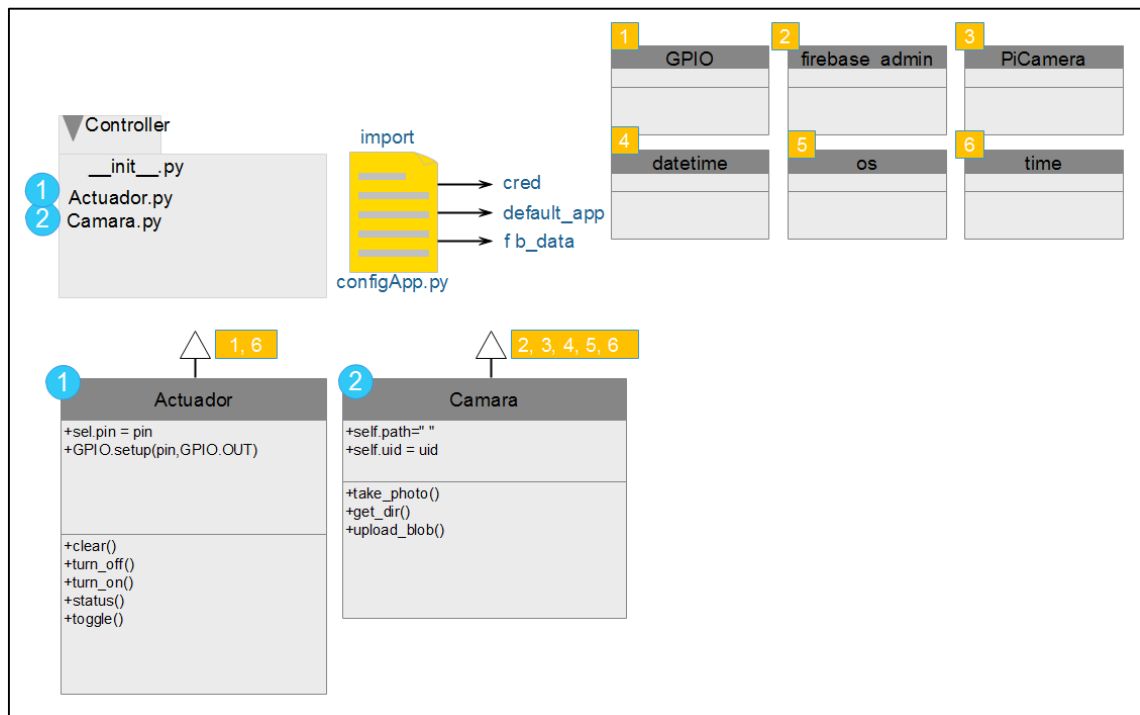


Figura 73. Estructura del paquete Controller y sus relaciones de herencia con las superclases mostradas.

Fuente: [El Autor]

▪ Paquete BootRaspSec

Cuando el archivo main.py llama a todo el paquete de Core, lo primero que hace es llamar a BootRaspSec el cual a través de su módulo Boot.py se encarga de manejar las operaciones conjuntamente con Loginuser.py y wraps.py (adornos o decoradores en python). Este paquete es el encargado de manejar las operaciones con todo lo relacionado

al mundo de Telegram, al ejecutarse el Bot, inmediatamente se encuentra escuchando, y cuando un usuario hace peticiones al API de Telegram, éste los re direcciona al Boot.py. La comunicación existente entre el Bot y el API de Telegram es bidireccional.

Al recibir las peticiones hechas por un usuario, Boot.py se convierte en el orquestador de funciones, encargado de designarlas a los paquetes y módulos creados anteriormente, tal es el caso del paquete Utils el cual permite verificar si el comando existe o no, o si fue escrito correctamente, también hace uso de services para identificar si el usuario que realiza la petición se encuentra logeado o no, desencadenando las operaciones descritas en el paquete services, quien en términos generales es el puente de conexión entre Firebase y el modelo de base de datos establecido en models. Boot.py también designa funciones a Controller para manejar los periféricos conectados a la Raspberry Pi como el módulo relé de 4 canales y la cámara los cuales son accedidos directamente desde la aplicación Telegram.

En otras palabras cuando se ejecuta el archivo principal main.py lo que se hace a través del módulo Boot.py es lanzar un demonio que tiene como función, escuchar las peticiones realizadas por un usuario a través del API de Telegram. El Bot creado se comunica con este API por medio de una sencilla interfaz HTTP, es decir, la comunicación que se describe en este punto, es la comunicación con los servidores de Telegram.

La estructura del paquete BootRaspSec se muestra en la Figura 74.

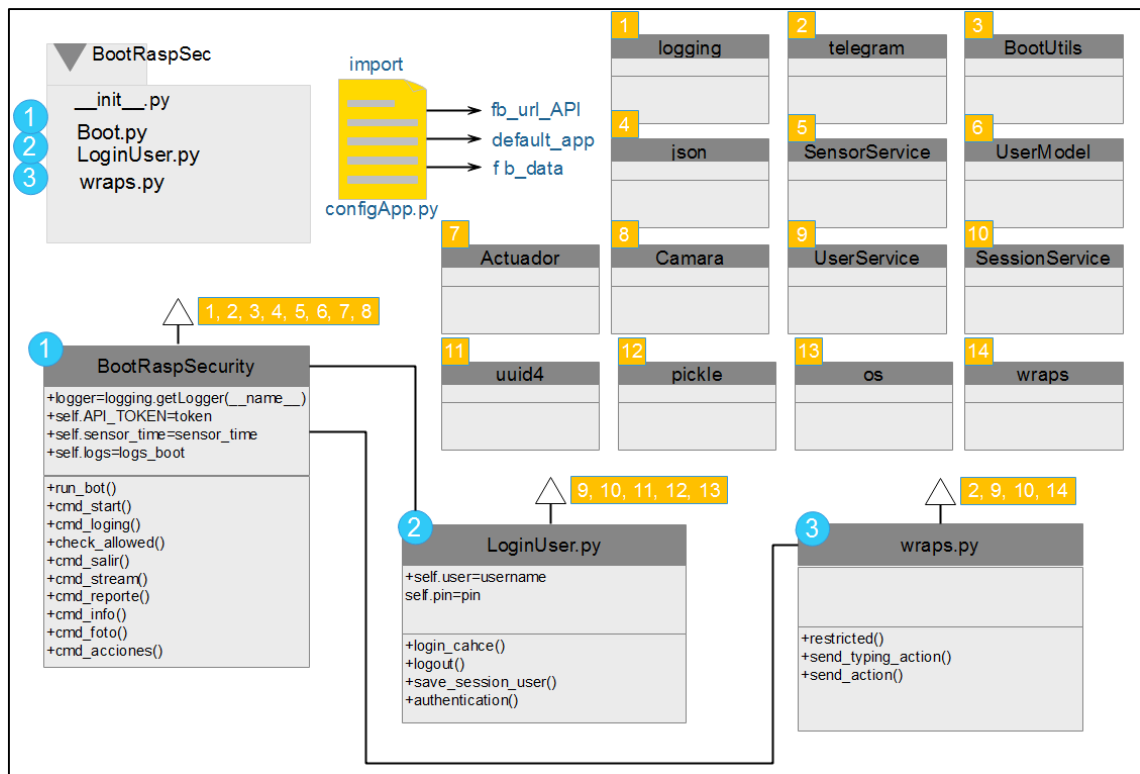


Figura 74. Composición del paquete BootRaspSec.

Fuente: [El Autor]

Habiendo realizado la estructura del código, correspondiente al núcleo del sistema (Core). Se procedió a la elaboración del archivo main.py que en este caso se lo denominó RaspSec.py el cual se encarga de lanzar toda la estructura de código desarrollada en los paquetes anteriores, conjuntamente con el módulo de configuraciones.py denominado configApp.py. La estructuración de la jerarquía en módulos y paquetes de un proyecto Python mostrada en la Figura 69, finalmente queda conformada como lo muestra la Figura 75. De igual forma, el diagrama general del funcionamiento del sistema se muestra en la Figura 76 en el cual se representan las interconexiones existentes entre los diversos paquetes y cómo se encuentra estructurado el programa de la central del sistema de seguridad para la adquisición, gestión y difusión de datos.

En esta representación, también se muestra el paquete de pruebas unitarias “Test”, utilizado para determinar si un módulo o un conjunto de módulos funcionan correctamente, mismo que fue implementado a la par con el desarrollo del proyecto. En otras palabras, este paquete es una forma de comprobar que un conjunto de módulos con sus respectivas clases, funcionen como se espera. En este caso se realizó las pruebas unitarias para verificar el funcionamiento del módulo de Logging con Firebase, la lectura

de los sensores, manejo de tramas y la autenticación de un usuario. Lógicamente, las pruebas unitarias nunca pueden garantizar completamente el correcto funcionamiento de una porción de código. No obstante, son capaces de detectar gran cantidad de anomalías y permiten ahorrar tiempo en la depuración del código.

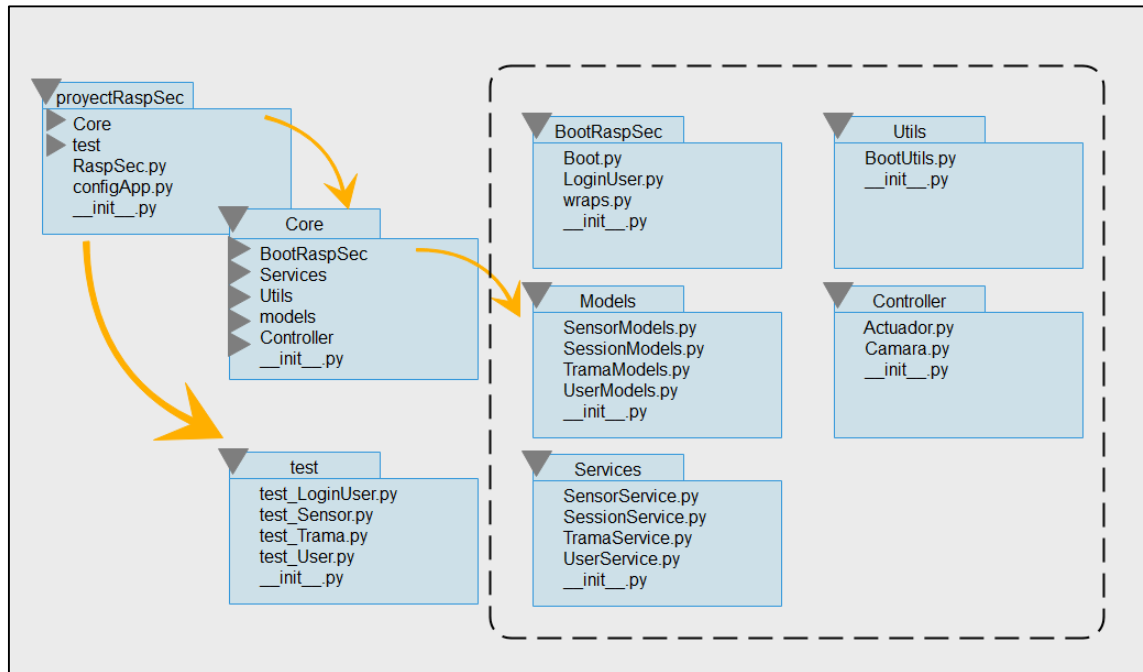


Figura 75. Estructuración jerárquica de módulos y paquetes adoptada, para el sistema de seguridad propuesto.

Fuente: [El Autor]

El código realizado se encuentra debidamente documentado y disponible para su análisis en el repositorio descrito en el Anexo VI.

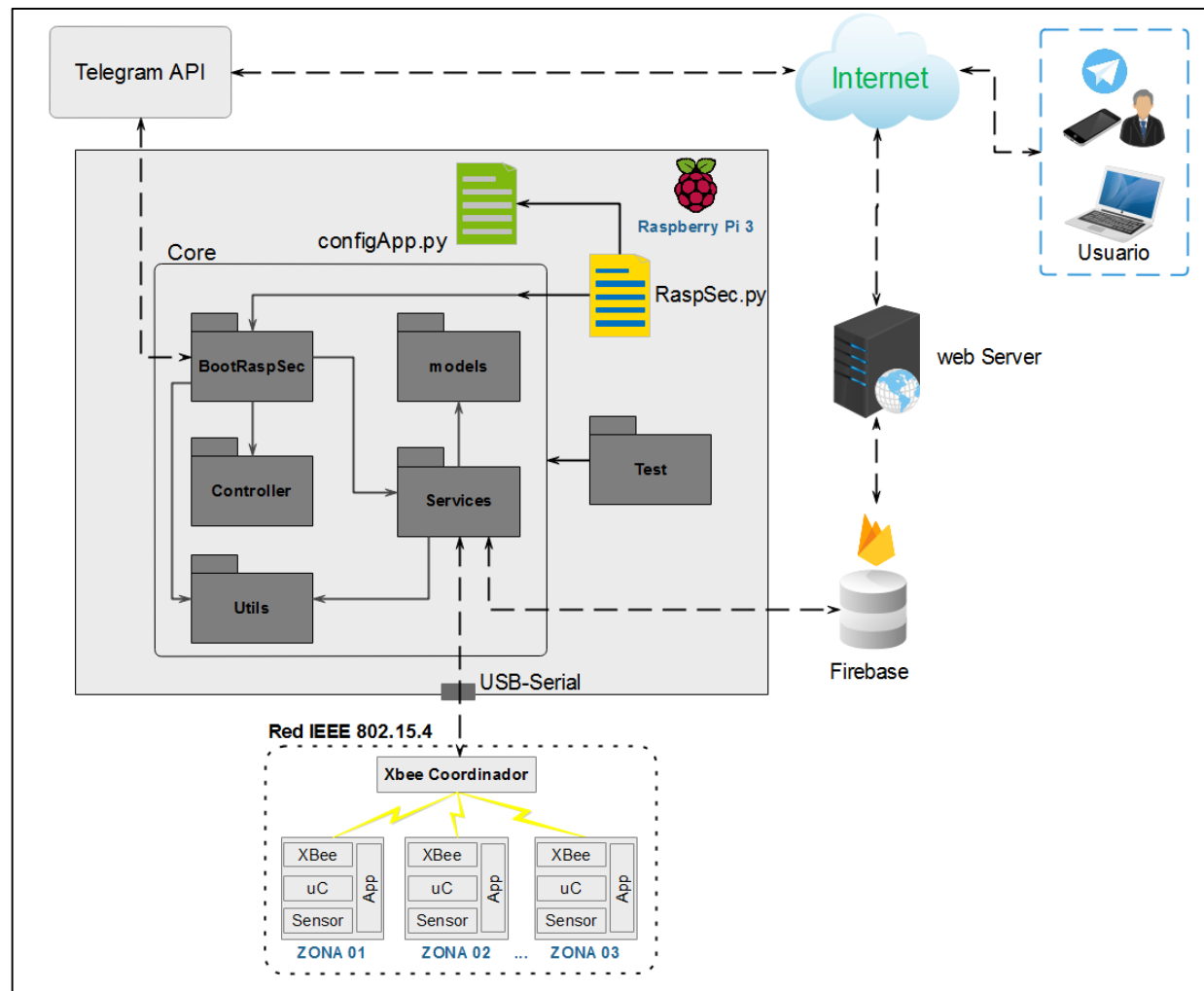


Figura 76. Diagrama general del funcionamiento del sistema, y su arquitectura de adquisición, gestión, y difusión de datos.

Fuente: [El Autor]

5.2.4.5. Servidor Web

Para realizar la interfaz gráfica de usuario se ha creído conveniente utilizar Angular como *framework* Web, ya que se ajusta mejor a las necesidades y con visión a escalabilidad a futuro. Angular es un *framework* en JavaScript que permite crear páginas web bajo “*Single Page Application*” (SPA) a través de Google. Entre las ventajas más significativas citamos: la velocidad de carga, al inicio un poco tardía, pero conforme se va gestionando la navegación va tornándose más rápida e instantánea, otra ventaja a mencionar es que angular lleva integrado un servidor web, lo cual permite visualizar y usar el proyecto en desarrollo sin la necesidad de cualquier otro software (excepto de un navegador).

- **Tecnologías implementadas**

Esta parte del proyecto se generó con las siguientes herramientas:

- ✓ Angular CLI: 7.0.2
- ✓ Node: 10.8.0
- ✓ Typescript: 3.2.4
- ✓ Angular: 7.0.2
- ✓ Firebase: 5.10.1

Para levantar la estructura del proyecto es necesario instalara las siguientes herramientas:

Tabla 20. Instalación de herramientas necesarias para empezar un proyecto en Angular.

Node JS 10.X (Incluye NPM)	
Windows	Linux
https://nodejs.org/es/download/	<pre>curl -sL https://db.nodesource.com/setup_10.x sudo -E bash - sudo apt-get install -y nodejs</pre>
CLI Angular	
<pre>npm install -g @angular/cli</pre>	<pre>npm install -g @angular/cli</pre>
Firebase	
<pre>npm install -g angularfire2</pre>	<pre>npm install -g angularfire2</pre>

Fuente: [El Autor]

Una vez instaladas estas herramientas se procede a construir un nuevo proyecto desde la línea de comandos digitando: **ng new raspsecurityweb**

El resultado se muestra en la Figura 77. Hasta este punto se ha creado una estructura de ficheros que se va a necesitar junto con una cantidad de herramientas ya configuradas. Como se puede observar existe una gran cantidad de ficheros en donde cada uno cumple una función específica, en este caso se abordará los ficheros fundamentales (Figura 78).

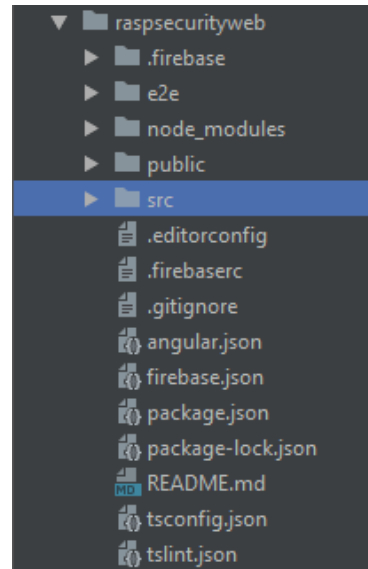


Figura 77. Estructura del proyecto para el servidor web generado con Angular.

Fuente: [El Autor]

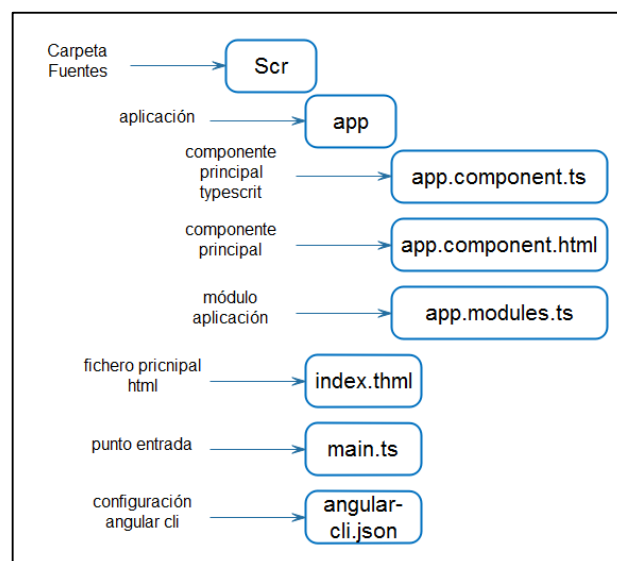


Figura 78. Ficheros fundamentales de un proyecto generado con Angular.

Fuente: [El Autor]

El primer archivo que destaca es el **index.html**. Este es el fichero de entrada a la aplicación y contiene varias líneas de código. El código más importante que contiene se encuentra ubicado en el *body*.

```
<body>
<app-root></app-root>
</body>
```

El *body* dispone de una etiqueta especial “<app-root></app-root>” la cual se encarga de definir el componente principal de Angular, esto debido a que angular es un *framework* que tiene un fuerte enfoque de construcción de componentes visuales con **TypeScript**. De esta forma el fichero Index.html referencia al componente raíz

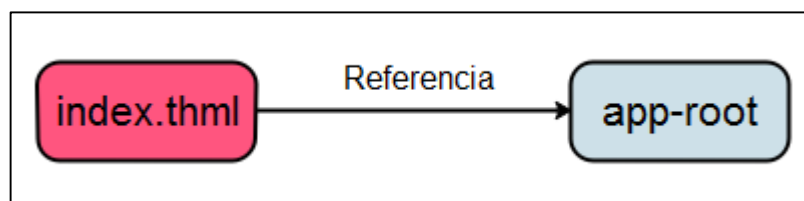


Figura 79. Referenciación del componente raíz por medio del fichero Index.

Fuente: [El autor]

El componente raíz se encuentra ubicado en la carpeta app denominado **app.component**

Angular divide el componente en varias partes

- **app.component.css:** Define el estilo del componente
- **app.component.html:** Define la plantilla del componente
- **app.component.ts:** Define la funcionalidad del Typescript del componente
- **app.component.ts:** Define el módulo al que pertenece el componente

Los componentes creados para el presente proyecto se alojan en las carpetas **pages** y **services** respectivamente, **pages** contiene la estructura de la interfaz de usuario compuesta por **config/dashboard** y **services** contiene la estructura de comunicación para el acceso a la base de datos Firebase.

Es en el fichero **app.component.ts** donde Angular enlaza la etiqueta <app-root> con el nombre del componente, o en este caso con los nombres de los componentes creados.

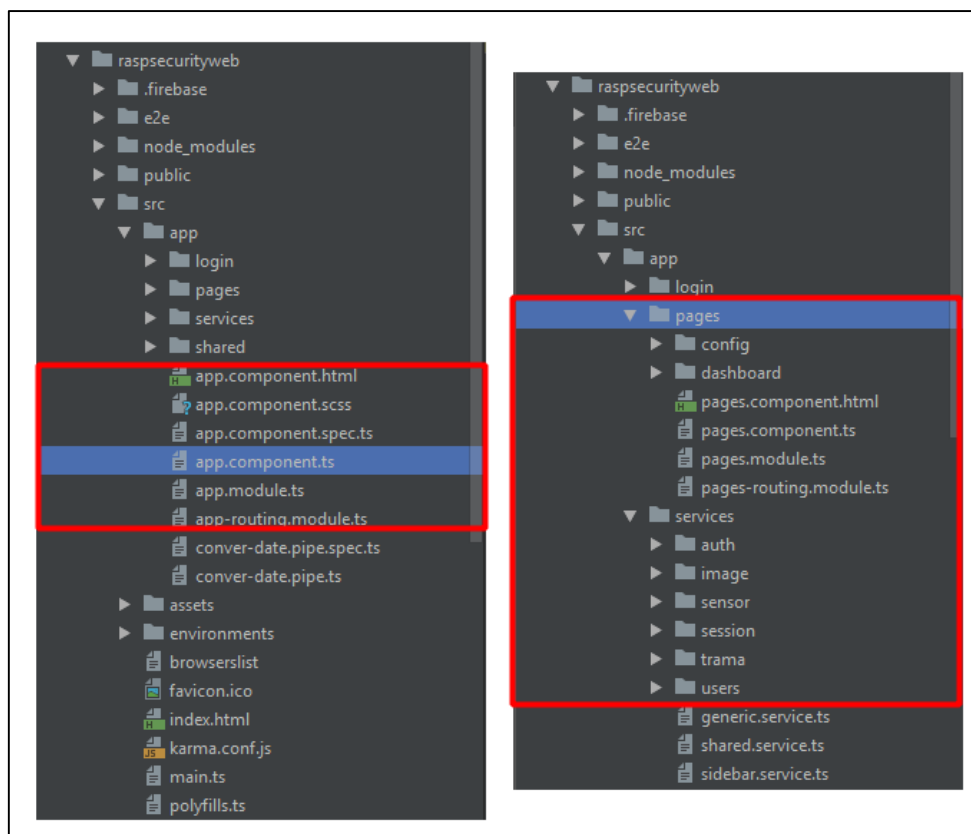


Figura 80. Componente raíz y componentes generados para el desarrollo del proyecto.

Fuente: [El Autor]

De esta forma es como se encuentran relacionados los componentes creados para el proyecto y el `index.html`. A continuación el fichero **main.ts** define cual módulo es el de arranque. Este fichero a través de la propiedad de *bootstrap*²⁸ indica que componente es el de arranque. En una aplicación Angular, en primera instancia se carga el fichero **main.ts** que define cual es el módulo de arranque. Angular accede al módulo de Arranque y busca cual es el componente inicial que se debe arrancar y lo arranca. Este componente dispone de un decorador “selector”: `app-root` que indica al *framework* que etiqueta le identifica. La etiqueta en mención es la etiqueta que se encuentra en el fichero `index.html` de cada componente creado.

²⁸ Es un framework desarrollado y liberado por twitter cuyo objetivo principal es facilitar el diseño web de forma rápida y sencilla, incluyendo plantillas de diseño basadas en HTML y CCS para diversos usos.

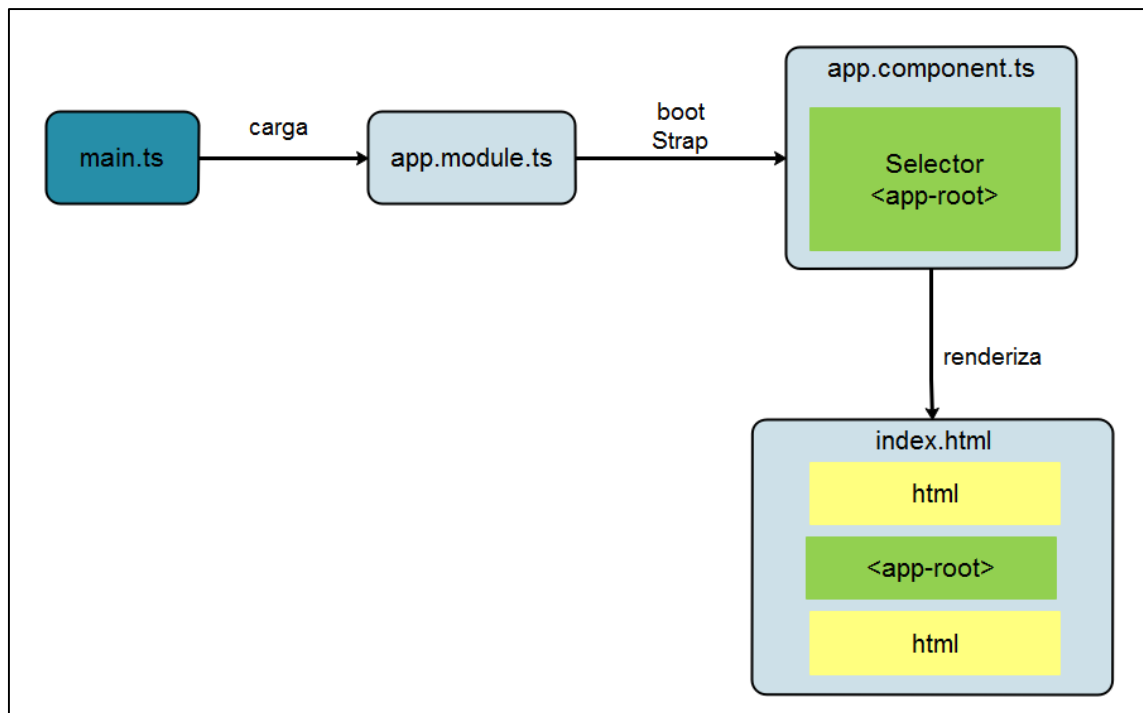


Figura 81. Función del fichero main.ts encargado del arranque de módulos en Angular.

Fuente: [El Autor]

El código realizado se encuentra debidamente documentado y disponible para su análisis en el repositorio descrito en el Anexo VII

- **Ejecución (Despliegue local)**

Desde la carpeta del proyecto se ejecuta el comando: **ng serve**. Al abrir el navegador, e ingresar la siguiente dirección **http://localhost:4200/** es posible visualizar la aplicación creada.

6. RESULTADOS

En este apartado, se presentan los resultados de los experimentos realizados para comprobar el funcionamiento del sistema y sus interfaces, verificando su consistencia. A continuación se detallan los resultados obtenidos que muestran los servicios más importantes desarrollados. En la Figura 82 se muestra la conformación final de la central del sistema y en las Figuras 83, 84, 85 se muestran los respectivos dispositivos finales implementados.



Figura 82. Central del sistema de la WSN basada en el estándar IEEE 802.15.4.

Fuente: [El Autor]

Se logró una operación completa por parte de la central de alarmas (tarjeta Raspberry Pi), que conecta diferentes dispositivos simultáneamente y ejecuta funciones en tiempo real intercambiando datos. Además, se comprobó el funcionamiento de la difusión de alertas en tiempo real y la ejecución de acciones de respuesta inmediata implementadas. Los resultados obtenidos, fueron positivos. En este trabajo se utilizaron diferentes tecnologías, lenguajes de programación, plataformas y sistemas operativos principalmente basados en el concepto de software libre. Los objetivos planteados se han logrado con gran éxito, desde una perspectiva práctica, como teórica. Es importante destacar que el trabajo

desarrollado es un primer enfoque en un área que tiene un alto potencial de crecimiento y que, en términos del objetivo principal de esta tesis, logró su propósito.



Figura 83. Dispositivos finales ZigBee: ZONA 01 (Sensor Magnético), ZONA 02 (Sensor de Humo).

Fuente: [El Autor]



Figura 84. Dispositivos finales ZigBee: ZONA 03 (Sensor PIR), ZONA 04 (Sensor PIR).

Fuente: [El Autor]



Figura 85. Dispositivos finales ZigBee: ZONA 05 (Sensor PIR), ZONA 06 (Botón de Pánico).

Fuente: [El Autor]

6.1. Pruebas de recepción de señal en la topología de Red IEEE 802.15.4 adoptada

Para tomar las lecturas del nivel de recepción de señal de cada ZONA respecto de su coordinador, se realizó las pruebas dentro de la Infraestructura de la Empresa TECSERLED teniendo en cuenta que las distancias de cada Nodo final hacia el elemento coordinador, están en función de la distribución mostrada en el Anexo I. Los resultados de estas pruebas se muestran en la Figura 86.

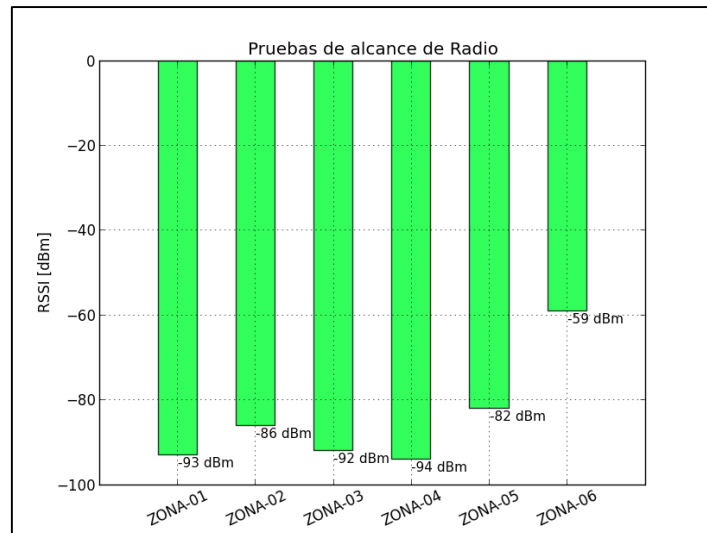


Figura 86. Nivel de RSSI obtenido por cada Zona perteneciente a la WSN implementada respecto de su coordinador.

Fuente: [El Autor]

Estas pruebas respectivas se realizaron haciendo uso de una Computadora portátil, dos módulos XBee-S2C (El elemento coordinador, dispositivo final) y una batería de 9V como fuente de alimentación del dispositivo final. Por medio del software XC-TU se obtuvieron los niveles *Receive Signal Strength Indication* (RSSI) a través del análisis “Range Test” presente en el menú de herramientas del programa. Los niveles de RSSI mostrados en la Figura 86, corresponden a los valores obtenidos por cada ZONA mediante este software. Adicional a ello, dentro del mismo análisis es posible obtener el porcentaje de paquetes recibidos con éxito, detalle que se muestra en la Figura 87.

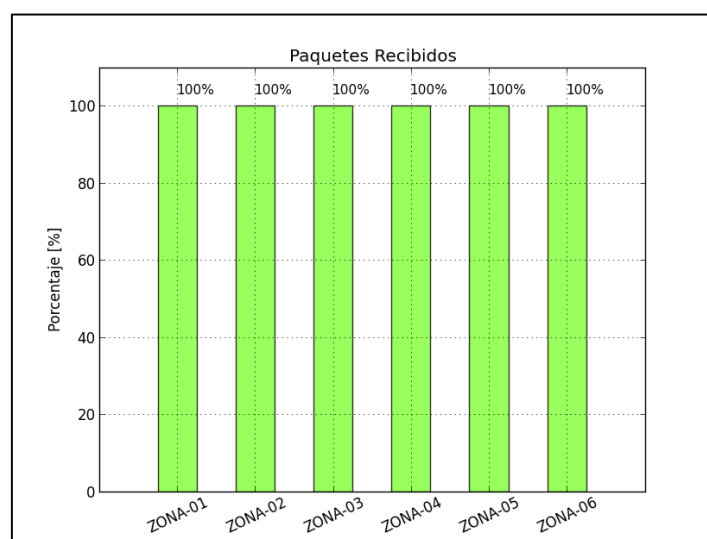


Figura 87. Nivel de confianza en la recepción de información, por cada zona perteneciente a la WSN.

Fuente: [El Autor]

Con el nivel de recepción mostrado y el porcentaje total de tramas recibidas por cada zona, se procedió a verificar la comunicación existente entre los dispositivos finales y la veracidad de los datos entregados por cada uno de los sensores hacia la central de alarmas.

6.2. Pruebas de verificación de comunicación entre dispositivos finales y la central de alarmas.

Para verificar el correcto desempeño del sistema se hicieron pruebas durante 7 días, con un número total de análisis correspondiente a 8 horas por día. Para realizar estas pruebas durante prolongados tiempos, fue necesaria la implementación de adaptadores de voltaje DC de 12V. La verificación de comunicación existente entre los dispositivos de la red de sensores inalámbricos bajo el estándar IEE 802.15.4 y la central de alarmas fue llevada a cabo, mediante la confirmación de recepción de tramas del estado normal de sensores, descrito en la etapa de software; relacionada a la programación del microcontrolador PIC18F25K20. En esta sección se detallaba el envío de datos del estado normal del sensor cada cierto intervalo de tiempo igual a 10 segundos.

Este dato preliminar, permitió establecer el número total de tramas a recibir (Teórico=2880) por cada sensor durante las 8 horas de análisis, de esta forma fue posible calcular el valor absoluto y relativo comparando con los valores obtenidos, y que se muestran en la Tabla 21. Adicional a ello durante el desarrollo del mismo análisis, se pudo obtener datos referentes a la veracidad de los estados de los sensores.

Debido a que la activación de alarmas puede ser generada por un estado en falso enviado por cualquier sensor, se hizo necesario contabilizar el número total de alarmas en falso, generadas por la red de sensores inalámbricos, para de esta forma validar el correcto funcionamiento del hardware y software implementado.

Tabla 21. Valores obtenidos durante los 7 días de observación realizada a la comunicación existente entre dispositivos finales y la central de alarmas.

ZONA ANALIZADA Y TIPO DE SESNOR		TOTAL DE TRAMAS DIA 1	Estado Falso	TOTAL DE TRAMAS DIA 2	Estado Falso	TOTAL DE TRAMAS DIA 3	Estado Falso	TOTAL DE TRAMAS DIA 4	Estado Falso	TOTAL DE TRAMAS DIA 5	Estado Falso	TOTAL DE TRAMAS DIA 6	Estado Falso	TOTAL DE TRAMAS DIA 7	Estado Falso
ZONA 1	Sensor MAG	2878	0	2877	0	2879	0	2880	0	2879	0	2879	0	2880	0
ZONA 2	Sensor HUM	2879	0	2880	0	2879	0	2878	0	2880	0	2880	0	2880	0
ZONA 3	Sensor MOV	2879	0	2878	0	2878	0	2878	0	2880	0	2879	0	2878	0
ZONA 4	Sensor MOV	2878	0	2878	0	2878	0	2878	0	2880	0	2878	0	2879	0
ZONA 5	Sensor MOV	2880	0	2879	0	2880	0	2880	0	2880	0	2879	0	2880	0
ZONA 6	Botón de PÁNICO	2880	0	2880	0	2880	0	2880	0	2879	0	2880	0	2880	0

Fuente: [El Autor]

La Tabla 21, describe en resumen, el análisis realizado durante los 7 días de prueba, que permitió la verificación de las tramas recibidas en la central de alarmas, provenientes de cada Zona perteneciente a la WSN implementada. Es posible también, observar en ella, la contabilización de falsas alarmas provocadas por los sensores, que en este caso da una totalidad de eventos igual a cero, describiendo una operación normal del sistema. Con base en estos datos estadísticos, se obtuvieron los siguientes resultados:

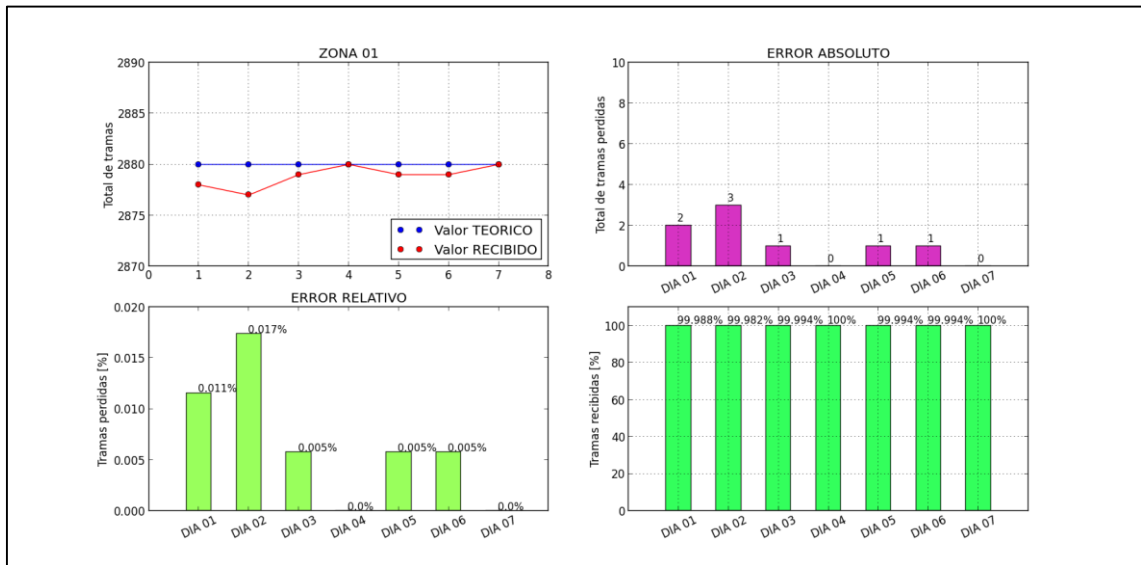


Figura 88. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 01.

Fuente: [El Autor]

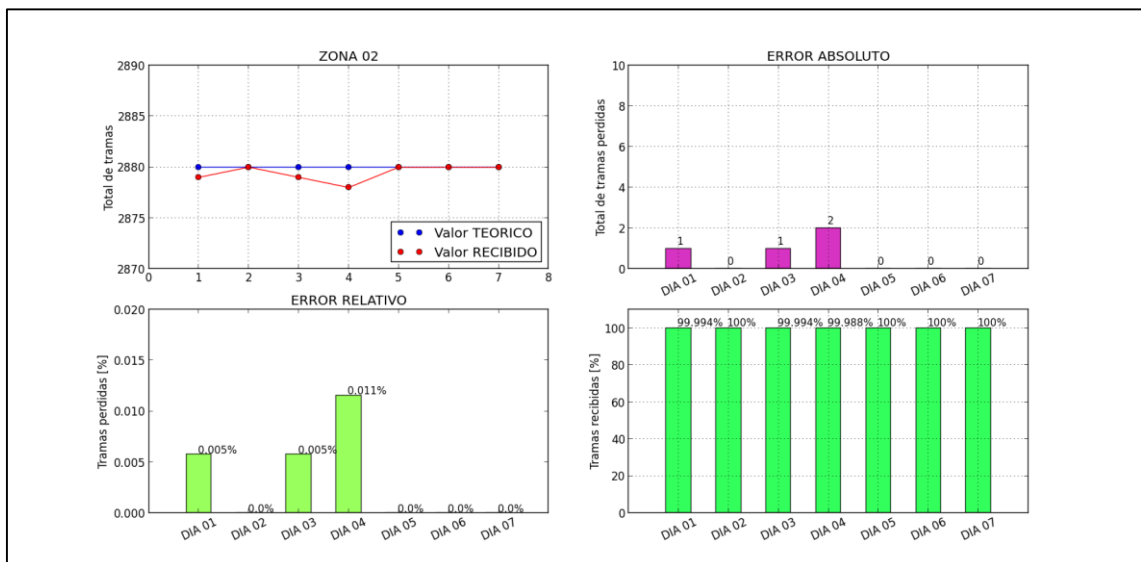


Figura 89. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 02.

Fuente: [El Autor]

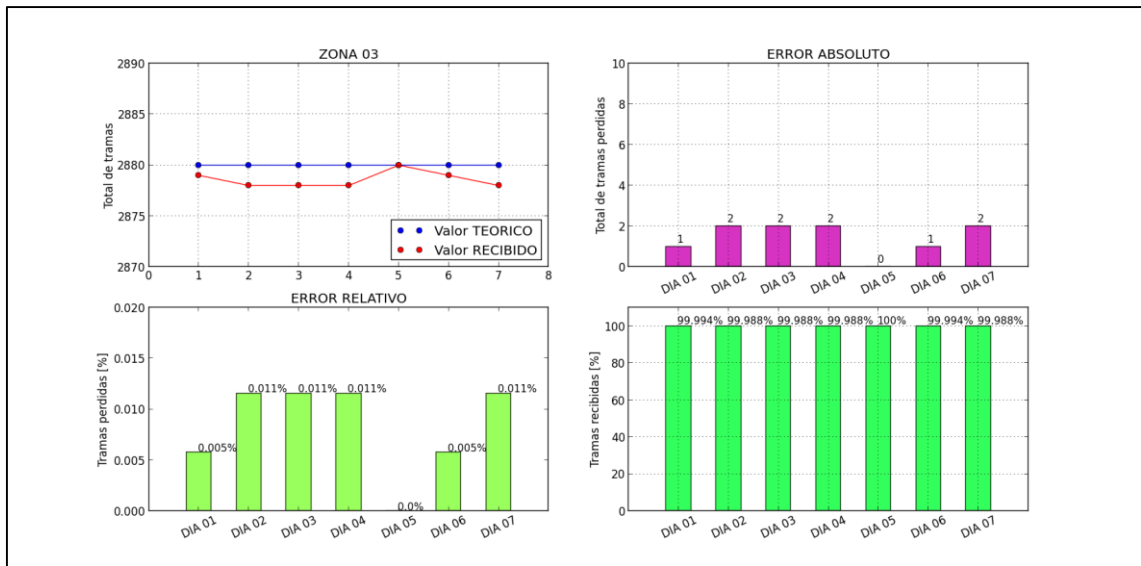


Figura 90. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 03.

Fuente: [El Autor]

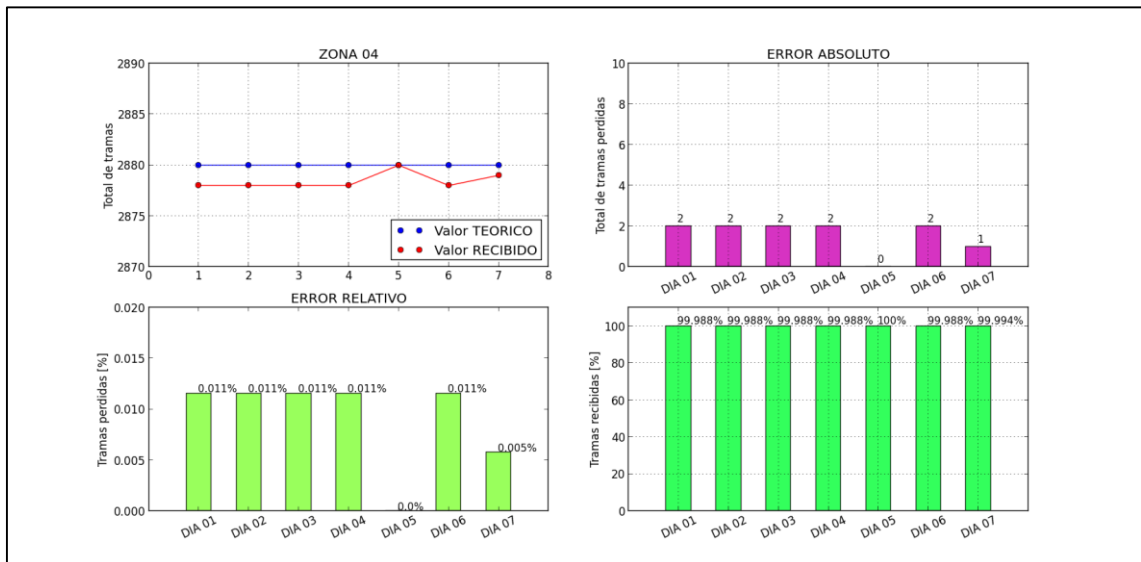


Figura 91. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 04.

Fuente: [El Autor]

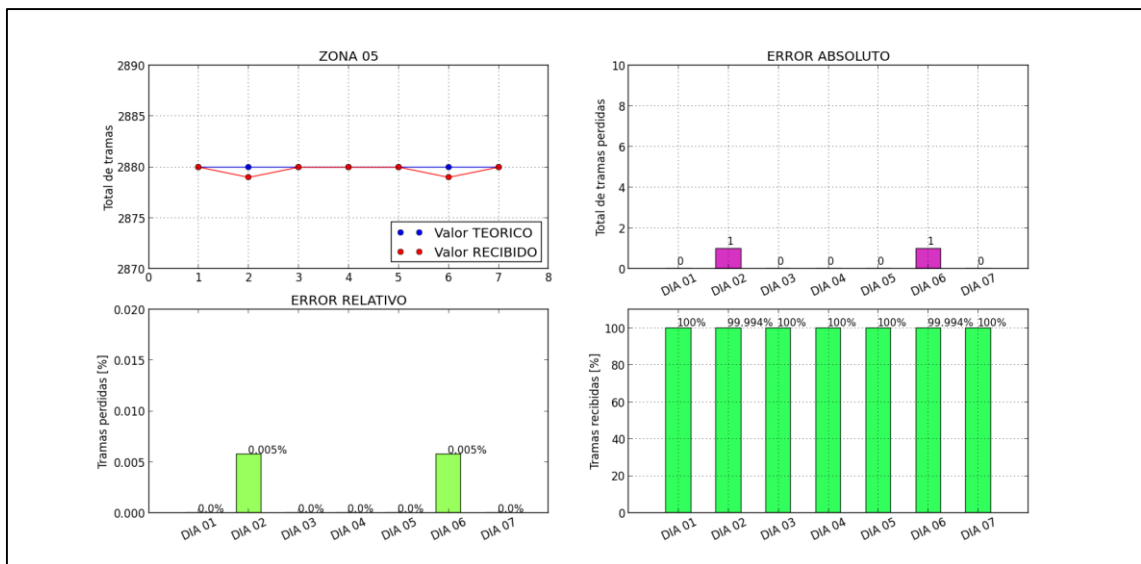


Figura 92. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 05.

Fuente: [El Autor]

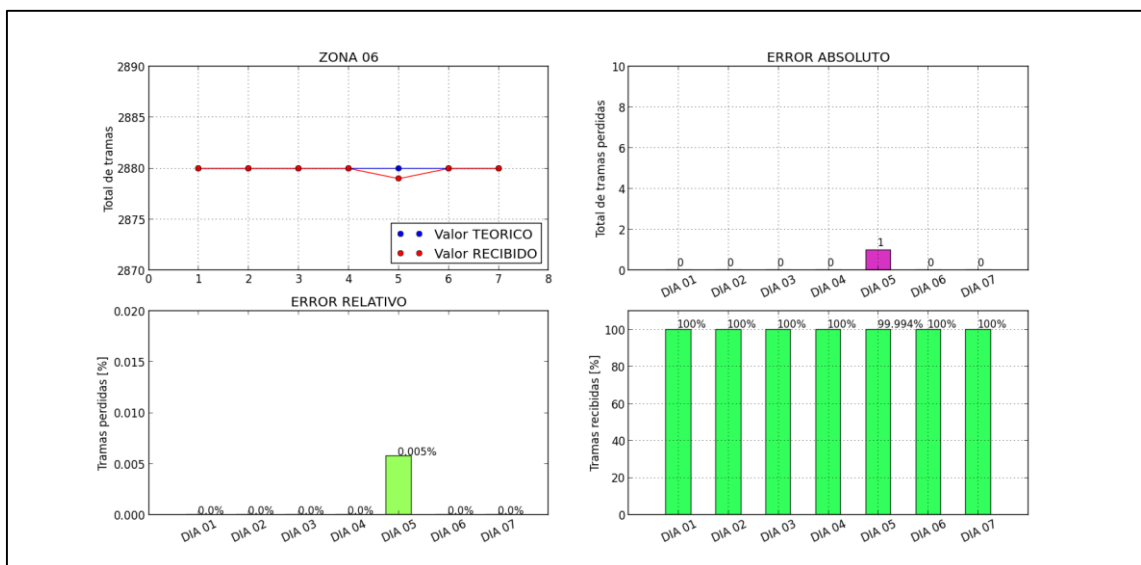


Figura 93. Error absoluto y relativo respecto a la recepción de tramas, en la central de alarmas provenientes desde la ZONA 06.

Fuente: [El Autor]

Al haber realizado el cálculo de los errores absolutos y relativos, los cuales son complementarios y se deben analizar conjuntamente. Muestran que el margen de error relativo en todos los casos no supera el 0.02%, teniendo como contraparte la superación del 99.98% ($\approx 99.99\%$ en la mayoría de los casos) en la entrega de tramas provenientes

desde cada sensor, por lo que se deduce que la central del sistema está receptando correctamente la información enviada desde cada dispositivo final, convirtiéndose estos resultados en un indicador, que muestra la comunicación constante entre los sensores y la central de alarmas.

6.3. Difusión de alertas a través de la aplicación Telegram

A través de la aplicación Telegram es posible recibir la información proveniente desde la central del sistema, la cual también permite ejecutar acciones de respuesta inmediata ante el desencadenamiento de un evento, las funcionalidades implementadas se muestran a continuación en las Figuras 94 y 95 respectivamente.

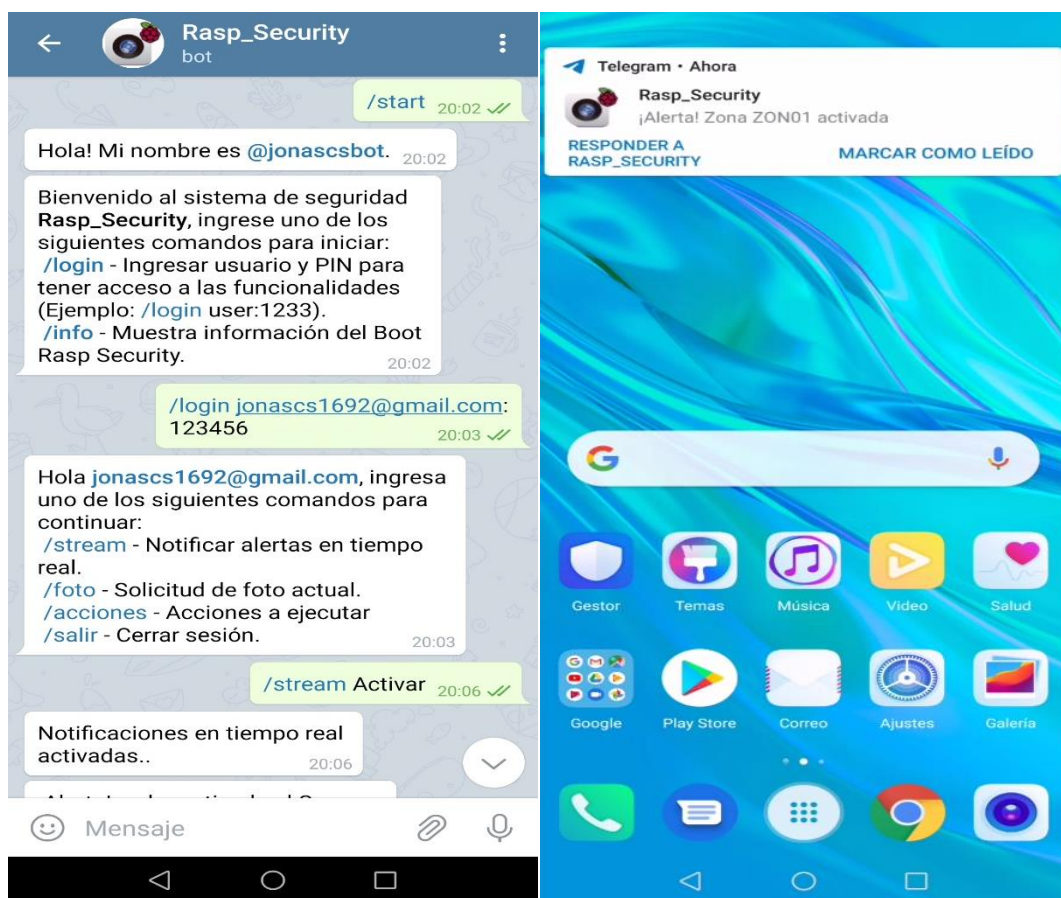


Figura 94. Funcionalidades implementadas en el Bot de Telegram creado para la difusión de alertas.

Fuente: [El Autor]

Tal como lo muestra la Figura 94 el bot creado, se comunica directamente con el usuario por medio de la aplicación Telegram, en él es posible acceder a una serie de comandos previo inicio de sesión con el sistema. Al estar registrado en la base de datos con un usuario y contraseña el sistema da paso y permite elegir una serie de comandos. Uno de

los principales y el más elemental de los comandos es “/stream Activar”, el cual es el que desencadena la notificación de alertas en tiempo real, inmediatamente ejecutado este comando, el sistema realiza la gestión para la difusión de alertas al usuario en caso de presentarse un evento no deseado dentro del espacio de monitoreo por parte de la WSN implementada. Al presentarse la activación de cualquier sensor, perteneciente a una zona conectada a la WSN, el sistema envía el mensaje de alerta mostrado en la Figura 95, especificando la zona activada, con una serie de botones que permiten ejecutar una acción de respuesta inmediata, todos estos botones excepto [Reportar al ECU 911] activan o desactivan un canal del módulo relé de 4 canales en el orden descrito en la sección anterior. Otros comandos disponibles son “/foto”, “/acciones”, y “/salir”. El comando “/foto” es otro de los comandos fundamentales ya que permite obtener un instantánea, de la vista frontal de la central de alarma representado en la Figura 95. El comando “/acciones” despliega el mismo menú de botones ideal para realizar cambios en los estados de los Pines GPIO administrados, si el comando salir es ejecutado se cierra la sesión con el bot y una vez ejecutado dicho comando, al querer ingresar alguna de las instrucciones anteriormente analizadas, el sistema responderá con el mensaje “Permisos Insuficientes” indicando que no se ha accedido al sistema, por lo que necesariamente toca iniciar nuevamente una sesión para obtener las funcionalidades. También es posible hacer uso del comando “/stream Desactivar” quien da la orden a la central del sistema para que las notificaciones en tiempo real se desactiven y de esta forma evitar el uso indebido cuando no se requiera de este servicio. En la Figura 95 se detalla los aspectos mencionados

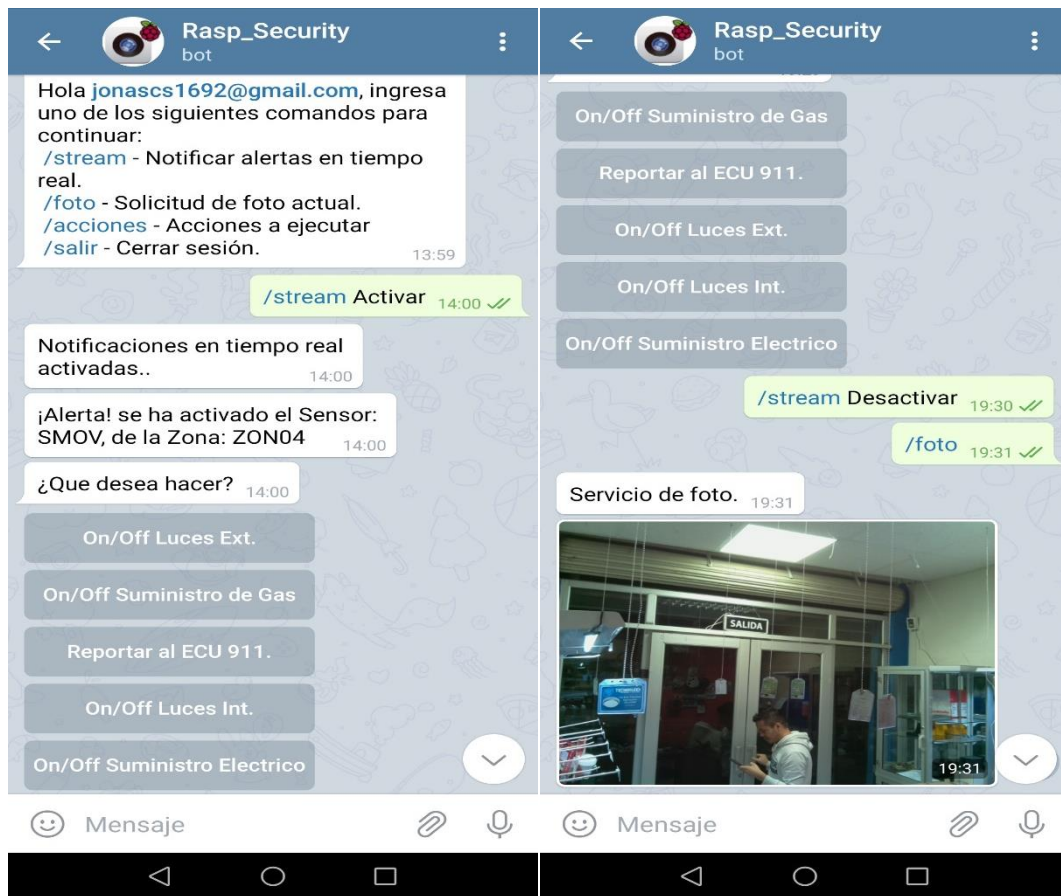


Figura 95. Gestión de alarmas mediante el uso del comando “/stream Activar” y “/foto”.

Fuente: [El Autor]

6.4. Funcionamiento del servidor Web

A través del servidor desarrollado con Angular CLI es posible visualizar de forma gráfica los acontecimientos suscitados en la operación y desempeño del sistema de seguridad, los cuales se encuentran almacenados en la base de datos desarrollada en Firebase, también es posible realizar en él, cambios que pueden ser guardados y almacenados en la base de datos. Las Figuras 96 y 97 muestran la interfaz desarrollada de los componentes: Config y Dashboard descritos en la Figura 80.

Dashboard despliega la información almacenada en la Base de datos Firebase referente a las ZONAS conectadas, los últimos eventos acontecidos por cada zona y una representación de las capturas realizadas por la cámara implementada.

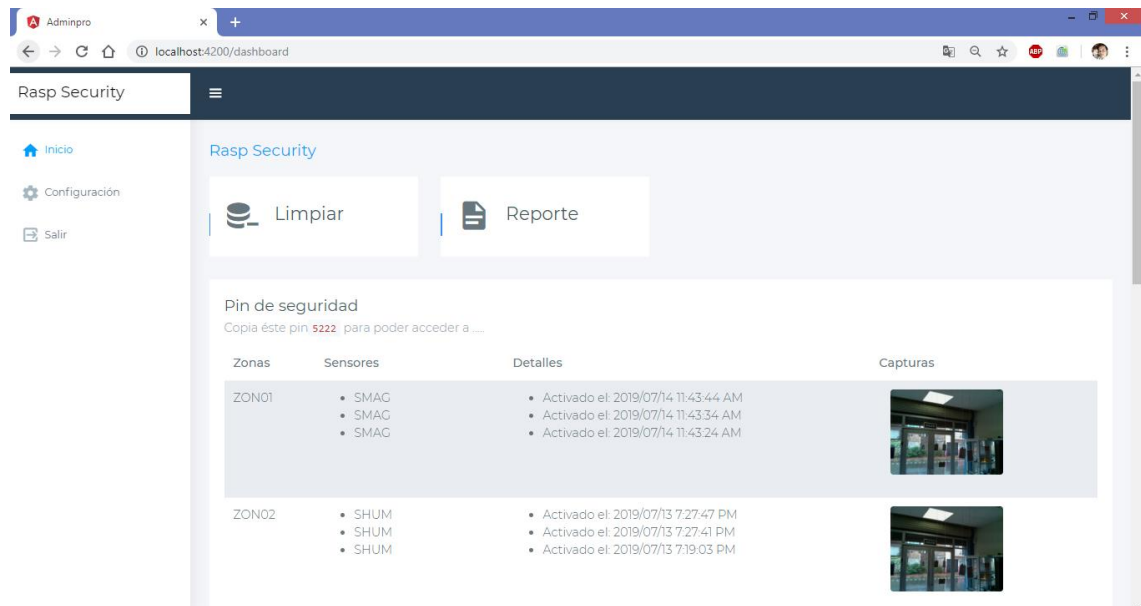


Figura 96. Despliegue de la interfaz web desarrollada (componente dashboard).

Fuente: [El Autor]

Config a su vez despliega una serie de opciones que permiten al usuario o administrador del sistema, modificar parámetros relacionados con la gestión de alarmas a través de la aplicación Telegram. Específicamente los campos “Username” y “Pin” son tomados para poder acceder desde Telegram, tal como lo indica la Figura 94.

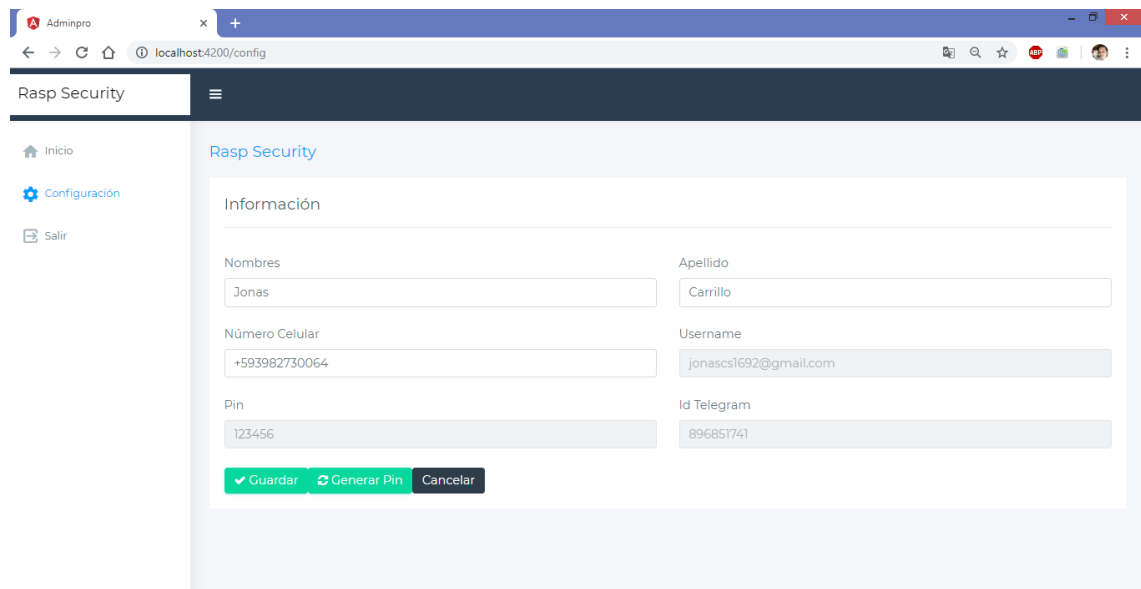


Figura 97. Despliegue de la interfaz web desarrollada (componente config).

Fuente: [El Autor]

Es posible levantar el servidor web en el hosting ofrecido por Firebase, el cual permite al usuario acceder al sistema de seguridad de forma remota. Teniendo en cuenta el factor seguridad se ha creído conveniente documentar este fragmento con los resultados obtenidos mediante las pruebas realizadas en el servidor local.

6.5. Limitaciones del sistema

El diseño del sistema está basado en un prototipo con elementos de *hardware* cuyas características se encuentran limitadas a las descripciones realizadas en la sección de materiales y métodos, si bien no podrían considerarse al nivel de los dispositivos industriales con mayor capacidad de resolución, o a los ofrecidos por una central de alarma comercial como BOSCH, DSC, HONEYWELL, etc., los sensores implementados permiten verificar plenamente el correcto desempeño de una central de alarma y sus periféricos, lo cual es precisamente el objeto de estudio en el presente trabajo de titulación.

Un problema existente y evidente es la falta de un circuito que permita la transición de la de energía de la red eléctrica convencional, hacia un sistema de alimentación ininterrumpida (alimentado por baterías), principalmente en la central de alarmas.

Otro aspecto a destacar y que no ha sido implementado, es una red de respaldo (*backup network*) que permita el envío de información, en caso de no existir conexión a través de internet en la central de alarmas, por parte de nuestro proveedor de servicios (ISP).

Adicional a ello, un aspecto fundamental a considerar para trabajos futuros es el factor ciberseguridad, debido a que se trata de un sistema conectado a Internet y al tratarse de un desarrollo basado principalmente en el uso de tecnología inalámbrica, es necesario considerar aspectos relevantes con el objetivo de minimizar la vulnerabilidad del sistema (evidente en diversos puntos del sistema planteado, Figura 29) y que no permitan a personas ajenas apoderarse de la información confidencial que transita por las redes de comunicación establecidas. Soluciones basadas en el manejo de contraseñas robustas, encriptación de los datos transmitidos por internet hacia la base de datos, certificados de seguridad para el servicio web y muchos más, son elementales para un producto que ofrece seguridad.

Si bien el objetivo principal en el desarrollo del presente trabajo de titulación, se enmarca dentro del contexto de la creación e implementación de un sistema de seguridad basado

en una *WSN* bajo el estándar IEEE 802.15.4, el uso de *software* libre, la integración de servicios adicionales y la validación de su funcionamiento. El desarrollo e implementación ha tomado el rumbo definido plenamente, para cumplir con los objetivos planteados obviando los aspectos mencionados anteriormente, pues no son considerados como objeto de estudio, sin embargo es necesario y prudente recalcar que son necesarios, los cuales deberían formar parte de un sistema de seguridad completo y sobretodo con fines comerciales.

7. DISCUSIÓN

Con base en los resultados obtenidos se confirmó la posibilidad de implementar un sistema de seguridad basado en una red de sensores inalámbricos. El sistema de seguridad basado en una red IEEE 802.15.4 permite de manera confiable la gestión de alarmas dentro de la infraestructura de la empresa de Tecnología y Servicios TECSERLED. El estudio realizado sobre el estándar de comunicación inalámbrica aplicado, las herramientas de *hardware* y principalmente del *software* libre, demuestran que las limitaciones que existían, referentes a la aplicación de redes de sensores inalámbricos hacia algunos años atrás como; los sistemas operativos destinados para este tipo de aplicaciones, *middelwares*, las abstracciones semánticas y el intercambio de datos, hoy en la actualidad se encuentran solucionadas en un alto grado, siendo posible a través del avance tecnológico en diversidad de áreas, desarrollar soluciones orientadas al monitoreo y control mediante su gestión remota.

Al haber realizado las pruebas referentes al nivel de recepción de señal y seguidamente las pruebas de verificación de comunicación con la central de alarmas, se refuta la implementación exitosa del sistema de seguridad basado en una red de sensores inalámbricos y el uso de *software* libre que ha permitido adicional a esto, la integración de servicios agregados al sistema, incluyendo el monitoreo y respuestas de acción inmediata de forma remota.

Referente a los aspectos técnicos, cabe mencionar que los dispositivos XBee-S2C implementados tienen un umbral de recepción de -100 dBm por lo que la distribución de cada uno de ellos dentro de la infraestructura, respecto a su coordinador se encuentra dentro del rango permitido y el nivel de entrega de paquetes lo confirma. Los diferentes niveles RSSI obtenidos se encuentran en función de las distancias existentes entre el coordinador y cada zona específica, considerando que en el trayecto que realiza la señal de RF se encuentra con obstáculos (paredes principalmente) e interferencia provocadas por otras redes que operan en la misma banda de frecuencia, tal como lo muestra la Figura 98, los niveles de recepción derivan en los valores obtenidos



Figura 98. Redes WLAN operando en la banda de 2.4 GHz, en el entorno de monitoreo de la WSN implementada.

Fuente: [El Autor]

Las gráficas obtenidas por cada zona (Figura 88-93) demuestran el porcentaje de entrega de tramas, o si se lo analiza desde el punto de vista de la comunicación; es el porcentaje de comunicación existente entre la central y cada dispositivo final, que es mayor al 99.99% en todos los casos, un valor aceptable al tratarse de una comunicación inalámbrica que comparte el espectro radioeléctrico con otras tecnologías inalámbricas que trabajan en el mismo rango de frecuencias.

Es importante mencionar, que al tratarse de un prototipo, la recopilación de datos, también evidencia la existencia de posibles fallas que generarían un desencadenamiento de falsas alarmas. Si bien en el periodo analizado no se presenta ningún evento de este tipo. A continuación se analiza lo que sucedería si se generara un único evento falso, y como influenciaría en la eficiencia y veracidad de los datos generados por el sistema. Un evento falso no es más que un dato erróneo del estado del sensor, entregado a la central de alarmas.

Para hacerlo evidente, se hace uso de la distribución de probabilidad de Poisson, la cual refleja el número de eventos y la probabilidad asociada a dichos eventos, representados en base a un espacio finito de observación. Suponiendo el caso de obtener tan solo un estado en Falso en la Tabla 21, perteneciente a “X ZONA” con “X Sensor” durante el periodo de prueba. Bajo esta suposición se puede obtener los siguientes resultados.

En un periodo de 56 horas, correspondiente a las 8 horas de análisis realizadas en los 7 días para una ZONA en específico, se ha detectado un estado en falso provocado por un sensor, por lo tanto, al hacer uso de la Distribución de Poisson se tiene:

1. La ecuación que define la función de densidad de probabilidad de la distribución Poisson es la Ec.(2):

$$P(x) = \frac{\lambda^x e^{-\lambda}}{x!} \text{ Ec. (2)}$$

2. x es el número de veces de ocurrencia del evento o fenómeno (en este caso es la existencia de estados falsos generados por un sensor)
3. λ es un parámetro positivo que representa la tasa de ocurrencias, o la media dividida entre la dimensión del espacio de observación. En este caso $\lambda = 1/56$ (horas)
4. e es la base de los logaritmos naturales ($e = 2,71828...$)

El suceso de estudio tal como se ha indicado tendría lugar en promedio $\lambda = 1/56$ veces (es decir en las 56 horas de análisis para una ZONA en concreto, se presentó un evento falso). Al estar interesados en la probabilidad de que un evento ocurra (x veces) dentro de un intervalo de tiempo superior se hace uso del modelo de distribución de Poisson con los siguientes valores de λ mostrados en la Tabla 22:

Tabla 22. Valores de lambda obtenidos, en base al suceso de estudio tomado para hacer la demostración.

Intervalo de tiempo (horas)	Valor de λ	Describe
168	$\lambda = 168(1/56) = 3$	1 Semana
336	$\lambda = 336(1/56) = 6$	2 Semanas
504	$\lambda = 504(1/56) = 9$	3 Semanas
672	$\lambda = 672(1/56) = 12$	4 Semanas

Fuente: [El Autor]

Teniendo como base el valor de λ , basado en el lapso de observación realizado, los nuevos valores de λ definen los niveles de probabilidad, para un número determinado de estados

falsos ocurridos en intervalos de tiempos correspondientes a una semana, dos semanas, tres semanas y un mes respectivamente, dando como resultado lo expuesto en Figura 99.

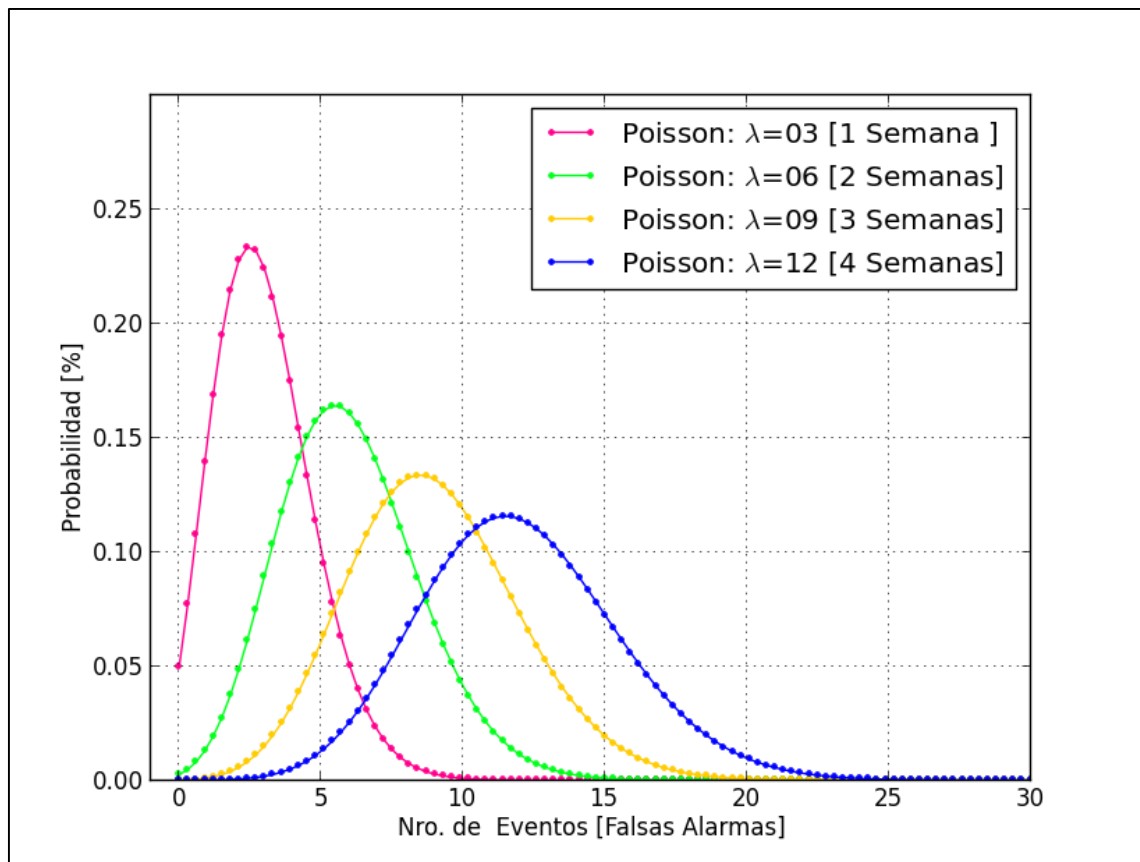


Figura 99. Número total de eventos falsos desencadenados, según la función de densidad de probabilidad de la distribución Poisson, aplicada a este caso en particular.

Fuente: [El Autor]

Si bien el tamaño de la muestra tomado como referencia para realizar esta distribución de probabilidad, aún no es lo suficientemente grande pero si considerable. Estos resultados reflejan lo crítico y a la vez lo importante que resulta ser, la realización correcta de las etapas de diseño a nivel de *hardware* y de *software*, ya que este tipo de sucesos es desencadenado por diseños o conexiones defectuosas, incluidas las lecturas efectuadas por dispositivos electrónicos como los microcontroladores, que conllevan al envío de información errónea, ajena a las situaciones del entorno de monitorización.

8. CONCLUSIONES

- El sistema de seguridad basado en una red de sensores inalámbricos bajo el estándar IEEE 802.15.4 y el uso de herramientas de *software* libre, fue implementado con éxito en la Empresa de Tecnología y Servicios TECSERLED en la ciudad de Loja, permitiendo monitorear la central del sistema de forma local y remota, minimizando la dependencia de elementos adicionales comunes en las centrales de alarmas; como teclados, pantallas, etc., facilitando su uso, al mismo tiempo que se ha agregado funcionalidades no disponibles en los sistemas de seguridad comunes.
- Se ha desarrollado un sistema de red de sensores inalámbricos con Raspberry Pi, XBee, el microcontrolador PIC18F25K20 y varios paquetes de *software* de código abierto. El sistema presenta una serie de características atractivas que incluyen; escalabilidad, facilidad en la implementación, compacto, fácil de mantener, siendo una de sus principales ventajas la integración del nodo *Gateway* de la red de sensores inalámbricos con una base de datos y un servidor *web* que permite la administración y la gestión de alarmas de forma eficiente. El diseño detallado y los resultados de medición presentados en este documento, demuestran claramente la utilidad de dicho sistema.
- Las WSN abren nuevas posibilidades, y trazan nuevos horizontes, hacia futuros sistemas de monitoreo y control, de entornos inteligentes y automáticos basados en redes de sensores inalámbricos, manteniendo siempre una íntegra relación de dependencia entre el *software* y *hardware* desarrollado.
- Los sistemas modernos de seguridad requieren de la presentación de resultados en diferentes terminales, y no únicamente en una PC o entornos aislados de la comunicación a Internet. El avance tecnológico de las nuevas generaciones de teléfonos inteligentes y las tecnologías de transmisión de datos disponibles, como *ZigBee*, pueden brindar soluciones más flexibles, que permiten la implementación de sensores portátiles y la visualización de datos en dispositivos móviles.
- A pesar de las interferencias causadas por otras tecnologías que operan en la banda de los 2.4 GHz, la red basada en el estándar IEEE 802.15.4 y principalmente los módulos XBee operan en condiciones normales con un alto grado de robustez,

demostrado con los resultados obtenidos, esto se debe principalmente a que el estándar ofrece una solución viable, ya que al disponer de 16 canales de comunicación, él establece el canal no ocupado y realiza el intercambio de datos, de esta forma se concluye que esta tecnología se encuentra diseñada para compartir plenamente el espectro radioeléctrico con otras tecnologías inalámbricas.

- Si bien la principal aplicación de *ZigBee*, está orientada a los sistemas de control inalámbrico, esta tecnología también presenta amplias capacidades de desarrollo que le permite ser utilizado en múltiples aplicaciones, como los sistemas de medición avanzada, medidores de agua, energía eléctrica y gas que forman parte de una red con otros dispositivos, incluso sistemas de alarma como queda demostrado mediante la implementación del proyecto de titulación realizado.
- Los índices de comunicación exitosa, entre la central y dispositivos finales obtenidos, demuestran una confiabilidad en la comunicación mayor al 99.99%, por lo que se deduce que la comunicación entre la central del sistema y los sensores es constante e ininterrumpida al menos que se produzca un corte de energía eléctrico. Los resultados de rendimiento confirman que el sistema de seguridad basado en esta WSN es prácticamente aplicable en entornos de edificios de trabajo y oficinas, en donde también operan entornos *Wifi Existing*.
- La plataforma Raspberry Pi es una potente herramienta que permite el desarrollo de sistemas embebidos, basados en la aplicación de ingeniería a nivel de *hardware* y *software*. Al usar un Kernel de Linux como sistema operativo se cuenta con los beneficios del *software* libre sin la necesidad de requerir una licencia comercial.
- El uso de las bases de datos NoSQL, se está popularizando cada vez más debido a sus grandes prestaciones. Con la llegada de las aplicaciones *IoT* se ha generado el problema de *BigData* derivado principalmente por la cantidad de datos generados por estas aplicaciones. Estos datos necesitan ser almacenados y gestionados con gran rapidez, los sistemas de gestión de bases de datos relacionales tradicionales no pueden manejar una cantidad tan grande de datos, es por ello que las bases de datos NoSQL han surgido como una solución para superar algunas limitaciones en las bases de datos relacionales tradicionales.
- La programación de scripts con un alto grado de complejidad en lenguaje de programación Python a través de la creación de módulos y paquetes, incidió positivamente en el desarrollo de los requerimientos del sistema desarrollado, ya

que se pudo cumplir con la realización del código desde sus etapas iniciales hasta sus etapas más complejas.

- El lenguaje de programación Python es un lenguaje multiplataforma y universal que permite integrar diversidad de aplicaciones a los desarrollos e investigaciones, convirtiéndolo en una herramienta fundamental para el desarrollo de aplicaciones basadas principalmente en el uso de *software* libre.
- Al no contar con infraestructura propia de intercomunicación con la base de datos, ni con servidores propietarios, el sistema desarrollado no permite al administrador un control total de la funcionalidad del sistema, esto se presenta principalmente en la solución de errores por fallas de comunicación con el servidor o por la indisponibilidad del mismo.

9. RECOMENDACIONES

- El correcto desarrollo en el diseño del *hardware* y *software* permitirá a futuro evitar inconvenientes relacionados con las falsas alarmas, producidas por malas conexiones o diseño defectuoso, hechos que pueden afectar significativamente la precisión en el proceso de envío de datos, por lo que es aconsejable dedicar el mayor de los esfuerzos en conseguir un diseño que evite este tipo de complicaciones.
- En caso de integrar nuevos dispositivos a la WSN generada, se debe tener en cuenta la actualización de los dispositivos XBee con la versión más reciente del *firmware*, para evitar conflictos entre versiones diferentes que afecten el rendimiento en general del sistema, ya que es muy posible que se presente este tipo de situaciones, pues la empresa DIGI se encuentra constantemente actualizando las base de los *firmware* para sus dispositivos XBee.
- Se recomienda hacer una correcta regulación de voltaje hacia 3.3V el cual es el voltaje de operación de los módulos XBee y el microcontrolador PIC18F25K20, para ello es preferible hacerlo con un regulador de voltaje específico como el LM1117 y evitar el uso de reguladores ajustables ya que pueden presentar fallas derivadas del mal funcionamiento o deterioro de los componentes electrónicos adicionales, que requiere para cumplir con la regulación de voltaje.
- Para realizar la programación en Python de sistemas muy complejos, con un sinnúmero de requerimientos, se recomienda utilizar el método de programación por capas y el uso de la jerarquización de un proyecto en Python, a través de la creación de módulos y paquetes, los cuales se conforman por una serie de clases y métodos, que permiten dinamizar la creación de scripts, la corrección de errores, y la agregación de servicios adicionales en el desarrollo de un proyecto. De esta forma se obtiene un sistema concebido y desarrollado en componentes que permite escalabilidad para futuras mejoras.
- Se recomienda hacer periódicamente la actualización del *software* y *firmware* de la Raspberry Pi, por lo menos una vez cada seis meses por un administrador, ya que las mejoras recibidas en sus actualizaciones permiten la ejecución eficiente de programas, del Kernel, y agregan seguridades a posibles vulnerabilidades presentes en la red.
- El sistema desarrollado como tal, al tratarse de un sistema informático conectado

a Internet, es susceptible a intentos de sabotaje para intentar apoderarse de los datos e información almacenada, es por ello que se recomienda un método como el mostrado en desarrollo del servidor *web*, que permite cambiar constantemente las claves de acceso al sistema. También es recomendable cambiar las contraseñas de acceso al dispositivo periódicamente, configurar correctamente el servidor de *firewall* de la Raspberry Pi (que tiene uno incluido) o el de la red local.

- Para trabajos futuros se recomienda, orientar el mayor de los esfuerzos en la protección de la información enviada desde la central del sistema hacia la base de datos, debido a que la información transmitida, al ser capturada, puede comprometer gravemente el funcionamiento del sistema, para evitar esto, es aconsejable agregar niveles de seguridad, de ser posible en todas las capas del modelo OSI; como restringir el acceso a los equipos a personal no autorizado, hacer uso de protocolos seguros, usar encriptación de los datos, activación de firewalls, etc., de tal forma que se brinde al usuario final un sistema completo que abarca la seguridad en toda su extensión, desde la generación de la información hasta su entrega.
- Someter al sistema desarrollado a etapas de pruebas más rigurosas, con el objetivo de tener resultados aún más reales e implementar medidas que aporten al mejoramiento del sistema.
- Para trabajos futuros, se recomienda diseñar una aplicación móvil multiplataforma que permita recibir la difusión de alertas al usuario, al mismo tiempo que implemente las características del servidor web desarrollado, para evitar la dependencia de infraestructura de terceros, en la implementación del sistema.
- Para la conformación de un dispositivo final existen dos métodos válidos y funcionales que pueden ser aplicables según los requerimientos del proyecto a trabajar. El primero es operar con los pines I/O del dispositivo XBee, sin embargo se recomienda utilizar un dispositivo XBee junto con un microcontrolador que permita la lectura de puerto serie, pues permite obtener una mejor potencialidad y liberar al dispositivo del coste computacional ejercido si se trabaja en modo API.
- Se recomienda para futuros trabajos elaborar circuitos de conmutación de la red eléctrica, hacia fuentes de alimentación ininterrumpidas, aplicables cuando exista un fallo de energía, tanto en la central del sistema como en cada dispositivo final, al igual que la creación de redes de respaldo que permitan enviar la información hacia el usuario en caso de existir fallas en la conexión a Internet por parte del

proveedor de servicios (ISP).

- Debido a que es posible en determinado momento, sufrir un grave daño a la infraestructura de la base de datos, es recomendable realizar periódicamente un respaldo de la información almacenada en este espacio. Para ello se debe establecer los mecanismos necesarios, de tal forma que al presentarse una situación de este tipo, se pueda regresar al último estado funcional de la base de datos, mediante el proceso conocido como recuperación de una base de datos a través de copias de respaldo generadas.
- Un nivel de seguridad adicional, que se puede implementar para el ingreso al sitio web desarrollado en trabajos futuros, es hacer uso de un sistema de seguridad CAPTCHA, diseñado para distinguir a las peticiones realizadas por los humanos del realizado por las computadoras, con el objetivo de evitar que spammers y robots de spam programados para atacar sitios web, inunden con tráfico el sitio, y de alguna forma comprometan la información presentada en él, se recomienda hacer uso de este sistema.

10. BIBLIOGRAFÍA

- [1] M. d. Interior, «Ministerio del Interior,» 9 Abril 2018. [En línea]. Available: <https://www.ministeriointerior.gob.ec/ecuador-disminuyo-en-un-12-los-delitos-de-mayor-afectacion-social-en-2018/>. [Último acceso: 03 02 2019].
- [2] R. Anderson, Security engineering, John Wiley & Sons, 2008.
- [3] D. M. Paez, «Diseño y evaluación de un sistema domótico para seguridad en viviendas bajo el estándar IEEE 802.15. 4/Zigbee,» *Revista Tekhnê*, pp. 27-36, 2014.
- [4] H. Bauge, Secdroid: An Improved Alarm Distribution System (Master's thesis), 2013.
- [5] J. R. Castro, Building a Home Security System with Arduino, Packt Publishing Ltd, 2015.
- [6] LAARCOM, «LAARCOM,» 2019. [En línea]. Available: <https://www.laarcom.com/que-es-la-seguridad-electronica>. [Último acceso: 21 03 2019].
- [7] J. R. Fernández, Circuito cerrado de televisión y seguridad electrónica, Ediciones Paraninfo, SA, 2013.
- [8] J. Salazar, Redes inalámbricas, República Checa., 2016.
- [9] E. & Y. LLP, «Cyber Security A necessary pillar of Smart Cities,» pp. 8-15, 2016.
- [10] S. B. M. & H. L. Paroutis, «A strategic view on smart city technology: The case of IBM Smarter Cities during a recession,» *echnological Forecasting and Social Change*, vol. 89, pp. 262-272, 2014.
- [11] E. Pievanelli, A. Plesca, R. Stefanelli y D. & Trincherro, «Dynamic wireless sensor networks for real time safeguard of workers exposed to physical agents in constructions sites,» *IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, pp. 55-57, 2013.
- [12] M. R. Ahmed, X. Huang, D. Sharma y & H. Cui, «Wireless Sensor Network: Characteristics and Architectures,» *World Academy of Science, Engineering and Technology International Journal of Information and Communication Engineering*, vol. 6, nº 12, pp. 1398-1401, 2012.
- [13] S. Farahani, ZigBee wireless networks and transceivers, Newnes, 2011.
- [14] R. G. Duque, Python para todos, 2011.
- [15] P. S. Foundation, 2019. [En línea]. Available: <https://www.python.org/>. [Último acceso: 21 03 2019].

- [16] P. S. Foundation, «Python Software Foundation [US],» 2019. [En línea]. Available: <https://docs.python.org/3/license.html?highlight=gpl%20python>. [Último acceso: 21 03 2019].
- [17] I. T. I. O. B. E., «TIOBE-The Software Quality Company,» TIOBE Index| TIOBE–The Software Quality Company [Electronic resource], 2019. [En línea]. Available: <https://www.tiobe.com/tiobe-index/>. [Último acceso: 10 07 2019].
- [18] EDUCBA, «educba.com,» EDUCBA (Corporate Bridge Consultancy Pvt Ltd), 2019. [En línea]. Available: <https://www.educba.com/python-vs-c-plus-plus/>. [Último acceso: 10 07 2019].
- [19] T. Cox, Raspberry Pi Cookbook for python programmers, Packt Publishing Ltd, 2014.
- [20] W. Harrington, Learning Raspbian, Packt Publishing Ltd, 2015.
- [21] D. Gislason, Zigbee wireless networking, Newnes, 2008.
- [22] J. P. Dignani, Análisis del protocolo ZigBee, (Doctoral dissertation, Facultad de Informática), 2012.
- [23] G. A.-J. P. & C. P. Thonet, «Zigbee-wifi coexistence,» *Schneider Electric White Paper and Test Report*, vol. 1, pp. 1-38, 2008.
- [24] K. T. Le, «Designing a ZigBee-ready IEEE 802.15. 4-compliant radio transceiver,» *RF Design*, vol. 27(11), pp. 42-50, 2004.
- [25] DIGI, «XBee/XBee-PRO S2C,» 2018. [En línea]. Available: <https://www.digi.com/resources/documentation/digidocs/pdfs/90002002.pdf>. [Último acceso: 25 03 2019].
- [26] C. REYES, Microcontroladores PIC Programación en Basic 16F62X, 16F81X, 16F87X, Quito Ecuador: Editorial Rispergraf CA, 2008.
- [27] «Dispositivos Lógicos Microprogramables,» 2004-2013. [En línea]. Available: <http://perso.wanadoo.es/pictob/indicemicroprg.htm>. [Último acceso: 29 03 2019].
- [28] P. J. F. C. N. & P. W. Villamarín, Construcción de un prototipo de sistema de seguridad inalámbrico mediante el uso de sensores de movimiento, magnético y de humo, utilizando el módulo de radiofrecuencia Xbee, Quito: Tesis, 2016.
- [29] A. P. Garrido, DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN WEB, PARA SU POSTERIOR ANÁLISIS MEDIANTE DOS TIPOS DE SISTEMAS DE GESTIÓN DE BASE DE DATOS, Madrid, 2016.
- [30] J. Molina, N. Loja, M. Zea y E. Loaiza, «Evaluación de los Frameworks en el Desarrollo de Aplicaciones Web con Python,» *Revista Latinoamericana de Ingeniería de Software*, vol. 4, nº 4, pp. 201-207, 2016.
- [31] R. Herranz Gómez, Bases de datos NoSQL: arquitectura y ejemplos de aplicación, Leganés: Tesis de Maestría, 2014.

- [32] ACENS, «Bases de datos NoSQL,» 2014. [En línea]. Available: <https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf>. [Último acceso: 28 03 2019].
- [33] A. C. S. J. S. G. & C. M. C. Romero, «Utilidad y funcionamiento de las bases de datos NoSQL,» *Facultad de Ingeniería*, vol. 21(33), pp. 21-32, 2012.
- [34] Microchip, Datasheet PIC18F25K20, Microchip Technology Inc, 2010.
- [35] Punto Flotante S.A., «HC-SR501, módulo infrarrojo pasivo PIR, sensor de movimiento, con alcance de 7 metros.,» 2017. [En línea]. Available: <https://www.puntoflotante.net/SENSOR-INFRRARROJO-PASIVO-PIR-MOVIMIENTO-HC-SR501.htm>. [Último acceso: 10 04 2019].
- [36] Bigtronica, «Pulsador Metálico para Chasis N-A,» 2019. [En línea]. Available: <https://www.bigtronica.com/poblado/componentes-mecanicos/suiches-pulsadores/1728-pulsador-metalico-para-chasis-n-a.html>. [Último acceso: 12 04 2019].
- [37] COMPONENTS101, «XBee S2C – RF Module,» 16 10 2018. [En línea]. Available: <https://components101.com/wireless/xbec-s2c-module-pinout-datasheet>. [Último acceso: 12 04 2019].
- [38] gizmojo, «Raspberry Pi Camera Module v1.3,» 2015. [En línea]. Available: <https://www.gizmojo.com.ar/products/raspberry-pi-camera-module-v1-3>. [Último acceso: 14 04 2019].
- [39] Amazon, «Raspberry Pi 3 Power Supply 5V 2.5A,» 2019. [En línea]. Available: <https://www.amazon.com/Raspberry-Power-Supply-Adapter-Charger/dp/B0719SX3GC>. [Último acceso: 14 04 2019].
- [40] Electronilab, «Modulo Relé De 4 Canales Salidas Optoacopladas 5V,» 2019. [En línea]. Available: <https://electronilab.co/tienda/modulo-rele-4-canales-salidas-optoacopladas-5v/>. [Último acceso: 15 04 2019].
- [41] C. AssociatesLimited, Proton Amicus18 BASIC Compiler, 2011.
- [42] R. P. FOUNDATION, «The Raspberry Pi Foundation,» 2019. [En línea]. Available: <https://www.raspberrypi.org/search/raspberry+pi+model+B>. [Último acceso: 26 03 2019].

11. ANEXOS

ANEXO I: PLANOS DE LOS ESPACIOS DE TRABAJO Y DISTRIBUCIÓN DE SENSORES EN LA EMPRESA TECSERLED

ANEXO II: PCB Y ESQUEMÁTICO DEL CIRCUITO DISPOSITIVO FINAL ZIGBEE

Disponible en: https://easyeda.com/jonascsl692/xbec_dispositivo_final,

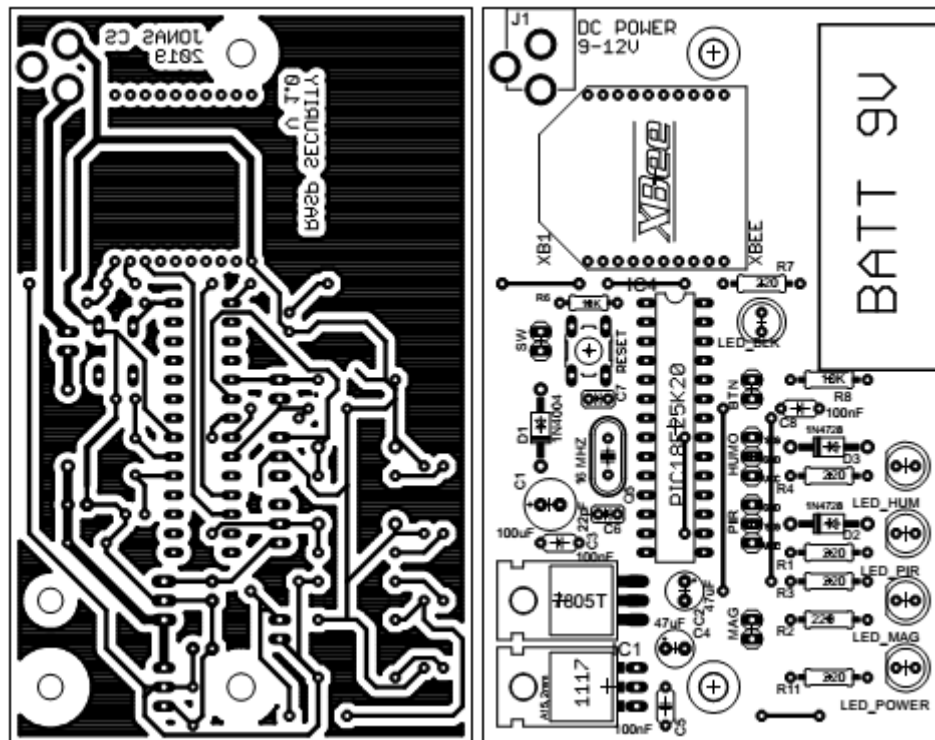
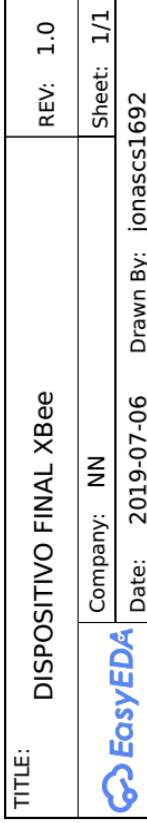
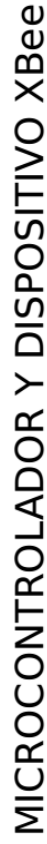


Figura 100. PCB del circuito correspondiente al dispositivo final ZigBee implementado.

Fuente: [El Autor]

ACONDICIONAMIENTO DE SEÑAL DE SENSORES



ANEXO III: CÓDIGO DEL MICROCONTROLADOR PIC18F25K20 CON LAS FUNCIONES DEL DISPOSITIVO FINAL ZIGBEE

Disponible en:

https://gitlab.com/JonasCS/proyecto_rasp_security/tree/master/Codigo_PIC18F25K20

```
'*****
' Proyecto : Programación del Dispositivo Final XBee      *
' Fecha : 13/06/2018(Inicio del desarrollo)              *
' Autor : Jonas Carrillo                                *
' Notas : Código de cada uno de los sensores de los dispo- *
' : sitivos finales PIC18F25K20 - XBee-S2C               *
'*****
'::::::::::::::::::LIBRERIAS::::::::::::::::::::::::::::

DelayMS 1000 ' Tiempo de estabilización de la fuente

Include "Amicus18.inc" ' Cargamos macros de AMICUS IDE

Include "Amicus18_Timers.inc" ' Cargamos macros Timer en el programa

'::::::::::::::::::CONFIGURACIONES DE COMUNICACIÓN::::::::::::::::::::
TRISA = 0
TRISB = 127 ' Configura PORTB como entrada EXCEPTO [RB 7]
PORTB = 127 ' Inicializa PORTB
Declare Hserial_Baud = 9600 ' Seteamos la velocidad de comunicación
Declare Hserial_Clear = On ' Buffer limpiado despues de recibir

'::::::::::::::::::VARIABLES Y CONSTANTES::::::::::::::::::::::::::::
Dim contador As Word ' Variable para control de registro del TIMER0
Dim tipo_sensor As Byte 'Identificar el tipo de sensor activado
Dim sensor_mag_f As Byte ' Variable utilizada como bandera
Dim sensor_pir_f As Byte ' Variable utilizada como bandera
Dim sensor_hum_f As Byte ' Variable utilizada como bandera
Dim btnp_f As Byte ' Variable utilizada como bandera
Dim pulso_Led As Byte ' Controla blink del LED conectado en PORTB.7
Dim cont_aux_led As Byte ' Almacena tiempo de encendido del LED
Symbol sensor_MAG = PORTB.0 ' Pin de lectura de sensor MAGNETICO
Symbol sensor_PIR = PORTB.1 ' Pin de lectura de sensor PIR
Symbol sensor_HUM = PORTB.2 ' Pin de lectura de sensor de HUMO
Symbol btnP = PORTB.3 ' Pin de lectura de Boton de PANICO
Symbol blk = PORTB.7 ' Pin de salida para conexión de LED

'----- Inicialización de variables -----
' Declaración del estado inicial de todas las variables
sensor_MAG = 1 ' Debido a que este sensor es activo en bajo
sensor_PIR = 0
sensor_HUM = 1 ' Debido a que este sensor es activo en bajo
btnP = 0
sensor_mag_f = 0
sensor_pir_f = 0
sensor_hum_f = 0
btnp_f = 0
contador = 0
tipo_sensor = 0
```



```

pulso_Led = 0
cont_aux_led = 0

On_Hardware_Interrupt GoTo Interrupt_blk ' Declaración de Punto de
interrupción
'::::::::::::::::::::: SALTO A LA RUTINA PRINCIPAL ::::::::::::::
GoTo Main
'::::::::::::::::::::::::::::: SUBROUTINAS ::::::::::::::::::::::
'===== Interrupción =====
Interrupt_blk:
Context Save
' Fue un desbordamiento Timer0 que activó la Interrupción?
If INTCONbits_T0IF = 1 Then
Clear INTCONbits_T0IF ' Borrar el indicador de desbordamiento de
Timer0
'===== ECUACION =====
' TMR0_Overflow = (2^n_bits-TMR0) (Prescaler) (4/FOsc)
'= (65536-65286) (128) (4/64MHz)
'----- base de tiempo-----
'-----10ms-----

WriteTimer0(64286)
contador = contador + 1
'----- base de tiempo-----

'-----10 s-----
If contador == 1000 Then
GoSub envio_dato_Normal ' Salto a subrutina envio_dato_Normal
Toggle blk
pulso_Led = 1 ' Enciende LED en PORTB.7
contador = 0
EndIf
If pulso_Led == 1 Then
cont_aux_led = cont_aux_led + 1
If cont_aux_led == 30 Then ' Mantengo encendido el LED durante 300ms
Toggle blk ' cambio el estado del PORTB.7
pulso_Led = 0
cont_aux_led = 0
EndIf
EndIf
EndIf
Context Restore ' Salgo de la interrupción, y restauro registros

'===== Envío normal de datos =====
' Descomentar línea de código
' Segun el sensor conectado al micro
envio_dato_Normal:
HRSOut "I_ZON01_SMAG_EST:00_F",13,10
' HRSOut "I_ZON01_SMOV_EST:00_F",13,10
' HRSOut "I_ZON01_SHUM_EST:00_F",13,10
' HRSOut "I_ZON01_BTNP_EST:00_F",13,10
Return

'===== Envío de datos de alerta =====
' Segun el sensor activado
envio_dato_Alerta: ' Se llama a esta subrutina únicamente cuando
' se ha producido un evento detectado por el
Select tipo_sensor ' Sensor conectado y se envía los datos por
' puerto serie

```

```

Case 1
HRSOut "I_ZON01_SMAG_EST:01_F",13,10

Case 2
HRSOut "I_ZON01_SMOV_EST:01_F",13,10
Case 3
HRSOut "I_ZON01_SHUM_EST:01_F",13,10
Case 4
HRSOut "I_ZON01_BTNP_EST:01_F",13,10
End Select
Return

':::::::::::::::::::::::::::::ROUTINA PRINCIPAL:::::::::::::::::::::::::::::
Main:
OpenTimer0(TIMER_INT_ON & T0_16BIT & T0_SOURCE_INT & T0_PS_1_128)
INTCONbits_GIE = 1 'Habilitamos interrupciones globales
WriteTimer0 (64286) 'Interrupcion cada 10ms
While 1 = 1

'::::::::::::::::::::::::::::: LECTURA DE SENSORES :::::::::::::::::::::::::::
'-----
'----- Lectura de sensor MAG -----
If sensor_MAG == 0 Then 'Lectura de sensor MAGNETICO
sensor_mag_f = 1 'Cargamos valor en bandera del sensor
EndIf
'----- Se ha activado el sensor MAG..?-----
If sensor_mag_f == 1 Then 'Existe cambio de estado
tipo_sensor = 1
GoSub envio_dato_Alerta 'Llamamos a subrutina
DelayMS 3000 'Demora de 3 segundos para nueva lectura
sensor_mag_f = 0 'Reiniciamos bandera para un nuevo uso
Else
tipo_sensor = 0 'Aseguramos siempre este estado por defecto
EndIf

'-----
'----- Lectura de sensor PIR -----
If sensor_PIR == 1 Then 'Lectura de sensor PIR
sensor_pir_f = 1 'cargamos valor en bandera del sensor
EndIf
'----- Se ha activado el sensor PIR..?-----
If sensor_pir_f == 1 Then 'Existe cambio de estado
tipo_sensor = 2
GoSub envio_dato_Alerta 'Llamamos a subrutina
DelayMS 3000 'Demora de 3 segundos para nueva lectura
sensor_pir_f = 0 'Reiniciamos bandera para un nuevo uso
Else
tipo_sensor = 0 'Aseguramos siempre este estado por defecto
EndIf

'-----
'----- Lectura de sensor HUMO -----
If sensor_HUM == 0 Then 'Lectura de sensor HUMO
sensor_hum_f = 1 'cargamos valor en bandera del sensor
EndIf
'----- Se ha activado el sensor HUMO..?-----
If sensor_hum_f == 1 Then 'Existe cambio de estado
sensor_hum_f = 0 'Reiniciamos bandera para un nuevo uso

```

```

tipo_sensor = 3
GoSub envio_dato_Alerta ' Llamamos a subrutina
DelayMS 5000 ' Demora de 5 segundos para hacer una nueva lectura
Else ' evitando envío masivo de datos a la central
tipo_sensor = 0 ' Aseguramos siempre este estado por defecto
EndIf

'-----
'----- Lectura de BTN de pánico-----
If btnP == 1 Then ' Lectura de botón de pánico
btnp_f = 1 ' cargamos valor en bandera del sensor
EndIf
'----- Se ha activado el BTN de Pánico..? -----
If btnP == 0 And btnp_f == 1 Then ' Boton pánico soltado (aseguramos antirebote)
btnp_f = 0 ' Reiniciamos bandera para un nuevo uso
tipo_sensor = 4
GoSub envio_dato_Alerta ' Llamamos a subrutina
DelayMS 3000
Else
tipo_sensor = 0 ' Aseguramos siempre este estado por defecto
EndIf
Wend ' Cerramos el ciclo while

```

ANEXO IV: CONEXIÓN DEL PICKIT3 CON EL MICROCONTROLADOR PIC18F25K20 PARA PROGRAMACIÓN MEDIANTE ICSP

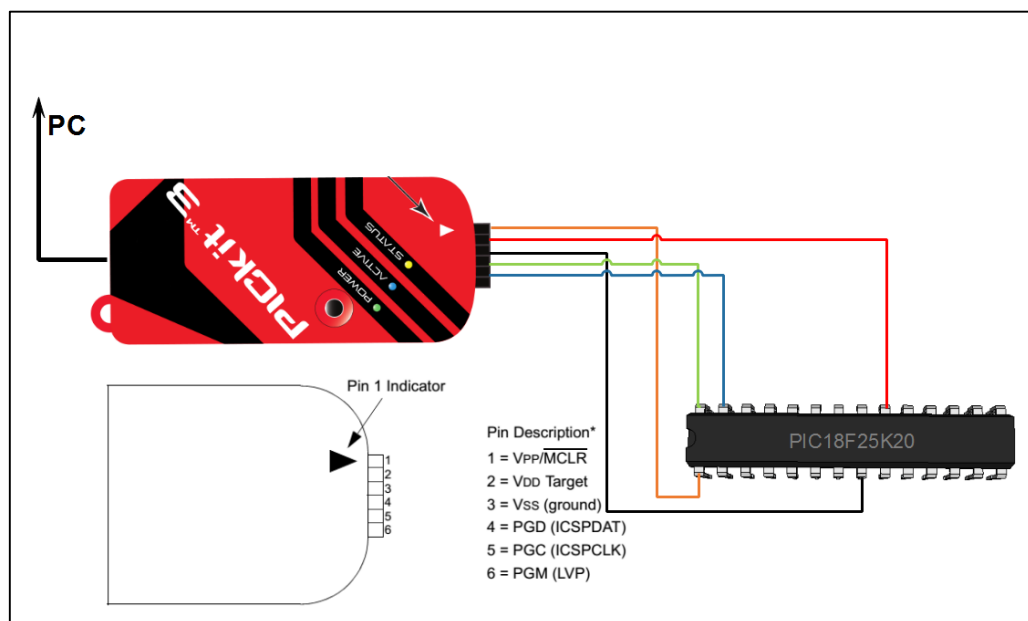


Figura 101. Conexión del PICKit 3 y el microcontrolador PIC18F25K20 para realizar la programación mediante ICSP.

Fuente: [El Autor]

ANEXO V: AGREGACIÓN DE UNA RED WI-FI A LA TARJETA RASPBERRY PI 3

A través del comando:

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Se abre el fichero “wpa_supplicant.conf”, y se añade lo siguiente al final del archivo, cambiando los datos por los de nuestra red Wi-Fi.

```
network={
    ssid="nombre-de-nuestra-red-wifi"
    psk="password-de-nuestra-red-wifi"
    key_mgmt=WPA-PSK
}
```

Al agregar una o varias redes al fichero, éste queda configurado como lo indica la Figura 102. Se guardan los cambios realizados y se ejecuta el comando **sudo reboot**.

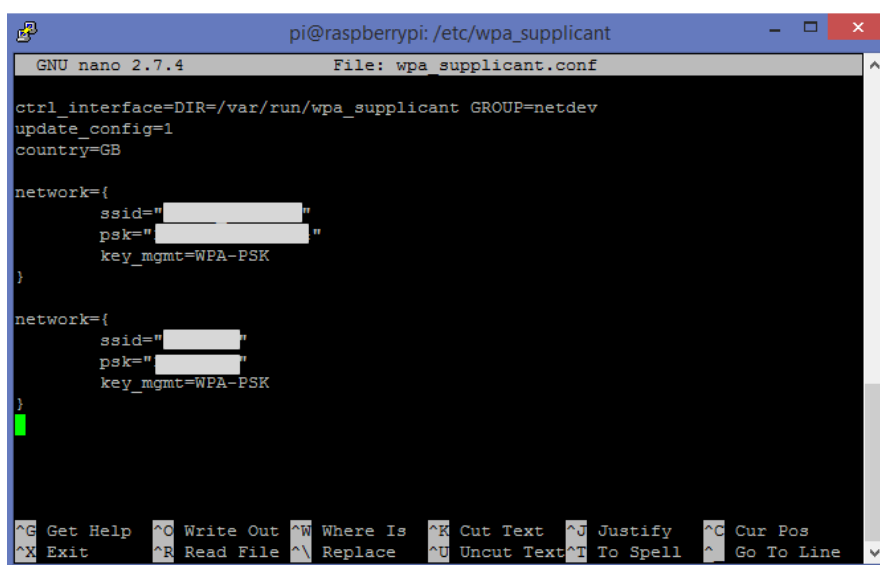


Figura 102. Agregación de una nueva red Wi-Fi a la tarjeta Raspberry Pi.

Fuente: [El Autor]

ANEXO VI: CÓDIGO EN PYTHON CORRESPONDIENTE A LA PROGRAMACIÓN DE LA CENTRAL DEL SISTEMA

Disponible en:

https://gitlab.com/JonasCS/proyecto_rasp_security?nav_source=navbar

ANEXO VII: CÓDIGO DEL SERVIDOR WEB DESARROLLADO

Disponible en: <https://gitlab.com/JonasCS/proyectoraspsecurityweb>

ANEXO VIII: DATASHEET DEL SENSOR PIR HC-SR501

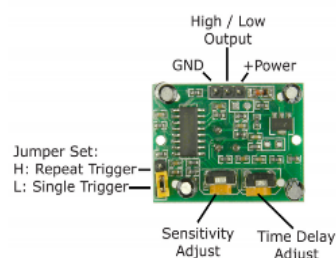
HC-SR501 PIR MOTION DETECTOR

Product Discription

HC-SR501 is based on infrared technology, automatic control module, using Germany imported LHI778 probe design, high sensitivity, high reliability, ultra-low-voltage operating mode, widely used in various auto-sensing electrical equipment, especially for battery-powered automatic controlled products.

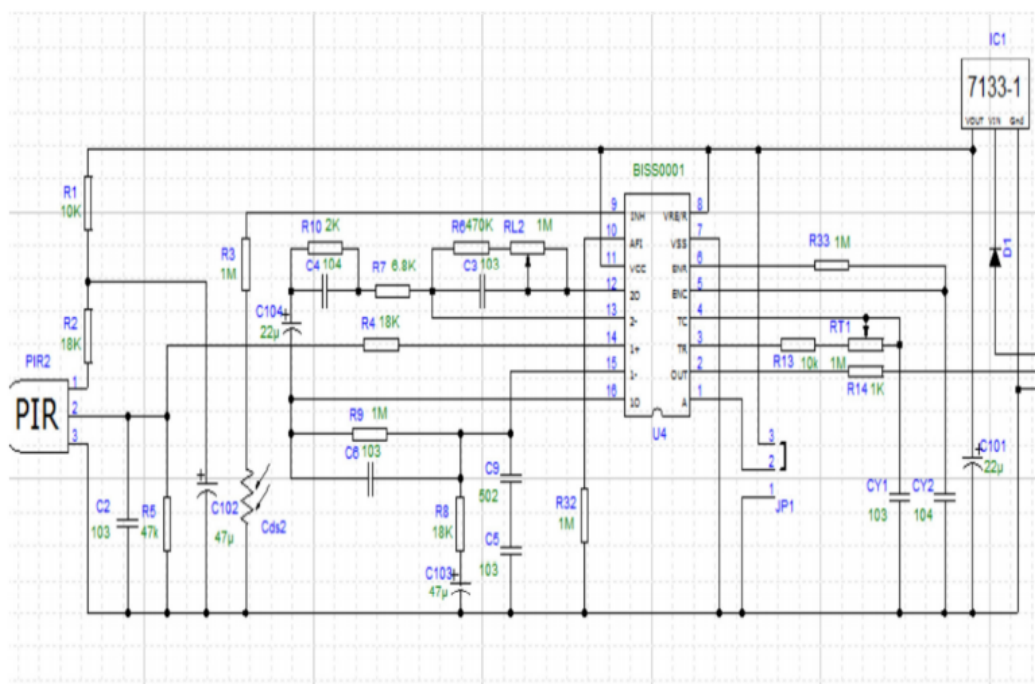
Specification:

- Voltage: 5V – 20V
- Power Consumption: 65mA
- TTL output: 3.3V, 0V
- Delay time: Adjustable (.3->5min)
- Lock time: 0.2 sec
- Trigger methods: L – disable repeat trigger, H enable repeat trigger
- Sensing range: less than 120 degree, within 7 meters
- Temperature: – 15 ~ +70
- Dimension: 32*24 mm, distance between screw 28mm, M2, Lens dimension in diameter: 23mm



Application:

Automatically sensing light for Floor, bathroom, basement, porch, warehouse, Garage, etc, ventilator, alarm, etc.



ANEXO IX: DATASHEET DEL SENSOR MQ2

MQ-2 Semiconductor Sensor for Combustible Gas

Sensitive material of MQ-2 gas sensor is SnO_2 , which with lower conductivity in clean air. When the target combustible gas exist, The sensor's conductivity is more higher along with the gas concentration rising. Please use simple electrocircuit, Convert change of conductivity to correspond output signal of gas concentration.

MQ-2 gas sensor has high sensitivity to LPG, Propane and Hydrogen, also could be used to Methane and other combustible steam, it is with low cost and suitable for different application.

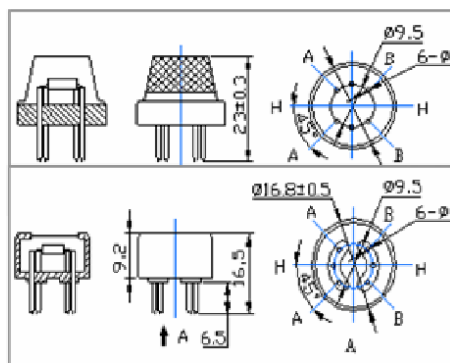
Character

- * Good sensitivity to Combustible gas in wide range
- * High sensitivity to LPG, Propane and Hydrogen
- * Long life and low cost
- * Simple drive circuit

Application

- * Domestic gas leakage detector
- * Industrial Combustible gas detector
- * Portable gas detector

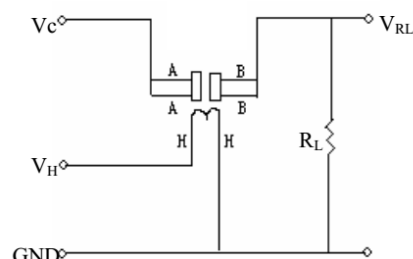
Configuration



Technical Data

Model No.		MQ-2	
Sensor Type		Semiconductor	
Standard Encapsulation		Bakelite (Black Bakelite)	
Detection Gas		Combustible gas and smoke	
Concentration		300-10000ppm (Combustible gas)	
Circuit	Loop Voltage	V_c	$\leq 24V$ DC
	Heater Voltage	V_H	$5.0V \pm 0.2V$ AC or DC
	Load Resistance	R_L	Adjustable
Character	Heater Resistance	R_H	$31\Omega \pm 3\Omega$ (Room Tem.)
	Heater consumption	P_H	$\leq 900mW$
	Sensing Resistance	R_s	$2K\Omega - 20K\Omega$ (in 2000ppm C_3H_8)
	Sensitivity	S	$R_s(\text{in air})/R_s(1000ppm \text{ isobutane}) \geq 5$
	Slope	α	$\leq 0.6(R_{5000ppm}/R_{3000ppm} CH_4)$
Condition	Tem. Humidity	$20^\circ C \pm 2^\circ C$; $65\% \pm 5\% RH$	
	Standard test circuit	$V_c: 5.0V \pm 0.1V$ $V_H: 5.0V \pm 0.1V$	
	Preheat time	Over 48 hours	

Basic test loop



The above is basic test circuit of the sensor. The sensor need to be put 2 voltage, heater voltage (V_H) and test voltage (V_C). V_H used to supply certified working temperature to the sensor, while V_C used to detect voltage (V_{RL}) on load resistance (R_L) whom is in series with sensor. The sensor has light polarity, V_c need DC power. V_C and V_H could use same power circuit with precondition to assure performance of sensor. In order to make the sensor with better performance, suitable R_L value is needed: Power of Sensitivity body (P_s):

$$P_s = V_c^2 \times R_s / (R_s + R_L)^2$$

ANEXO X: CERTIFICACIÓN DE LA EMPRESA TECSERLED

CORPORACION TECSERLED Tecnología y Servicios ®

Lourdes entre Olmedo y Juan José Peña – Loja

tecserled@hotmail.com | +593.7.2565881 | 0996842879 | 0994969766 / ventas y cotizaciones: 0960777711 (whatsapp/telegram)

CERTIFICADO

Por medio de la presente certificamos que el proyecto de grado para la obtención del título en Ingeniería en Electrónica y Telecomunicaciones titulado "Diseño e implementación de un sistema integral de seguridad utilizando software libre, basado en el estándar IEEE 802.15.4 para la empresa de tecnología y servicios TECSERLED de la ciudad de Loja" fue desarrollado e implementado en su totalidad en nuestra instalación ubicada en la ciudad de Loja calle Lourdes 162-30 entre Olmedo y Juan José Peña por el señor JONAS ELISEO CARRILLO SISALIMA

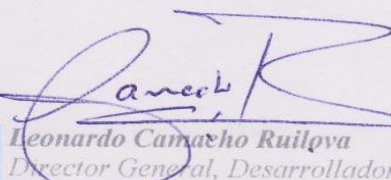
Es todo cuanto puedo certificar en honor a la verdad y quedo a su disposición cualquier inquietud. El solicitante del presente certificado puede hacer uso del mismo a su entera disposición.



Loja, 15 de julio de 2019

Atentamente,




Leonardo Camacho Ruilova
Director General, Desarrollador

Tel: (593) 7.2572-333 | Móvil: (593) 996842879



Venta, distribución y desarrollo de productos y sistemas basados en LED's (Diodo Emisor de Luz) • Proyectos de Iluminación con tecnología LED • Repuestos, componentes y suministros electrónicos • Servicio de Capacitación Tecnológica y Educativa • Venta, mantenimiento y reparación de maquinaria de computación y equipo periférico • Servicio de Laboratorio especializado de electrónica • Desarrollo de Proyectos Electrónicos y Eléctricos

ANEXO XI: DISPOSITIVOS IMPLEMENTADOS



Figura 103. Dispositivos implementados: ZONA 01 (Sensor Magnético), ZONA 02 (Sensor de Humo).

Fuente: [El Autor]

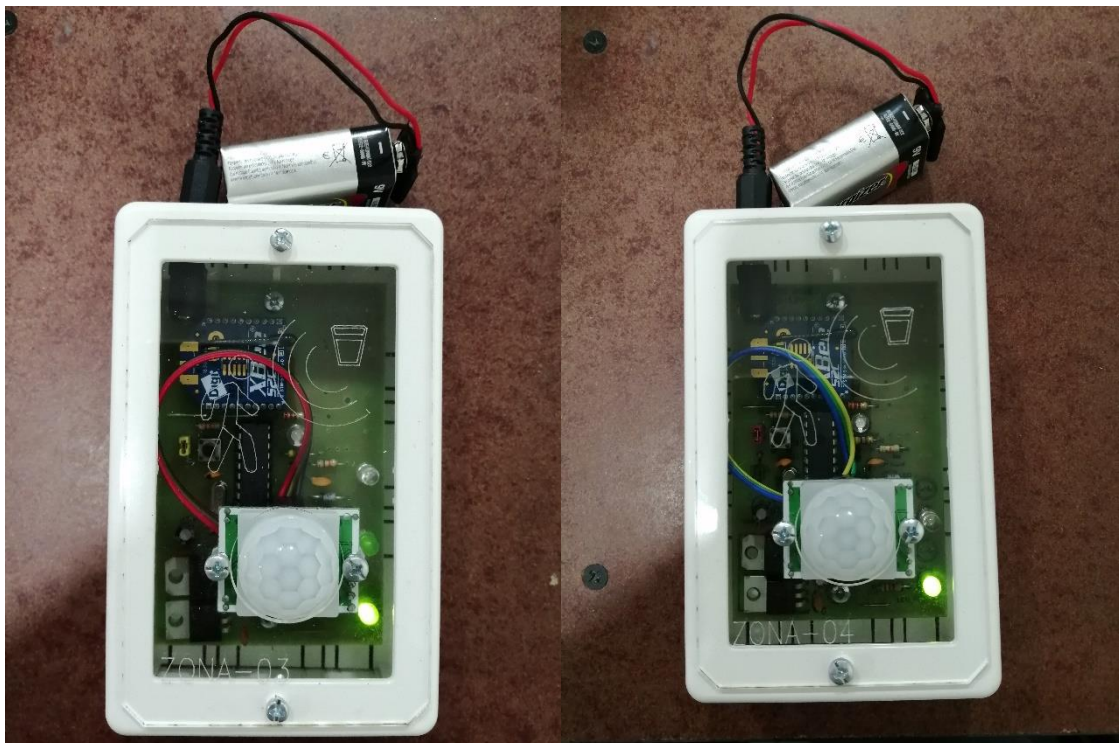


Figura 104. Dispositivos implementados: ZONA 03 (Sensor PIR), ZONA 04 (Sensor PIR).

Fuente: [El Autor]

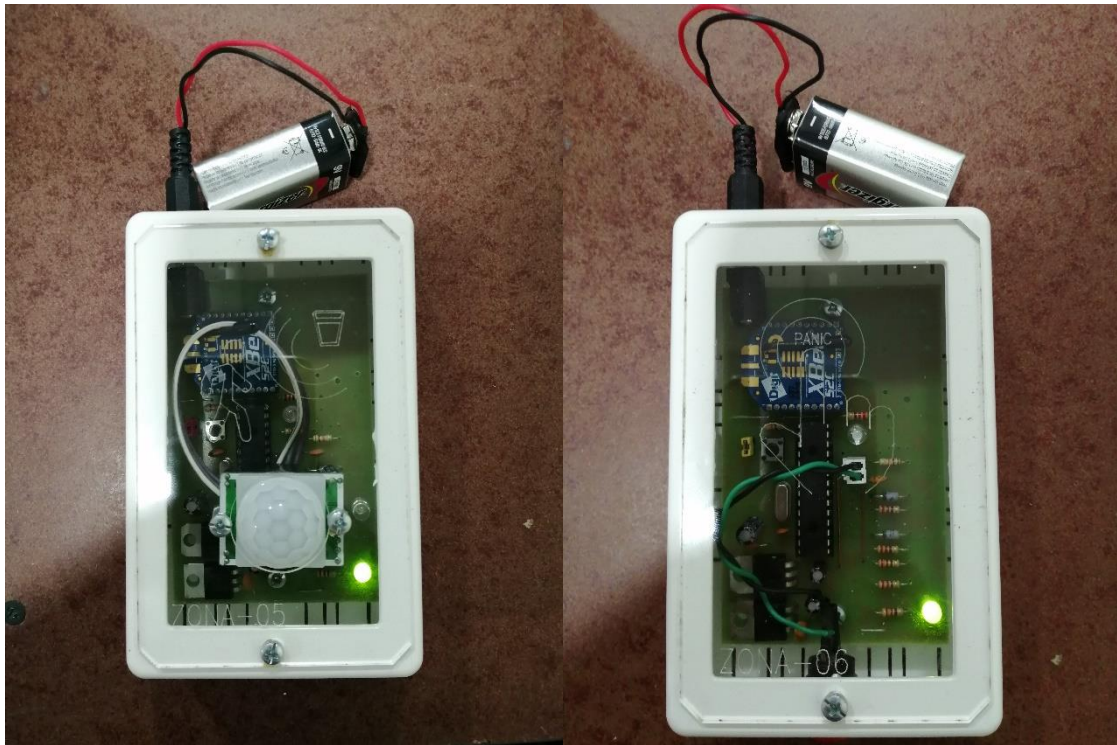


Figura 105. Dispositivos implementados: ZONA 05 (Sensor PIR), ZONA 06 (Botón de Pánico).

Fuente: [El Autor]