Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE NOTIFICACIÓN DE SERVICIO E INFORMACIÓN PARA LA UBICACIÓN DE LAS UNIDADES Y ESTACIONES DE UN SISTEMA DE TRANSPORTE PÚBLICO

Gerson Estuardo Alvarado Hernández
Asesorado por la Inga. Ingrid Salomé Rodríguez de Loukota

Guatemala, mayo de 2023

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE NOTIFICACIÓN DE SERVICIO E INFORMACIÓN PARA LA UBICACIÓN DE LAS UNIDADES Y ESTACIONES DE UN SISTEMA DE TRANSPORTE PÚBLICO

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

GERSON ESTUARDO ALVARADO HERNÁNDEZ
ASESORADO POR EL INGA. INGRID SALOMÉ RODRÍGUEZ DE LOUKOTA

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO ELECTRÓNICO

GUATEMALA, MAYO DE 2023

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANA	Ing. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Kevin Vladimir Armando Cruz Lorente
VOCAL V	Br. Fernando José Paz González
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANA	Inga. Aurelia Anabela Cordova Estrada
EXAMINADOR	Ing. Byron Odilio Arrivillaga Méndez
EXAMINADOR	Ing. Marvin Marino Hernández Fernández
EXAMINADOR	Ing. Carlos Alberto Navarro Fuentes
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE NOTIFICACIÓN DE SERVICIO E INFORMACIÓN PARA LA UBICACIÓN DE LAS UNIDADES Y ESTACIONES DE UN SISTEMA DE TRANSPORTE PÚBLICO

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica, con fecha 4 de septiembre de 2019.

Gerson Estuardo Alvarado Hernández

Guatemala 16 de septiembre 2022

Ingeniero
Julio César Solares Peñate
Coordinador del Área de Electrónica
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC.

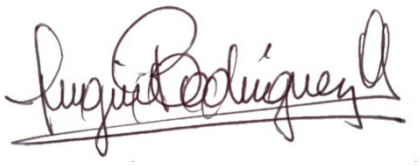
Apreciable Ingeniero Solares,

Me permito dar aprobación al trabajo de graduación titulado ¨**Diseño e implementación de un sistema de notificación de servicio e información para la ubicación de las unidades y estaciones de un sistema de transporte público**¨, del señor **Gerson Estuardo Alvarado Hernández**, por considerar que cumple con los requisitos establecidos.

Por tanto, el autor de este trabajo de graduación y, yo, como su asesora, nos hacemos responsables por el contenido y conclusiones de este.

Sin otro particular, me es grato saludarle.

Atentamente,



MSc. Ingrid Rodríguez de Loukota
Ingeniera en Electrónica
Colegiada 5,356
Asesora

Ingrid Rodríguez de Loukota
Ingeniera en Electrónica
colegiado 5356

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

Guatemala, 12 de octubre de 2022

Señor director
Armando Alonso Rivera Carrillo
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC

Estimado Señor director:

Por este medio me permito dar aprobación al Trabajo de Graduación titulado: **DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE NOTIFICACIÓN DE SERVICIO E INFORMACIÓN PARA LA UBICACIÓN DE LAS UNIDADES Y ESTACIONES DE UN SISTEMA DE TRANSPORTE PÚBLICO,** desarrollado por el estudiante **Gerson Estuardo Alvarado Hernández**, ya que considero que cumple con los requisitos establecidos.

Sin otro particular, aprovecho la oportunidad para saludarlo.

Atentamente,

ID Y ENSEÑAD A TODOS

Ing. Julio César Solares Peñate
Coordinador de Electrónica

ESCUELA DE INGENIERÍA MECÁNICA ELÉCTRICA
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

REF. EIME 22.2023

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto Bueno del Coordinador de área, al trabajo de Graduación del estudiante Gerson Estuardo Alvarado Hernández: **DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE NOTIFICACIÓN DE SERVICIO E INFORMACIÓN PARA LA UBICACIÓN DE LAS UNIDADES Y ESTACIONES DE UN SISTEMA DE TRANSPORTE PÚBLICO**, procede a la autorización del mismo.

Ing. Armando Alonso Rivera Carrillo

Guatemala, 24 de marzo de 2023

Facultad de Ingeniería

Decanato
24189101-
24189102
secretariadecanato@ingenieria.usac.edu.gt

LNG.DECANATO.OI.478.2023

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Mecánica Eléctrica, al Trabajo de Graduación titulado: **DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE NOTIFICACIÓN DE SERVICIO E INFORMACIÓN PARA LA UBICACIÓN DE LAS UNIDADES Y ESTACIONES DE UN SISTEMA DE TRANSPORTE PÚBLICO,** presentado por: **Gerson Estuardo Alvarado Hernández,** después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANA
FACULTAD DE INGENIERÍA

Inga. Aurelia Anabela Cordova Estrada
Decana

Guatemala, mayo de 2023

AACE/gaoc

ACTO QUE DEDICO A:

Dios	Por permitirme superar los obstáculos durante el camino para poder alcanzar este logro personal.
Mis padres	Cecilia de Alvarado, Sergio Alvarado (q. e. p. d.), por los valores inculcados por todo el apoyo, cariño y sacrificios realizados.
Mi hermano	Alan Alvarado, por el apoyo, cariño y el acompañamiento en las desveladas, cuando me ayudabas con los proyectos.
Mis abuelos	Sonia de Oliva (q. e. p. d.), Concepción de Alvarado, German Oliva y Vicente Alvarado, por todo su cariño.
Mi tío	César Ramírez, por ser un ejemplo de superación y perseverancia, por los momentos compartidos, por todo, muchas gracias.
Mi familia	Por contribuir de distintas formas en mi formación como persona.

Mis amigos

Victor Pérez, Ligia Aguilar, Mónica Marroquín, Helmuth Palacios, José Carlos Quiñonez, Juan Valdez y todos aquellos con los que conviví, por todas las experiencias vividas, por escucharme y apoyarme en mis momentos más bajos, por animarme a seguir intentando y no abandonar cuando todo parecía perdido, los considero parte de mi familia.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por proveer las instalaciones necesarias para el desarrollo de las actividades académicas y permitirme formarme.
Facultad de Ingeniería	Por proveer el personal necesario para mi formación académica.
Laboratorios de Electrotecnia	Por brindarme la oportunidad de crecer profesionalmente.
Ingeniera Ingrid de Loukota	Por su apoyo para realizar el trabajo de graduación y durante los cursos en la carrera.
Ingeniera María Zaghi	Por invitarme a participar en eventos donde expandí mis horizontes, descubrí nuevos temas de estudio y campos de aplicación.
Unidad de Salud de La Universidad de San Carlos	En especial a la Dra. Rita María Figueroa Figueroa y a la enfermera Basilia Lorena Matías Oxlaj.
Hospital Temporal Parque de la Industria	A todo su personal, por los servicios médicos prestados durante mi estancia en sus instalaciones.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
ACK	Acuse de recibido
ARM	*Advanced* RISC *Machines*
PoE	Alimentación a través de *Ethernet*
USB	Bus serial universal
ASCII	Código Estándar Estadounidense para el intercambio de información
NTSC	Comité de Sistemas de Televisión Nacionales de Estados Unidos
ADC	Convertidor analógico digital
dBm	Decibelio mili vatio
RPD	Detector de potencia recibida
GPIO	Entrada/salida de propósito general
RS-232	Estándar recomendado 232 (puerto)
TCP/IP	Familia de protocolos de Internet
GHz	Gigahercio
CE	Habilitador de chip
CSN	Habilitador/selector de chip
Hz	Hercio
APT	Herramienta de empaquetado avanzado
PID	Identificador del paquete
UUID	Identificador único universal
ISO	Imagen de disco
ANSI	Instituto Nacional Estadounidense de Estándares

OSI	Interconexión de sistemas abiertos
HDMI	Interfaz multimedia de alta definición
SPI	Interfaz Serial de Periféricos
IoT	Internet de las cosas
SSH	Intérprete seguro de comandos
kbps	*Kilobit* por segundo
kHz	Kilohercio
SQL	Lenguaje de Consulta Estructurada
XHTML	Lenguaje de marcado de hipertexto extensible
XML	Lenguaje de marcado extensible
URL	Localizador de recursos uniforme
Mbps	*Megabit* por segundo
MHz	Megahercio
RAM	Memoria de acceso aleatorio
m/s	Metro por segundo
µA	Microamperio
mA	Miliamperio
ms	Milisegundo
AM	Modulación de amplitud
PM	Modulación de fase
FM	Modulación de frecuencia
PAM	Modulación por amplitud de pulsos
PCM	Modulación por codificación de pulsos
GFSK	Modulación por desplazamiento de frecuencia gaussiana
PDM	Modulación por duración de pulsos
PPM	Modulación por posición de pulsos
CDM	Multiplexado por división de código
FDM	Multiplexado por división de frecuencia

WDM	Multiplexado por división de longitud de onda
TDM	Multiplexado por división de tiempo
nA	Nano amperio
nm	Nanómetro
JSON	Notación de objeto de JavaScript
FIFO	Primero en entrar, primero en salir
IP	Protocolo de Internet
HTTP	Protocolo de transferencia de hipertexto
DHCP	Protocolo dinámico de configuración del cliente
RX	Receptor
PRX	Receptor primario
SCK	Reloj serial
MISO	Salida de datos del esclavo y entrada al maestro
MOSI	Salida de datos del maestro y entrada al esclavo
s	Segundo
DNS	Servidor de nombres de dominio
IRQ	Solicitud de interrupción
TX	Transmisor
PTX	Transmisor primario
CPU	Unidad de procesamiento central
GPU	Unidad gráfica de procesamiento
CRC	Verificación de redundancia cíclica
V	Voltaje

GLOSARIO

Adafruit Es una compañía de *hardware* libre establecida en 2006 por Limor Fried en la ciudad de Nueva York.

Adaptador Python Traduce los comandos desde el intérprete Python a la interfaz nativa del controlador en cuestión.

Ancho de banda Longitud de la extensión de frecuencias en donde se encuentra la mayor potencia de la señal.

Arduino Compañía de *hardware* y *software* abierto que diseña placas de desarrollo con microcontroladores para la fabricación de dispositivos electrónicos.

Bit Se le denomina así a un digito individual del sistema de numeración binario que representa la unidad mínima de información.

Broadcom Es uno de los principales fabricantes de circuitos integrados.

Búfer de datos Espacio de memoria en donde se almacenan datos de manera temporal.

Byte Unidad base de información que se encuentra compuesta por un conjunto ordenado de ocho *bits*.

C	Lenguaje de programación de propósito general desarrollado entre 1969 y 1972.
Canal	Medio por el cual viajan las señales que transportan información entre transmisor y receptor.
Carga útil	Conjunto de datos seleccionados que se transmiten durante el proceso de comunicación, se excluyen de esta denominación las cabeceras, metadatos y otro tipo de información adicional que sea utilizada en el proceso de transmisión.
CircuitPython	Proyecto derivado de MicroPython el cual es más amigable con los usuarios principiantes.
Cliente	En una aplicación informática, se refiere a la aplicación u ordenador que consume un servicio remoto proporcionado por un servidor.
Codificador	Sistema o dispositivo que regula una serie de transformaciones sobre una señal o fragmento de información para representarla en un formato específico.
Consola	Conocida también como emulador de terminal, permite al usuario ingresar instrucciones a un programa o sistema operativo utilizando una línea de texto simple.

Cookie	Término utilizado para llamar a la información almacenada por un sitio web en el navegador con el fin de recordar configuraciones u opciones del usuario entre visitas.
Decibelio	Unidad que se utiliza para representar la relación entre dos valores de presión sonora, tensión o potencia eléctrica.
Decodificador	Sistema o dispositivo que a partir de un formato especifico puede recuperar la mayor parte de la información original manipulada por un codificador.
Django	*Framework* de desarrollo web de alto nivel.
Enhaced ShockBurst	Protocolo de banda base diseñado especialmente para aplicaciones inalámbricas de ultra bajo consumo.
Entrono virtual Python	Directorio que contiene un intérprete de Python así como librerías y scripts de forma que funcionen aislados de los demás elementos del sistema operativo.
Framework	Entorno de trabajo que puede contener programas, bibliotecas entre otras herramientas para facilitar el desarrollo de los componentes de un proyecto.
FreeCad	Aplicación de diseño asistido en tres dimensiones de código abierto.

Función *hash* Función utilizada para convertir uno o varios elementos a cadenas de información de longitud fija.

Fundación Raspberry Pi Organización sin fines de lucro con sede en Reino Unido encargada de promover avances en la educación relacionados al campo de la computación, informática, entre otros.

Git *Software* de control de versiones.

GNU/Linux Familia de sistemas operativos tipo Unix, usualmente clasificado como *software* libre y código abierto.

Hardware Conjunto de componentes físicos, tangibles de un sistema informático.

Inkscape Editor de gráficos vectoriales de código abierto.

Interfaz gráfica Programa informático que sirve como interfaz de usuario utilizando un conjunto de elementos gráficos para representar información y acciones disponibles.

KiCad Conjunto de programas para la automatización del diseño electrónico de código abierto.

Licencia BSD Licencia de *software* utilizada en los sistemas BSD, es una licencia de *software* libre permisiva.

Licencia MIT	Licencia de *software* utilizada por el Instituto de Tecnología de Massachusetts, es una licencia de *software* libre permisiva.
Microcontrolador	Circuito integrado programable, capaz de ejecutar instrucciones almacenadas en la memoria, se compone de una unidad central de procesamiento, memoria o memorias y pines de propósito general de entrada y salida.
MicroPython	Es una implementación de la versión 3 del lenguaje programación Python escrita en C, optimizada para su uso en microcontroladores.
OSHPark	Fabricante de placas de circuitos impresos.
Petición *GET*	Solicita la representación de un recurso web específico.
Petición *POST*	Envía una entidad a un recurso web específico.
PGXN	Sistema centralizado de distribución de extensiones para PostgreSQL.
Placa circuito impreso	Superficie compuesta por caminos de un material conductor sobre una base no conductora utilizados para conectar eléctricamente y sostener mecánicamente un conjunto de componentes electrónicos.

PostgreSQL	Sistema de gestión de bases de datos relacional orientado a objetos, de código abierto.
Receptor	Dispositivo electrónico que utiliza las ondas electromagnéticas para recibir la información enviada por un transmisor.
Rust	Lenguaje de programación compilado de propósito general y multiparadigma desarrollado inicialmente en 2006.
Señal eléctrica	Resultado de un fenómeno electromagnético, puede ser continuo en el tiempo o variar de forma discreta.
Servidor	Conjunto de computadoras que pueden procesar las peticiones de un cliente y devolver una respuesta acorde a una solicitud.
Software	Conjunto de componentes lógicos que permiten la realización de tareas específicas dentro de un sistema informático.
***Software* Libre**	*Software* que permite el acceso a su código fuente para ser estudiado, modificado, utilizado con cualquier finalidad e incluso redistribuirlo con los cambios aplicados.

SparkFun Electronics	Compañía que vende componentes electrónicos, además diseña y fabrica placas de desarrollo que son liberadas como *hardware* abierto.
Transmisor	Dispositivo electrónico que utiliza las ondas electromagnéticas para transmitir información como audio, televisión o telefonía móvil.
Wifi	Tecnología que permite la conexión inalámbrica entre dispositivos electrónicos.

RESUMEN

El mundo de la electrónica de consumo avanza rápidamente cada año, este avance continuo es producto del constante trabajo de las empresas encargadas de manufacturar o desarrollar nuevos componentes, de diseñar nuevos equipos y procesos para su fabricación.

Similar es el caso del Internet que, desde sus inicios en 1969 ha evolucionado hasta convertirse en parte primordial de las actividades del mundo moderno, un campo de aplicación que surge como consecuencia de esta asociación es el denominado Internet de las cosas (IoT), término usualmente atribuido a Kevin Ashton que en 1999 describía un mundo en el cual varios herramientas, dispositivos o sistemas enteros estarían conectados a Internet sin la intervención humana, se estima que en 2025 la industria del IoT tendrá un impacto de aproximadamente 11,1 billones de dólares en la economía mundial.

Según el informe del 2018 por parte de The World´s Cities de Naciones Unidas, se estima que un 55,3 % de la población mundial vive en asentamientos urbanos y para 2030 se proyecta que la cifra llegue al 60 %, con este aumento crece la necesidad de automatizar ciertos procesos dentro de las ciudades, al utilizar IoT para resolver estos problemas se crea un nuevo campo de aplicación comúnmente denominado Ciudades Inteligentes.

Este trabajo comprende el desarrollo de un sistema IoT que ayuda a los gestores de sistemas de transporte público a manejar y notificar a los usuarios del estado actual del servicio en casi tiempo real.

OBJETIVOS

General

Diseñar un sistema que muestre información relacionada con el estado del servicio de transporte público de uso simple para los usuarios y para los operadores de los servicios de transporte.

Específicos

1. Facilitar el acceso a la información del estado de la ruta a la mayor cantidad de usuarios dentro del sistema de transporte.

2. Facilitar a los operadores el proceso de gestión de información y notificaciones relacionadas con el sistema de transporte.

3. Determinar las opciones disponibles de sistemas embebidos que pueden ser utilizados en el proceso de modernización de los sistemas de transporte públicos.

4. Diseñar los componentes del sistema de forma que sean compactos y de fácil adquisición.

INTRODUCCIÓN

El tráfico urbano es uno de los problemas que más influyen en la calidad de vida de los habitantes de las áreas urbanas y sus alrededores, esto se debe a la expansión territorial de las ciudades que afectan entre otros a la infraestructura y servicios como el transporte público o privado, en algunos casos pueden llegar a representar cuantiosos gastos por parte de los operadores de los servicios, así como problemas de gestión para las autoridades municipales o de gobierno.

Los avances tecnológicos en temas de fabricación y desarrollo de microcontroladores junto a su bajo coste y gran número de entornos de desarrollo disponibles algunos con licencia libre han impulsado un mercado emergente de nuevas implementaciones de soluciones tecnológicas en temas cotidianos.

Muchas ciudades en el mundo han adoptado o se encuentran en proceso de adoptar nuevos sistemas a través del desarrollo de nuevas tecnologías. Los beneficios de la modernización del transporte público incluyen, entre otros, facilitar el pago del importe del pasaje, así como acceso a información casi en tiempo real de la ruta o cualquier posible incidencia que surgiera durante el viaje o proporcional al viajero información sobre el tiempo estimado en llegar a la próxima parada en pantallas dentro de las unidades.

1. SISTEMAS DE TRANSPORTE PÚBLICO

El término de transporte público se utiliza para describir a los servicios de transporte compartidos entre personas que no guardan relación entre ellas o que no han tenido un acuerdo previamente fijado para compartir dicho servicio, estos se encuentran disponibles al público en general, pueden incluir buses, trenes de superficie o subterráneos, entre otros.

Se excluye de este término a los taxis, los servicios de alquiler de automóviles y renta de buses, que son utilizados realizando un acuerdo previo.

1.1. Clasificación de los sistemas de transporte urbano

El transporte urbano puede clasificarse de forma básica tomando en cuenta el tipo de servicio que prestan o por el volumen de viajes que manejan.

Tabla I. **Clasificación de los sistemas de transporte público**

Tipo de servicio	**Por el volumen de viajes que manejan**
Transporte privado	Individual
Transporte de alquiler	Grupal
Transporte público	

Fuente: elaboración propia, empleando Microsoft Word.

Al referirse a transporte privado, se hace mención del que es operado por el propietario de la unidad, circulando en la vía pública que es operada y

mantenida por el Estado o Gobierno, entre estos podemos incluir automóviles, bicicletas, transporte de tracción animal, entre otros.

En cambio, el transporte de alquiler se refiere al que es utilizado por cualquier persona que pague una tarifa en vehículos proporcionados por un operador, chofer o empleado, ajustándose a los deseos de movilidad del usuario, entre estos se puede mencionar servicios de taxi particular, taxi de empresas, Uber y similares.

El transporte público, opera con rutas fijas y horarios predeterminados y que pueden ser utilizados por cualquier persona a cambio del pago de una tarifa previamente establecida, como, por ejemplo: los sistemas Transmetro o Transurbano.

Al referirse a los sistemas de transporte urbano por el volumen de viajes que manejan se tiene el utilizado para transporte individual, es cuando un vehículo sirve a una persona o un grupo organizado de usuarios que se dirigen a un mismo destino o el transporte en grupos, se refiere a cuando una unidad traslada a personas sin ninguna relación entre sí y con destinos diferentes.

1.2. Características de los medios de transporte públicos

Las diferencias entre los sistemas de transporte público se pueden establecer a partir de tres características principales.

1.2.1. Tipo de derecho de vía

Porción de la vía pública por donde circulan las unidades de transporte, incluyendo el peatón.

Tabla II. **Características de los sistemas de transporte público por su derecho de vía**

Tipo de operación o separación	Cruce compartido	Separación longitudinal	Separación vertical
Mixto	Si	No	No
Longitudinal	Si	Si	No
Longitudinal y vertical	No	Si	Si

Fuente: elaboración propia, empleando Microsoft Word.

La operación con tránsito mixto es cuando la vía pública es compartida entre varios medios de transporte, esta operación puede incluir tratos preferenciales en todo o algunas partes de su desarrollo, incluyendo aquellas calles por donde se tienen acciones de preferencia hacia el transporte público de pasajeros.

Figura 1. **Ejemplo de tipo de vía con operación de tránsito mixto**

Fuente: Municipalidad de Guatemala. *Expreso UMG, ahora puedes tomar Transmetro para ir a la U*. https://tmblr.co/ZIHb7t2M8RI1m. Consulta: 13 de agosto de 2020.

Figura 2. **Ejemplo de vía de tránsito mixto con señalización para transporte público**

Fuente: Municipalidad de Guatemala. *Señalización en eje nororiente del Transmetro*. https://tmblr.co/ZIHb7t2BshHY6. Consulta: 13 agosto 2020.

La operación con separación física longitudinal se refiere a utilizar elementos físicos fijos, tales como barreras o guarniciones para separar la operación de las unidades del servicio de los demás vehículos en la vía pública. Se mantienen los cruces a nivel con otros vehículos como con los peatones.

Figura 3. **Ejemplo de vía con separación física longitudinal**

Fuente: Municipalidad de Guatemala. *Finalizada pavimentación en pista de Transmetro*. https://tmblr.co/ZIHb7t2VSFbVf. Consulta: 13 agosto 2020.

Figura 4. **Ejemplo de vía con separación física longitudinal**

Fuente: Dirección de obras Municipalidad de Guatemala. *Señalización en "Centra Atlántida", zona 18*. http://www.muniguate.com/blog/2020/02/14/senalizacion-en-centra-atlantida-zona-18/. Consulta: 13 agosto 2020.

En la operación con separación física longitudinal y vertical, se evita cualquier interferencia entre vehículos y peatones. Estos sistemas pueden ser subterráneos, elevados o a nivel.

Figura 5. **Ejemplo de vía con separación física longitudinal y vertical**

Fuente: Johnson, Hans. *Línea Yutorito Japón*. https://www.flickr.com/photos/hansjohnson/29749140406. Consulta: 13 agosto 2020.

Figura 6. **Ejemplo de vía con separación física longitudinal y vertical**

Fuente: Oka21000, *Línea Yutorito Japón*.
https://www.flickr.com/photos/hansjohnson/29749140406. Consulta: 13 agosto 2020.

1.2.2. Tipo de tecnología utilizada

Se asocia con las propiedades mecánicas de las unidades de transporte y las del camino mismo, estas dos propiedades están relacionadas entre sí y se tienen cuatro componentes principales a considerar.

Soporte, es el contacto vertical entre la unidad de transporte y la superficie de rodamiento sobre la que se transfiere el peso del mismo vehículo.

Guía, forma que permite controlar al vehículo en sus movimientos laterales. Se presentan dos tipos fundamentales:

- Sistemas dirigidos desde el vehículo o a través de un volante.

- Sistemas que su control lateral viene dado por las guías o rieles con que cuenta.

Propulsión, se refiere al tipo de unidad motriz con que cuenta el vehículo, así como el método de transferir las fuerzas de aceleración y desaceleración.

Control, forma que permite regular los movimientos de las unidades de transporte que operan en un sistema, manual-visual; manual-señal, completamente automático.

1.2.3. Tipo de servicio

Básicamente se refiere a los tipos de rutas que se presentan en el sistema y a la forma y horario en que opera el sistema de transporte.

Tabla III. **Clasificación de los sistemas de transporte público por el tipo de servicio**

Tipo de ruta	**Tipo de operación**	**Horario de operación**
Frecuencia intensiva	Local	Regular
Transporte urbano	Paradas alternas	Punta
Regionales o suburbanas	Expreso	Valle
		Especiales

Fuente: elaboración propia, empleando Microsoft Word.

Cuando se hace referencia al tipo de ruta de frecuencia intensiva se refiere a los servicios de baja velocidad con altas intensidades de viajes dentro de pequeñas áreas, las rutas de transporte urbano se refieren a los servicios que se encuentran comúnmente en una ciudad, en cambio al referirse a las rutas de transporte regionales o suburbanas se habla de rutas que permiten obtener altas velocidades con pocas paradas a lo largo del trayecto y sirviendo a viajes de cierta longitud dentro de un área metropolitana.

En cuando a los tipos de operación el servicio local se refiere a los sistemas que hacen uso extensivo a todas las paradas a lo largo de la ruta, en los servicios de paradas alternas como su nombre lo indica se alterna el servicio en las paradas a lo largo de una ruta con el fin de acelerar el servicio, en cambio en el servicio expreso se busca lograr velocidades comerciales altas mediante el espaciamiento de las paradas por arriba del promedio del sistema.

Entre los distintos horarios de operación de los sistemas de transporte público se encuentra el horario regular es el que se refiere al horario normal de operación, dentro de este podemos encontrar el horario punta que se refiere a las franjas horarias en que el sistema tiene una mayor demanda, por el contrario el horario valle son las zonas horarias de baja demanda pero en las cuales se debe mantener una regularidad en el servicio, por último los servicios especiales se refiere a la operación de las unidades del sistema durante eventos especiales o emergencias cuyo horario se encuentra delimitado por el evento mismo.

1.3. Componentes físicos de los sistemas de transporte

Los sistemas de transporte se componen principalmente de tres elementos físicos:

- Vehículo: unidades de transporte, su conjunto se describe como parque vehicular en el caso de autobuses o trolebuses y equipo rodante para el caso de transporte férreo.

    - Unidad de transporte: se le puede denominar de esta forma a un solo vehículo o a un agrupamiento de vehículos que formen un tren, que cumplan con la condición de operar conjuntamente como uno solo.

- Infraestructura: derecho de vías en que operan los sistemas de transporte, sus paradas o estaciones. Estaciones normales, terminales, puntos de transbordo, garajes, depósitos, talleres de mantenimiento y reparación. Sistemas de control, detección, comunicación, señalización. Sistema de suministro de energía.

- Red de transporte: está compuesta por las rutas de las unidades de transporte que operan en una ciudad.

1.4. Antecedentes históricos del transporte colectivo en la Ciudad de Guatemala

En el año 1882, se inició en la Nueva Guatemala de la Asunción, el servicio de tranvías tirado por caballos. Había cinco líneas que tenían como punto de partida la Plaza Central hacia el Calvario, la Estación del Ferrocarril del sur, el Cementerio General, El Hipódromo y la Parroquia Vieja, cubriendo toda la ciudad. Después del terremoto de 1918 se sustituyeron en algunas líneas, por tranvías accionados por motor.

Luego de que en la Municipalidad se presentarán algunos reclamos por parte de uno de los concejales, en 1927 la municipalidad capitalina realizó una licitación pública para instalar un servicio moderno y eficiente de nuevos tranvías en la ciudad. La empresa ganadora nunca llevó a cabo el proyecto por lo que el antiguo sistema continúo hasta el año 1929.

La Empresa Guatemalteca de Autobuses, con el servicio de tranvías aún funcionando, inició desde finales del año 1927 el servicio con buses colectivos, que fueron fabricados utilizando motores de camión importados y adaptándole carrocería artesanal de madera. Así surge el sistema de transporte colectivo de

autobuses como una iniciativa privada, sin contar con autorización municipal, la cual fue tramitada posteriormente.

Debido al desorden y deficiencias que presentaba el sistema de transporte colectivo surgido sin ninguna regulación por parte de la Municipalidad, se crea el primer reglamento para normal el servicio elaborado por el Juzgado de Tránsito, de la Dirección General de la policía, el cual fue puesto en vigencia en mayo de 1930, cuyo desacato no se hizo esperar.

En el año 1940 aún daban servicio algunos pocos carruajes, que podían localizarse en el Incienso, el Amate, el Callejón del Conejo y el Botellón. La ciudad de pequeñas dimensiones permitía aún la movilización de la población caminando o por medio de bicicletas. El transporte extraurbano se componía de no más de 100 buses, que se utilizaban casi exclusivamente para transportar a los comerciantes de víveres para el consumo de la población capitalina.

En 1944 la Intendencia Municipal designó las rutas con números. Para entonces habían aumentado a 16 rutas para cubrir la ciudad y sus cantones. Con la revolución 1944, el Gobierno ante las constantes quejas del servicio de autobuses colectivos promovió por medio de un proyecto de ley la municipalización del servicio. El proyecto de ley de municipalización del servicio de transporte colectivo no fue aceptado por el Congreso.

En las décadas de los años cincuenta y sesenta, la expansión de la Ciudad de Guatemala se produjo a través de la ocupación de zonas cada vez más alejadas del núcleo urbano central, lo que provocó que el proceso de expansión se situará en áreas periféricas del municipio de Guatemala e incluso fuera de su jurisdicción. Ello sin enmarcarse en una mínima planificación urbana.

Luego del primer intento del alza a la tarifa en 1969, en los años siguientes se sucedieron una serie de incrementos por parte de las autoridades del gobierno de turno a las que precedieron múltiples actos de protesta por parte de los usuarios.

En 1994 la municipalidad inició su propio servicio de transporte público, fue denominado Munitrans, contaba con una tarifa menor pero la incomodidad para los usuarios lo hacía poco atractivo, el servicio funcionó solamente alrededor de un año debido a que los costos de servicio eran muy altos.

El 3 de febrero de 2007 comienza el funcionamiento del servicio de transporte de la Municipalidad de Guatemala denominado Transmetro, que en la actualidad cuenta con 7 líneas de servicio. El 3 de julio de 2010 con el apoyo del Gobierno de turno inicia operaciones el servicio de transporte público denominado Transurbano.

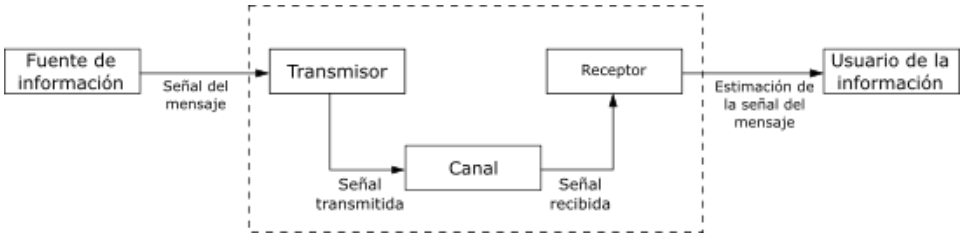
2. FUNDAMENTOS DE LA COMUNICACIÓN INALÁMBRICA

2.1. Proceso de comunicación

De forma generalizada, la comunicación es el proceso por el cual se manifiesta la transmisión de información de un punto a otro a través de una serie de pasos o procesos, los cuales se pueden definir como:

- La generación de la señal del mensaje.
- La descripción de esa señal del mensaje con cierta medida de precisión, mediante un conjunto de símbolos que pueden ser auditivos, eléctricos o visuales.
- La codificación de esos símbolos en una forma que sea adecuada para la transmisión por un medio físico de interés.
- La transmisión de estos símbolos codificados al destino deseado.
- La decodificación y la reproducción de los símbolos originales.
- La recreación de la señal del mensaje original, con una degradación definible en la calidad; la degradación la ocasionan las imperfecciones del sistema.

Figura 7. **Elementos de un sistema de comunicación**



Fuente: elaboración propia, empleando Inkscape 1.0.

Independientemente de la forma del proceso de comunicación que se esté analizando, existen tres elementos básicos para todo sistema de comunicación: transmisor ubicado en algún punto en el espacio, receptor ubicado en otro punto en el espacio separado del transmisor y el canal es el medio físico que los conecta, ver figura 7.

El transmisor es el encargado de convertir la señal del mensaje producida por la fuente de información en una forma adecuada para la transmisión por el canal, la señal al propagarse por el canal se degrada debido a las imperfecciones de este, el ruido y las señales de interferencia originadas en otras fuentes también se suman a la corrupción de la señal transmitida.

El receptor tiene la tarea de actuar sobre la señal recibida de manera que pueda reconstruir de forma entendible para el usuario la señal del mensaje original.

Existen dos modos básicos de comunicación:

- Transmisión, implica el uso de un transmisor poderoso y de numerosos receptores cuya construcción es relativamente económica, es este caso

las señales que contienen la información solamente fluyen en una dirección.

- Comunicación punto a punto, los procesos de comunicación se llevan a cabo por un enlace entre un solo transmisor y un receptor. En la mayoría de las veces que se utiliza este modo, existe un flujo bidireccional entre las señales que llevan la información que requiere el emisor como el receptor.

Un ejemplo del modelo de comunicación por transmisión es la radio y televisión, mientras que el teléfono o el control a distancia de robots es un ejemplo de comunicación punto a punto.

2.2. Elementos de un proceso de comunicación

Se puede afirmar que en un sistema de comunicación se emplean dos recursos primarios: la potencia transmitida y el ancho de banda del canal. La potencia transmitida se refiere a la potencia promedio de la señal que se desea transmitir, el ancho de banda se define como la sección de las frecuencias que se utilizan para la transmisión de la señal del mensaje.

Uno de los objetivos para tener en cuenta cuando se realiza el diseño general del sistema es el de utilizar los recursos antes mencionados en la forma más eficiente posible, debido a esto los canales de comunicación se pueden clasificar como limitados por potencia o limitados por ancho de banda.

El ruido es un elemento importante que se debe tener presente en un sistema de comunicación, este se refiere a ondas indeseables que tienden a

perturbar la transmisión y el procesamiento de las señales dentro del sistema de comunicación, estas pueden tener origen dentro o fuera del sistema.

La forma en la que se cuantiza el efecto del ruido en el sistema es introducir el uso de la relación señal a ruido como una característica del sistema, esta relación se define como la proporción entre la potencia promedio de la señal y la potencia promedio del ruido, ambas cantidades deben obtenerse en el mismo punto. Es común que se exprese en decibeles, los cuales son definidos como 10 veces el logaritmo decimal de la proporción de potencia.

2.3. Grupos de información

La información que se maneja en el mundo de las telecomunicaciones se puede separar en cuatro grupos: voz, música, imágenes y datos, se puede identificar la fuente de la información guiándose por la señal que porta la información.

Una señal se define como una función de un solo valor de tiempo que desempeña el papel de la variante independiente; en cualquier instante, la función tiene un valor único, la señal puede ser unidimensional como la voz, música o datos de computadora; bidimensional, como en el caso de las imágenes; tridimensional, como el caso de los datos de vídeo; y cuadrimensional, como en el caso de datos de volumen a lo largo del tiempo.

2.3.1. Voz

El habla es la forma de comunicación por excelencia de los seres humanos. El proceso de comunicación del habla involucra la transmisión de información

desde un ente a otro, procedimiento que en la mayoría de los casos ocurre en al menos tres etapas sucesivas:

- Producción: ocurre en la mente del emisor que desea expresarse, se representa mediante una señal del habla que es compuesta por sonidos que se generarán dentro del tracto vocal y cuyo arreglo está definido por las reglas del lenguaje que ocupa el emisor.

- Propagación: luego de que son emitidas, la velocidad de propagación de las ondas sonoras en el aire es de 300 m/s, llegando al receptor del mensaje.

- Percepción: El receptor debe descifrar los sonidos del mensaje recibido que llegan a sus oídos, de esta forma se completa el proceso de transferencia de información entre el emisor y receptor.

A este proceso se le puede considerar como una forma de filtrado.

2.3.2. Música

Se origina en instrumentos musicales, estos producen notas que quizá duren unos instantes o sean sostenidas por un intervalo de tiempo largo, comúnmente la música tiene dos componentes fundamentales: la melodía, es una secuencia de tiempo y de sonidos y una armonía, que se conforma por una serie de sonidos simultáneos.

A diferencia de la señal de la voz, la música posee un espectro que ocupa un ancho de banda mayor al del primero llegando hasta a valores cercanos de 15 KHz.

2.3.3. Imágenes

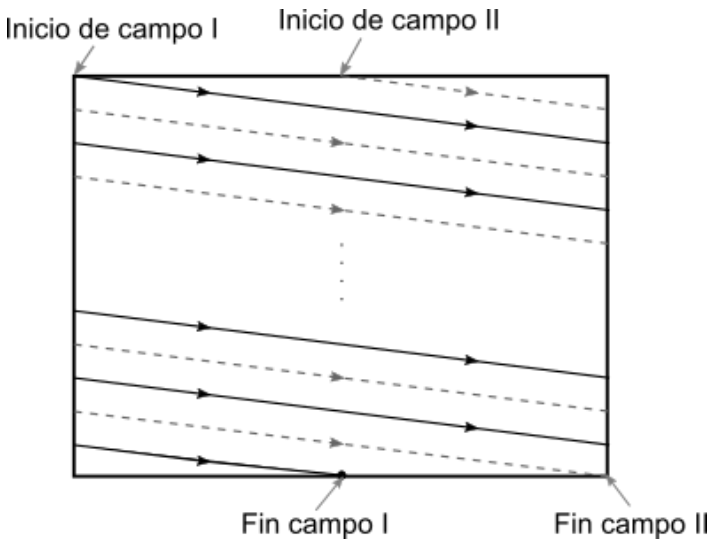
Estas son percibidas por el sistema visual humano, pueden ser estáticas como las fotos que se muestran en los marcos digitales, vallas publicitarias o dinámicas como los vídeos.

Tomando como ejemplo la transmisión analógica de imágenes en televisión, estas se transforman en señales eléctricas lo que permite que sean transportadas fácilmente entre transmisor y receptor, para realizar dicho proceso las cámaras de televisión escanean las imágenes en forma secuencial, utilizando una óptica especial para proyectar la imagen sobre un fotocátodo negro, al realizar esto se crea un patrón de carga que luego es escaneado por un haz de electrones generando así una corriente de salida que varía dependiendo del cambio de brillantez de la imagen de un punto a otro.

A esta corriente resultante se le denomina señal de video, cada una de las imágenes que conforman dicha señal se les denomina tramas, una trama dependiendo del sistema que se implemente puede contar con diferentes características, en el sistema utilizado en Estados Unidos una trama posee 525 líneas divididas en dos campos, cada uno consta de 262,5 líneas.

El proceso de escaneo consiste en que el primer campo se mueve de izquierda a derecha, cuando se alcanza el límite de la línea en cuestión el punto de escaneo regresa al principio de la siguiente línea, a esto se le llama trazado horizontal, este proceso se repite hasta finalizar el recorrido por el primer campo, cuando se finaliza este proceso el punto de escaneo se mueve de forma vertical al inicio del segundo campo y su comportamiento es el mismo que el del primer campo.

Figura 8. **Ejemplo del proceso de escaneo de trama**



Fuente: elaboración propia, empleando Inkscape 1.0.

El tiempo que le toma a cada campo ser recorrido por el punto de escaneo es de 1/60 s, lo que indica que para una escanear una imagen completa se necesitan 1/30 s, esto da como resultado una frecuencia de exploración de 15,75 KHz además, indica que en un segundo se transmiten 30 imágenes que, debido al fenómeno conocido como persistencia de la visión, el ojo humano lo puede percibir como imágenes en movimiento.

En los eventos de trazado horizontal y vertical ocurren eventos generados por el transmisor en los que el receptor se hace inoperativo, a estos se les denomina pulsos de anulación o de blanqueo en los cuales ocurren diversas acciones de sincronización entre el emisor y receptor mediante el envío de pulsos especiales. La calidad de la imagen de TV está delimitada por dos factores básicos:

- EL número de líneas disponibles en una exploración de trama, esto limita la resolución de la imagen en sentido vertical.

- El ancho de banda del canal disponible para transmitir la señal de vídeo, esto limita la resolución de la imagen en sentido horizontal.

En cuanto al color los seres humanos poseen tres receptores de color en el ojo llamados conos, los cuales son sensibles a longitudes de onda de 570nm para el color rojo, 535 nm para el color verde y 445 nm para el color azul, a estos colores se les denomina colores primarios debido a que se puede generar una representación de cualquier color en la naturaleza mediante una adición de estos. En las transmisiones de TV se utilizan estos colores se representan con las señales de vídeo $m_{R(t)}$, $m_{G(t)}$ y $m_{B(t)}$, para mantener un ancho de banda convencional y producir una imagen que pueda ser representada en una televisión monocromática se observa que las señales de color se pueden representar por medio de tres señales que son combinaciones lineales independientes de estas, las señales generadas son:

- La señal de luminancia $m_{L(t)}$, produce una versión en blanco y negro de la imagen en color cuando esta se recibe en un receptor monocromático.

- Las señales de crominancia denominadas $m_{I(t)}$ y $m_{Q(t)}$, estas indican la forma en la que el color de la imagen se desvía de las sombras de grises.

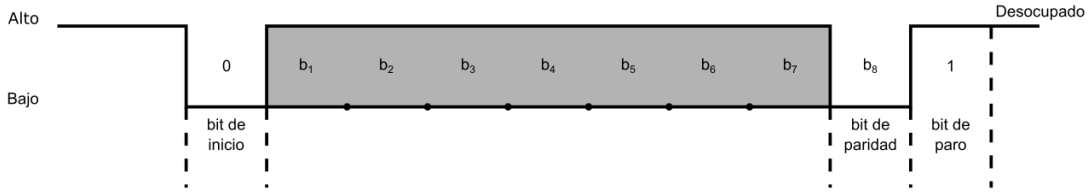
A la señal de luminancia se le asigna según el estándar NTSC un ancho de banda de 4,2 MHz y debido a ciertas propiedades en la visión humana a las señales de crominancia se les asigna un ancho de banda de 1,6 MHz y 0,6 MHz respectivamente.

2.3.4. Datos

Las computadoras han sido parte importante en la vida diaria de los seres humanos se utilizan para diversas tareas tales como envío de correos electrónicos, intercambio de *software* o recursos entre dispositivos, el texto que se transmite entre computadoras en un inicio se codificaba utilizando el estándar ASCII en donde cada carácter se representa por medio de siete *bits* de datos que conforman un patrón único de ceros y unos, de esta manera se pueden representar hasta 128 caracteres diferentes con el estándar ASCII.

Para realizar la transferencia de información entre computadoras se puede tomar como ejemplo el estándar RS-232, en el cual los *bits* de datos se ordenan empezando por el *bit* más significativo hasta el menos significativo, al final de los *bits* de datos se agrega un *bit* adicional que se utiliza para detectar errores, este *bit* se denomina *bit* de paridad y su valor se determina para que la cantidad de unos sea par o impar dependiendo de la paridad seleccionada, al conjunto de 8 *bits* se le denomina *byte* u octeto.

Figura 9. **Ejemplo del formato utilizado por el estándar RS-232 para el envío de datos asíncronos**



Fuente: elaboración propia, empleando Inkscape 1.0.

Los datos que generan las computadoras son un ejemplo de señal de banda ancha, debido a que su contenido ocupa una gama amplia de frecuencias. Además del intercambio de información en forma de texto se pueden descargar formas comprimidas de datos de audio, video o texto de un proveedor remoto, la compresión de datos es una forma eficiente de almacenar y transferir datos entre dispositivos.

Un sistema de compresión de datos se compone de dos partes llamadas codificador y decodificador, existen dos formas de compresión de datos:

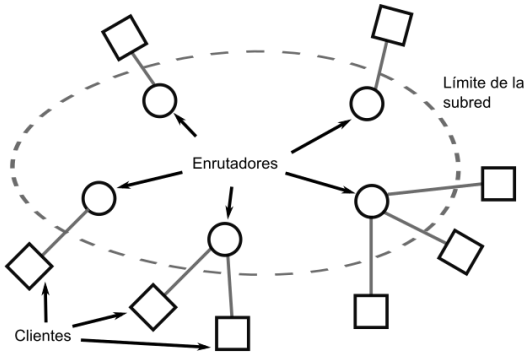
- La compresión sin pérdida elimina la información redundante y es completamente reversible, lo que permite recuperar el estado original de los datos antes de ser sometidos al algoritmo de compresión.

- La compresión con pérdida implica la pérdida de información de forma controlada, no es posible revertirla totalmente, pero a cambio permite tazas de compresión mucho mayores a las que ofrecen los métodos sin pérdidas.

2.4. Redes de comunicación

Una red de comunicación, comúnmente denominada solo red es el conjunto de interconexiones de diversos dispositivos en algunos casos con ayuda de dispositivos intermedios como los enrutadores, a cada cliente de una red se le denomina *host*.

Figura 10. **Ejemplo de una red de comunicación**



Fuente: elaboración propia, empleando Inkscape 1.0.

El diseño de una red de datos se simplifica al manejar el concepto de arquitectura de capas, las cuales indican que los procesos o dispositivos dentro de un sistema están diseñados para realizar una función específica. A nivel de un usuario las capas pueden ser vistas como cajas negras que poseen entradas y salidas, las cuales al estar conectadas entre sí y tener interfaces bien definidas tomando en cuenta las capacidades de sus capas vecinas son capaces de realizar el complejo proceso de comunicación, un ejemplo de esto es el sistema de referencia de interconexión de sistemas abiertos o modelo OSI desarrollado por la International Organization for Standardization.

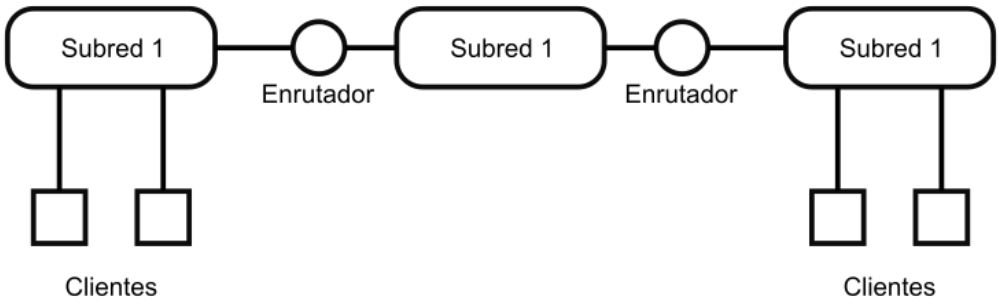
El modelo OSI consta de siete capas, tomando como ejemplo la existencia de dos sistemas conectados entre sí, la capa k del modelo A se comunica con la capa k del modelo B, de conformidad con un grupo de reglas y convenciones que componen el protocolo k de la capa, la comunicación se consigue al contar con procesos iguales en ambos sistemas, en este modelo solo existe comunicación física entre la capa 1, para las demás capas se utiliza la comunicación virtual con sus equivalentes, no obstante cada una de las capas restantes puede

intercambiar datos e información con sus vecinas superior e inferior utilizando interfaces de capa a capa.

2.4.1. Internet

La arquitectura de Internet cuenta con tres elementos utilizables, los clientes o hosts, subredes y enrutadores. En los clientes es donde se generan y reciben los datos, los enrutadores son los encargados de servir entre las fronteras de las subredes para direccionar el tráfico de estas, dentro de una subred los clientes o computadoras intercambian datos de forma directa.

Figura 11. **Ejemplo de una red interconectada de subredes**



Fuente: elaboración propia, empleando Inkscape 1.0.

Internet también utiliza protocolos y capas, estos están definidos por el protocolo de Internet (IP), este es un protocolo universal que ocupa la capa de red, el cual define un plan de direccionar datos en forma de paquetes de un nodo a otro, esto da como resultado que Internet ofrezca el servicio del mayor esfuerzo en donde la entrega de cada paquete de datos se garantiza, pero no el tiempo de entrega o si el receptor es el que ha solicitado la información.

2.5. Canales de comunicación

El envío de información en una red de comunicación se realiza utilizando la capa física a través de un canal de comunicación, dependiendo del modo de transmisión que se desee utilizar se pueden agrupar en dos tipos de canales de información: los basados en propagación guiada como los canales telefónicos, fibras ópticas y cables coaxiales o los basados en propagación libre como lo son las ondas de TV y radio o la comunicación por satélite.

2.5.1. Canales de transmisión inalámbrica

La señal que se encarga de trasladar la información que representan la información que se desee enviar debe ser modulada sobre una frecuencia portadora que a su vez identifica al transmisor. La transmisión tiene su origen entre el acoplamiento de la señal modulada y las ondas electromagnéticas en el espacio libre al utilizar una antena cuyo objetivo es estimular dichas ondas en la dirección o direcciones requeridas teniendo en cuenta que su funcionamiento debe ser del modo más eficiente que se pueda obtener.

Por norma general, las antenas se sitúan en lugares elevados y sin obstáculos entre el emisor y receptor, debido al fenómeno de la difracción es posible transmitir ondas por medio inalámbricos más allá de la línea de vista, aunque esto conlleva a un poco más de pérdidas.

En el extremo del receptor se emplea una antena para obtener las ondas irradiadas por el emisor, con esto se establece el enlace de comunicación, la mayoría de los dispositivos receptores son de categoría superheterodino, estos dispositivos funcionan convirtiendo la señal recibida dentro de cierta banda de frecuencias denominada banda de frecuencias intermedia y obtienen de ahí la

señal deseada implementando algún tipo de detector configurado para dicha señal.

La capacidad de la red puede ser ampliada al utilizar un canal de radio móvil, esto abarca escenarios en los cuales el transmisor o receptor se encuentran en movimiento o no, es utilizada en lugares poblados donde normalmente no es posible tener una trayectoria de línea de vista entre emisor y receptor, en cambio la propagación de la señal se realiza utilizando la difracción en las construcciones o alrededor de estas y la dispersión en las superficies.

Esto da como resultado que múltiples señales puedan ser captadas por el receptor con diferente tiempo de retardo e intensidad, a esto se le denomina fenómeno de trayectorias múltiples, un canal de radio móvil se considera como uno lineal variante con el tiempo que es de naturaleza estadística.

Otros canales de comunicación inalámbricos utilizan los canales satelitales ofrecidos por satélites geosíncronos, estos satélites se encuentran inmóviles sobre un punto de la superficie terrestre, entre los beneficios están: amplia cobertura alrededor del 40 % de la superficie terrestre, enlaces de transmisión son confiables, cuentan con anchos de banda amplios, el equipo de recepción y transmisión es relativamente barato, poseen baja atenuación debido a la lluvia y el ruido de fondo es casi insignificante.

2.6. Modulación de la información

EL objetivo de un sistema de comunicación es el intercambio de información entre el emisor y el receptor de forma en que ambos elementos pueda descifrarla de forma satisfactoria. Para lograr su objetivo el transmisor debe adaptar la información para que sea apta para su transmisión por el canal seleccionado,

este proceso se llama modulación y se realiza modificando alguna de las características de la onda portadora en relación con la señal del mensaje.

El receptor debe ser capaz de recrear la señal original a partir de la versión modificada del emisor que ya ha viajado por el canal de comunicación, a este proceso se le llama demodulación, uno de los problemas más comunes es la degradación de la señal al ser enviada por el canal de comunicación, esto dependerá del esquema de modulación, algunos esquemas de modulación se ven menos afectados por el ruido y la distorsión que otros.

Los procesos de modulación se pueden clasificar en:

- Modulación de onda continua que utiliza una onda senoidal como portadora en donde en función de la señal del mensaje se puede variar: la amplitud (AM), la frecuencia (FM) y la fase (PM).

- Modulación por pulsos, esta consiste en una serie de pulsos rectangulares que puede ser generados de forma análoga o digital, en la modulación por pulsos análogos utiliza la señal del mensaje para manipular la amplitud del pulso (PAM), la duración del pulso (PDM) y la posición del pulso (PPM), la forma de modulación por pulsos en forma digital se conoce como modulación por codificación de pulsos o PCM.

La PCM maneja la señal del mensaje tomando el valor de la amplitud en intervalos regulares, este valor se aproxima al valor más cercano de una colección de datos discretos establecidos previamente llamados niveles de cuantización, que se encuentran representados por una secuencia equivalente en números binarios.

Al realizar el proceso de cuantización siempre perderá parte de la información en la señal del mensaje, esto se puede mitigar al utilizar varios niveles, en algunos casos como lo es la voz o imágenes las pérdidas pueden ser indetectables para los seres humanos.

Otro beneficio de la modulación es que hace posible el uso de la multiplexación, esto consiste en enviar varias señales de mensajes por un solo canal, los métodos de multiplexado utilizados comúnmente son:

- Multiplexado por división de frecuencias (FDM): consiste en utilizar la modulación de onda continua para trasladar la señal del mensaje con el objetivo que esta resida en una ranura de frecuencia específica dentro de la paso banda del canal asignándole una frecuencia de portadora distinta, el receptor utiliza un conjunto de filtros para separar las diferentes señales moduladas y prepararlas para su demodulación, debido a que las señales se traslapan entre sí en la entrada es posible que el sistema sufra de diafonía.

- Multiplexado por división de tiempo (TDM): se emplea para colocar muestras de señales de mensajes diferentes en ranuras de tiempo que no se traslapen, en este sistema se utiliza todo el ancho de banda del canal.

- Multiplexado por división de código (CDM): cada señal del mensaje se identifica mediante un código distintivo, permite que las señales se puedan traslapar entre sí ya sea en tiempo como en frecuencia en el canal.

- Multiplexado por división de longitud de onda (WDM): se utiliza en la comunicación por fibra óptica utiliza diferentes longitudes de onda para

transportar distintas señales por un mismo medio, es por decirlo así un recíproco del FDM para comunicaciones ópticas.

2.7. Comunicación análoga y digital

Al diseñar un sistema de comunicación suelen estar definidos el canal de comunicación, la fuente de información y el receptor, debido a estas especificaciones se debe diseñar el transmisor y receptor teniendo en cuenta las siguientes pautas:

- Codificar/modular la señal del mensaje que se desea transmitir, enviar esa señal a través del canal seleccionado y reproducir la estimación de la señal original en el receptor de forma que complazca los requerimientos del cliente.

- Efectuar el proceso de comunicación a un costo que sea viable y sostenible para el cliente.

Se puede implementar el sistema de comunicación de forma análoga o digital, el primero es simple en términos conceptuales pero difícil de construir debido a los exigentes requerimientos de linealidad y ajustes del sistema, la simplicidad conceptual es resultado de que las técnicas utilizadas al modular de forma analógica realizan cambios relativamente superficiales a la señal del mensaje.

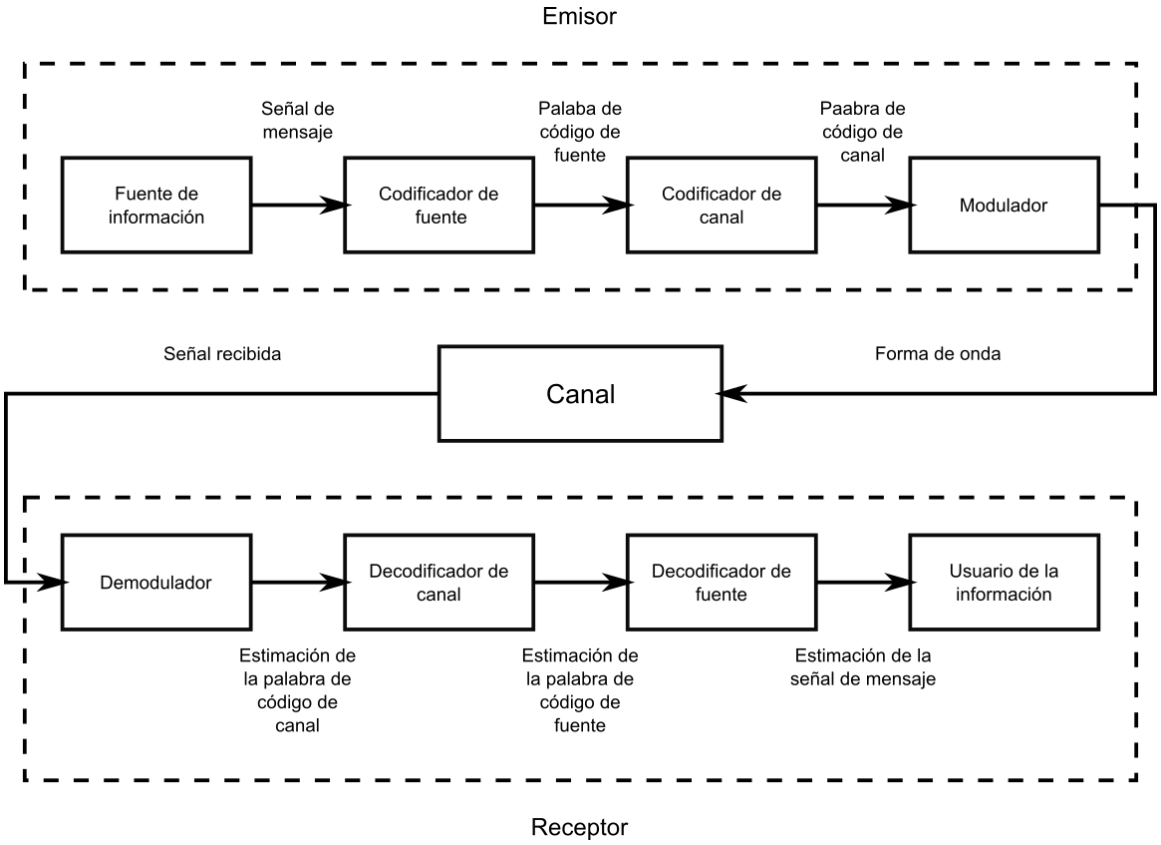
Al implementar el sistema de comunicación de forma digital se intenta definir una serie de valores finitos para representar una señal de mensaje, que en consecuencia correspondan a las especificaciones del canal para hacer más tolerante el mensaje a los deterioros de este último, esto se puede representar

en una relación codificador-decodificador de fuente, codificador-decodificador de canal y un proceso de modulación-demodulación.

El codificador se encarga de eliminar la información no esencial para la comprensión del mensaje, esto da como resultado un flujo de datos llamado palabra de código fuente que el codificador del canal se encarga de procesar dando como resultado una palabra de código de canal, esta es más grande que la primera debido a la redundancia que se utiliza en el proceso de creación.

Durante el proceso de modulación cada símbolo de la palabra de código de canal se representa por el equivalente análogo del conjunto finito de valores utilizados, al conjunto de estos valores se le denomina forma de señal la cual es adecuada para su transmisión por el canal, en el receptor se realiza el mismo proceso solo que en sentido inverso.

Figura 12. **Diagrama de bloques de un sistema de comunicación digital**



Fuente: elaboración propia, empleando Inkscape 1.0.

3. DISPOSITIVO NRF24L01+

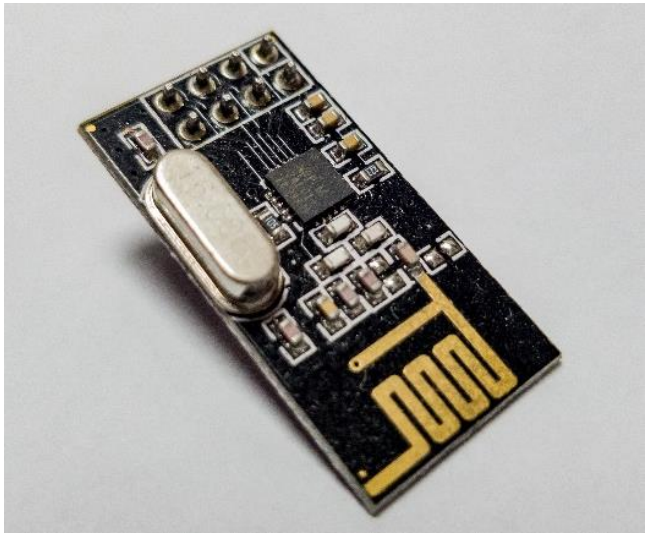
3.1. Descripción del dispositivo NRF24L01+

El dispositivo nRF24L01+ es un transceptor contenido en un solo chip que utiliza la banda de frecuencia ISM mundial entre 2,4000 – 2,4835 GHz, además, utiliza el protocolo de banda base denominado Enhaced ShockBurst, el cual está diseñado para aplicaciones inalámbricas de ultra bajo consumo, para diseñar un sistema de radio con el dispositivo nRF24L01+, se necesita un microcontrolador y unos pocos componentes pasivos externos.

El protocolo de banda base Enhaced ShockBurst basa su funcionamiento en la comunicación por paquetes, admitiendo varios que pueden variar desde el funcionamiento manual hasta un funcionamiento autónomo avanzado. Las estructuras de datos utilizadas por el protocolo aseguran que el flujo de datos ocurre de forma clara entre el transceptor y el microcontrolador, permitiendo así que el impacto de rendimiento en el sistema sea reducido.

El componente encargado de las transmisiones de radio del dispositivo nRF24L01+ utiliza modulación GFSK, dentro de los parámetros que pueden ser configurados por el usuario se encuentra el canal seleccionado para la transmisión y recepción de datos, la potencia de salida en la antena y la velocidad de transmisión de datos cuyos valores pueden ser 250 kbps, 1 Mbps y 2 Mbps. La alta transferencia de datos junto con los dos modos de ahorro de energía hace que este dispositivo sea adecuado para aplicaciones de ultra baja energía.

Figura 13. **Dispositivo nRF24L01+**



Fuente: elaboración propia, zona 18, Guatemala.

3.2. Características del dispositivo nRF24L01+

- Interfaz de radio:
    - Opera en la banda internacional ISM de 2,4 GHz.
    - Puede seleccionar uno de los 126 canales disponibles para realizar el proceso de comunicación.
    - Puede funcionar como emisor y receptor.
    - Utiliza la modulación GFSK.

- La velocidad de transmisión por aire puede variar entre 250 kbps y 2 Mbps.

- La separación entre los canales de comunicación es de 1 MHz cuando se utiliza una velocidad de transmisión de datos de 1 Mbps o menor y el espacio entre canales de comunicación es de 2 MHz en caso de transmitir datos a 2 Mbps.

- Transmisor:

  - Potencia de salida programable entre 0, -6, -12 o -18 dBm.

  - Consumo de corriente eléctrica de 11,3 mA cuando se transmite a una potencia de 0 dBm.

- Receptor:

  - Control automático de ganancia.

  - Posee filtros integrados para los canales.

  - Su consumo de corriente eléctrica es de 13,5 mA cuando se reciben datos a 2 Mbps.

  - Sensibilidad de -82 dBm a una velocidad de recepción de 2 Mbps.

  - Sensibilidad de -85 dBm a una velocidad de recepción de 1 Mbps.

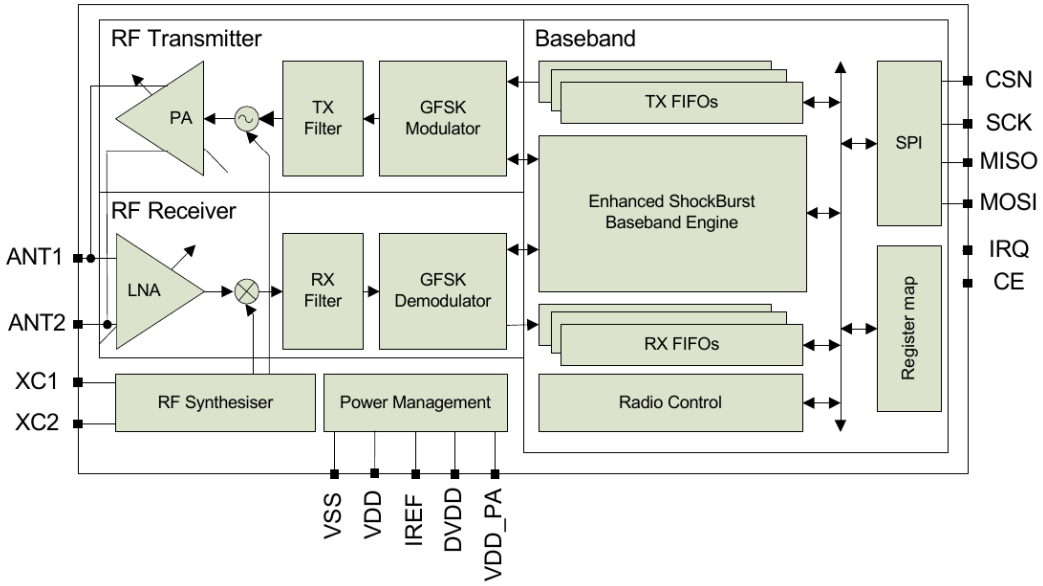
  - Sensibilidad de -94 dBm a una velocidad de recepción de 250 Kbps.

- Gestión de energía:
    - Posee un regulador de voltaje integrado.
    - El voltaje de alimentación del dispositivo se encuentra entre 1,9 y 3,6 V.
    - Posee estado de espera con tiempos de arranque rápidos.
    - Consume solo 26 µA en el modo de suspensión y 900 nA en el modo de baja potencia.
    - 1,5 ms como máximo de tiempo de restablecimiento desde el modo de baja potencia.
    - 130 µs como máximo de tiempo de restablecimientos desde el modo de suspensión.
- Interfaz de comunicación
    - Comunicación por medio de interfaz SPI de 4 pines.
    - Máxima velocidad de comunicación 10 Mbps.
    - Los pines de propósito general soportan hasta 5V.

3.3. Diagrama de bloques del dispositivo

La estructura interna del dispositivo nRF24L01+ se encuentra representada por el diagrama de bloques en la figura 14.

Figura 14. **Diagrama de bloques del dispositivo nRF24L01+**



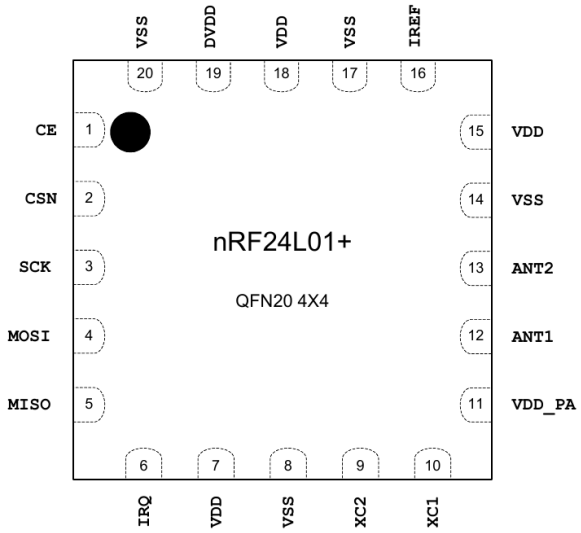
Fuente: Nordic Semiconductor. *nRF24L01+ Single Chip 2.4GHz Transceiver Preliminary Product Specification v1.0*. http://www.sparkfun.com/datasheets/Components/SMD/ nRF24L01Pluss_Preliminary_Product_Specification_v1_0.pdf. Consulta: 18 agosto 2020.

3.4. Información de los pines del dispositivo

El dispositivo nRF24L01+ utiliza el empaquetado QFN20 4x4, entre los beneficios de este empaquetado se puede decir que los *pads* de cobre que se utilizan en el circuito impreso ofrecen un buen rendimiento térmico y eléctrico,

además de ser una de las mejores opciones para dispositivos donde el tamaño, peso y el rendimiento térmico son importantes.

Figura 15. **Asignación de los pines con el empaquetado QFN20 4x4**



Fuente: Nordic Semiconductor. *nRF24L01+ Single Chip 2.4GHz Transceiver Preliminary Product Specification v1.0*. http://www.sparkfun.com/datasheets/Components/SMD/nRF24L01Pluss_Preliminary_Product_Specification_v1_0.pdf. Consulta: 18 agosto 2020.

Tabla IV. **Funciones de los pines del dispositivo**

Pin	Nombre	Función	Descripción
1	CE	Entrada digital	Activa o desactiva el modo RX o TX
2	CSN	Entrada digital	SPI selector de chip
3	SCK	Entrada digital	SPI reloj
4	MOSI	Entrada digital	SPI ingreso de datos

Continuación de la tabla IV.

Pin	Nombre	Función	Descripción
5	MISO	Salida digital	SPI salida de datos, con opción triestado
6	IRQ	Salida digital	Pin de interrupción enmascarable. Activo en estado bajo
7	VDD	Alimentación	Fuente de alimentación (+1,9V - +3,6V DC)
8	VSS	Alimentación	Referencia (0V)
9	XC2	Salida analógica	Oscilador Pin 2
10	XC1	Entrada analógica	Oscilador Pin 1
11	VDD_PA	Fuente de alimentación	Fuente de alimentación (+1,8V) para el amplificador de potencia interno del nRF24L01+. Debe estar conectado a ANT1 y ANT2.
12	ANT1	Radiofrecuencia	Interfaz de antena 1
13	ANT2	Radiofrecuencia	Interfaz de antena 2
14	VSS	Alimentación	Referencia (0V)
15	VDD	Alimentación	Fuente de alimentación (+1,9V - +3,6V DC)
16	IREF	Entrada analógica	Corriente de referencia. Conectar a través de un resistor con valor 22kΩ a referencia 0V.
17	VSS	Alimentación	Referencia (0V)
18	VDD	Alimentación	Fuente de alimentación (+1,9V - +3,6V DC)
19	DVDD	Fuente de alimentación	Salida de alimentación interna para propósitos de desacoplamiento.
20	VSS	Alimentación	Referencia (0V)

Fuente: elaboración propia, empleando Microsoft Word.

Tabla V. **Valores máximos y mínimos de operación del dispositivo**

Condiciones de operación	Mínima	Máxima	Unidades
Voltaje de alimentación			
VDD	-0,3	3,6	V
VSS		0	V
Voltaje de entrada			
VI	-0,3	5,25	V
Voltaje de salida			

Continuación de la tabla V.

Condiciones de operación	Mínima	Máxima	Unidades
VO	Entre VSS y VDD	Entre VSS y VDD	V
Disipación total de potencia			
PD (TA=85°C)		60	mW
Temperatura operativa	-40	+85	°C
Temperatura de almacenamiento	-40	+85 +125	°C

Fuente: elaboración propia, empleando Microsoft Word.

En el dispositivo nRF24L01+ el voltaje que necesita ser suministrado para operar debe ser seleccionado dependiendo de si el valor de voltaje de las señales de entrada provenientes del microcontrolador es mayor a 3.6 V o no, como se puede observar en la tabla VI.

Tabla VI. **Condiciones de operación del dispositivo nRF24L01+**

Valor	Parámetro	Mínimo	Típico	Máximo	Unidades
VDD	Fuente de alimentación	1,9	3,0	3,6	V
VDD	Fuente de alimentación si la señal de entrada es mayor a 3,6V	2,7	3,0	3,3	V
TEMP	Temperatura ambiente	-40	+27	+85	°C

Fuente: elaboración propia, empleando Microsoft Word.

3.5. Modos de operación del dispositivo nRF24L01+

El dispositivo nRF24L01+ posee una máquina de estados que se encarga de la transición entre los distintos modos de operación, toma los valores de entrada del registro definido por el usuario y los valores de las señales internas

que tenga. Los modos de operación disponibles son modo de baja potencia, modo de suspensión, modo de transmisión o recepción.

3.5.1. Diagrama de estados del dispositivo

Existen tres tipos de estados incluidos en este diagrama:

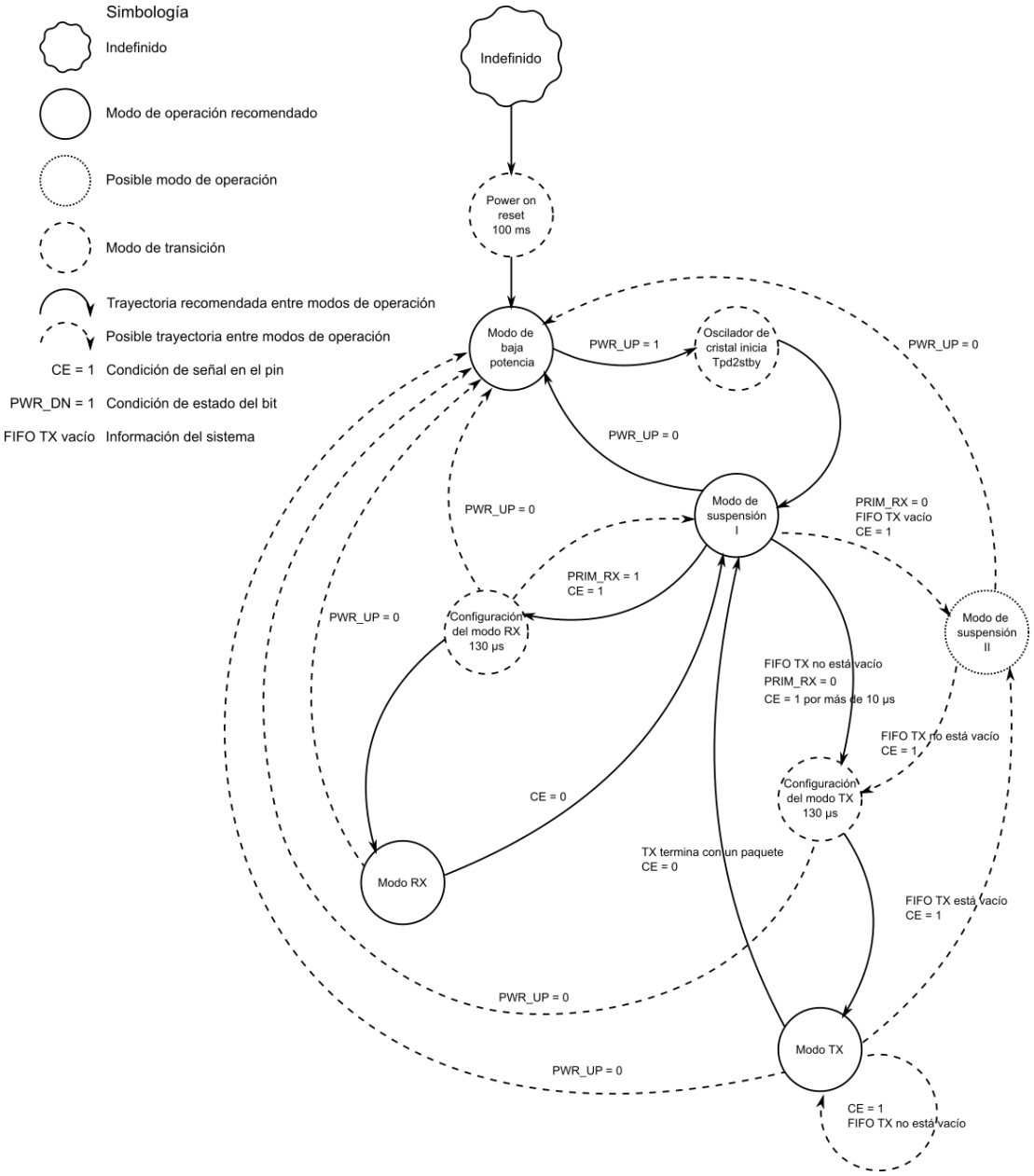
- Modo de operación recomendado: es el estado recomendado cuando el equipo opera de forma convencional.

- Posible modo de operación: es un posible estado, pero no es utilizado si el equipo opera de forma convencional.

- Estado de transición: es un estado cuya duración es corta, usualmente se encuentra entre el encendido del dispositivo y la configuración del PLL.

Cuando VCC alcanza un valor de 1,9 V o superior el dispositivo nRF24L01+ ingresa al estado de *Power on reset* el cual hace que el dispositivo inicie su funcionamiento en un estado conocido, donde permanece hasta que entra al estado de baja potencia.

Figura 16. **Diagrama de estado del control de radio del dispositivo nRF24L01+**



Fuente: elaboración propia, empleando Inkscape 1.0.

3.5.2. Modo de baja potencia

En este modo el dispositivo nRF24L01+ se encuentra desactivado, lo que hace que su consumo energético sea mínimo, los valores de registros establecidos previamente se mantienen, la interfaz SPI se mantiene activa lo que permite que la configuración del dispositivo a través del cambio de registros pueda ser modificada según se requiera. El dispositivo ingresa en este estado al establecer el *bit* PWR_UP en el registro CONFIG a un nivel bajo.

3.5.3. Modo de suspensión

En este modo de operación los valores de los registros se mantienen y la interfaz SPI puede ser activada en cualquiera de las versiones disponibles.

3.5.3.1. Modo de suspensión I

Este modo es utilizado para minimizar el consumo de corriente eléctrica del dispositivo mientras se mantiene un tiempo de inicio desde la inactividad corto, en este modo de operación solo la parte correspondiente al oscilador de cristal está activa. El dispositivo ingresa en este estado al establecer el *bit* PWR_UP en el registro CONFIG a un nivel alto.

Para cambiar entre modos activos se necesita que en el pin correspondiente a CE se encuentre a un nivel alto y este cambie a un nivel bajo, el dispositivo nRF24L01+ regresa a este estado luego de terminar las operaciones correspondientes al modo de transmisión y recepción.

3.5.3.2. Modo de suspensión II

En este modo de operación se incluyen *buffers* extras de reloj a la configuración del modo de suspensión I, comparado con este su consumo de corriente eléctrica es mayor. El dispositivo nRF24L01+ entra en este modo cuando el pin correspondiente a CE se encuentra a un nivel alto y el FIFO TX se encuentra vacío, si un nuevo paquete es ingresado al FIFO TX, se iniciará inmediatamente el PLL y el paquete será transmitido luego del retraso normal de operación del PLL (130 µs).

3.5.4. Modo de recepción

Es un modo activo en el cual el sistema de radio del dispositivo nRF24L01+ es utilizado como receptor, para ingresar a este modo los *bits* PWR_UP, PRIM_RX y el pin correspondiente a CE deben estar a nivel alto.

En el modo de recepción el receptor demodula las señales del canal de radiofrecuencia, presentando constantemente los datos demodulados al protocolo de banda base. El motor del protocolo de banda base busca constantemente un paquete válido, en caso de encontrarlo la información contenida es almacenada temporalmente en los FIFOS de RX, si estos se encuentran ocupados el paquete recibido se descarta.

El dispositivo nRF24L01+ permanece en modo de recepción hasta que el microcontrolador cambie su modo de operación a suspensión I o de baja potencia. Sin embargo, si se activan las funciones de protocolo automático Enhaced ShockBurst en el motor del protocolo de banda base, el dispositivo nRF24L01+ puede ingresar a otros modos para completar las operaciones del protocolo mencionado.

En el modo de recepción se encuentra un detector de potencia recibida que envía una señal en alto cuando se detecta una señal de radiofrecuencia con potencia superior a -64 dBm dentro del canal de recepción. La señal de radiofrecuencia debe estar presente durante al menos 40 µs antes que el detector de potencia recibida envíe la señal en alto.

3.5.5. Modo de transmisión

Es un modo activo para la transmisión de paquetes. Para ingresar en este modo, en el dispositivo nRF24L01+ se debe configurar el *bit* PWR_UP a nivel alto, el *bit* PRIM_RX a nivel bajo, deben existir datos en el FIFO TX para transmitir y el pin correspondiente a CE debe estar a nivel alto por más de 10 µs.

El dispositivo nRF24L01+ se mantiene en el modo de transmisión hasta que termine la transmisión del paquete, si el pin correspondiente a CE se encuentra a nivel bajo, el dispositivo regresará al estado de suspensión I, si el pin correspondiente a CE se encuentra a nivel alto, el estatus del FIFO TX determinará la siguiente acción.

En caso de que el FIFO TX no se encuentre vacío el dispositivo se mantendrá en el modo de transmisión y transmitirá el siguiente paquete, en caso de que el FIFO TX se encuentre vacío el dispositivo entrará al modo se suspensión II. Es importante nunca mantener el dispositivo en modo de transmisión por más de 4 ms ininterrumpidos, si se activa el protocolo Enhanced ShockBurst esto nunca ocurrirá.

3.5.6. Configuración de los modos de operación del dispositivo nRF24L01+

En la tabla VII se muestran los requisitos que deben cumplirse para que el dispositivo opere en el modo deseado.

Tabla VII. **Modos de operación disponibles para el dispositivo nRF24L01+**

Modo	Registro PWR_UP	Registro PRIM_RX	Pin asignado a CE	Estado del FIFO
Modo de recepción	1	1	1	-
Modo de transmisión	1	0	1	Con datos, se vaciarán todos los niveles del FIFO TX
Modo de transmisión	1	0	Pulso en alto por al menos 10µs	Con datos, se vaciará solo el primer nivel del FIFO TX
Modo de suspensión II	1	0	1	Sin datos FIFO TX vacío
Modo de suspensión I	1	-	0	No hay transmisión de paquetes en curso
Modo de baja potencia	0	-	-	-

Fuente: elaboración propia, empleando Microsoft Word.

3.5.7. Información de sincronización

La información de la tabla VIII se relaciona con el tiempo requerido por el dispositivo nRF24L01+ para realizar las transiciones entre modos y el tiempo que se requiere que el pin asignado a CE tenga un nivel determinado. La transición

entre el modo de transmisión y recepción tiene una duración igual que la transición del modo de suspensión al modo de transmisión o al modo de recepción un máximo de 130 µs.

Tabla VIII. **Información de sincronización en el cambio de modos de operación del dispositivo nRF24L01+**

Name	nRF24L01+	Max.	Min.	Comments
Tpd2stby	Transición entre el modo de baja potencia y el modo de suspensión	150µs		Con fuente de reloj externa
		1,5ms		Cristal externo, Ls < 30mH
		3ms		Cristal externo, Ls = 60mH
		4,5ms		Cristal externo, Ls = 90mH
Tstby2a	Transición entre el modo de suspensión y el modo de transmisión o recepción	130µs		
Thce	Tiempo mínimo en el cual el pin asignado a CE debe estar a un nivel alto		10µs	
Tpece2csn	Retraso en el cambio de nivel alto en el pin asignado a CE a nivel bajo en pin asignado a CSN		4µs	

Fuente: elaboración propia, empleando Microsoft Word.

Para que el dispositivo nRF24L01+ cambie del modo de baja potencia al modo de transmisión o recepción debe cambiar primero al modo de suspensión este cambio tiene un retraso de Tpd2stby antes de colocar a nivel alto el pin asignado a CE, para ingresar al modo de operación requerido.

3.6. Velocidad de transferencia de datos

En el dispositivo nRF24L01+ esta puede ser 250 kbps, 1 Mbps o 2 Mbps, el uso de bajas velocidades de transmisión permite que el receptor tenga una mayor sensibilidad a la seña, en cambio una alta velocidad de transmisión de datos permite un consumo menor de corriente eléctrica y reduce la probabilidad de que existan colisiones en la transferencia de los paquetes.

La velocidad de transferencia de datos se configura modificando el *bit* RF_DR del registro RF_SETUP. Para que exista una comunicación exitosa entre transmisor y receptor ambos deben utilizar la misma velocidad de transferencia de datos.

3.7. Frecuencia del canal de transmisión

Determina el centro del canal utilizado por el dispositivo nRf24L01+, el canal ocupa un ancho de banda de al menos 1 MHz con una velocidad de transferencia de datos de 250 kbps o 1 Mbps y de al menos 2 MHz al utilizar una velocidad de transmisión de 2 Mbps. Las frecuencias en la que puede operar el dispositivo abarcan desde los 2,400 GHz hasta los 2,525 GHz.

La resolución de los canales de radiofrecuencia por defecto es de 1 MHz, debido a esto al utilizar velocidades de transmisión de 2 Mbps que ocupan anchos de banda mayores a la resolución de los canales, el espacio entre canales debe ser de 2 MHz o más.

La frecuencia del canal es configurada por medio del registro RF_CH de acuerdo con la siguiente formula:

$$F_o = 2400 + RF_{CH}[MHz]$$

Para que exista una comunicación exitosa entre transmisor y receptor ambos deben utilizar la misma frecuencia de canal.

3.8. Detector de potencia recibida

RPD por sus siglas en inglés, se encuentra localizado en el registro 09 en el *bit* 0, cambia a estado activo cuando la potencia recibida por el dispositivo en el canal que se encuentre actico es mayor a -64 dBm, si la potencia recibida es menor a -64 dBm RDP se mantiene en estado bajo.

El RPD puede ser leído en cualquier momento mientras el dispositivo nRF24L01+ se encuentre en modo de recepción, esto ofrece una forma de monitorizar en tiempo real la potencia recibida en el canal que se esté utilizando.

El estado de RPD se mantiene cuando se recibe un paquete válido, si no se recibe ningún paquete el RPD puede mantener su valor por otras condiciones tales como, al final de un periodo de recepción como resultado de que el microcontrolador establezca en el pin correspondiente a CE un nivel bajo o a un tiempo de espera del modo de recepción controlado por el protocolo Enhaced ShockBurst.

3.9. Control de potencia de transmisión

Se usa para establecer la potencia con la cual el dispositivo nRF24L01+ operará en el modo de transmisión, existen cuatro niveles de potencia programables, estos se pueden seleccionar utilizando los *bits* correspondientes a RF_PWR en el registro RF_SETUP, en la tabla IX se muestran los valores

esperados de potencia de transmisión al igual que el consumo de corriente eléctrica que consume cada potencia.

Tabla IX. **Opciones de configuración de la potencia de transmisión en el dispositivo nRF24L01+**

Valores de los *bits* de RF_PWR en el registro RF_SETUP	**Potencia de transmisión**	**Consumo de corriente eléctrica**
11	0dBm	11,3mA
10	-6dBm	9,0mA
01	-12dBm	7,5mA
00	-18dBm	7,0mA

Fuente: elaboración propia, empleando Microsoft Word.

3.10. Enhaced ShockBurst

Su funcionamiento se basa en el manejo de paquetes de datos, entre sus características se encuentra el ensamble automático de paquetes, la notificación automática de acuses de recibido, la retransmisión de paquetes. Permite su implementación en sistemas de ultra baja potencia, gran desempeño de comunicación, incrementa la eficiencia de transmisión y recepción en sistemas direccionales y bidireccionales sin implementar interfaces de control muy complejas o costosas.

3.10.1. Vista general del funcionamiento

Enhanced ShockBurst utiliza ShockBurst para el manejo de forma automática de tiempos de sincronización y paquetes. Durante el modo de transmisión ShockBurst ensambla el paquete y sincroniza el paquete para su transmisión.

Durante el modo de recepción, ShockBurst se encuentra analizando la señal demodulada en búsqueda de direcciones válidas, cuando encuentra una procesa el resto del paquete y valida su contenido por medio de la verificación de redundancia cíclica o CRC por sus siglas en inglés, si esto resulta exitoso la carga útil del paquete es almacenada en uno de los FIFOs RX, las operaciones correspondientes al manejo de *bits* y tiempos de sincronización son controlados por ShockBurst.

Enhanced ShockBurst cuenta con manejo automático de transacción de paquetes para la fácil implementación de un enlace bidireccional de comunicación. Una transacción de Enhanced ShockBurst se refiere al intercambio de paquetes entre dos transceptores, uno actúa como receptor primario (PRX) y el otro como transmisor primario (PTX).

Las transacciones son iniciadas siempre por el PTX, la transacción se completa cuando el PTX recibe el paquete de acuse de recibido (ACK) por parte del PRX. El PRX puede adjuntar datos adicionales al ACK estableciendo un enlace de comunicación bidireccional.

El manejo automático de paquetes ocurre de la siguiente forma:

- Se inicia la transmisión del paquete del PTX al PRX, Enhanced ShockBurst configura el PTX en modo de recepción en espera del ACK.

- Si el paquete es recibido por el PRX, Enhanced ShockBurst ensambla y transmite de forma automática el paquete ACK al PTX antes de regresar al modo de recepción.

- Si PTX no recibe el paquete ACK inmediatamente, Enhanced ShockBurst automáticamente retransmitirá el paquete original luego de un retardo que puede ser de duración programable y configura el PTX en modo de recepción en espera del paquete ACK.

Al utilizar Enhanced ShockBurst es posible configurar algunos parámetros tales como el número máximo de retransmisiones que se realizan y el retardo entre la transmisión actual y la siguiente. Todos estos procesos son manejados por el dispositivo nRF24L01+ sin involucrar al microcontrolador conectado por medio de la interfaz SPI.

3.10.2. Características

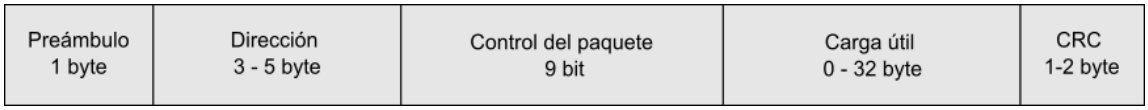
Las principales características de Enhanced ShockBurst son:

- Carga útil de tamaño dinámico con valores entre 1 y 32 *bytes*.
- Manejo automático de paquetes.
- Manejo automático de transacciones de paquetes.
    - Envío automático de acuse de recibido cuando se envían cargas útiles.
    - Retransmisión automática de paquetes en caso sea necesario.
- *MultiCeiver* de hasta 6 clientes, para formar redes en estrella bidireccionales de 1 a 6.

3.10.3. Formato del paquete en Enhanced ShockBurst

Consiste en un preámbulo, una dirección, un control del paquete, una carga útil y un campo CRC.

Figura 17. **Estructura del paquete utilizado por Enhanced ShockBurst**



Fuente: elaboración propia, empleando Inkscape 1.0.

3.10.3.1. Preámbulo del paquete

Es una secuencia de *bits* que se utiliza para sincronizar el demodulador del receptor con la señal entrante. Su tamaño es de un *byte* y puede ser 01010101 o 10101010. Su valor dependerá si el primer *bit* en el segmento de dirección, en caso de ser 1, el preámbulo será 10101010 y si el primer *bit* es 0 el preámbulo será 01010101. Esto se realiza con el fin de garantizar que existen la cantidad suficiente de transiciones en el preámbulo para estabilizar el receptor.

3.10.3.2. Dirección del paquete

Se refiere a la dirección asignada al receptor, con esto se asegura que el paquete es detectado y recibido por el receptor deseado evitando que existan comunicaciones no deseadas entre distintos sistemas que utilicen dispositivos nRF24L01+, puede ser configurada en el registro AW puede tener una longitud de 3, 4 o 5 *bytes.*

3.10.3.3. Campo de control del paquete

Contiene una sección de 6 *bits* que corresponden a la longitud de la carga útil del paquete, 2 *bits* corresponden al identificador del paquete PID por sus siglas en inglés y NO_ACK es una sección de 1 *bit* que corresponde a la opción de enviar o no un acuse de recibido en cada transferencia exitosa.

Figura 18. **Estructura del control del paquete utilizado por Enhanced ShockBurst**

Fuente: elaboración propia, empleando Inkscape 1.0.

3.10.3.3.1. Longitud de la carga útil

Este campo de 6 *bits* especifica la longitud de la carga útil en el paquete, esta puede ser desde 0 hasta 32 *bytes* de información.

- Si se utiliza un valor 000000 = 0 *byte,* indica que es un paquete vació y solo se utiliza en paquetes vacíos de ACK.

- Para indicar que la carga útil utiliza el tamaño máximo posible se utiliza un valor 100000 = 32 *byte*.

- Si se utiliza la opción de longitud dinámica de carga útil se puede asignar el valor 100001 = no importa.

3.10.3.3.2. Identificador del paquete

Este campo de 2 *bits* se utiliza para saber si el paquete recibido es un paquete nuevo o uno que ha sido retransmitido. El PID impide que el dispositivo envíe más de una vez la información del paquete al microcontrolador que se encarga de operar el dispositivo. El valor del PID se incrementa en el lado del transmisor con cada nuevo conjunto de datos que recibe a través de la interfaz SPI de comunicación.

Los campos correspondientes al PID y el CRC son utilizados por el dispositivo PRX para determinar si el paquete recibido es uno nuevo o uno retransmitido, cuando se pierde una cantidad significativa de paquetes en el enlace de comunicación es posible que el valor de PID sea igual al del último paquete válido recibido, cuando ocurre esto el dispositivo nRF24L01+ compara la suma de cada valor CRC de ambos paquetes, si el resultado es el mismo el paquete recibido se considera como una copia del anterior y por lo tanto se descarta.

3.10.3.3.3. Bandera de no acuse de recibido (NO_ACK)

Esta bandera de configuración solo puede ser utilizada cuando se utiliza la característica del acuse de recibido automático, al colocar como alto el valor de la bandera se le indica al receptor que no debe enviar un acuse de recibido.

Para configurar esta bandera en el dispositivo PTX, se puede utilizar el comando:

W_TX_PAYLOAD_NOACK

Para utilizar el comando anterior primero debe habilitar en el registro FEATURE el *bit* correspondiente a EN_DYN_ACK, cuando se utiliza esta opción el dispositivo PTX cambia automáticamente al modo de suspensión I después de transmitir el paquete. El PRX no transmite un acuse de recibido al PTX cada vez que recibe un paquete.

3.10.3.4. Carga útil del paquete

Se le denomina así a la información que el usuario puede definir dentro del paquete, puede ser de una longitud de 0 *bytes* hasta 32 *bytes* que es transferida por el aire cuando es enviada al dispositivo nRF24L01+.

Enhanced ShockBurst permite dos alternativas para manejar la longitud de la carga útil de los paquetes:

- Estático: es la configuración que se utiliza de forma predeterminada, en este modo todos los paquetes que son enviados entre transmisor y receptor deben tener la misma longitud, se puede configurar la longitud en el receptor utilizando los registros RX_PW_TX.

Para configurarlo en el transmisor la longitud de carga útil se establece por el número de *bytes* contabilizados en los registros TX_FIFO y debe ser igual al valor en el registro RX_PW_Px del lado del receptor.

- Dinámico: DPL por sus siglas en inglés, permite la transmisión de paquetes con longitud de carga útil variable, esto significa que para sistemas con diferentes longitudes de carga útil en el receptor y transmisor ya no es necesario utilizar la longitud mayor disponible.

Con esta característica el dispositivo nRF24L01+ puede conocer de forma automática la longitud de la carga útil del mensaje en vez de usar la configuración con los registros RX_PW_PX, el microcontrolador puede conocer la longitud de la carga útil recibida con el comando: R_RX_PL_WID.

Para activar esta característica el *bit* correspondiente a EN_DPL en el registro FEATURE debe ser activado, en el modo recepción se debe activar el registro DYNPD, el dispositivo PTX que transmita a un dispositivo PRX con DPL debe activar también el *bit* DPL_P0 en el registro DYNPD.

3.10.3.5. Verificación de redundancia cíclica

Es llamado CRC por sus siglas en inglés, es el mecanismo de detección de errores para los paquetes. Es de 1 o 2 *bytes* y se calcula utilizando la dirección, el campo de control y la carga útil del paquete.

El polinomio para un *byte* de CRC es $X^8 + X^2 + X + 1$, con un valor inicial de 0xFF.

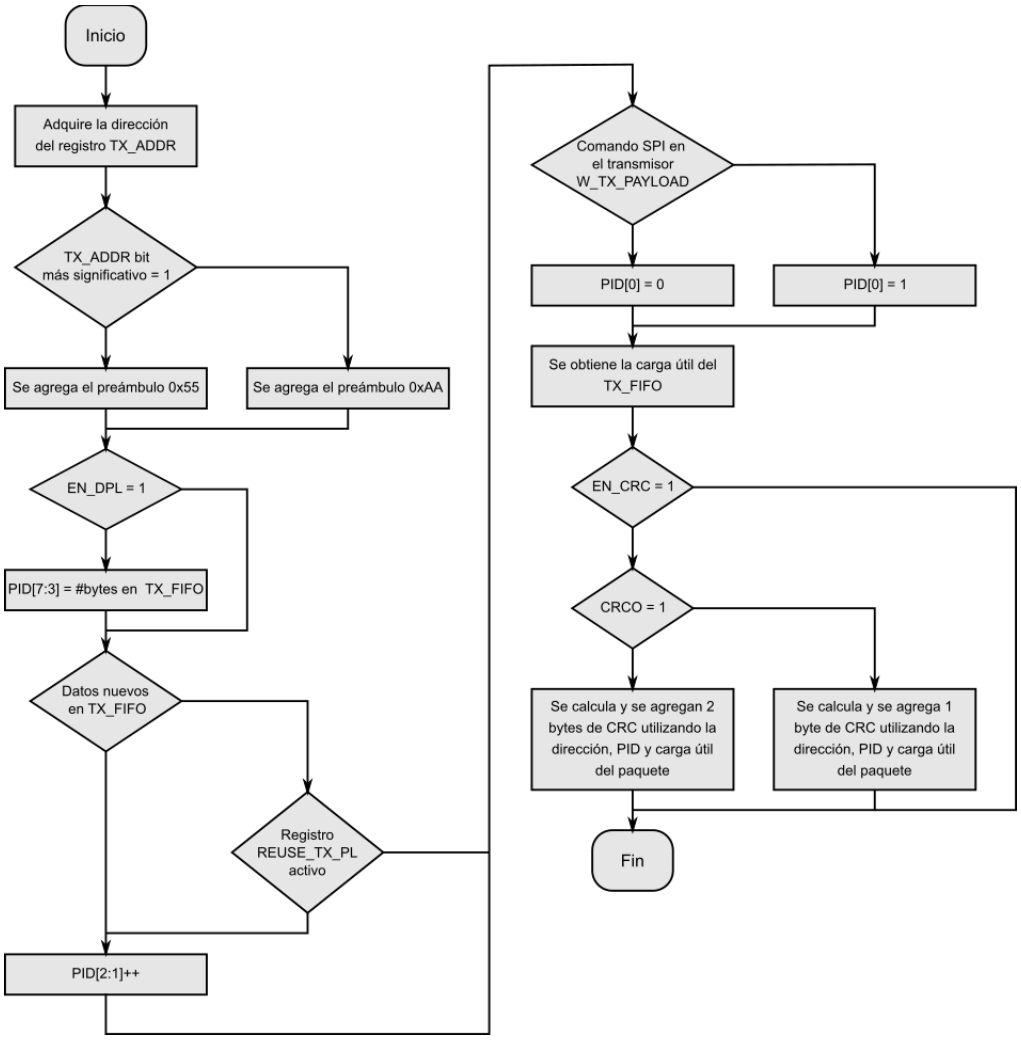
El polinomio para dos *bytes* de CRC es $X^{16} + X^{12} + X^5 + 1$, con un valor inicial de 0xFFFF.

El número de *bytes* en el CRC se configura con el *bit* CRCO en el registro CONFIG, ningún paquete es aceptado por Enhanced ShockBurst si la verificación CRC falla.

3.10.3.6. Ensamblado automático de paquetes

Este proceso toma los valores generados para el preámbulo, la dirección, el campo de control del paquete, la carga útil y el CRC para generar un paquete que luego será transmitido.

Figura 19. **Proceso de creación automática de un paquete**

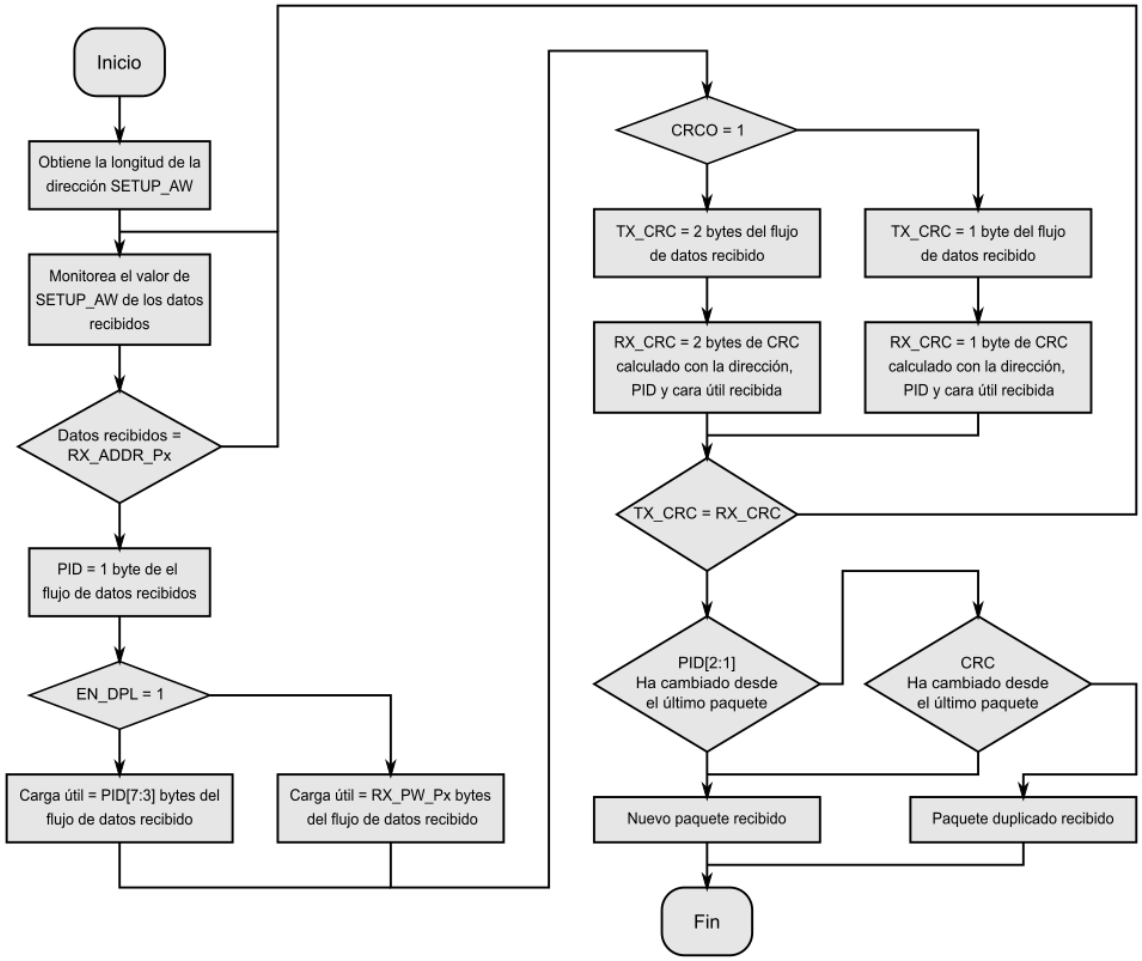


Fuente: elaboración propia, empleando Inkscape 1.0.

3.10.3.7. Extracción automática de paquetes

Luego de que un paquete es validado, Enhanced ShockBurst se encarga de extraer del paquete recibido la carga útil, almacenarla en el FIFO RX y activar la interrupción RX_DR.

Figura 20. **Proceso de extracción automática de un paquete**



Fuente: elaboración propia, empleando Inkscape 1.0.

3.10.4. Manejo automático de transacciones de paquetes

Enhanced ShokBurst posee dos funciones automáticas que se utilizan en el manejo de transacciones de paquetes.

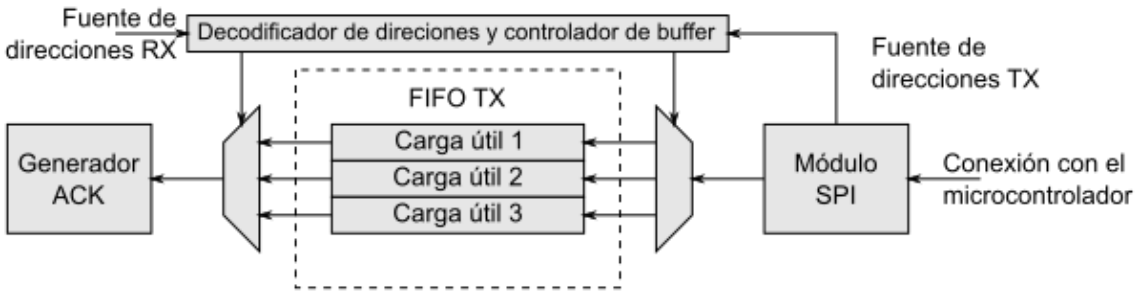
3.10.4.1. Acuse de recibido automático

Es una función que de forma automática envía un paquete ACK al PTX luego de recibir y validar un paquete, esto reduce la carga sobre el microcontrolador y el consumo de corriente eléctrica. Esta función se activa utilizando el registro EN_AA, si el paquete recibido tiene la bandera NO_ACK, no se enviará ningún acuse de recibido.

Un paquete ACK puede tener una carga útil opcional que tiene como origen el PRX y destino el PTX. Para utilizar esta característica se debe activar la longitud dinámica de carga útil (DPL) y el *bit* correspondiente a EN_ACK_PAY en el registro FEATURE. El microcontrolador en el dispositivo PRX debe sincronizar la carga útil con el FIFO TX utilizando el comando W_ACK_PAYLOAD.

La carga útil se encuentra pendiente en el FIFO TX del dispositivo PRX, hasta que se reciba un nuevo paquete del PTX, el dispositivo nRF24L01+ puede terne hasta tres cargas útiles de paquetes ACK pendientes en el FIFO TX del dispositivo PRX al mismo tiempo.

Figura 21. **FIFO TX con cargas útiles pendientes**



Fuente: elaboración propia, empleando Inkscape 1.0.

La figura 21 muestra como el FIFO TX maneja las cargas útiles de los paquetes ACK, el microcontrolador sincroniza la carga útil con el comando W_ACK_PAYLOAD, el decodificador de direcciones y controlador de buffer se asegura de que esta sea almacenada en un espacio disponible dentro del FIFO TX del dispositivo PRX, al mismo tiempo recibe la dirección del dispositivo PTX con el fin de utilizar la carga útil correspondiente en el generador de paquetes ACK.

Si el FIFO TX contiene más de una carga útil destinada a un dispositivo PTX, estas son procesadas siguiendo el principio de primero en llegar, primero en salir. El FIFO TX se bloqueará si todas las cargas útiles deben ser enviadas a un dispositivo PTX con el cual se ha perdido la comunicación, en este caso el FIFO TX puede ser vaciado utilizando el comando FLUSH_TX.

3.10.4.2. Retransmisión automática

Se le denomina ART por sus siglas en inglés, es una función que permite que un paquete sea retransmitido si no se recibe el paquete ACK, es usado en la función de acuse de recibido automático en el PTX.

Cuando no se confirma la recepción de un paquete, se puede configurar el número de veces que será retransmitido el paquete configurando el valor de los *bits* correspondientes a ARC en el registro SETUP_RETR. PTX entrará al modo de recepción y esperará un periodo corto de tiempo que el paquete ACK sea transmitido, este tiempo dependerá de las siguientes condiciones:

- Ha transcurrido el retardo de retransmisión automática (ARD).

- No se reconoce una dirección válida en 250 µs, 500 µs si se utiliza el modo de 250 kbps.

- Luego de recibir un paquete, sin importar si su CRC es correcto o no.

El dispositivo nRF24L01+ activa la interrupción TX_DS cuando el paquete ACK es recibido.

El dispositivo nRF24L01+ entra en modo de suspensión I si no existen más datos por transmitir en el FIFO TX y el pin correspondiente a CE se encuentra a un valor bajo, si el paquete ACK no es recibido el dispositivo regresará al modo de transmisión luego de un retardo definido por ARD y retransmitirá los datos, esto continua hasta que se recibe paquete ACK o se alcanza el número máximo de intentos de retransmisión.

Los contadores ARC_CNT y PLOS_CNT se encargan de llevan el registro de cuantos paquetes no se han transmitido de forma satisfactoria, se encuentran en el registro OBSERVE_TX.

El contador ARC_CNT corresponde al número de retransmisiones del paquete actual, este se reinicia al enviar un nuevo paquete. El contador

PLOS_CNT se encarga de registrar el número total de retransmisiones realizadas desde el último cambio de canal, este se reinicia cambiando el valor del registro RF_CH, es posible utilizar la información del registro OBSERVE_TX para realizar una evaluación de la calidad del canal seleccionado.

El ARD define el tiempo desde que se termina la transmisión del paquete hasta que empieza la retransmisión en el PTX, se configura en el registro SETUP_REGISTER en múltiplos de 250 µs, no existe una restricción en la longitud del ARD cuando se utilizan paquetes ACK con carga útil, aunque este no debe ser menor a la suma del tiempo de inicio y el tiempo de transmisión del paquete ACK.

Para velocidades de transmisión de 2 Mbps, 5 *bytes* utilizados en la dirección y un ARD igual a 250 µs, 15 *bytes* es el tamaño máximo que puede tener la carga útil del paquete ACK.

Para velocidades de transmisión de 1 Mbps, 5 *bytes* utilizados en la dirección y un ARD igual a 250 µs, 5 *bytes* es el tamaño máximo que puede tener la carga útil del paquete ACK. Si se utiliza un ARD igual a 500 µs el tamaño de la carga útil del paquete ACK puede tener cualquier valor al utilizar las velocidades de transmisión de 1 Mbps y 2 Mbps.

En la tabla X se muestran los valores necesarios de ARD y longitud de la carga útil del paquete ACK si se utiliza una velocidad de transmisión de 250 kbps y 5 *bytes* para las direcciones.

Tabla X. **Valores máximos de la longitud de carga útil del paquete ACK para distintos valores de ARD con 250 kbps como velocidad de transmisión**

ARD	Tamaño carga útil del paquete ACK (en *bytes*)
1500µs	Cualquier tamaño de carga útil
1250µs	< 24
1000µs	< 16
750µs	< 8
500µs	Paquete ACK sin carga útil

Fuente: elaboración propia, empleando Microsoft Word.

Como alternativa a la función de retransmisión automática es posible configurar el dispositivo nRF24L01+ para que retransmita un paquete una determinada cantidad de veces, esto se configura utilizando el comando REUSE_TX_PL, el microcontrolador deberá iniciar la retransmisión de cada paquete al enviar un pulso al pin correspondiente a CE cada vez que se utiliza este comando.

3.10.5. Operación del dispositivo PTX

Para que un dispositivo funcione como PTX se debe colocar el pin correspondiente a CE a un nivel alto, si se encuentra un paquete disponible en el FIFO TX, el dispositivo nRF24L01+ entra en modo de transmisión y en envía el paquete.

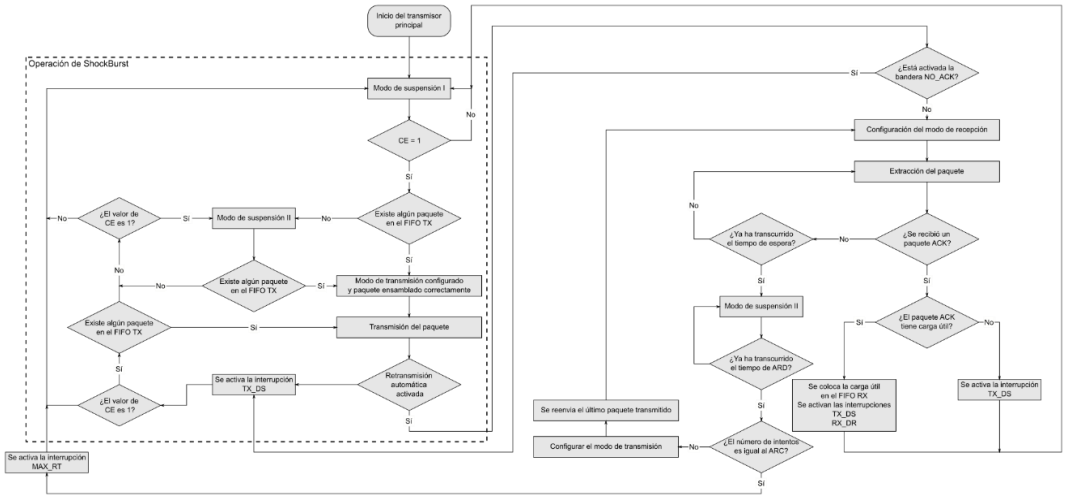
Si se encuentra activada la función de retrasmisión automática, la máquina de estados verifica si la bandera NO_ACK se encuentra activada, si no se encuentra activada el dispositivo entrará en modo de recepción a la espera del paquete ACK, si se encuentra una carga útil en el paquete ACK se activarán las

interrupciones TX_DS y RX_DR, en caso contrario solamente se activará la interrupción TX_DS antes de que el dispositivo regrese al modo de suspensión I.

Si no se recibe el paquete ACK en el tiempo establecido, el dispositivo cambia al modo de suspensión II, se mantiene en ese estado hasta que termine el tiempo estipulado por ARD, si el número máximo de retransmisiones no ha sido alcanzado el dispositivo entra en el modo de transmisión y envía nuevamente el paquete.

Cuando se utiliza la función de retransmisión automática y se alcanza el límite de transmisiones definidas en el ARC, se activa la interrupción MAX_PT y se regresa al modo de suspensión I. Si se coloca a nivel alto el pin correspondiente a CE y el FIFO TX se encuentra vacío el dispositivo nRF24L01+ entra en el modo de suspensión II.

Figura 22. **Funcionamiento del dispositivo PTX en Enhanced ShockBurst**



Fuente: elaboración propia, empleando Inkscape 1.0.

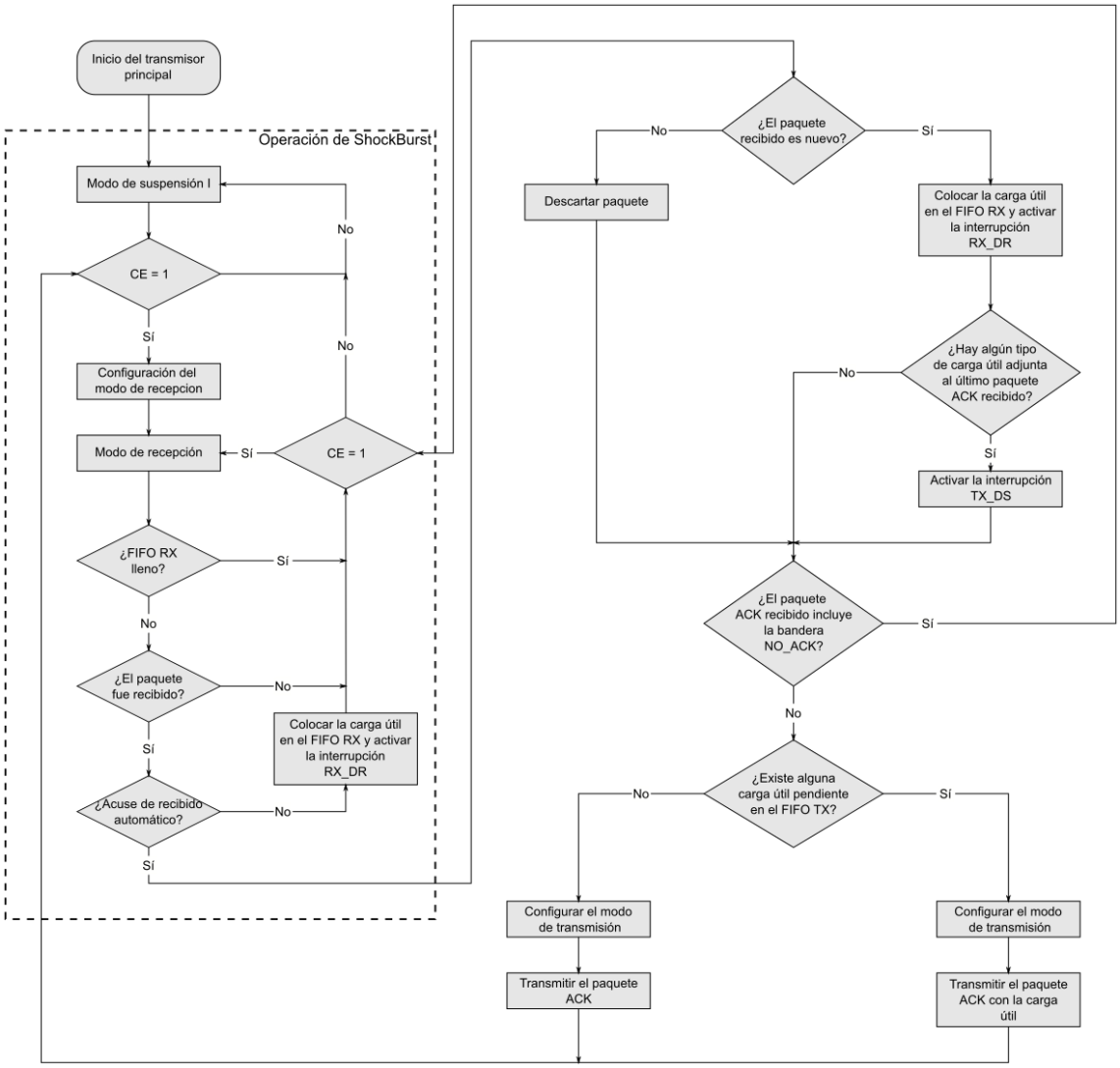
3.10.6. Operación del dispositivo PRX

Para activar este modo se necesita colocar el pin correspondiente a CE a un nivel alto, el dispositivo nRF24L01+ entra en modo de recepción y empieza a buscar paquetes, si un paquete es recibido y se encuentra activada la opción de acuse de recibido automático, el dispositivo decidirá si el paquete es uno nuevo o la copia de un que ha sido recibido previamente.

Si se recibe un nuevo paquete, su carga útil se coloca en el FIFO RX, la interrupción RX_DR indica que el PTX ha recibido un paquete ACK con carga útil, cuando la bandera NO_ACK no se encuentra en el paquete recibido, el PRX entra en modo de transmisión, si existe alguna carga útil en el FIFO TX es adjuntada en el paquete ACK, luego de transmitir el paquete ACK el dispositivo nRF24L01+ entra en modo de recepción.

Es posible que se reciban paquetes duplicados si el paquete ACK no es entregado, en este caso el PRX descarta el paquete recibido y transmite un paquete ACK antes de regresar al modo de recepción.

Figura 23. **Funcionamiento del dispositivo PRX en Enhance ShockBurst**



Fuente: elaboración propia, empleando Inkscape 1.0.

3.10.7. Compatibilidad con ShockBurst

Para utilizar el dispositivo nRF24L01+ en conjunto con otros dispositivos que solo sean compatibles con ShockBurst se debe desactivar Enhanced

ShockBurst colocando el registro EN_AA = 0x00 y el registro ARC = 0, adicional a esto la velocidad de transferencia de datos solamente puede ser de 1 Mbps o 250 kbps.

3.11. Interfaz de control y datos

Esta interfaz permite el acceso a todas las características del dispositivo nRF24L01+, la interfaz consiste en las siguientes líneas de comunicación digital que pueden tolerar señales de 5 voltios:

- IRQ: esta señal se activa con un pulso bajo y controlada por tres interrupciones enmascarables.

- CE: esta señal se activa con un pulso alto y se utiliza para colocar el dispositivo en modo de transmisión o recepción.

- CSN: señal SPI

- SCK: señal SPI

- MOSI: señal SPI

- MISO: señal SPI

Enviando comandos de 1 *byte* por la interfaz SPI se pueden activar los FIFOS de datos del dispositivo o el mapa de registros durante todos los modos de operación.

3.11.1. Características

- Comandos SPI especiales para el acceso rápido a las funciones más utilizadas.

- Interfaz SPI de 4 conectores con velocidad de transferencia de datos de hasta 10 Mbps.

- Conjunto de comandos de 8 *bits*.

- Mapa de registro de configuración simple.

- Tres FIFO que pueden ser utilizados para recibir y transmitir datos.

3.11.2. Comandos SPI

Los comandos SPI disponibles se encuentran en la tabla XI, para enviar cada comando primero se debe existir una transición entre el estado alto y bajo en el pin correspondiente a CSN. El registro STATUS será enviado en forma serial por el pin correspondiente a MISO y el comando SPI será enviado de igual forma por el pin correspondiente a MOSI.

Al enviar los comandos se envía del *bit* más significativo al menos significativo, cuando se envía los *bytes* de datos se envía del *byte* menos significativo al más significativo, cada *byte* se transmite del *bit* más significativo al menos significativo.

Tabla XI. **Comandos SPI para el dispositivo nRF24L01+**

Nombre del comando	**Palabra del comando (binario)**	**# *bytes* de datos**	**Operación**
R_REGISTER	000A AAAA	1 a 5, *byte* menos significativo primero	Lee los registros COMMAND y STATUS. AAAAA = dirección del mapa de registros de 5 *bits.*
W_REGISTER	001A AAAA	1 a 5, *byte* menos significativo primero	Escribe los registros COMMAND y STATUS. AAAAA = dirección del mapa de registros de 5 *bits.* Se puede ejecutar solo en los modos de baja potencia y suspensión.
R_RX_PAYLOAD	01100001	1 a 32, *byte* menos significativo primero	Lee la carga útil del modo de recepción: 1 – 32 *bytes*. La operación de lectura iniciará siempre en el *byte* 0. La carga útil es eliminada del FIFO al ser leída.
W_TX_PAYLOAD	10100000	1 a 32, *byte* menos significativo primero	Escribe la carga útil del modo de transmisión: 1 – 32 *bytes*. La operación de escritura empezará siempre en el *byte* 0.
FLUSH_TX	11100001	0	Elimina los datos del FIFO TX, utilizado en el modo de transmisión.
FLUSH_RX	11100010	0	Elimina los datos del FIFO TX, utilizado en el modo de recepción. No debe ser utilizado mientras se realiza la transmisión del paquete de acuse de recibido.

Continuación de la tabla XI.

Nombre del comando	**Palabra del comando (binario)**	**# *bytes* de datos**	**Operación**
REUSE_TX_PL	1110 0011	0	El dispositivo PTX reutilizará a última carga útil transmitida, esta se encontrará activa hasta que se ejecute W_TX_PAYLOAD o FLUSH TX No se debe activar o desactivar durante la transmisión de un paquete.
R_RX_PL_WID	0110 0000	1	Obtiene el tamaño de la carga útil del FIFO RX superior de R_RX_PAYLOAD. Si el valor obtenido es mayor a 32 *bytes*, el contenido del FIFO RX debe ser purgado con Flush_RX
W_ACK_PAYLOAD	1010 1PPP	1 a 32, *byte* menos significativo primero	Se utiliza en el modo de recepción. Le agrega carga útil al paquete ACK en PIPE PPP. (PPP e válido en un rango desde 000 hasta 101). Puede haber hasta un máximo de tres cargas útiles para el paquete ACK pendientes. Las cargas pendientes con el mismo PPP serán manejadas utilizando el principio primero en entrar, primero en salir. La operación de escritura inicia siempre en el *byte* 0.

Continuación de la tabla XI.

Nombre del comando	**Palabra del comando (binario)**	**# *bytes* de datos**	**Operación**
W_TX_PAYLOAD_NO_ACK	10110000	1 a 32, *byte* menos significativo primero	Se utiliza en el modo de transmisión, desactiva AUTOACK en ese paquete.
NOP	11111111	0	Ninguna operación realizada, se utiliza para leer el registro STATUS.

Fuente: elaboración propia, empleando Microsoft Word.

Los comandos W_REGISTER y R_REGISTER pueden operar en registros de uno o varios *bytes*, cuando se accede a registros de varios *bytes* se lee o escribe el *bit* más significativo del *byte* menos significativo.

3.12. FIFO de datos

Los FIFOs de datos son accesibles desde el modo PTX y PRX, los FIFOs que se encuentran presentes en el dispositivo nRF24L01+ son:

- Transmisión de tres niveles, 32 *bytes*.
- Recepción de tres niveles, 32 *bytes*.

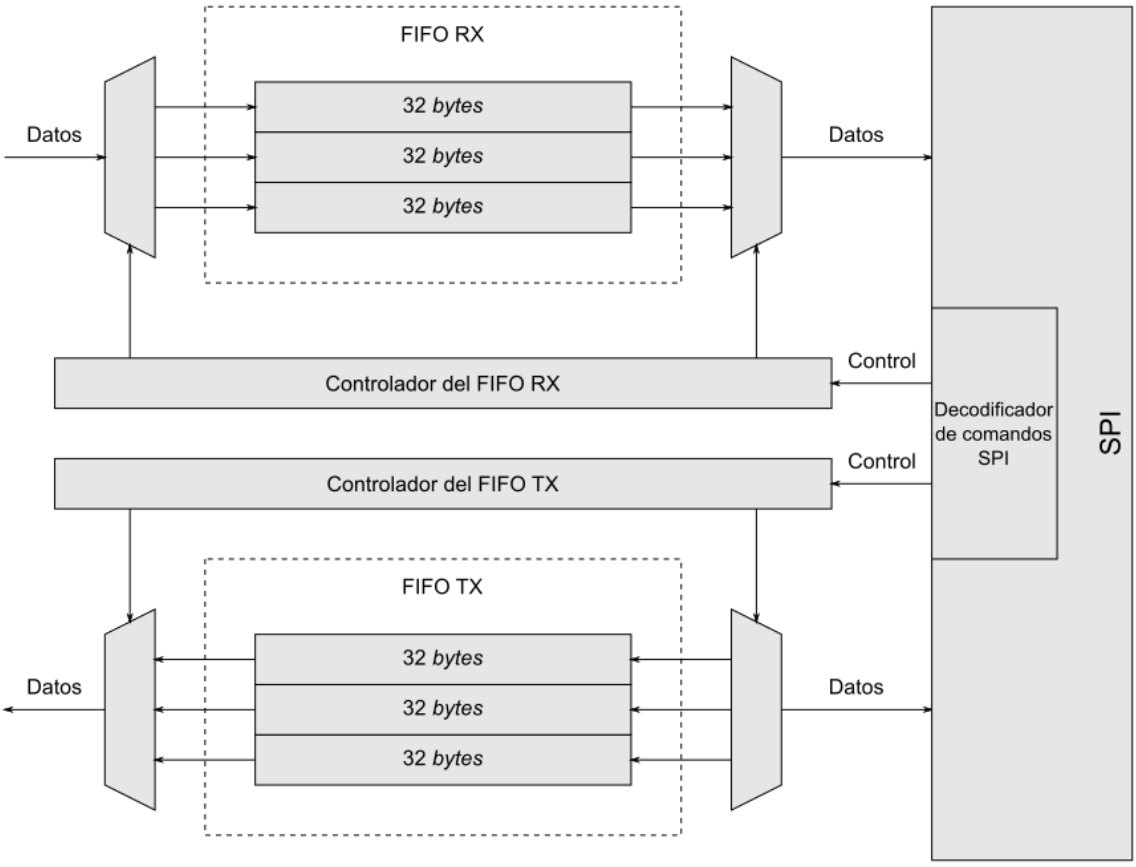
Ambos FIFOs pueden ser controlados utilizando la interfaz SPI a través de los comandos SPI, un FIFO TX en un dispositivo PRX puede almacenar hasta tres cargas útiles para los paquetes ACK de tres dispositivos PTX diferentes, si el FIFO TX contiene más de una carga útil estas son procesadas utilizando el principio de primero en entrar, primero en salir.

El FIFO TX en un dispositivo PRX se bloquea si existen cargas útiles pendientes de ser enviadas debido a que se ha perdido la conexión con el dispositivo PTX, en este caso el controlador puede vaciar el FIFO TX utilizando el comando FLUSH_TX.

Se puede acceder al FIFO TX desde el modo PTX utilizando los comandos W_TX_PAYLOAD y W_TX_PAYLOAD_NO_ACK o desde el modo PRX con el comando W_ACK_PAYLOAD, estos comandos permiten acceder al registro TX_PLD. Para acceder al FIFO RX se puede utilizar el comando R_RX_PAYLOAD desde el modo PTX o PRX, este comando permite acceder al registro RX_PLD.

La carga útil en el FIFO TX no se eliminará si la interrupción MAX_RT se encuentra declarada, para comprobar si el FIFO TX o RX se encuentra vacío o con datos se utiliza el registro FIFO_STATUS.

Figura 24. **Diagrama de bloques del FIFO (TX y RX)**



Fuente: elaboración propia, empleando Inkscape 1.0.

3.13. Interrupciones

El dispositivo nRF24L01+ tiene un pin correspondiente a las interrupciones o IRQ que se encuentra a nivel bajo cuando está activo, este pin se activa cuando las interrupciones TX_DS, RX_DR o MAX_RT se encuentran habilitadas en el registro STATUS.

La máscara de interrupciones en el registro CONFIG se utiliza para seleccionar las fuentes de interrupción que pueden activar el pin de interrupción, al colocar en la máscara el *bit* correspondiente a una fuente de interrupción a nivel alto esta fuente se desactiva, por defecto todas las fuentes de interrupción están activadas.

El valor de los tres *bits* correspondientes a las interrupciones en el registro STATUS es actualizado durante la transición del estado alto a bajo en el pin IRQ, si se lee este registro durante dicha transición la información obtenida no es confiable.

3.14. Antena del dispositivo nRF24L01+

Los pines correspondientes a la antena son ANT1 y ANT2, estos proveen de una salida balanceada para la conexión de dicha antena, estos pines deben estar conectados al pin VDD_PA ya sea por medio de una bobina de choque o por el punto central de una antena dipolo balanceada.

Es recomendable utilizar una carga de $15\,\Omega + j88\Omega$ para obtener una potencia máxima de transmisión (0 dBm).

3.15. Oscilador de cristal

El tiempo que el oscilador demora en iniciar su operación es proporcional a la inductancia equivalente del cristal, en el diseño moderno de osciladores se toma el camino de reducir el tamaño físico del oscilador esto deriva en un aumento de la inductancia equivalente del cristal.

El tiempo que se demora en iniciar su operación es denominado Tpd2stby y debe tener un valor máximo de 1,5 ms, esto se consigue utilizando un cristal con una inductancia en serie equivalente a un máximo de 30 mH, en el caso de que Ls exceda los 30 mH el valor máximo de Tpd2stby se puede calcular con:

$$Tpd2stby = \frac{Ls}{30mH} * 1.5ms$$

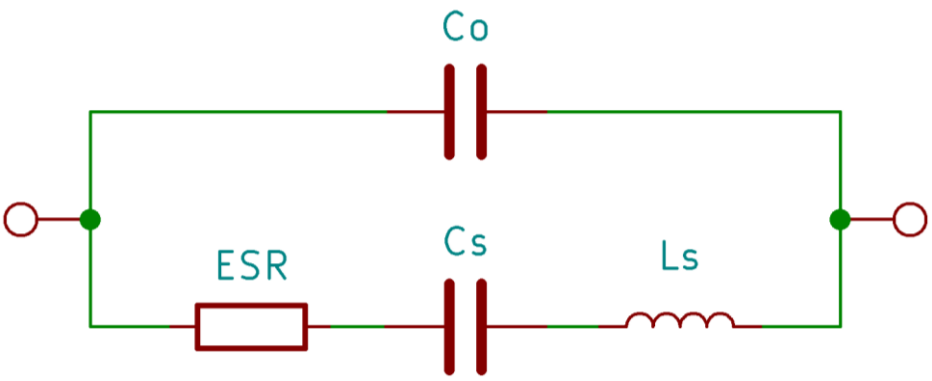
Las especificaciones del cristal a utilizar en el dispositivo nRF24L01+ se encuentran en la tabla XII y el circuito equivalente del cristal se encuentra en la figura 25.

Tabla XII. **Especificaciones del cristal**

Symbol	Parámetro	Min.	Típico	Máx.	Unidades
Fxo	Frecuencia del cristal		16		MHz
ΔF	Tolerancia			± 60	ppm
C0	Capacitancia equivalente en paralelo		1,5	7,0	pF
Ls	Inductancia equivalente en serie		30		mH
CL	Capacitancia de la carga	8	12	16	pF
ESR	Resistencia equivalente en serie			100	Ω

Fuente: elaboración propia, empleando Microsoft Word.

Figura 25. **Circuito equivalente del oscilador de cristal**



Fuente: elaboración propia, empleando Kicad 5.1.5.

4. RASPBERRY PI

4.1. Ordenadores de una sola placa

Se denomina así a los sistemas de computación que incluyen el procesador o procesadores, la memoria RAM, las entradas/salidas de propósito general, los elementos para comunicarse a la red, entre otros. en una sola placa de circuito impreso. Al contrario de los sistemas de computación tradicionales, el *hardware* de estos ordenadores no es modular por lo tanto no puede ser actualizado.

Los ordenadores de una sola placa son utilizados como ordenadores de bajo coste para el sector de la educación, la investigación y el desarrollo de sistemas embebidos, estos últimos son muy comunes en la industria varios productos comerciales han sido desarrollados, fabricados y comercializados utilizando ordenadores de una sola placa.

El microordenador Trainer MMD-1 diseñado por John Titus en 1976 es considerado como el primer ordenador de una sola placa, su diseño se basaba en el microprocesador C8080A de Intel, se le llamaba *dyna-micro* en su fase de diseño y al empezar su producción se le denominó MMD-1 acrónimo para *Mini Micro Designer 1*.

Los ordenadores de una sola placa formaron una parte importante en la historia de la computación doméstica, siendo algunos el BBC Micro fabricado por Acorn Computers en 1981 o su versión más económica la Acorn Electronic lanzada al mercado en 1983.

4.2. Historia y antecedentes de la Raspberry Pi

Durante décadas, las escuelas habían enseñado cómo usar los programas de computadora, no cómo hacerlos. Al mismo tiempo, las tecnologías digitales se habían vuelto menos abiertas a modificaciones por parte de los usuarios. El resultado fue que, sin importar el entorno, ya sea en la escuela o en el hogar, muchas personas se convirtieron en consumidores digitales en lugar de creadores digitales.

La inspiración para realizar lo que hoy se conoce como Raspberry Pi nació mientras Eber Upton se encontraba trabajando con estudiantes de la carrera de Ciencias de la Computación en la Universidad de Cambridge, Upton se percató de una caída en la solicitud de inscripciones en dicha carrera en la década de los 2000 y que los estudiantes necesitaban tener más oportunidades para adquirir experiencia en la programación antes de ingresar a la universidad.

Los primeros conceptos de lo que se convertiría en una Raspberry Pi se materializaron en 2006, la Fundación Raspberry Pi fue establecida en 2008 como una organización benéfica con sede en el Reino Unido con el propósito de promover el avance de la educación de adultos y niños, particularmente en el campo de las computadoras, la informática y temas relacionados.

Las primeras placas en versión alfa fueron mostradas al público por error en 2011 luego de que Upton fuera a visitar a Rory Cellan-Jones en la BBC, con la intención de solicitar el uso de la marca descontinuada BBC Micro y este último publicara un vídeo del prototipo en su blog alcanzando en dos días 600 000 visitas en YouTube y haciendo de forma accidental, según palabras de Upton ¨la promesa a más de medio millón de personas de fabricar un ordenador cuyo precio sea $25¨.

Las primeras 10 unidades de Raspberry Pi se subastaron al inicio del 2012, logrando recaudar £16 000, la primera producción de 10 000 unidades se puso a la venta el 29 de febrero de 2012, la imagen ISO del sistema operativo se había publicado a finales de 2011 y para la fecha de lanzamiento ya había sido descargada más de 50 000 veces, los dos distribuidores oficiales en ese entonces reportaron más de 100 000 órdenes ese día.

El desarrollo de la Raspberry Pi fue enfocado principalmente a la educación el Pi en el nombre proviene del uso de Python en el ordenador, sin embargo, llamó la atención a programadores y *makers* avanzados. Los primeros modelos en ser lanzados al mercado fueron el Modelo A y el Modelo B nombres similares a los utilizados en la BBC Micro, siendo el Modelo B lanzado primero.

4.3. Modelos disponibles de Raspberry Pi

Existen tres series de Raspberry Pi y varias generaciones han sido lanzadas, la serie Raspberry Pi utilizan como base un sistema en un chip SoC por sus siglas en inglés de la marca Broadcom que integra una unidad de procesamiento central (CPU) compatible con ARM y una unidad gráfica de procesamiento (GPU), mientras que la serie Raspberry Pi Pico utiliza el SoC RP2040 con una unidad de procesamiento (CPU) integrada compatible con ARM.

4.3.1. Raspberry Pi

La primera generación fue lanzada en 2012 siendo el Modelo B el primero seguido por el Modelo A, la principal forma de diferenciar entre ambos modelos es la falta de puerto *Ethernet* del Modelo A, en 2014 se realizó una revisión de diseño en los modelos denominados Modelo B+ y Modelo A+ estos incluían un procesador ARM11 y su tamaño era aproximadamente el mismo que el de una

tarjeta de crédito volviendo este tamaño un estándar en las siguientes versiones. En 2014 se lanzó un módulo de computación enfocado principalmente para aplicaciones embebidas.

Figura 26. **Imagen de una Raspberry Pi 1 Modelo A+**

Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-1-model-a-plus/. Consulta: 14 enero 2022.

La Raspberry Pi 2 fue lanzada en febrero de 2015, inicialmente contaba con un procesador ARM Cortex-A7 de cuatro núcleos y 32 *bits* con una velocidad de operación de 900 MHz y 1 GB de memoria RAM, la revisión 1,2 incluye un procesador ARM Cortex-A53 de cuatro núcleos y 64 *bits* (el mismo que se incluiría posteriormente en la Raspberry Pi 3 pero con su velocidad de operación reducida a 900 MHz).

Figura 27. **Imagen de una Raspberry Pi 2 Modelo B+**

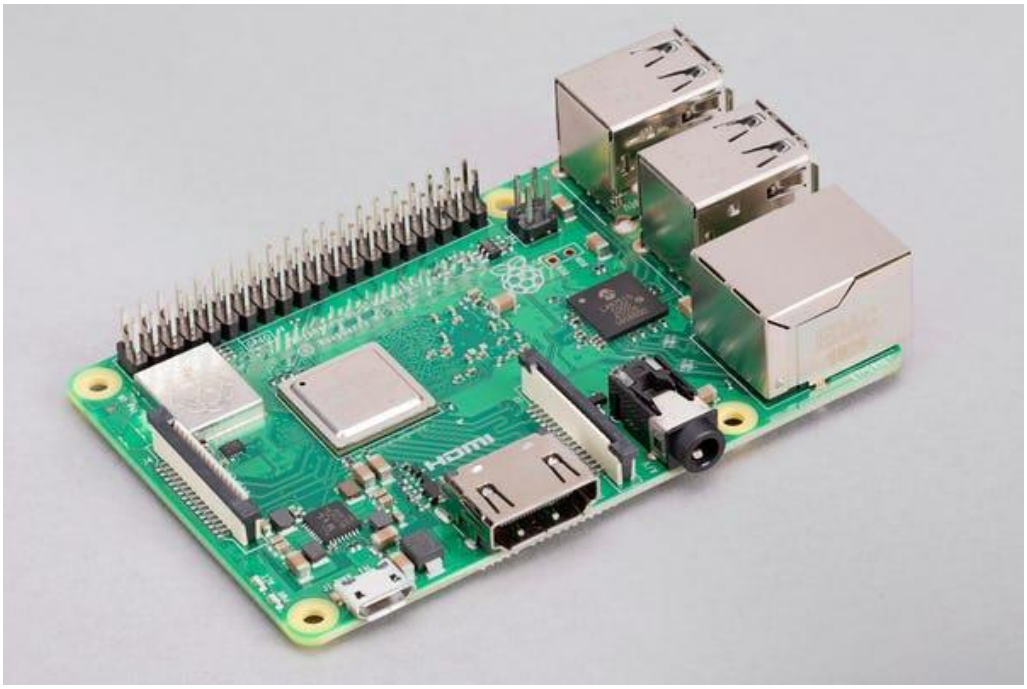
Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-1-model-b-plus/. Consulta: 14 enero 2022.

La Raspberry Pi 3 Modelo B fue lanzada en febrero de 2016 los cambios más notables son el uso de un procesador ARM Cortex-A53 de cuatro núcleos y 64 *bits* con una velocidad de operación de 1,2 GHz, la inclusión de capacidad de conectarse a redes inalámbricas utilizando el protocolo 802.11n, un módulo *Bluetooth* y la posibilidad de iniciar el ordenador desde los puertos USB.

Durante el día Pi de 2018 se lanzó la Raspberry Pi 3 Modelo B+ mejorando así la velocidad de operación del procesador que aumentaba a 1,4 GHz, un puerto *Gigabit Ethernet* tres veces más rápido que el anterior pero limitado a aproximadamente 300 Mbit/s debido a la conexión USB 2,0 que utiliza, la conexión a redes inalámbricas ahora utiliza las bandas de 2,4 y 5 GHz además

de utilizar el protocolo 802.11ac, es posible utilizar otras características como alimentación por internet (PoE por sus siglas en Inglés) o iniciar el ordenador a través de Internet.

Figura 28. **Imagen de una Raspberry Pi 3 Modelo B+**



Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/. Consulta: 14 enero 2022.

En noviembre del 2018 se lanzó la Raspberry Pi 3 Modelo A+, siendo esta la última iteración del modelo 3, mantiene las mismas características del procesador incluido en el Modelo B+ pero la memoria RAM se reduce a 512 MB, es compatible con todos los accesorios diseñados para las versiones anteriores del Modelo A.

Figura 29. **Imagen de una Raspberry Pi 3 Modelo A+**

Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-3-model-a-plus/. Consulta: 14 enero 2022.

El modelo actual es la Raspberry Pi 4 Modelo B, fue lanzada en junio de 2019 posee un procesador ARM Cortex-A72 de cuatro núcleos y 64 *bits* que opera a una velocidad de 1,8 GHz, la conexión del puerto *Gigabit Ethernet* ya no se encuentra limitada, sigue utilizando el protocolo 802.11ac para las conexiones a redes inalámbricas, el módulo *Bluetooth* se actualizó a la versión 5, posee dos puertos micro HDMI tipo D con una resolución máxima de 4k, dos de sus puertos USB son 2,0 y los dos restantes son 3,0.

Se encuentran disponibles versiones con 1, 2, 4 y 8 GB de memoria RAM, la versión con 1 GB de memoria RAM se descontinuó en marzo del 2020 en favor de la de 2 GB manteniendo está el precio de la primera, debido a la crisis iniciada

en 2020 de escases de semiconductores se ha reanudado su producción en octubre de 2021. A mediados de 2019 se lanzó la revisión 1,2 que resuelve un problema de diseño en el cual algunos cables USB no funcionaban para alimentar la Raspberry Pi, a mediados de 2021 todos los modelos de Raspberry Pi 4 comenzaron a utilizar el procesador mejorado BCM2711C0 de Broadcom.

Figura 30. **Imagen de una Raspberry Pi 4 Modelo B**

Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. Consulta: 14 enero 2022.

La Raspberry Pi 400 fue lanzada en noviembre de 2020, es un diseño personalizado derivado de la Raspberry Pi 4 que incluye un teclado similar al producto oficial llamado Raspberry Pi Keyboard, como consecuencia de poseer más espacio dentro de la estructura del teclado el sistema de enfriamiento y fuente de alimentación fueron mejorados permitiendo así que la velocidad de

operación del procesador BCM2711C0 aumentara a 1,8 GHz, esta versión se encuentra disponible solamente con 4 GB de memoria RAM.

Figura 31. **Imagen de una Raspberry Pi 400**

Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-400-unit/. Consulta: 14 enero 2022.

4.3.2. Raspberry Pi Zero

Es la versión de tamaño y características reducidas de la versión estándar de la Raspberry Pi, fue lanzada en noviembre de 2015 a un precio de $5 utiliza el procesador BCM2835 que fue utilizado en la Raspberry Pi 1, 512 MB de memoria RAM.

Figura 32. **Imagen de una Raspberry Pi Zero**

Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-zero/. Consulta: 14 enero 2022.

El 28 de febrero de 2017 se lanzó la Raspberry Pi Zero W cuyo cambio principal fue la incorporación de la posibilidad de conectarse a redes inalámbricas a un precio de $10, en enero de 2018 se incluyó la versión WH de la Raspberry Pi Zero que es idéntica a la versión W pero esta última incluye los pines GPIO soldados desde la fábrica.

Figura 33. **Imagen de una Raspberry Pi Zero W**

Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-zero-w/. Consulta: 14 enero 2022.

El 28 de octubre de 2021 fue lanzada la Raspberry Pi Zero 2 W que utiliza un sistema en un paquete SiP por sus siglas en inglés desarrollado por Raspberry Pi y basado en la Raspberry Pi 3, a diferencia de las versiones anteriores el procesador de la Raspberry Pi Zero 2 W es capaz de manejar instrucciones de 64 bits, a un precio de $15.

Figura 34. **Imagen de una Raspberry Pi Zero 2 W**

Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/. Consulta: 14 enero 2022.

4.3.3. Raspberry Pi Pico

Fue lanzada en enero de 2021, utiliza el microcontrolador RP2040 cuyo diseño es propio de Raspberry Pi fue desarrollado en el Reino Unido, entre sus características se encentran 264 KB de memoria RAM, 2 MB de memoria de almacenamiento y 26 pines de propósito general para realizar conexiones con diferentes dispositivos.

Es compatible con MicroPython, CircuitPython, C y Rust, se ha asociado con Vilros, Adafuirt, Pimoroni, Arduino y SparkFun para el desarrollo de

accesorios y placas de desarrollo utilizando el RP2040 como base, su concepto según sus desarrolladores es similar al de Arduino.

Figura 35. **Imagen de una Raspberry Pi Pico**

Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/products/raspberry-pi-pico/. Consulta: 14 enero 2022.

Tabla XIII. **Comparación de los modelos de Raspberry Pi**

Familia	**Modelo**	**SoC**	**Memoria**	***Ethernet***	**Wifi**	**GPIO**
Raspberry Pi	B	BCM2835	256 / 512 MB	Sí	No	26 pines
	A		256 MB	No		
	B+		512 MB	Sí		40 pines
	A+			No		
Raspberry Pi 2	B	BCM2836/7	1 GB	Sí	No	

Continuación de la tabla XIII.

Familia	**Modelo**	**SoC**	**Memoria**	***Ethernet***	**Wifi**	**GPIO**
Raspberry Pi Zero	Zero	BCM2835	512 MB	No	Sí	
	W/WH					
	2 W	BCM2710A1 (RP3A0)				
Raspberry Pi 3	B	BCM2837A0/B0	1 GB	Sí	Sí	
	A+	BCM2837B0	512 MB	No		
	B+		1 GB	Sí		
Raspberry Pi 4	B	BCM2711	1 GB	Sí	Sí	
			2 GB			
			4 GB			
			8 GB			
	400		4 GB			
Raspberry Pi Pico		RP2040	264 KB	No	No	26 pines

Fuente: elaboración propia, empleando Microsoft Word.

4.4. Configuración de Raspberry Pi

Los modelos estándar de la Raspberry Pi no poseen ningún tipo de almacenamiento incluido por defecto dentro de la placa de circuito impreso, es necesario utilizar un dispositivo de almacenamiento externo, en este caso una memoria microSD o SD en las versiones más antiguas, en dónde se procederá a instalar el sistema operativo y almacenar los archivos generados por el usuario, exceptuando la Raspberry Pi Pico los demás modelos son compatibles con algunas distribuciones GNU/Linux.

4.4.1. GNU/Linux

GNU/Linux es la denominación técnica generalizada que reciben los sistemas operativos tipo UNIX, que suelen ser de código abierto, multiusuario y multitarea. Su rol principal es proveer al usuario de todo lo necesario para realizar operaciones dentro de su plataforma de ejecución.

Los sistemas operativos GNU/Linux consisten en distintos componentes que a su vez poseen distintas variantes, todos estos componentes se agrupan en las denominadas distribuciones que en su mayoría son diseñadas con un campo de aplicación predeterminado.

En términos generales se puede describir los componentes de un sistema operativo GNU/Linux como:

- Entorno de escritorio.
- Administrador de ventanas.
- X Window System (X11) / Wayland.
- Interfaz de línea de comandos.
- *Kernel* o núcleo.

Siendo el *kernel* o núcleo el cerebro de toda operación, se comunica con el *hardware* directamente, por ejemplo, el acceso a la entrada del teclado o el acceso al almacenamiento en los discos duros tendrá que utilizar de alguna forma el núcleo para realizar sus operaciones correctamente, además puede ser

compilado para distintas versiones de CPU como el CPU ARM que posee la Raspberry PI.

Entre los sistemas operativos disponibles para las versiones de la Raspberry Pi a excepción de la Raspberry Pi Pico se encuentra:

- Raspberry Pi OS
    - Con entorno de escritorio.
    - Con entorno de escritorio y programas recomendados.
    - Versión ligera, sin interfaz gráfica.
    - Versión de compatibilidad para equipos no soportados en la versión más reciente.
- Fedora, en sus distintas versiones estación de trabajo, servidor, IoT o con distintos entornos de escritorio como KDE Plasma, GNOME, LXQT, entre otros.
- Ubuntu, en su versión de escritorio, servidores o IoT.
- LibreElec.
- RetroPie.

4.4.2. Tarjetas Secure Digital

Abreviadas oficialmente como SD, es un tipo propietario no volátil de tarjeta de memoria desarrollado por la Asociación SD (SDA por sus siglas en inglés) para su uso en dispositivos portátiles.

El estándar fue introducido en agosto de 1999 a través de los esfuerzos conjuntos de SanDisk, Panasonic y Toshiba como una mejora al formato de las Multi Media Cards (MMCs), las tres compañías formaron la SD-3C LLC que se encarga de otorgar licencias y hacer cumplir los derechos de propiedad intelectual asociados con las tarjetas de memoria SD y todo lo relacionado a estas.

Estás compañías fueron las encargadas de fundar en el año 2000 la Asociación SD, una organización sin fines de lucro encargada de promover y crear estándares para las tarjetas SD. La SDA utiliza varios logotipos de marcas registradas propiedad de SD-3C y autorizados por esta para exigir el cumplimiento de sus especificaciones y asegurar a los usuarios la compatibilidad.

Figura 36. **Ejemplo de las distintas capacidades de las tarjetas SD**

Fuente: Deshmukh, Supratik. *Memory Card Evolutions*. https://unsplash.com/photos/hLZYqn5Ae2s. Consulta: 13 enero 2022.

4.4.3. Raspberry Pi OS

Raspberry Pi OS es un sistema operativo gratuito basado en Debian, optimizado para el *hardware* Raspberry Pi es el sistema operativo recomendado para uso normal en una Raspberry Pi. El sistema operativo tiene disponible más de 35 000 paquetes de *software* pre compilado incluido en un formato agradable para una fácil instalación en su Raspberry Pi.

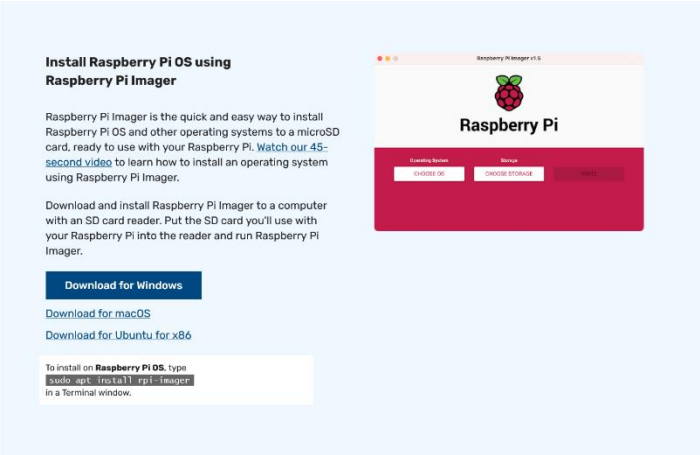
El sistema operativo Raspberry Pi se encuentra en desarrollo activo, con énfasis en mejorar la estabilidad y el rendimiento de tantos paquetes Debian como sea posible en Raspberry Pi.

Para la instalación de Raspberry Pi OS se recomienda una memoria SD con capacidad mínima de 8 GB para la versión con entorno de escritorio o 4 GB para la versión ligera, debido a una limitación de *hardware* en la Raspberry Pi Zero, 1 y 2 la partición *boot* no puede ser mayor a 256 GB de lo contrario el dispositivo no iniciará, en modelos que utilizan el procesador BCM2837 o posterior no poseen esta limitación.

4.4.3.1. Instalación

La forma más simple de instalar el sistema operativo Raspberry Pi OS en una tarjeta SD es utilizar el programa desarrollado por Raspberry Pi llamado Raspberry Pi Imager, se encuentra disponible en la dirección https://www.raspberrypi.com/software/ donde se encuentra la versión para Windows, Mac OS y Ubuntu x86.

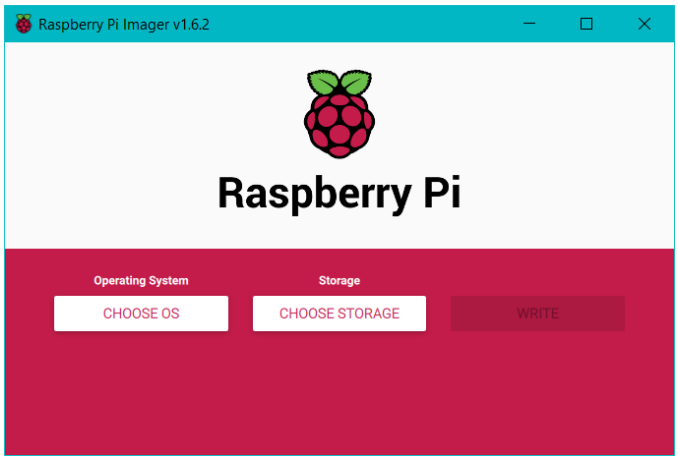
Figura 37. **Opciones disponibles de descarga para Raspberry Pi Imager**



Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/software/. Consulta: 13 enero 2022.

Luego de instalar Raspberry Pi Imager, se procede a insertar la tarjeta SD en la computadora y se ejecuta el programa.

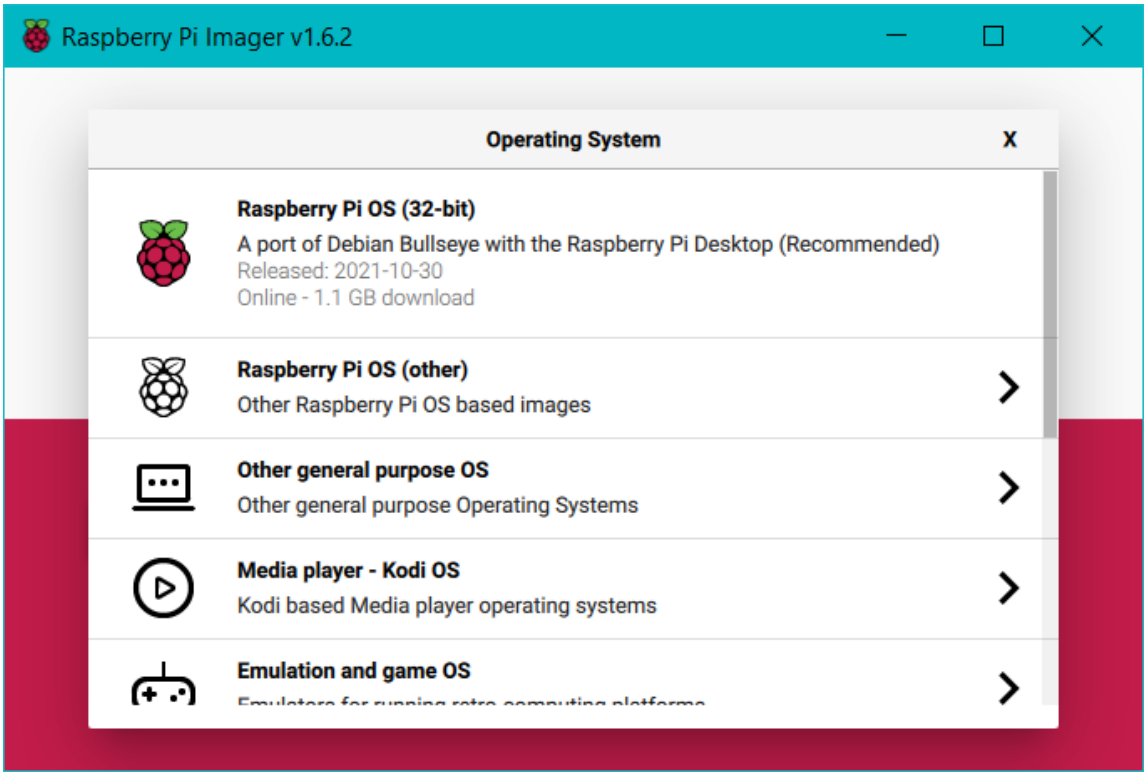
Figura 38. **Interfaz de Raspberry Pi Imager**



Fuente: elaboración propia, empleando Raspberry Pi Imager 1.6.2

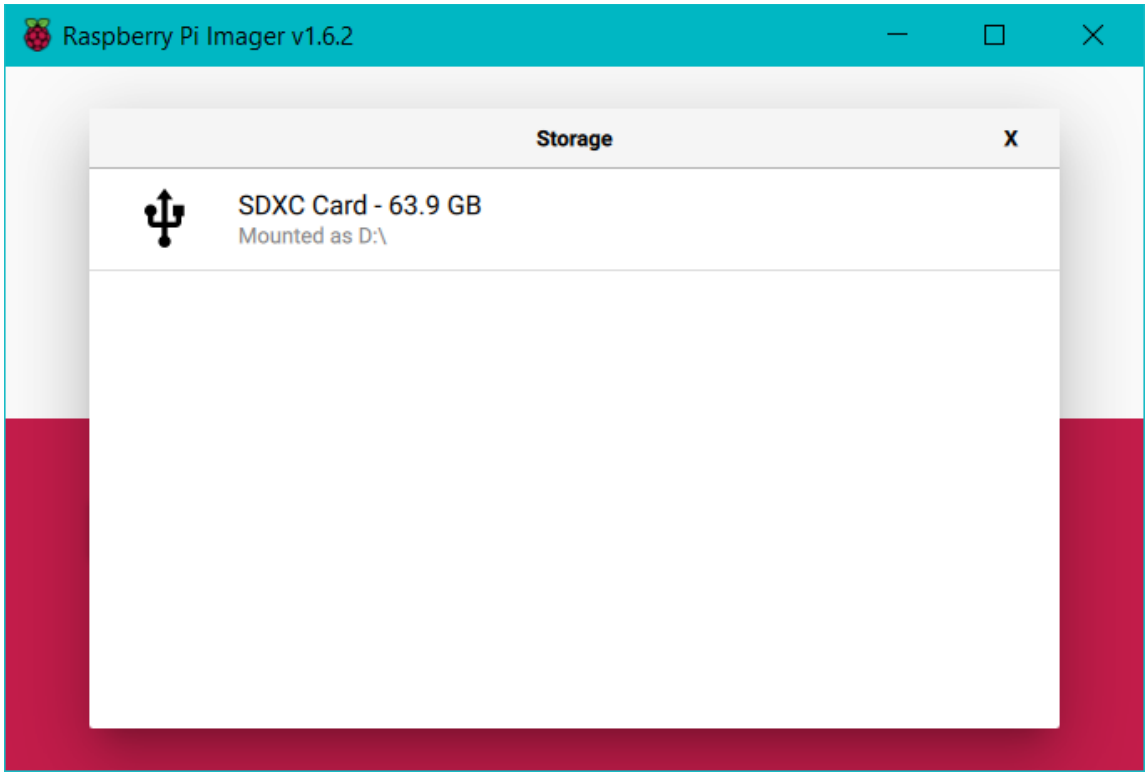
Se debe seleccionar el sistema operativo que se desee instalar desde las opciones que se presentan, también puede utilizar una imagen descargada de alguna otra fuente, luego se selecciona el dispositivo de almacenamiento que se desea utilizar, en el caso de la Raspberry Pi sería la tarjeta SD que se insertó previamente, para iniciar el proceso de escritura en la tarjeta SD se debe hacer clic en el botón escribir.

Figura 39. **Sistemas operativos disponibles para descargar desde Raspberry Pi Imager**



Fuente: elaboración propia, empleando Raspberry Pi Imager 1.6.2

Figura 40. **Selección del dispositivo de almacenamiento para instalar el sistema operativo en Raspberry Pi Imager**



Fuente: elaboración propia, empleando Raspberry Pi Imager 1.6.2

Cuando termine el proceso de escritura y comprobación la tarjeta SD se podrá utilizar en una Raspberry Pi, por defecto el nombre de usuario es pi, la contraseña es raspberry y la disposición del teclado es la de Reino Unido (UK). Dentro del programa Raspberry Pi Imager es posible acceder a las configuraciones avanzadas, entre las cuales se encuentra:

- Cambiar el nombre en la red que tendrá la Raspberry Pi

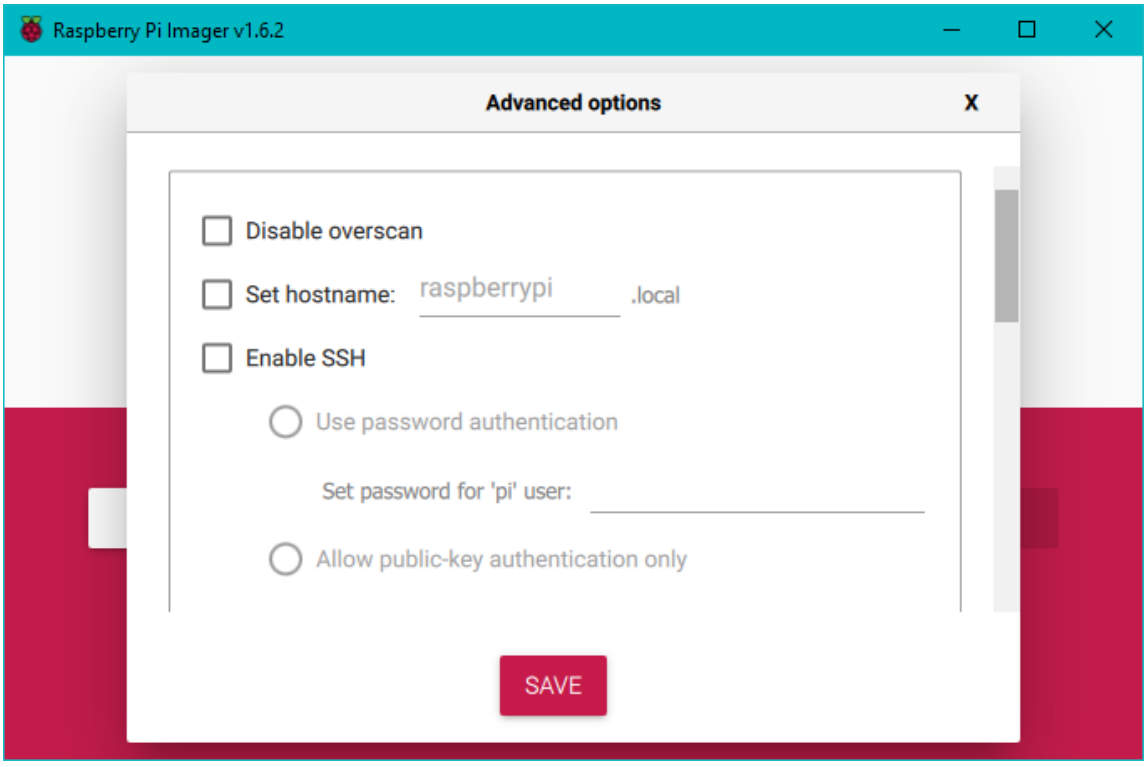
- Activar la conexión remota por SSH y cambiar la contraseña por defecto o agregar una llave de autenticación.

- Configurar la conexión wifi del dispositivo.

- Cambiar las opciones de zona horaria y disposición del teclado, entre otros.

Se accede a este menú presionando las teclas ctrl, shift y x al mismo tiempo.

Figura 41. **Opciones avanzadas de Raspberry Pi Imager**



Fuente: elaboración propia, empleando Raspberry Pi Imager 1.6.2.

4.4.3.2. Operaciones básicas

Entre estas operaciones se puede nombrar la actualización, instalación o desinstalación de programas para realizar tareas con el dispositivo. Es importante mantener la Raspberry Pi actualizada por dos razones importantes:

- La seguridad, el sistema operativo se compone de millones de líneas de código de las que depende, con el tiempo se encuentran vulnerabilidades en las implementaciones de código de algunos programas que son documentadas y se encuentran disponibles en bases de datos públicas por lo tanto son fáciles de explotar.

- La segunda guarda relación con la primera, algunos programas pueden contener errores algunos de estos podrían ser críticos, otros podrían estar afectando a la funcionalidad de los dispositivos sin estar relacionados con la seguridad, al mantener el *software* actualizar se reduce la probabilidad de encontrarse con estos errores.

La forma más fácil de realizar estas acciones es utilizar la herramienta de Debian llamada herramienta de empaquetado avanzado o APT por sus siglas en inglés.

APT mantiene una lista con las fuentes de los programas instalados en la Raspberry Pi, este archivo se encuentra en */etc/apt/sources.list*, antes de actualizar los programas es necesario actualizar dicha lista, esto se hace ejecutando en una terminal el comando *sudo apt update*, cuando este termine se procede a realizar la actualización ejecutando *sudo apt full-upgrade*, se prefiere utilizar el comando *full-upgrade* en vez de *upgrade* debido a que el primero también maneja los cambios en las dependencias de los programas.

Para buscar programas o paquetes con APT se utiliza el comando *apt-cache search* seguido por el nombre del programa o paquete que se desee buscar, para mostrar los resultados de la búsqueda anterior se utiliza el comando *apt-cache show*.

Para instalar el programa o paquete con APT se utiliza el comando sudo *apt install* seguido por el nombre del programa o paquete que se desee instalar, se deberá ingresar la contraseña de administrador, APT mostrará cuantos datos necesita descargar y cuanto espacio ocupara en disco el paquete o programa a instalar, luego pedirá la confirmación por parte del usuario, para aceptar de forma automática esta petición basta con agregar un -y al final del comando de instalación.

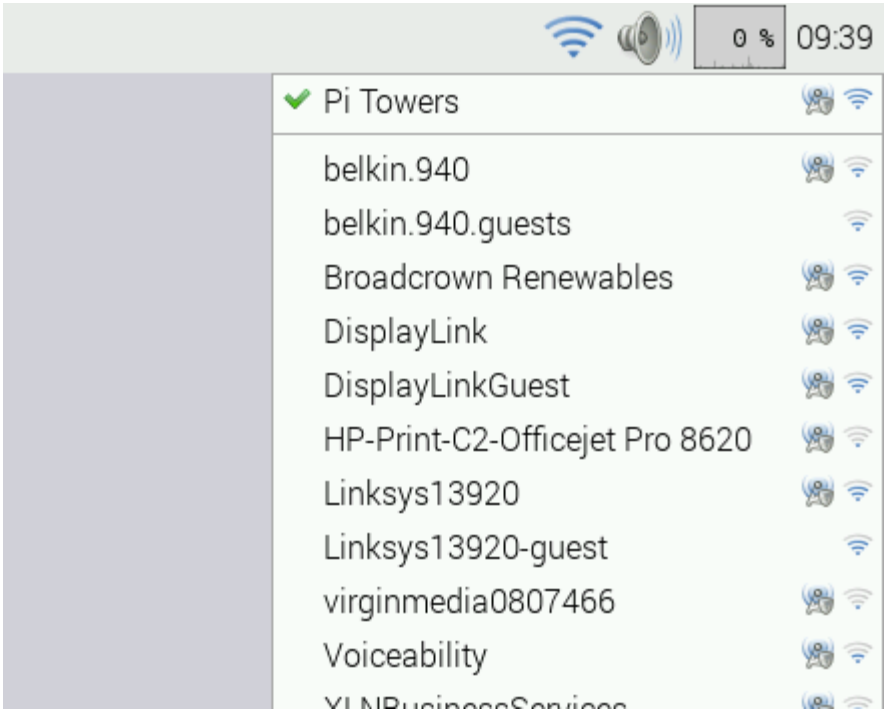
El proceso para desinstalar un programa o paquete es similar al de instalación, siendo este *sudo apt remove* seguido del nombre del programa o paquete que se desea desinstalar, nuevamente se mostrará el espacio en disco que será liberado y se pedirá la confirmación por parte del usuario, al igual que en el caso de la instalación agregar -y al final de comando aceptará de forma automática esta acción.

4.4.3.3. Configuración de conexiones inalámbricas

Dependiendo del modelo de Raspberry Pi a utilizar se tendrá acceso a redes cuya frecuencia de operación es 2,4 y 5 GHz, por disposiciones regulatorias es necesario actualizar en las opciones de localización el país para el manejo de las conexiones inalámbricas, para realizar esto se puede utilizar la herramienta *rasp-config* en su versión con interfaz gráfica o en la versión para la línea de comandos.

Para realizar la conexión a una nueva red utilizando la interfaz gráfica, en el escritorio de Raspberry Pi OS en la esquina superior derecha se encuentra un ícono que al hacer clic en él se mostrará un listado con las redes inalámbricas disponibles y si estas necesitan una contraseña para poder conectarse a ellas, si la red necesita contraseña aparecerá una ventana adicional solicitando que el usuario la ingrese.

Figura 42. **Listado de redes inalámbricas disponibles en Raspberry Pi OS**



Fuente: Raspberry Pi. *Productos raspberrypi.* https://www.raspberrypi.com/documentation/computers/configuration.html#configuring-networking. Consulta: 14 enero 2022.

Para realizar la conexión a una red inalámbrica utilizando la línea de comandos la forma más simple de hacerlo es utilizando como administrador la

herramienta *raspi-config*, en las opciones de localización se debe selecciona el país en el que se utilizará el dispositivo en caso de no haberlo hecho anteriormente, esto con el fin de cumplir con las normas de regulación de cada país, en las opciones de red de la herramienta *raspi-config* se ingresa el nombre de la red y la contraseña para realizar la conexión.

Otra forma de configurar las conexiones a redes inalámbricas es editando el archivo *wpa_supplicant.conf* que se encuentra en el directorio */etc/wpa_supplicant/*, la edición de este archivo se realiza ejecutando el comando *sudo nano /etc/wpa_supplicant/wpa_supplicant.conf* en la línea de comandos, se debe agregar al final del archivo los datos de la red a la que se desea conectar siguiendo el siguiente formato:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country= Código ISO 3166-1 del país, GT para Guatemala.
network={
    ssid="prueba"
    psk="Contraseñadeprueba"
}
```

Las primeras dos líneas son necesarias si la Raspberry Pi ejecuta una versión de Raspberry Pi OS basada en Debian Buster o alguna versión posterior, en la tercera línea se indica el código ISO del país para el correcto manejo de las frecuencias de 5 GHz según las regulaciones de cada país.

La sección *network* se refiere a cada una de las conexiones que el usuario desee guardar, cada red almacenada puede tener las siguientes opciones:

- *ssid*, es el nombre de la red.

- *psk*, es la contraseña para conectarse a dicha red, se puede utilizar la herramienta *wpa_passphrase* para colocar la contraseña de forma encriptada.

- *id_str*, se utiliza para diferenciar entre las redes que se tengan almacenadas,

- *priority*, asigna un nivel de prioridad a la conexión cuando más de una red almacenada se encuentre en el alcance. La red dentro del alcance, con la prioridad más alta, será a la que se conecte la Raspberry Pi.

- *key_mgmt=NONE*, si la red a la que se desea conectar no tiene configurada una contraseña de acceso.

Los cambios en el archivo se guardan con crtl+x y luego confirmando los cambios escribiendo y, luego se presiona la tecla intro.

4.4.3.4. Configuración de una dirección IP estática

En algunos casos es importante mantener una misma dirección IP para la Raspberry Pi dentro de una red, lo recomendable es realizar este control y asignación desde la interfaz del enrutador o servidor DHCP de la red, si se tiene acceso limitado a estos es posible asignar una dirección IP estática modificando el contenido del archivo *dhcpcd.conf* que se encuentra en el directorio / del sistema, puede ser editado utilizando la herramienta *lxplug-network* que posee una interfaz gráfica o utilizando un editor en la línea de comandos, por ejemplo nano. La estructura del archivo *dhcpcd.conf* es la siguiente:

- *interface*, se refiere a la interfaz del dispositivo a la que se le asignará la dirección IP, se utiliza el comando *ip link* para listar el nombre de todas las interfaces disponibles, por ejemplo, *eth0*.

- *static ip_address*, se refiere a la dirección IP y a la máscara de subred en su forma de longitud de prefijo, por ejemplo 192.168.0.4/24

- *static routers*, se refiere a la dirección IP del dispositivo que servirá como puerta de enlace hacia otras redes para el dispositivo, por ejemplo 192.168.0.254

- *static domain_name_servers*, se refiere a la dirección IP de los servidores de nombres de dominio o DNS, que se encargan de realizar la traducción entre direcciones IP y nombres que son más simples de recordar, por ejemplo 192.168.0.254 8.8.8.8.

5. POSTGRESQL Y DJANGO

5.1. Bases de datos

El término base de datos puede ser definido como una colección o repositorio de datos, posee una estructura predeterminada y es manejada por un sistema de administración de bases de datos o DBMS por sus siglas en inglés. Los datos pueden estar estructurados en forma tabular, semi estructurados como en los archivos XML o no poseer una estructura que encaje en algún modelo predeterminado.

Con la introducción de lenguajes de programación orientados a objetos, se crearon nuevos paradigmas en los sistemas de manejo de bases de datos como las bases de datos relacionales de objetos o las bases de datos orientadas a objetos. Con la introducción de las aplicaciones web como redes sociales y similares, se necesitó que las bases de datos soportarán una gran cantidad de peticiones de forma distribuida, esto llevó a la creación de un nuevo paradigma llamado bases de datos NoSQL, que se deriva del término *Not only SQL* que posee entre sus características un mejor rendimiento sobre tolerancia a las fallas o la capacidad de ser escalable de forma horizontal.

En general la línea del tiempo de la evolución de las bases de datos ha sido afectada por factores como:

- Requisitos funcionales: debido a la naturaleza de las aplicaciones que utilizan las bases de datos se han desarrollado extensiones de las bases de datos relacionales como PostGIS para datos espaciales o sistemas de

manejo de bases de datos dedicados como SCI-DB que se utiliza para el análisis científico de datos.

- Requisitos no funcionales: la popularidad de los lenguajes de programación orientados a objetos ha creado tendencias como lo son las bases de datos orientadas a objetos y con ello sistemas de gestión de bases de datos relacionales de objetos que funcionan como un puente entre las bases de datos y los lenguajes de programación. De la misma forma el incremento exponencial de la cantidad de datos y la necesidad de manejar *terabytes* de datos en dispositivos de baja potencia han sido clave en el desarrollo de bases de datos columnares las cuales son escalables de forma horizontal con facilidad.

5.1.1. Categorías de bases de datos

Actualmente las categorías predominantes en la industria son las bases de datos relacionales, las bases de datos relacionales de objetos y las bases de datos NoSQL.

5.1.1.1. Bases de datos NoSQL

Proveen un medio para almacenar, manipular y consultar datos no relacionales, comúnmente adoptan el modelo BASE que es el acrónimo para *basically available soft-satate, eventual-consistency* que prioriza la disponibilidad de los datos sobre la consistencia de estos, e informalmente garantiza que, si no se realizan nuevas actualizaciones en un elemento de datos, eventualmente todos los accesos a ese elemento de datos devolverán la última versión de ese elemento de datos. Entre sus ventajas se encuentra las siguientes:

- Simplicidad en su diseño.

- Son de fácilmente replicables y escalables horizontalmente.

- No posee un esquema predeterminado.

- Soporta una gran cantidad de datos.

Las bases de datos NoSQL son afectadas por el teorema CAP, también conocido como el teorema de Brewer, en el año 2002 S. Gilbert y N. Lynch publicaron una prueba formal del teorema CAP en su artículo ¨La conjetura de Brewer y la viabilidad de servicios web consistentes, disponibles y tolerantes al particionado¨ según datos de https://hostingdata.co.uk/nosql-database/ existen actualmente 225 sistemas de manejo de bases de datos NoSQL.

5.1.1.1.1. El teorema CAP

Su nombre se deriva de los términos en inglés *consistency, availability and partition tolerance.* Indica que para un sistema de computación distribuido es imposible que pueda garantizar de forma simultánea los siguientes aspectos:

- Consistencia: todos los clientes podrán ver de forma inmediata los datos más actualizados incluso en caso de actualizaciones.

- Disponibilidad: todos los clientes pueden acceder a una copia de los datos incluso durante el fallo de un nodo, significa que si alguna parte del sistema falla los clientes aún podrán acceder a los datos.

- Tolerante al particionado: el sistema sigue funcionando incluso si un número arbitrario de mensajes son descartados (o retrasados) entre nodos de la red.

La decisión de que característica se debe descartar dependerá de la naturaleza del sistema, por ejemplo, se puede sacrificar la consistencia para tener un sistema de manejo de bases de datos escalable, simple y de alto desempeño. La diferencia más común entre una base de datos relacional y una NoSQL es la consistencia, en las bases de datos relacionales se impone ACID, que es el acrónimo para las propiedades: *Atomicity, Consistency, Isolation and Durability*.

5.1.1.2. Bases de datos relacionales y bases de datos relacionales de objetos

Son los sistemas de manejo de bases de datos más utilizados en el mundo, los programas informáticos las utilizan a través de un servidor dedicado o motores ligeros de sistemas de manejo de bases de datos incluidos entre sus dependencias o en las librerías compartidas.

Las capacidades de estos sistemas varían entre implementaciones, pero la mayoría se adhiere al estándar ANSI SQL. Una base de datos relacional puede ser definida por álgebra relacional que puede ser definida como un conjunto de operaciones que describen paso a paso cómo computar una respuesta sobre las relaciones, tal y como éstas son definidas en el modelo relacional. Las bases de datos orientadas a objetos son similares a las bases de datos relacionales, pero soportan conceptos orientados a objetos como:

- Tipos de datos complejos definidos por el usuario.

- Herencia de datos entre objetos.

5.1.1.2.1. Propiedades ACID

En una base de datos relacional, una operación lógica es llamada transacción, cada transacción debe ser garantizada sin importar factores como errores, fallos en los sistemas de energía entre otros. La traducción técnica de una transacción se puede definir como el conjunto de operaciones en una base de datos que son crear, leer, actualizar y borrar o CRUD por sus siglas en inglés. El ejemplo más simple se puede observar en una transacción bancaria entre dos cuentas, la cual involucra debitar a una cuenta y acreditar a otra, las propiedades ACID en este contexto serían:

- Atomicidad: todo o nada, lo que significa que si una parte de la transacción falla entonces la transacción falla como un todo.

- Consistencia: cualquier transacción lleva la base de datos de un estado a otro, esta consistencia es controlada normalmente por limitaciones en los datos debido a las posibles combinaciones entre los estos, por ejemplo, si se desea eliminar una cuenta, se deberán eliminar todos los datos relacionados a la misma como lo son direcciones, números de teléfono, entre otros.

- Aislamiento: comúnmente las transacciones son ejecutadas de forma simultánea, esta característica se encarga de que estas ejecuciones dejen la base de datos en un estado como si hubieran sido ejecutadas de forma secuencial.

- Durabilidad: las transacciones que son realizadas con éxito son persistentes en la base de datos incluso si se produce una falla de energía en el servidor o parte de este se daña, esto se logra comúnmente con la técnica llamada *write-ahead log* o WAL.

5.1.1.2.2. Lenguaje SQL

Las bases de datos relacionales usualmente son asociadas con el lenguaje de consulta estructural o SQL por sus siglas en inglés, es un lenguaje de programación declarativo. El Instituto Nacional Estadounidense de Estándares (ANSI) y la Organización Internacional para la Estandarización (ISO) publicaron en 1986 el estándar para el lenguaje SQL, con el tiempo ha recibido distintas revisiones.

El lenguaje SQL se compone de las siguientes partes:

- Lenguaje de definición de datos o DLL por sus siglas en inglés: define y modifica la estructura relacional de los datos.

- Lenguaje de manipulación de datos o DML por sus siglas en inglés: permite consultar y extraer la información sobre las relaciones de los datos.

- Lenguaje de control de datos o DCL por sus siglas en inglés: controla los permisos de acceso a las relaciones de datos.

5.2. PostgreSQL

Es un sistema de manejo de bases de datos relacionales de código abierto, compite con proveedores de servicios similares como Oracle, MySQL, entre otros. Es multiplataforma por lo tanto se encuentra disponible para los sistemas operativos de Windows, Mac OS y GNU/Linux, se ajusta a la especificación de SQL y posee las propiedades ACID.

El nombre PostgreSQL se deriva de la base de datos post-ingres, su historia puede resumirse de la siguiente manera:

- Académico: Desarrollo en la Universidad de California en Berkeley
    - 1977 – 1985, el proyecto Ingress: fue creado por Michael Stonebraker, era un sistema de manejo de bases de datos relacionales.
    - 1986 – 1994, postgres: es desarrollado por Michael Stonebraker para brindar soporte a tipos de datos más complejos y al modelo relacional de objetos.
    - 1995, Postgres95: Andrew Yu y Jolly Chen cambian el lenguaje de consultas de postgres pasando de POSTQUEL a un subconjunto extendido de SQL.
- Industria
    - 1996, PostgeSQL: Luego del trabajo en conjunto de varios desarrolladores para estabilizar el funcionamiento de Postgres95,

se lanzó la primera versión de código abierto el 29 de enero de 1997. Debido al inicio de proyecto como proyecto de código abierto se cambió el nombre de Postgres95 a PostgreSQL.

El proyecto PostgreSQL utiliza la licencia PostgreSQL que es similar a las licencias BSD o MIT, es altamente permisiva y PostgreSQL no puede ser controlado o comprado por un tercero, esto otorga las siguientes ventajas a los usuarios de PostgreSQL.

- No se debe pagar una licencia para utilizar PostgreSQL.
- El usuario puede implementar una cantidad ilimitada de instancias de PostgreSQL.
- PostgreSQL sigue los estándares de SQL por lo tanto, la búsqueda de mano de obra calificada no debería ser tan complicado.
- Posee una curva de aprendizaje reducida y el traslado de otro sistema de manejo de bases de datos a PostgreSQL resulta beneficioso.
- Los procesos administrativos son fácilmente automatizables.
- Posee integración con varios lenguajes de programación.
- Es escalable y ofrece un alto rendimiento.
- Debido a que cumple con las propiedades ACID puede soportar algunos fallos de *hardware*.

- Posee una documentación muy completa y una comunidad muy activa, el manual de uso posee aproximadamente 2 500 páginas.

- Tiene una gran cantidad de extensiones disponibles.

- Se puede integrar fácilmente con otros sistemas de manejo de bases de datos de forma simple lo cual le da flexibilidad para la implementación de diseños específicos.

Debido a que PostgreSQL puede ser utilizado con una gran variedad de aplicaciones, estas se pueden agrupar en dos categorías:

- Procesamiento de transacciones en línea, OLTP por sus siglas en inglés: se caracteriza por una cantidad enorme de operaciones CRUD, una rápida velocidad de procesamiento de las operaciones y mantiene la integridad de la base de datos en un sistema de acceso múltiple. Su rendimiento se mide en el número de transacciones que realizan por segundo.

- Procesamiento analítico en línea, OLAP por sus siglas en inglés: se caracteriza por una cantidad pequeña de transacciones, complejas consultas que involucran la adición de datos, gran cantidad de datos provenientes de distintas fuentes y en distintos formatos, minería de datos y análisis histórico de datos.

Las aplicaciones OLTP se utilizan en operaciones de modelo de negocios como los sistemas de gestión de relaciones con los clientes o CRM por sus siglas en inglés, normalmente siguen conceptos como la normalización de datos cuando se diseña la base de datos, las aplicaciones OLAP son utilizadas en inteligencia empresarial, soporte para la toma de decisiones, generación de

reportes y planeamiento, los datos manejados por estas aplicaciones tienen menos relación, al final los datos son normalizados.

PostgreSQL por defecto puede ser utilizada para aplicaciones OLTP, para utilizarla en una implementación OLAP se deben utilizar algunas extensiones o herramientas como el comando PostgreSQL COPY y contenedores de datos externos o FDW por sus siglas en inglés.

5.2.1. Arquitectura de PostgreSQL

Utiliza el modelo cliente servidor, en el cual el cliente y el servidor pueden estar en distintos dispositivos. La conexión entre el cliente y el servidor usualmente realizada utilizando el protocolo TCP/IP o Linux Sockets. PostgreSQL puede manejar múltiples conexiones para un mismo cliente, comúnmente un programa que utiliza PostgreSQL consiste en lo siguiente:

- Proceso o programa cliente (*frontend*): se encarga de realizar la acción en la base de datos, puede ser un servidor web que desea mostrar el contenido de una página web o un programa de la línea de comandos que se encarga de realizar tareas administrativas o de mantenimiento. PostgreSQL provee de herramientas *frontend* como *psql*, *createdb*, *dropdb* y *createuser*.

- Proceso del servidor (*backend*): este proceso maneja la base de datos, acepta las conexiones de los clientes y realiza las acciones que estos solicitan, el nombre del proceso del servidor es postgres.

PostgreSQL bifurca un nuevo proceso para cada nueva conexión, con esto el cliente y el servidor se pueden comunicar entre ellos sin la intervención del

proceso principal del servidor, estos nuevos procesos tienen un tiempo de ejecución determinado al aceptar y terminar la conexión de un cliente.

5.2.2. Arquitectura abstracta de PostgreSQL

La arquitectura mencionada anteriormente brinda una descripción general de las capacidades de PostgreSQL y describe la interacción con el cliente y el sistema operativo. El servidor de PostgreSQL puede dividirse aproximadamente en cuatro subsistemas:

- Gestor de procesos: se encargar de manejar los procesos como bifurcar el proceso principal cuando un cliente nuevo de conecta.

- Procesador de consultas: cuando un cliente envía una consulta esta es analizada determinando su tipo, luego ejecutada y por último el resultado es enviado de regreso al cliente.

- Utilidades: proporciona herramientas para realizar mantenimiento a la base de datos como reclamar espacio de almacenamiento, actualizar estadísticas o exportar e importar datos con un cierto formato y registro.

- Gestor de almacenamiento: maneja la memoria caché, el búfer en los discos y la asignación de almacenamiento.

Casi todos los componentes de PostgreSQL pueden ser configurados, incluido el registro, el planeador, el analizador de estadísticas y el gestor de almacenamiento. La configuración de PostgreSQL se rige por la naturaleza de la aplicación ya sea OLAP u OLTP.

5.2.3. Capacidades de PostgreSQL

Entre las capacidades que posee para brindar el correcto funcionamiento de servicios empresariales se encuentra:

5.2.3.1. Replicación

Permite que los datos de una base de datos en un servidor puedan ser replicadas en otro servidor, esto con el objetivo de conseguir lo siguiente:

- Alta disponibilidad: un segundo servidor puede reemplazar al primero en caso de fallo.

- Balance de cargas: todos los servidores pueden resolver las mismas consultas, gracias a esto la carga de trabajo se distribuye de forma equitativa entre ellos.

- Rápida ejecución: una consulta es ejecutada en diferentes servidores al mismo tiempo para obtener un mejor desempeño.

PostgreSQL soporta la replicación por defecto, para realizar dicha replicación utiliza un modelo de maestro esclavo denominado *Streaming replication*, es una técnica de replicación binaria debido a que las declaraciones SQL no son analizadas. Consiste en tomar una instantánea del nodo maestro para luego enviar los cambios a los nodos esclavos en dónde se replicarán dichos cambios. El nodo maestro se puede utilizar para operaciones de lectura y escritura a diferencia de los nodos esclavos que solo se pueden utilizar para operaciones de lectura.

5.2.3.2. Seguridad

PostgreSQL soporta distintos métodos de autenticación entre ellos contraseña, LDAB, GSSAPI, SSPI, Kerberos, entre otros, las vulnerabilidades que se descubren en PostgreSQL se encuentran listadas en el sitio web https://www.postgresql.org/support/security/ indicando la versión afectada, la clase de vulnerabilidad y el componente al que afecta dicha vulnerabilidad.

Las actualizaciones de seguridad de PostgreSQL se encuentran disponibles como actualizaciones menores, al utilizar este método es más fácil para el administrados mantener su versión de PostgreSQL segura y actualizada sin necesidad de dar de baja él servicio durante demasiado tiempo, además, los problemas de seguridad conocidos son siempre corregidos en los lanzamientos de una versión mayor.

En PostgreSQL se pueden controlar el acceso a las bases de datos en distintos niveles incluyendo a nivel de base de datos, tabla, vista, función, secuencia y columna. En PostgreSQL se pueden encriptar los datos para protegerlos utilizando encriptación por *hardware* o si se desea encriptar solo parte de estos se puede utilizar la extensión *pgcrypto*.

5.2.3.3. Extensión

PostgreSQL puede ser extendido para soportar nuevos tipos de datos para dicho fin se utiliza el comando *CREATE EXTENSION* para agregar nuevas extensiones a la base de datos en la que se esté trabajando. PostgreSQL posee una red de distribución central llamada PGXN por sus siglas en inglés en donde los usuarios pueden explorar y descargar distintos tipos de extensiones.

Durante la instalación de PostgreSQL se instala también un paquete llamado *postgresql-postgresql-contrib* el cual contiene muchas extensiones útiles como *tablefunc* o la extensión *pgcrypto* mencionada en la sección anterior. La habilidad de PostgreSQL de soportar extensiones es el resultado de tener las siguientes características:

- Tipos de datos de PostgreSQL: soporta desde tipos de datos primitivos, algunas estructuras primitivas de datos, adicionalmente soporta tipos complejos de datos como:

    - Tipos de datos geométricos: incluyen puntos, segmentos de líneas, caminos, polígonos y cajas.

    - Tipos de datos de red: incluye tipos como *cidr*, *inet* y *macaddr.*

    - *tsvector*: es una lista ordenada de lexemas que habilita a PostgreSQL a realizar búsquedas de texto.

    - Identificadores únicos universales UUID por sus siglas en inglés: los UUID solucionan muchos problemas relacionados con las bases de datos como la generación de datos fuera de línea.

    - NoSQL: soporta varios tipos de datos NoSQL incluyendo XML, Hstore y JSONB.

    - Enumeración, rango y dominio son tipos de datos definidos por el usuario con restricciones específicas, como un conjunto de valores permitidos, restricción en el rango de datos y restricción en la verificación.

- Tipos de datos compuestos que son definidos por el usuario, donde un atributo se compone de varios atributos más.

- Leguajes soportados: PostgreSQL permite que las funciones para realizar consultas o manejar la base de datos sean escritas en diferentes lenguajes de programación, la comunidad brinda soporte para los siguientes lenguajes de programación: SQL, C, Phyton, PL/pgSQL, Perl y Tcl, además existen otros lenguajes de programación soportados por contribuidores externos como Java, PHP, Ruby entre otros.

5.2.3.4. Capacidades NoSQL

Con el tiempo PostgreSQL se ha convertido en más que una base de datos relacional, actualmente soporta diferentes tipos de datos NoSQL lo cual ha brindado a los desarrolladores una herramienta confiable y flexible para el desarrollo de aplicaciones de forma ágil.

PostgreSQL soporta la notación de objetos simples Javascript o JSON por sus siglas en inglés, la cual es utilizada en el intercambio de información entre diferentes sistemas en las aplicaciones web RESTful, en el lanzamiento de la versión 9,4 de PosrgreSQL se introdujo una estructura binaria para almacenar documentos JSON denominada JSONB, este nuevo tipo de dato elimina la necesidad de analizar los documentos JSON antes de almacenarlos en la base de datos, con este cambio los documentos JSON puede ser procesados a una velocidad comparable al de las bases de datos de archivos, mientras se mantiene la compatibilidad con ACID.

Los pares atributo, valor son soportados por la extensión *hstore*, puede ser utilizada también con otros tipos de datos semi estructurados en diferentes

escenarios para reducir el número de atributos que son raramente utilizados y contienen un valor nulo.

PostgreSQL soporta diversas funciones para generar o crear documentos que utilizan el lenguaje de marcado extensible o XML por sus siglas en inglés, es un tipo de dato muy utilizado y flexible usualmente utilizado en la definición del formato de los documentos, también soporta *xpath* que se utiliza para encontrar información en un documento XML. XML es utilizado en RSS, Atom, SOAP y XHTML.

5.2.3.5. Contenedor de datos externos

En 2011 la versión 9,1 de PostgreSQL fue lanzada con el soporte solamente de lectura para el estándar de ISO/IEC 9075-9;2003 SQL sobre el manejo de datos externo o MED por sus siglas en inglés que define los contenedores de datos externos o FDW por sus siglas en inglés que permiten que las bases de datos relacionales manejen datos externos. Los FDW pueden ser utilizados para alcanzar la integración sistema federado de bases de datos, entre los FDW soportados por PostgreSQL se encuentran Oracle, Redis, Mongodb y archivos delimitados.

Desde la versión 9,3 de PostgreSQL se soporta *postgres_fdw* que es utilizado para habilitar el acceso e intercambio de información entre diferentes bases de datos de PostgreSQL, soporta las operaciones *SELECT*, *INSERT*, *UPDATE* y *DELETE* en tablas externas.

5.2.3.6. Rendimiento

PostgreSQL emplea distintas técnicas para mejorar su concurrencia y escalabilidad entre ellos se encuentran:

- Sistema de bloqueo de PostgreSQL: provee distintas formas de bloquear los datos de las tablas llegando al nivel de filas, generando así bloqueos granulares con esto se evita bloquear más de lo necesario para realizar la acción, esto incrementa la concurrencia y reduce los tiempos de bloqueo.

- Índices: PostgreSQL provee cuatro distintos tipos estos son: *B-Tree*, *hash*, *generalized inverted index* (IGN) y *Generalized Search Tree* (GiST), cada uno se aplica en un escenario en concreto. PostgreSQL soporta índices únicos, parciales o multi columna, así como índice para operaciones y operadores de clases.

- *EXPLAIN*, *ANALYZE*, *VACUUM* Y *CLUSTER*: son comandos utilizados para mejorar el rendimiento del servidor. El comando *EXPLAIN* muestra el plan de ejecución de la declaración SQL, se pueden realizar cambio en las opciones de ejecución y luego comparar los planes de ejecución antes y después de los cambios. El comando *ANALYZE* se utiliza para recolectar estadísticas sobre las tablas y las columnas, el comando *VACUUM* se encarga para la recolección de basura y reclamar espacio de almacenamiento en el disco duro. El comando *CLUSTER* es utilizado para organizar los datos de forma física en el disco duro, estos comandos pueden ser configurados basándose en la carga de trabajo de la base de datos.

- Herencia de tablas: permite la creación de nuevas tablas con la misma estructura a partir de otras, esas tablas se utilizan para almacenar subconjuntos de datos en función de ciertos criterios. esto permite una recuperación muy rápida de información en ciertos escenarios, porque solo se accede a un subconjunto de datos al responder una consulta.

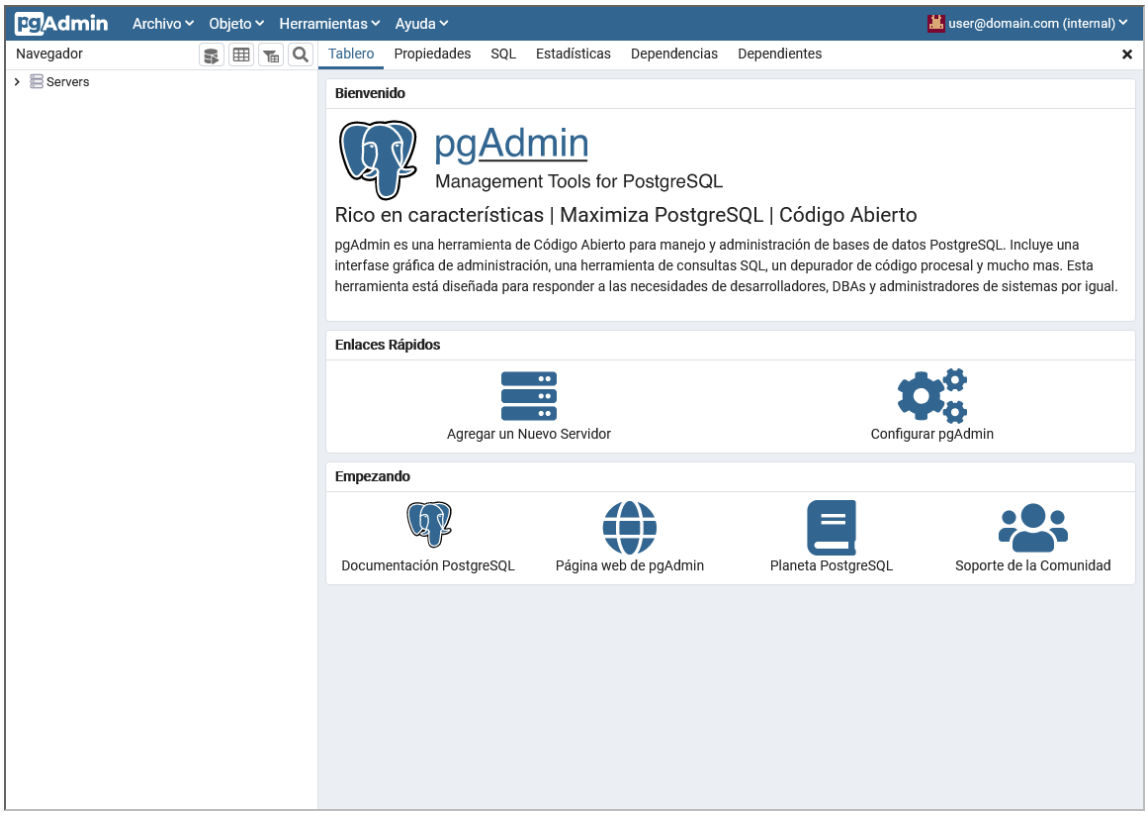
5.2.4. Instalación de PostgreSQL en sistemas ATP

Para instalar PostgreSQL en sistemas GNU/Linux que utilizan el sistema avanzado de paquetería es necesario seleccionar los siguientes paquetes:

- *postgresql-client*: es el paquete encargado de instalar el cliente para PostgreSQL se instala con el comando *sudo apt install postgresql-client*, además de una serie de programas útiles como *psql*, *clusterdb*, *createdb*, *createlang*, *createuser*, *dropdb*, *droplang*, entre otros. Para realizar la conexión a un servidor PostgreSQL utilizando *psql* se debe especificar el *host,* la base de datos, el puerto y el nombre de usuario.

- *pgadmin*: es una poderosa herramienta *frontend* se instala con el comando *sudo apt install pgadmin4*, se utiliza para el desarrollo y mantenimiento de bases de datos de PostgreSQL, a diferencia de *psql* que es una aplicación que se ejecuta en la terminal de comandos, *pgadmin* posee una interfaz gráfica que puede ser más amigable con los principiantes.

Figura 43. **Vista web de pgAdmin versión 4**



Fuente: Katakoda. *Enterprisedb.* https://www.katacoda.com/enterprisedb/scenarios/pgadmin-sandbox. Consulta: 17 enero 2022.

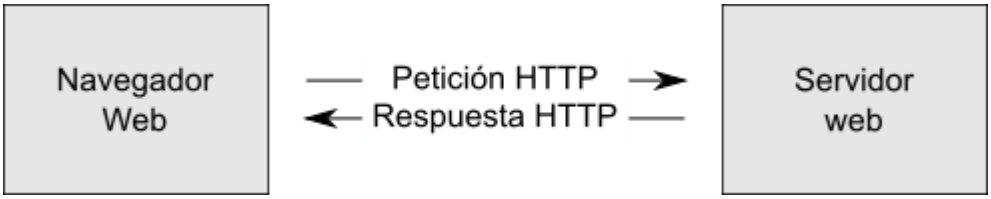
- *postgresql*: es el proceso del servidor se instala con el comando *sudo apt install postgresql*, durante el proceso de instalación se le muestra al usuario la ubicación de los archivos de configuración, datos, configuración de idioma, puerto de PostgreSQL, para confirmar que se ha instalado correctamente se puede ejecutar el comando *pgrep – a postgres*, que muestra la ruta al ejecutable y al archivo de configuración de PostgreSQL.

5.3. Protocolo de transferencia de hipertexto

Se denomina HTTP por sus siglas en inglés, es un protocolo de comunicación sin estado, es decir, no guarda información sobre las conexiones anteriores. Fue desarrollado por el World Wide Web Consortium y la Internet Engineering Task Force, en 1991 se inicia la publicación de una serie de RFC, siendo el RFC 2616 publicado en 1999 en dónde se especifica el funcionamiento de la versión 1,1 de HTTP.

Cuando un usuario visita una página web, el navegador web se encarga de realizar una petición HTTP la cual es enviada desde el cliente hasta el servidor, cuando el servidor recibe esta petición puede interpretarla para luego enviar una respuesta. Esta respuesta puede ser simple como leer un archivo HTML o una imagen desde su disco duro y enviarla o puede ser más compleja en donde es necesario utilizar *software* que se ejecuta en el servidor para generar el contenido de forma dinámica antes de enviarlo.

Figura 44. **Petición HTTP y respuesta HTTP**



Fuente: elaboración propia, empleando Inkscape 1.0.

5.3.1. Petición HTTP

Tomando como ejemplo una petición utilizando la versión 1,1 de HTTP, esta se encuentra compuesta por cuatro partes principales: el método, la ruta de

acceso, cabecera y cuerpo. Algunas peticiones no poseen un cuerpo al ser realizadas por ejemplo al visitar un sitio web, el navegador web no enviará nada en el cuerpo de la petición en cambio al enviar información utilizando un formulario o similar, estos datos se enviarán en el cuerpo de la petición HTTP.

Como ejemplo se muestra parte de la petición utilizando la versión 1,1 de HTTP a la dirección web https://alertafuegogt.sparkgt.org, para poder obtener la página web correspondiente el navegador web envía la siguiente petición:

GET / HTTP/1.1
Host: alertafuegogt.sparkgt.org
User-Agent: Mozilla/5.0 (X11; Fedora; Linux x86_64; rv:96.0) Gecko/20100101 Firefox/96.0
Cookie: mzaghi=spaceapps

La primera línea contiene el método utilizado, la ruta web y la versión de HTTP. A partir de la segunda línea se muestran las cabeceras HTTP, estas permiten al cliente y al servidor enviar información adicional junto a una petición o respuesta.

La cabecera de la segunda línea corresponde al *host,* opcionalmente se puede agregar el puerto en caso de no hacerlo el servidor utilizará los valores por defecto como el puerto 80 para una URL HTTP o el puerto 443 para una URL HTTPS, es necesaria para que el servidor web identifique que página web o aplicación debe responder a la petición, en el caso de que existan múltiples sitios web alojados en el mismo servidor.

La tercera línea corresponde a la cabecera del agente de usuario, este es cualquier herramienta que actúa en representación del usuario en su mayoría se

refieren a los navegadores web, también pueden ser programas utilizados para depurar, desarrollar sitios web o aplicaciones. Se puede utilizar para enviar una versión optimizada de la página o aplicación dependiendo del dispositivo que se utilice para acceder a esta.

En la cuarta línea se encuentra la cabecera correspondiente a las *coockies* que son pequeñas piezas de información que un sitio web puede almacenar en el navegador del cliente para poder identificarlo o cargar las preferencias específicas del cliente cuando este vuelva a acceder al sitio web o aplicación, esta cabecera puede ser omitida por completo si en las preferencias de privacidad del navegador se ha configurado bloquearlas.

5.3.1.1. Métodos de petición HTTP

HTTP define un conjunto de métodos de petición para indicar la acción que se desea realizar para un recurso determinado, cada uno de ellos implementan una semántica diferente pero algunas características son compartidas, entre estos métodos se encuentra:

- *GET*: solicita una representación de un recurso específico. Las peticiones que usan el método *GET* sólo deben recuperar datos.

- *POST*: se utiliza para enviar una entidad a un recurso en específico, causando a menudo un cambio en el estado o efectos secundarios en el servidor.

- *PUT*: reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.

- *DELETE*: borra un recurso en específico.

El servidor puede escoger como responder a los diferentes métodos, es importante tener en cuenta que si un servidor soporta un método en particular es probable que el usuario necesite poseer ciertos permisos para realizar dicha acción, en las aplicaciones web la mayor parte del tiempo se manejan solo peticiones *GET* y *POST* para interactuar con el usuario.

5.3.1.2. Cabeceras HTTP

Una cabecera se compone por su nombre que no es sensible a las mayúsculas seguido de dos puntos y su valor, en este no deben de existir saltos de línea.

En el pasado se podían agregar cabeceras propietarias utilizando el prefijo ¨X-¨, pero esta práctica se encuentra obsoleta desde julio de 2012 en el RFC 6648 debido a los problemas cuando se estandarizaron los campos no estándar, en el RFC 4229 se encuentran definidas otros tipos de cabeceras.

Las cabeceras pueden agruparse de acuerdo con su contexto en:

- Cabecera general: son las que se aplican tanto a peticiones como a respuestas, pero no tienen relación con los datos que se transmiten en el cuerpo.

- Cabeceras de consulta: contienen información sobre el contenido que se obtendrá o sobre el cliente.

- Cabeceras de respuesta: contienen información sobre el contenido, como su origen o datos del servidor (nombre, versión, entre otros).

- Cabeceras de entidad: contienen información sobre el cuerpo de la entidad, como su tamaño o su tipo MIME.

5.3.2. Respuesta HTTP

La respuesta de una petición HTTP resulta ser muy similar al ejemplo anterior, posee tres partes principales, el estatus, las cabeceras y el cuerpo, utilizando como ejemplo una respuesta exitosa de HTTP 1,1 el cliente recibiría lo siguiente:

HTTP/1.1 200 OK
server: GitHub.com
Content-Length: 18132
content-type: text/html; charset=utf-8
Set-Cookie: mzaghi=spaceapps

<!DOCTYPE html><html lang=¨es¨><head>…

La primera línea contiene la versión de HTTP, un código numérico de estado seguido de una pequeña descripción de lo que el código significa, en las líneas 2 a la 5 se encuentran las cabeceras de forma similar a como se encuentran en la petición.

La cabecera de la segunda línea es lo opuesto a la cabecera del agente de usuario, en esta el servidor le indica al cliente cual es el *software* utilizado en su petición, en la tercera línea se refiere al tamaño de la entidad del cuerpo en *bytes*,

la cuarta línea se refiere el tipo de datos que se envían, con esto el cliente puede decidir cómo mostrar dichos datos, en la quinta línea se muestra un ejemplo de cómo el servidor envía una *cookie* al cliente. Después de las cabeceras se deja una línea en blanco antes del contenido del cuerpo de la respuesta.

5.3.2.1. Códigos de estado de respuesta HTTP

El código numérico indicado en el ejemplo anterior recibe el nombre de código de estado de respuesta HTTP, las respuestas se agrupan en cinco clases:

- Respuestas informativas 100-199: sirve para indicar cambios en el protocolo del servidor o que más datos son requeridos.

- Respuestas satisfactorias 200-299: indica el correcto manejo de la solicitud en el servidor.

- Redirecciones 300-399: un código en este rango indica que la página web solicitada ha sido trasladada a otra dirección web, cuando se envía una respuesta de redirección el servidor incluye la cabecera *Location* que contiene la URL a la que debe ser redirigido el cliente.

- Errores de los clientes 400-499: indica que la petición no pudo ser procesada debido a un problema con la petición que ha enviado el cliente.

- Errores de los servidores 500-599: indica un error del lado del servidor, el cliente no puede ajustar los datos de su petición para solucionar estos problemas.

5.4. Django

Es un *framework* utilizado en el desarrollo web de alto nivel, se encarga de gran parte de las complicaciones del desarrollo web, su desarrollo inicio entre los años 2003 y 2005 por un grupo de programadores responsables de utilizar Python para crear aplicaciones en el periódico Lawrence Journal-World, luego de crear una cantidad considerable de sitios web se percataron que reutilizaban muchas partes de código y patrones de diseño.

Este código repetitivo evolucionó en lo que se convertiría en *framework* genérico de desarrollo web que posteriormente fue liberado como un proyecto de código abierto con el nombre Django en julio de 2005, su curva de aprendizaje no es demasiado elevada, entre las características del *software* que resulta ser producto de Django se puede enumerar las siguientes:

- Completo: sigue la filosofía de ¨baterías incluidas¨ es decir que provee de todo lo necesario para el desarrollo web por defecto, sigue consistentes principios de diseño y cuenta con una amplia y actualizada documentación.

- Versátil: se puede utilizar para construir cualquier tipo de sitio web, desde *wikis*, sistemas de manejo de contenido, redes sociales, sitios de noticias, entre otros. Es compatible con cualquier *framework* del lado del cliente y puede proveer su contenido en casi cualquier formato incluyendo HTML, JSON, XML, entre otros. Además, puede ser extendido para utilizar otros componentes de ser necesario.

- Seguro: ayuda a los desarrolladores a evitar errores comunes relacionados con la seguridad, dado que protege de forma automática al

sitio web, por ejemplo, Django provee una forma segura de manejar nombres de usuarios y contraseñas evitando así errores comunes como, almacenar dichos datos en *cookies* en dónde estarían vulnerables, o almacenar la contraseña en texto plano en la base de datos en lugar de un *hash* de esta.

El *hash* de una contraseña es un valor de longitud fija que se crea a partir de enviar la contraseña a través de una función criptográfica, Django puede verificar si la contraseña ingresada es correcta comparando el *hash* generado por la contraseña ingresada con el *hash* almacenado en la base de datos, debido a que esta es una operación en un solo sentido, incluso si un atacante obtiene acceso al valor de un *hash* almacenado en el servidor, le será difícil obtener el valor real de la contraseña.

Por defecto Django posee protección contra algunas vulnerabilidades como la inyección SQL, *scripts* entre sitios, falsificación de solicitudes entre sitios y *clickjacking*.

- Escalable: Django utiliza componentes basados en la arquitectura *shared-nothing* se refiere a que cada parte de la arquitectura es independiente de las demás y puede ser cambiada o reemplazada por otra. Tener una clara separación entre sus diferentes partes significa que su capacidad puede ser escalada agregando *hardware* en cualquier nivel como servidores caché, servidores de bases de datos o servidores de aplicaciones.

- Mantenible: el código de Django se encuentra escrito siguiendo principio y patrones que incentivan la creación de código reutilizable y de fácil mantenimiento, hace uso del principio ¨no te repitas tú mismo¨ o DRY por sus siglas en inglés.

- Portable: al estar escrito en Python puede ser ejecutado en diversas plataformas y el desarrollador no se encuentra atado a una plataforma en específico, muchos proveedores de servicios en la nube poseen infraestructura dedicada para el alojamiento de sitio web que utilizan Django.

Los *frameworks* utilizados para el desarrollo web pueden ser clasificados según la libertad que dan a los programadores de utilizar sus elementos de forma creativa o si deben seguir ciertos consensos establecidos previamente, son los llamados *frameworks* dogmáticos o no dogmáticos.

Los *frameworks* dogmáticos son aquellos que opinan sobre la manera correcta de realizar una tarea en particular, ofrecen soporte realizar un desarrollo rápido enfocándose en un área en específico, cada acción se encuentra bien documentada, sin embargo, pueden ser menos flexibles para resolver problemas que se encuentren fuera de su área de dominio principal y ofrecen menos opciones para seleccionar que componentes y enfoques pueden usarse.

Los *frameworks* no dogmáticos no poseen estas sugerencias en el flujo de trabajo, para los desarrolladores es más simple utilizar las herramientas que deseen o que mejor se adecuen a su flujo de trabajo con el coste que deben encontrar esas herramientas por ellos mismos.

Django es dogmático parcialmente intenta entregar lo mejor de ambos esquemas, proporciona un conjunto de componentes para gestionar la mayoría de las tareas de desarrollo web y una o dos maneras predefinidas de utilizarlos, sin embargo, por su arquitectura el desarrollador tiene la libertad de seleccionar entre las numerosas opciones o incluso añadir soporte para nuevas características.

5.4.1. Funcionamiento de Django

Un patrón de diseño común en el diseño de aplicaciones es el de modelo, vista y controlador o MVC por sus siglas en inglés, en dónde el modelo de la aplicación son sus datos son mostrados en una o más vistas mientras un controlador gestiona la interacción entre el modelo y la vista, Django sigue un paradigma parecido llamado modelo, vista, plantilla o MVT por sus siglas en inglés.

En los sitios web convencionales el servidor espera por una petición HTTP del navegador web, utiliza la dirección URL, si existen datos o no en el cuerpo de la petición y el tipo de petición para realizar la acción requerida. Dependiendo del tipo de petición recibida se realizará una acción de lectura o escritura en la base de datos, luego el sitio web enviará entonces la respuesta usualmente creando una página HTML al utilizar una plantilla previamente definida e ingresando en ella de forma dinámica la información solicitada.

Las aplicaciones web creadas con Django usualmente agrupa el código encargado de manejar cada uno de los pasos mencionado anteriormente en archivos separados, estos son:

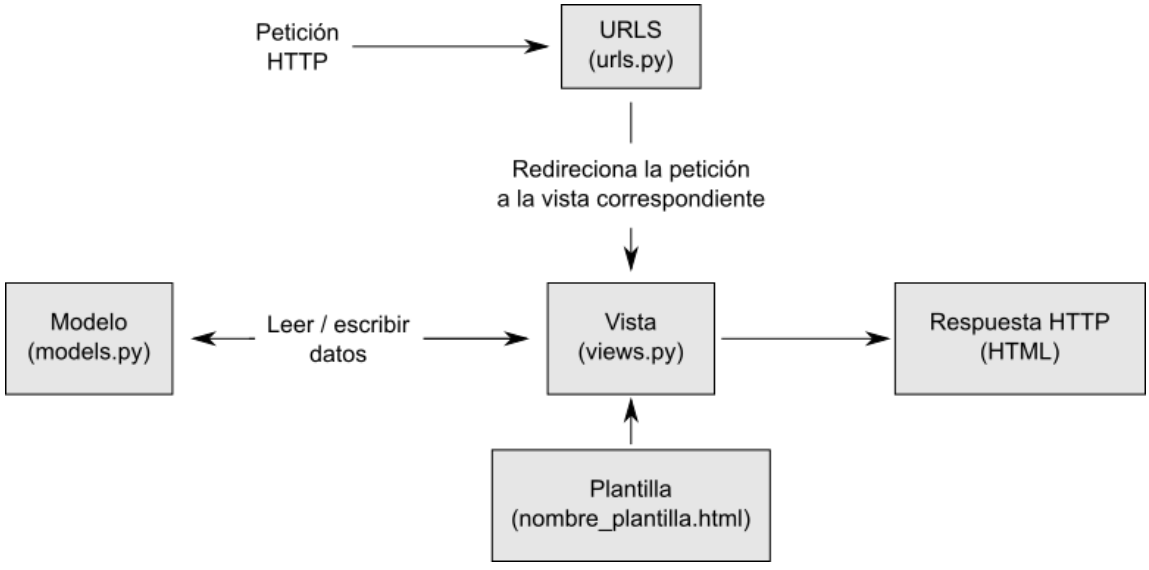
- *URLS*: es posible manejar las peticiones de todas las direcciones web de forma individual, resulta ser más simple de mantener el escribir una función de visualización para manejar cada recurso. Usualmente se realiza un mapeo de *URL* en donde se redireccionan las peticiones HTTP a la función de visualización correspondiente y si en caso existieran algunos patrones de números o dígitos en particular estos se envían a la función como datos adicionales.

- Vista: es una función que maneja las peticiones HTTP y da como resultado una respuesta HTTP, las vistas pueden acceder a los datos necesarios requeridos utilizando modelos y delegando el formato de visualización de la respuesta a las plantillas.

- Modelos: son objetos de Python que definen la estructura de los datos de la aplicación, proveen mecanismos para administrarlos y consultar registros en la base de datos.

- Plantillas: son archivos de texto que definen la estructura o disposición de un archivo como una página HTML, con marcadores que se utilizan para representar la ubicación de los datos.

Figura 45. **Diagrama del manejo de las peticiones HTTP en Django**



Fuente: elaboración propia, empleando Inkscape 1.0.

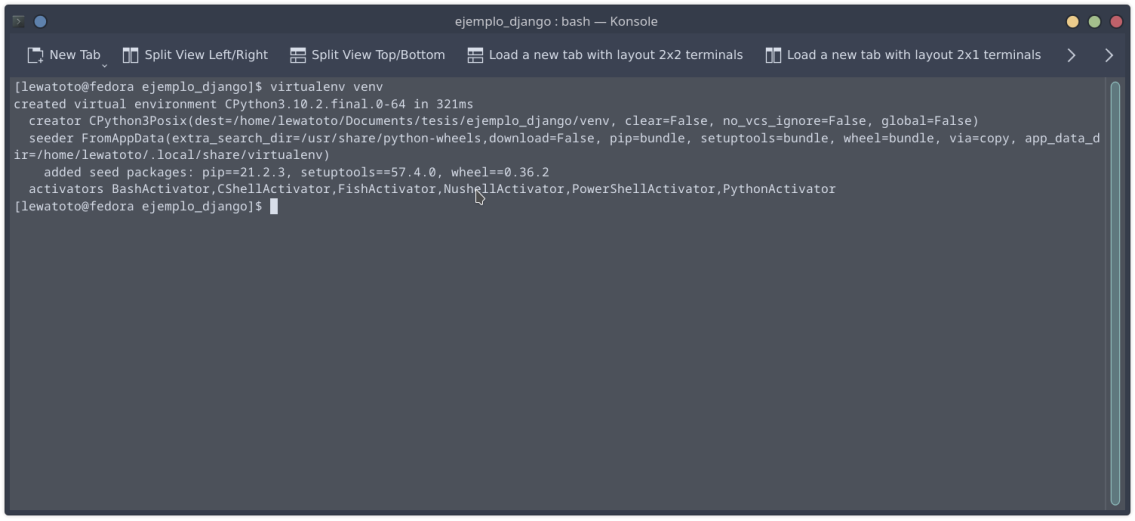
5.4.2. Instalación de Django

Para poder utilizar Django se debe instalar Python y con esto usualmente se instala la herramienta Python Package Index o *pip*, en los sistemas operativos GNU/Linux suele ser común la inclusión de Python por defecto, para evitar que exista un conflicto con las dependencias de los paquetes que se encuentran instalados por defecto con los que se necesitan instalar o se utiliza un entorno virtual en Python.

El término entorno virtual se refiere a que el intérprete de Python, las librerías, los scripts son instalados de forma aislada de otros entornos de ejecución y por lo tanto también de cualquier librería instalada por defecto en el sistema con esto se evitan los posibles conflictos que puedan existir, un entorno virtual es un directorio en el sistema que contiene los archivos ejecutables de Python y otros archivos que indican que es un entorno virtual. Las herramientas de instalación comunes como *setuptools* y *pip* funcionan con normalidad en los entornos virtuales.

Para instalar el paquete que nos permite crear entornos virtuales en sistemas ATP se ejecuta en la línea de comandos lo siguiente, *sudo apt install python3-virtualenv*, para crear un entorno virtual se utiliza el comando *virtualenv nombre-del-entorno-virtual*, si el comando se ejecuta con éxito se mostrará algo similar a lo siguiente, *created virtual environment CPython3.10.2.final.0-64 in 321ms* seguido de más información sobre la instalación.

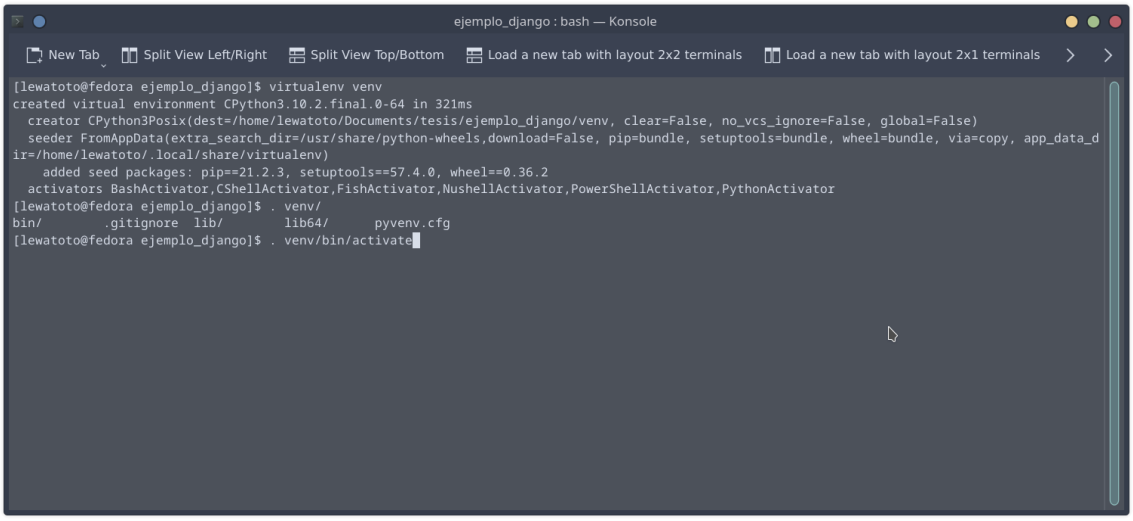
Figura 46. **Creación del entorno virtual para la instalación de Django**



Fuente: elaboración propia, empleando Konsole en Fedora 37 KDE.

Para poder utilizar el entorno virtual se debe utilizar el emulador de terminal para ejecutar el script llamado *activate* que se encuentra dentro del directorio del entorno virtual, al activarlo se mostrará entre paréntesis el nombre del entorno virtual a la izquierda del nombre de usuario y equipo en la terminal, por ejemplo: si el entorno se llama ejemplo_django, el nombre de usuario y pc es [usuario@equipo]$ al activar dicho entorno en la consola se mostrará de la siguiente forma (ejemplo_django) [usuario@equipo]$.

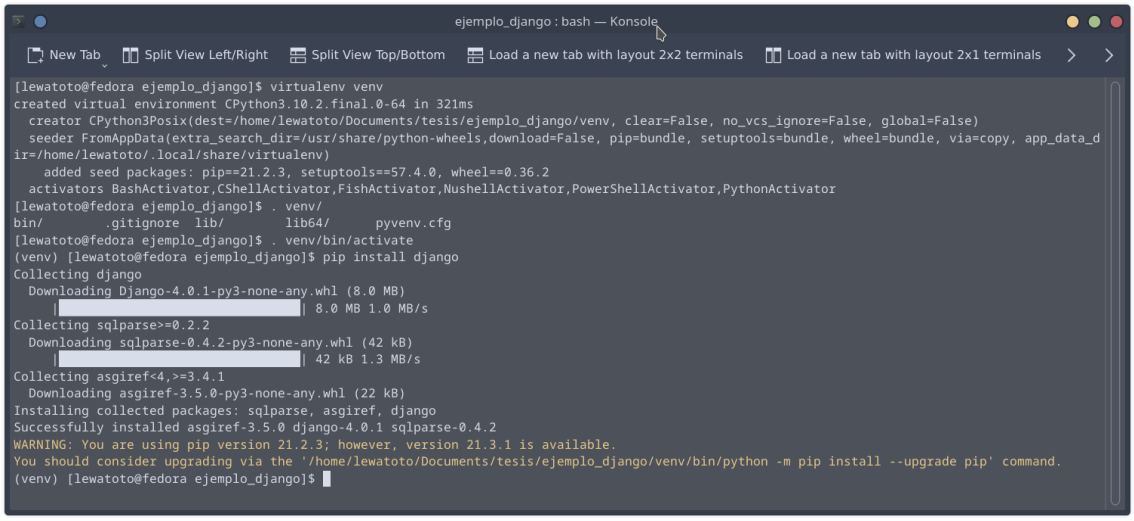
Figura 47. **Comando necesario para activar el entorno virtual**



Fuente: elaboración propia, empleando Konsole en Fedora 37 KDE.

Para instalar Django basta con ejecutar el comando *pip install django*, el cual descargará las dependencias necesarias, para comprobar la correcta instalación de Django se puede ejecutar lo siguiente *python3 -m django –version*, este regresará un texto que contiene el número de versión de Django instalada en el sistema.

Figura 48. **Instalación de Django con pip**



Fuente: elaboración propia, empleando Konsole en Fedora 37 KDE.

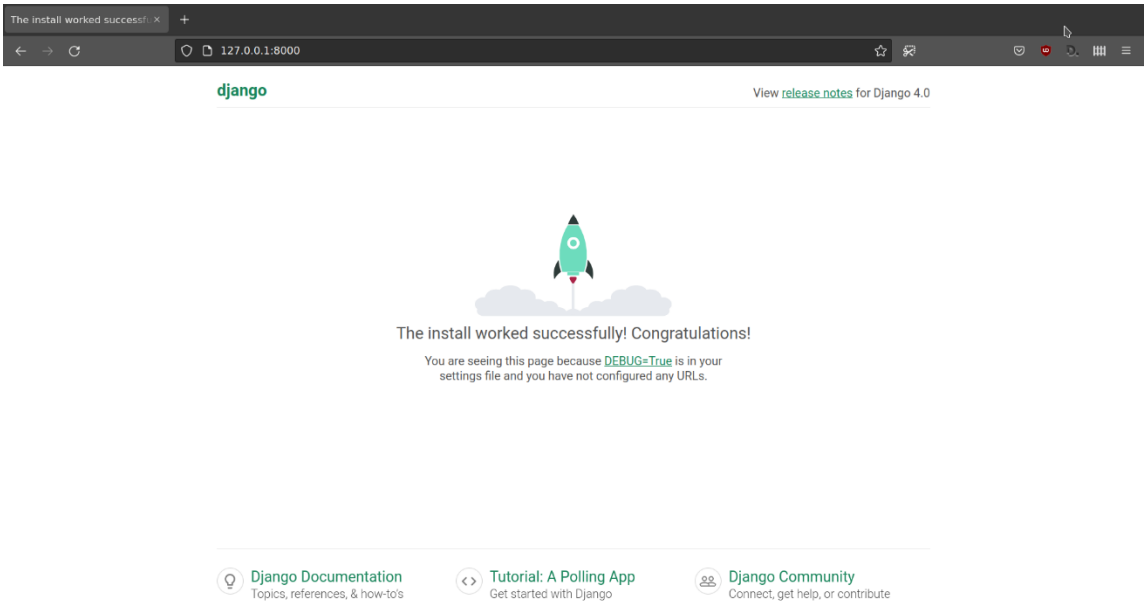
Otra forma de comprobar la instalación de Django es creando un proyecto por defecto y ejecutarlo, si algo salió mal durante la instalación se mostrará en esta prueba, para crear un proyecto por defecto se utiliza la herramienta *django-admin* seguido de la opción *startproject* y el nombre del proyecto, esto creará un directorio llamado como el nombre del proyecto con la estructura y los archivos necesarios para ejecutar Django.

Figura 49. **Estructura de los archivos y carpetas creado por django-admin**

Fuente: elaboración propia, empleando Dolphin en Fedora 37 KDE.

Para iniciar el servidor se debe ejecutar el archivo *manage.py* seguido de la opción *runserver*, el comando completo es: *python3 manage.py runserver*. Esto mostrará en la consola la dirección donde se ejecuta el servidor de Django en este caso al utilizar las opciones por defecto, la dirección del servidor será http://127.0.0.1:8000, al acceder a esta dirección desde un navegador web se mostrará un menaje diciendo que la instalación ha sido exitosa y que el modo de depuración se encuentra activo, para detener la ejecución del servidor se deben presionar las teclas control y c al mismo tiempo.

Figura 50. **Proyecto Django por defecto ejecutándose en un servidor local**



Fuente: elaboración propia, empleando Firefox 96.0.2.

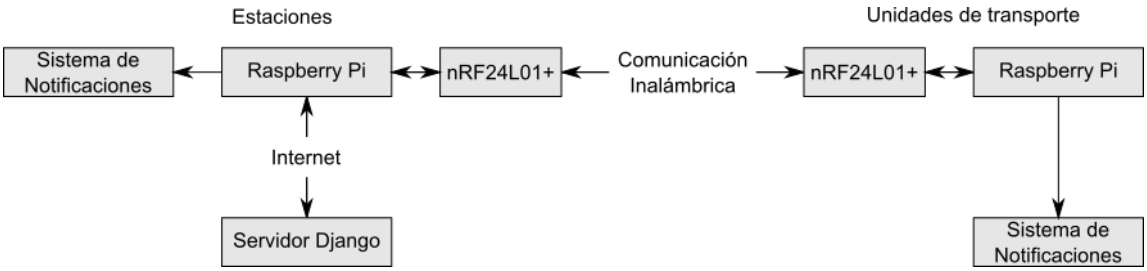
6. DESARROLLO DEL PROTOTIPO

Las herramientas ya mencionadas se utilizarán en conjunto para el desarrollo del sistema de notificación, consiste en dos partes principales el *software* en donde se incluye el servidor y los programas que se ejecutan en las Raspberry Pi del sistema y el *hardware* que corresponde a las conexiones entre los diferentes dispositivos a utilizar.

Para el desarrollo del prototipo se asumirá un escenario ficticio en el cual el sistema de transporte cuenta con 6 estaciones, de las cuales dos se encuentran en alerta por eventos externos y una se encuentra deshabilitada. Los servidores no se encuentran configurados de forma que estén listos para producción, antes de implementar este sistema en un escenario real es necesario tomar todas las precauciones correspondientes en cuanto a la configuración de los servidores.

Los archivos necesarios para el funcionamiento del prototipo, así como los archivos necesarios para la fabricación de los componentes de este se encuentran en el repositorio https://github.com/Lewatoto/Archivos_tesis en cada sección se indicará la ubicación dentro del repositorio de los archivos necesarios.

Figura 51. **Diagrama de bloques del funcionamiento del prototipo**



Fuente: elaboración propia, empleando Inkscape 1.0.

6.1. *Software*

El *software* utilizado se puede dividir en dos grupos, el que se ejecuta en el servidor, este incluye el servidor PostgreSQL utilizado para almacenar la información sobre el estado del sistema, el servidor Django que se utiliza como interfaz para realizar cambios o consultas de los datos almacenados y el que se ejecuta en la Raspberry Pi el cual se encargada de consultar los datos del estado del sistema y realizar la comunicación entre la estación y la unidad de transporte.

6.1.1. El servidor PostgreSQL y Django

En servidor PostgreSQL luego de su instalación es necesario crear la base de datos a utilizar en el proyecto, dentro del código en el repositorio del proyecto la base de datos es llamada bd_sistema_n el usuario a utilizar tiene por nombre pi y la contraseña es una generada de forma aleatoria, para la creación y configuración de esta base de datos se utilizó la interfaz web de PgAdmin.

En el servidor Django se crearon dos aplicaciones o submódulos, uno se utiliza para el manejo del acceso a las páginas y la forma en que se muestran los

datos en ellas y el otro se utiliza para manejar los datos relacionados con las estaciones.

La aplicación encargada del manejo de las páginas se encarga de manejar las peticiones HTTP que se realicen al servidor, utiliza la información almacenada en la base de datos para mostrar el estado del sistema de forma que sea entendible para los usuarios, además puede recibir peticiones de la Raspberry Pi instaladas en las estaciones las cuales responde enviando un archivo JSON que contiene toda la información del sistema que luego es procesado y enviado a las unidades del sistema.

Figura 52. **Vista de la página principal que muestra el estado del sistema**

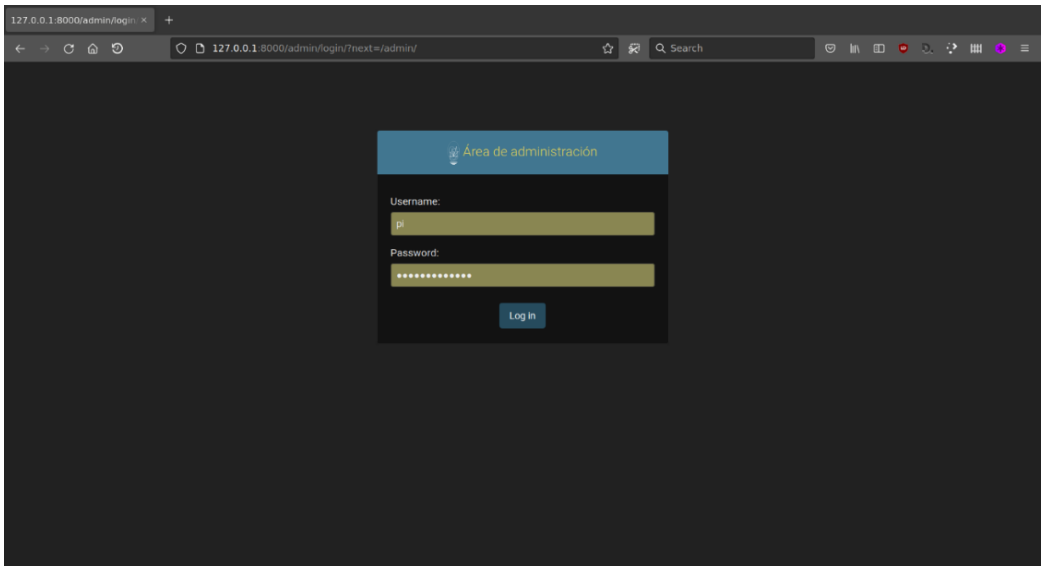
Fuente: elaboración propia, empleando Firefox 96.0.2.

La aplicación encargada de manejar la información de las estaciones utiliza un modelo que consiste en los siguientes elementos:

- *ID*: elemento incluido por defecto en la configuración del modelo.
- Nombre: correspondiente a cada estación.
- Estado: indica la situación en la que se encuentra la estación este puede tener tres valores:
    - Disponible: la estación funciona de forma correcta y no presenta ningún inconveniente.
    - Atención: la estación puede sufrir de algún inconveniente el cual puede ser provocado por un agente interno o externo al sistema.
    - No disponible: la estación se encuentra cerrada al público en general.
- Estado_extra: información adicional respecto al estado reportado.

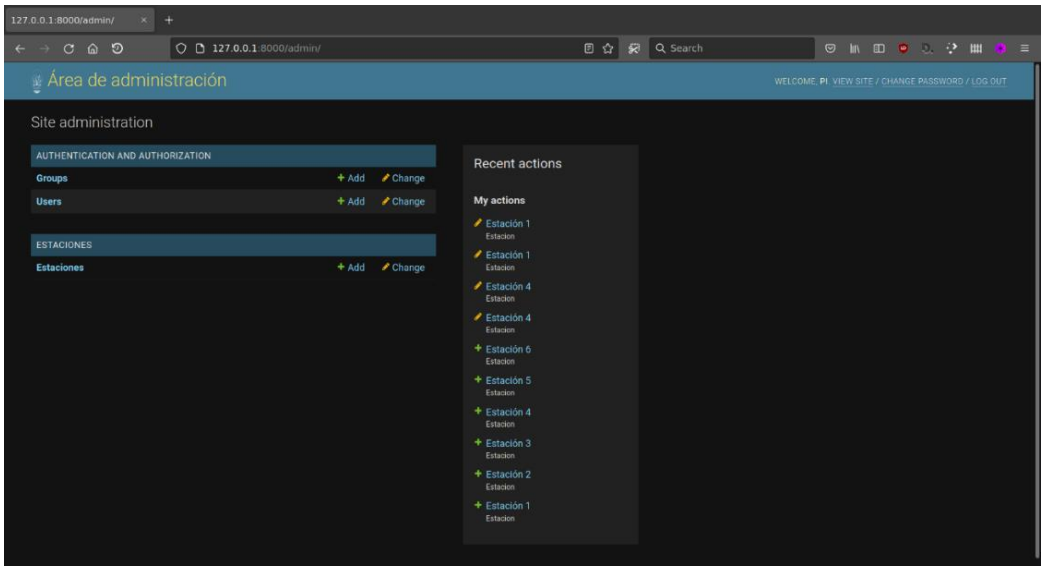
Las cuentas de usuarios, la cantidad de estaciones y los datos relacionados a estas pueden ser modificados desde el panel de administración de Django, por defecto el acceso a esta área administrativa se encuentra en la página */admin* de la dirección del servidor utilizado.

Figura 53. **Inicio de sesión del área administrativa del servidor Django**



Fuente: elaboración propia, empleando Firefox 96.0.2.

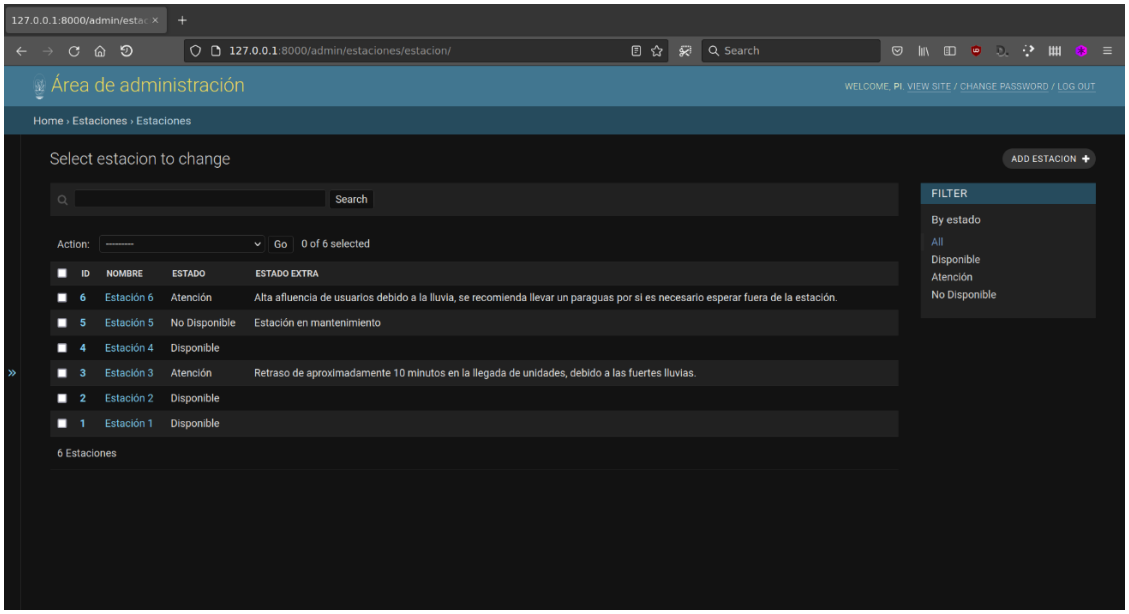
Figura 54. **Área administrativa del servidor Django**



Fuente: elaboración propia, empleando Firefox 96.0.2.

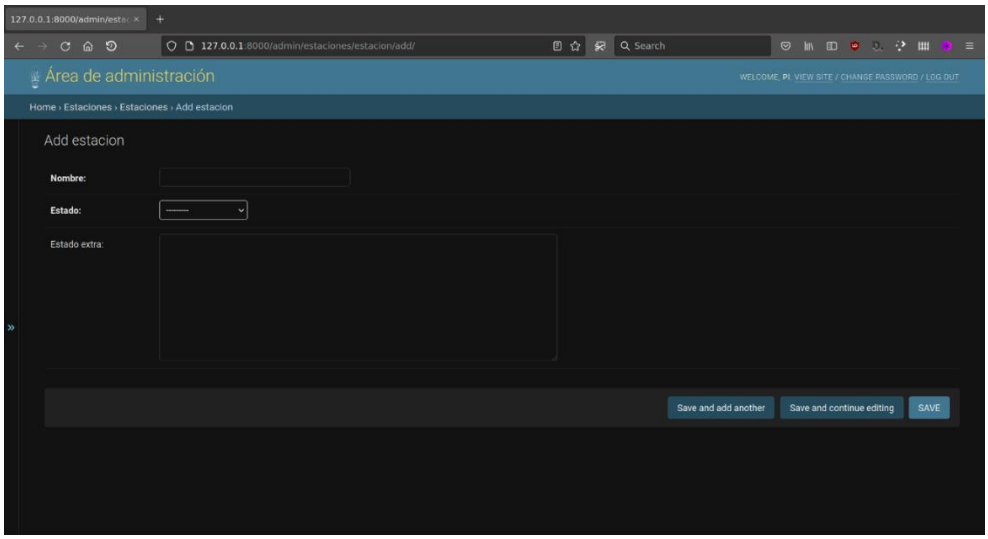
Para comprobar la información que se tiene almacenada en el sistema sobre las estaciones se puede hacer clic en el enlace Estaciones, dependiendo de la página en la que se encuentre el usuario la información puede ser editada o agregada utilizando los enlaces denominados *add* o *change* como se observa en la figura 54, haciendo clic en el nombre de una estación que desee modificar o haciendo clic en el botón *Add Estación* como se observa en la figura 55.

Figura 55. **Vista general de la información sobre las estaciones del sistema**



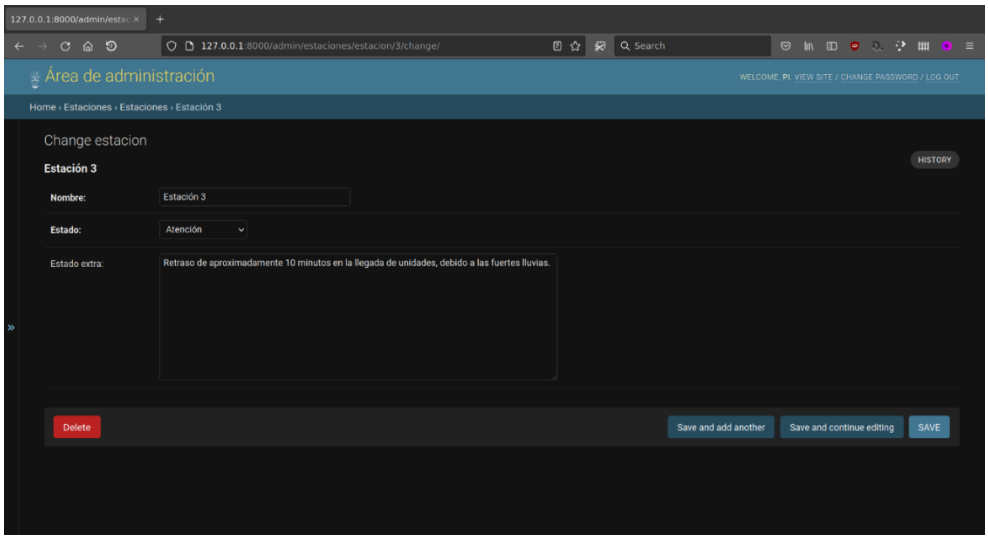
Fuente: elaboración propia, empleando Firefox 96.0.2.

Figura 56. **Página de ingreso de los datos para una nueva estación**



Fuente: elaboración propia, empleando Firefox 96.0.2.

Figura 57. **Página de edición de los datos de una estación**



Fuente: elaboración propia, empleando Firefox 96.0.2.

6.1.2. *Software* que se ejecuta en la Raspberry Pi

Para que exista una comunicación rápida y eficiente entre la Raspberry Pi y el módulo NRF24L01+ se utiliza el controlador disponible en el repositorio NRF24 junto con sus variantes NFR24Network y NRF24Mesh dependiendo de la forma en que se desee manejar la comunicación entre dispositivos.

6.1.2.1. Instalando los controladores NRF24, NRF24Network y NRF24Mesh

Estos controladores pueden ser instalados de dos formas, una es utilizando el *script* provisto en el repositorio que puede ser descargado con el comando *wget http://tmrh20.github.io/RF24Installer/RPi/install.sh*, para ejecutarlo se le deben de asignar permisos de ejecución eso se realiza con el comando *chmod +x install.sh*, al ejecutar el *script* se le preguntará al usuario si desea instalar *git* como dependencia, si se desea instalar *git* antes de ejecutar el script se puede hacer con el comando *sudo apt update && apt install git*.

Luego se le pregunta al usuario que librería desea utilizar para manejar los puertos de propósito general de la Raspberry Pi por defecto se utiliza SPIDEV que es la opción 2 por último, se deben seleccionar los componentes que se deseen instalar escribiendo y o n en cada pregunta y presionando la tecla enter para confirmar la selección, al finalizar la preguntas el *script* realizará la instalación.

Si existe un error que impide la instalación de los controladores este se mostraría en la consola y la ejecución del *script* se interrumpe, para retomar la instalación es necesario eliminar la carpeta rf24libs con todo su contenido y volver a ejecutar el *script*.

La otra opción es instalar de forma individual los paquetes que se utilizarán, estos se encuentran en el repositorio del proyecto en la carpeta de software/dependencias como submódulos del proyecto, para descargar estas dependencias al momento de clonar el repositorio existen varios métodos dependiendo de la versión de git que se utilice, por ejemplo:

- Para versiones posteriores a la 2,13 se puede utilizar el comando *git clone --recurse-submodules -j8 https://github.com/Lewatoto/Archivos_tesis*, con esto se descarga el repositorio y sus dependencias, la opción j8 se refiere a la cantidad de submódulos que se descargarán de forma simultánea.
- Para versiones 1,9 a 2,12 se utiliza el comando *git clone –recursive https://github.com/Lewatoto/Archivos_tesis*.
- Si el repositorio se descarga de forma normal y luego se desean descargar los submódulos se ejecuta el comando *git submodule update --init –recursive* dentro de la carpeta con los archivos descargados.

Para instalar los distintos módulos se ejecuta en cada una de las carpetas descargadas los siguientes comandos:

- *./config*, si se desea cambiar la librería que maneja los puertos de propósito general de la Raspberry Pi se debe agregar solo un elemento de los siguientes:
  - --*driver*=SPIDEV (opción utilizada por defecto).
  - --*driver*=MRAA
  - --*driver*=BCM2835
  - --*driver*=WiringPi
  - --*driver*=LittleWire
- *sudo make install*

El orden en el que deben ser instalados los controladores es NRF24, NRF24Network y por último NRF24Mesh.

Luego de este proceso los controladores son accesible desde cualquier programa escrito en C++, para facilitar su uso es necesario instalar el adaptador de Python, debido a que la versión de Python 2,x ha dejado de recibir soporte desde el 1 de enero de 2020 las instrucciones se enfocarán en la versión 3 de Python, las dependencias necesarias se instalan con los siguientes comandos:

- *sudo apt install python3-dev libboost-python-dev python3-pip python3-rpi.gpio*
- *python3 -m pip install --upgrade pip setuptools*
- *sudo ln -s $(ls /usr/lib/$(ls /usr/lib/gcc | tail -1)/libboost_python3*.so | tail -1) /usr/lib/$(ls /usr/lib/gcc | tail -1)/libboost_python3.so*

El último comando crea un enlace simbólico a la librería *boost.python* necesario para el proceso de instalación, el orden en que deben ser instalados los adaptadores Python es el mismo en que se instalaron los controladores.

La ubicación del instalador del adaptador Python se encuentra en una ubicación distinta para cada componente de NRF24, las ubicaciones son las siguientes:

- NRF24: RF24/pyRF24/*setup.py*
- NRF24Network: RF24Network/RPi/pyRF24Network/*setup.py*
- NRF24Mesh: RF24Mesh/pyRF24Mesh/*setup.py*

Para instalar cada uno de los adaptadores se ejecuta el archivo *setup.py* con el siguiente comando *sudo python3 setup.py install*.

6.1.2.2. Comprobando la instalación de los controladores

Durante el proceso de instalación se muestran en la consola los posibles errores que puedan surgir por falta de dependencias, archivos no encontrados, entre otros. Para comprobar que tanto los controladores como el adaptador Python funcionan correctamente es necesario contar con dos Raspberry Pi cada una debe estar conectada a un NRF24L01+ siguiendo las indicaciones de la sección de *hardware* correspondiente a la conexión entre dispositivos, para controlar las pruebas y visualizar posibles problemas en ambas Raspberry Pi debe activar el servidor SSH para su gestión remota.

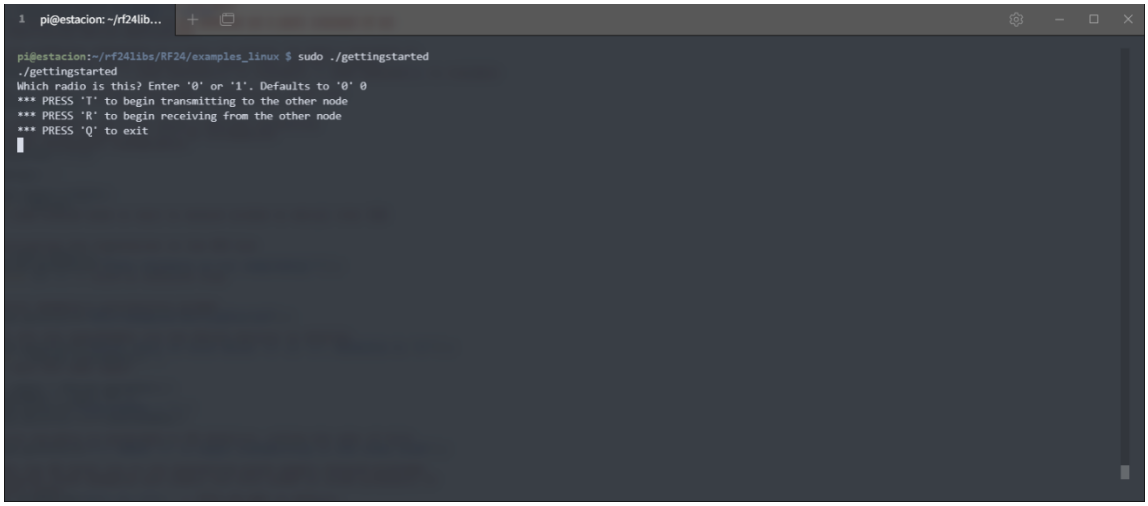
Para acceder a cada una de las Raspberry Pi se debe conocer su dirección IP dentro de la red, luego desde una consola se escribe el comando *ssh usuario@ip*, en donde el usuario es el que se utiliza para iniciar sesión e IP la dirección IP de la Raspberry Pi, luego de eso se solicitará la contraseña para poder iniciar sesión.

Las dependencias cuentan con archivos que pueden ser utilizados para comprobar el correcto funcionamiento de las conexiones y los dispositivos, estos archivos se encuentran en la carpeta *software/dependencias/NRF24/examples_linux*, en su mayoría cuentan con dos versiones una .cpp y otra .py, dependiendo del lenguaje que se desee manejar.

El archivo que contiene el programa más simple que se puede utilizar para comprobar el estado de los dispositivos y sus conexiones se llama *getting_started.py*, su funcionamiento es el siguiente:

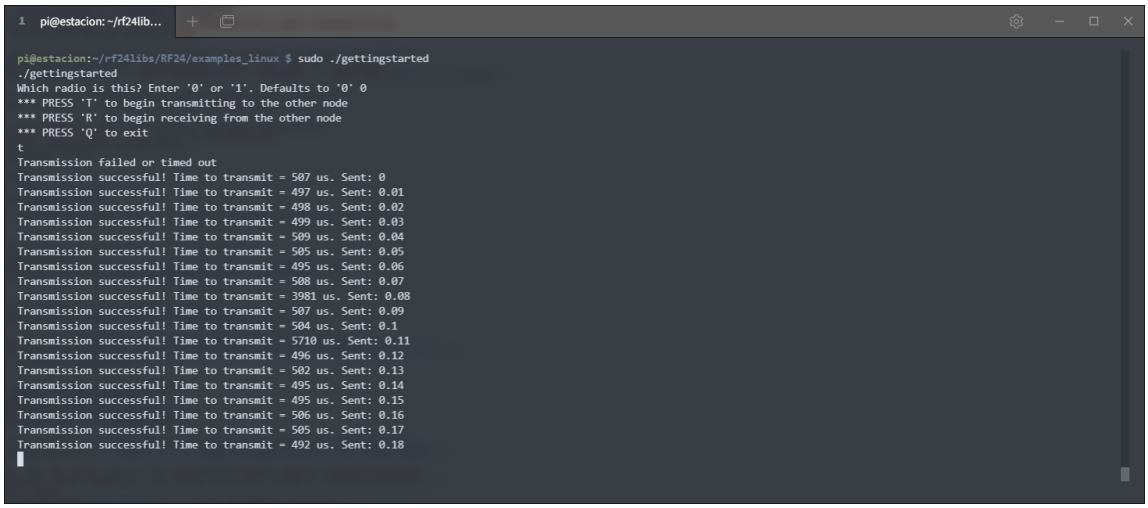
- Al ejecutarse pregunta al usuario el rol que debe asumir el dispositivo en el proceso de comunicación, T para empezar la transmisión, R para iniciar la recepción de mensajes o Q para salir.
- Para que funcione correctamente un dispositivo debe ser configurado como transmisor y el otro como receptor.
- Si existe un error se indicará en la consola con el mensaje, transmisión fallida para el dispositivo con el rol de transmisor o tiempo de espera agotado para el dispositivo con el rol de receptor.
- Cuando se conecten ambos dispositivos se mostrarán los siguientes mensajes:
    - En el transmisor: ¡transmisión exitosa! Tiempo para transmitir = xx us. Enviado: x.
    - En el receptor: se recibieron 4 *bits* en el FIFO x: x
    - Las equis corresponden al tiempo en microsegundos que le toma al mensaje ser transmitido, al número que es enviado, el FIFO en el cual se recibió la carga y el número que es recibido.
- Para detener el funcionamiento del programa se puede presionar Q o Ctrl+C para forzar la detención.

Figura 58. **Selección del rol del dispositivo en la ejecución del archivo getting_started.py**



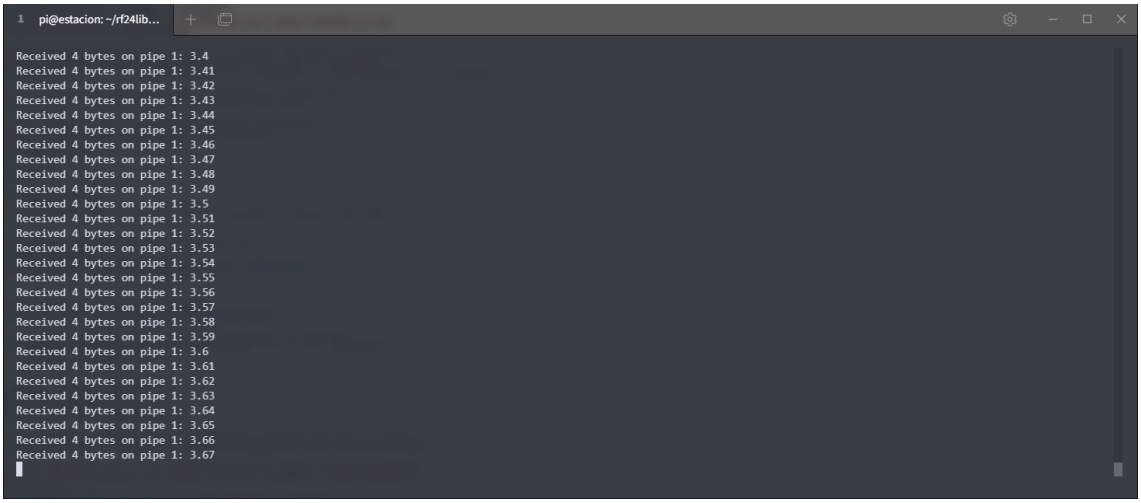
Fuente: elaboración propia, empleando Konsole en Fedora 37 KDE.

Figura 59. **Ejemplo de los mensajes de consola del rol de transmisor en la ejecución del archivo getting_started.py**



Fuente: elaboración propia, empleando Konsole en Fedora 37 KDE.

Figura 60. **Ejemplo de los mensajes de consola del rol de receptor en la ejecución del archivo getting_started.py**



Fuente: elaboración propia, empleando Konsole en Fedora 37 KDE.

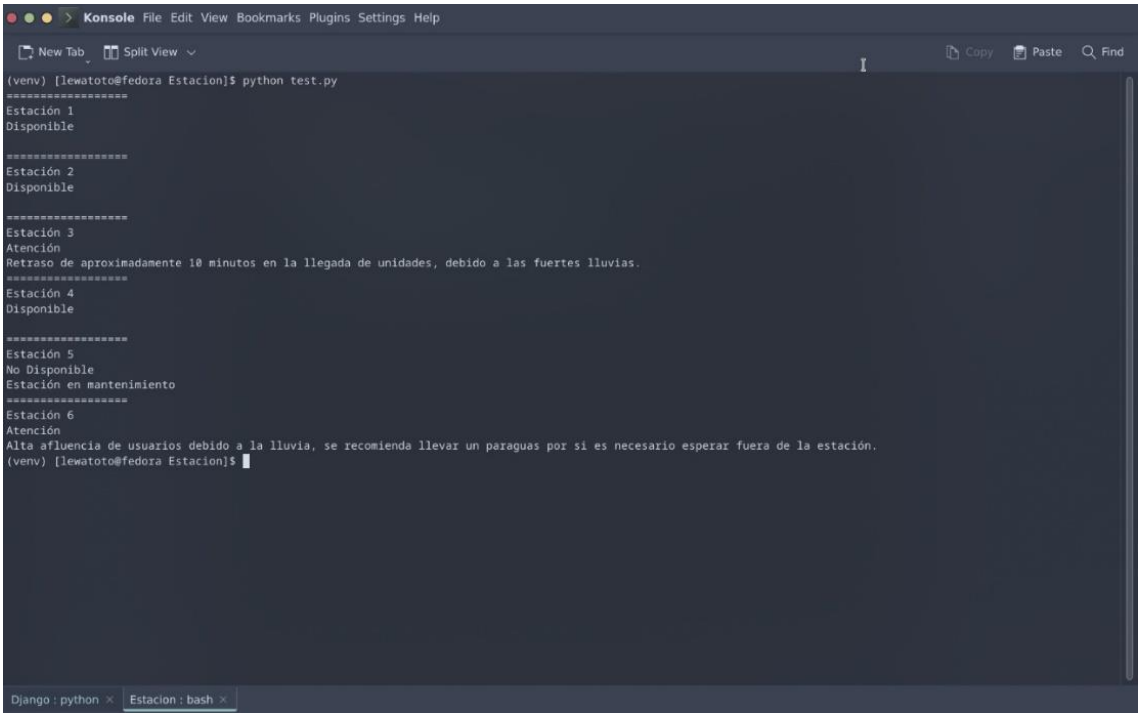
6.1.3. Funcionamiento en del programa para sincronizar datos entre estación y unidad de transporte

El funcionamiento general de sistema se sincronización de datos se puede resumir de la siguiente forma:

- El dispositivo en la estación realiza la consulta al servidor Django en un intervalo predeterminado para obtener la información del sistema, esta información se almacena de forma local.
- El dispositivo de la estación codifica la información minimizando así la cantidad de información que es enviada entre dispositivos.
- El dispositivo de la estación se mantiene en modo de escucha permanente, esperando la interacción de algún dispositivo en las unidades de transporte.

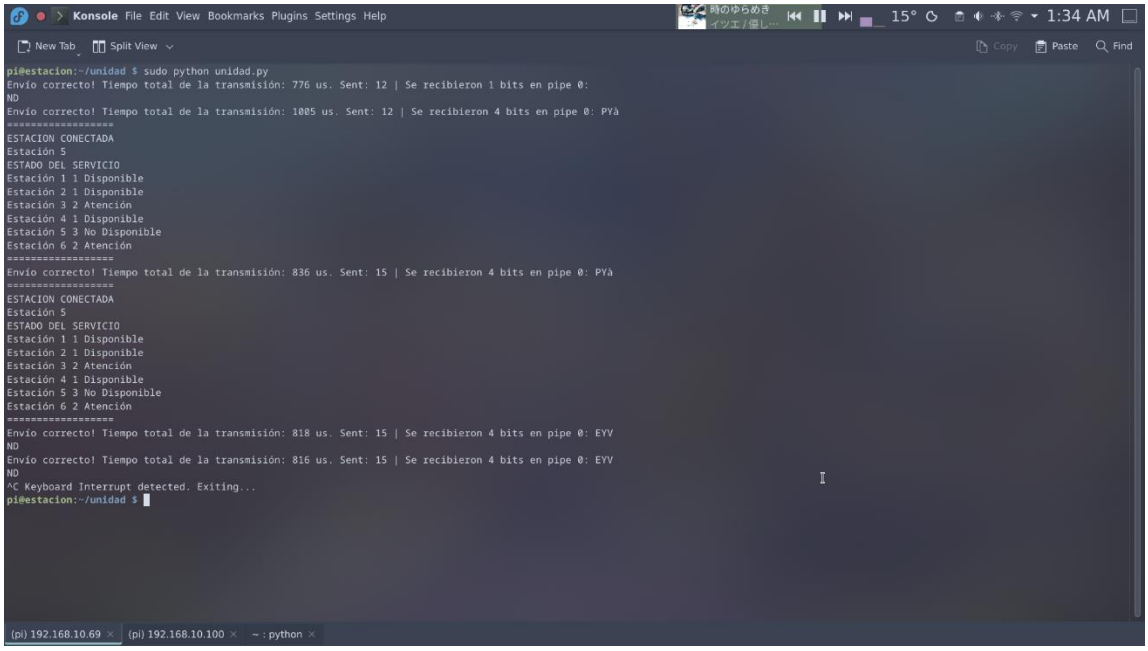
- Cuando una unidad de transporte establece un canal de comunicación con la estación, la primera le envía su información y recibe como respuesta el estado actual del sistema.
- Luego de recibir la información la unidad la decodifica y si la información recibida pertenece a la ruta establecida de la unidad, esta se almacena junto con los datos de la estación de la cual se obtuvo.
- Los datos obtenidos son mostrados en la consola o en su defecto en el dispositivo de visualización designado para mostrar la información de la ruta.
- Si la unidad vuelve a recibir información de la estación más reciente visitada, la omite hasta encontrar una estación diferente.
- El proceso se repite para cada estación del sistema.

Figura 61. **Consola de la estación mostrando la información del sistema obtenida del servidor**



Fuente: elaboración propia, empleando Konsole en Fedora 37 KDE.

Figura 62. **Consola de la unidad mostrando la información obtenida de la estación y mensajes extra de depuración**



Fuente: elaboración propia, empleando Konsole en Fedora 37 KDE.

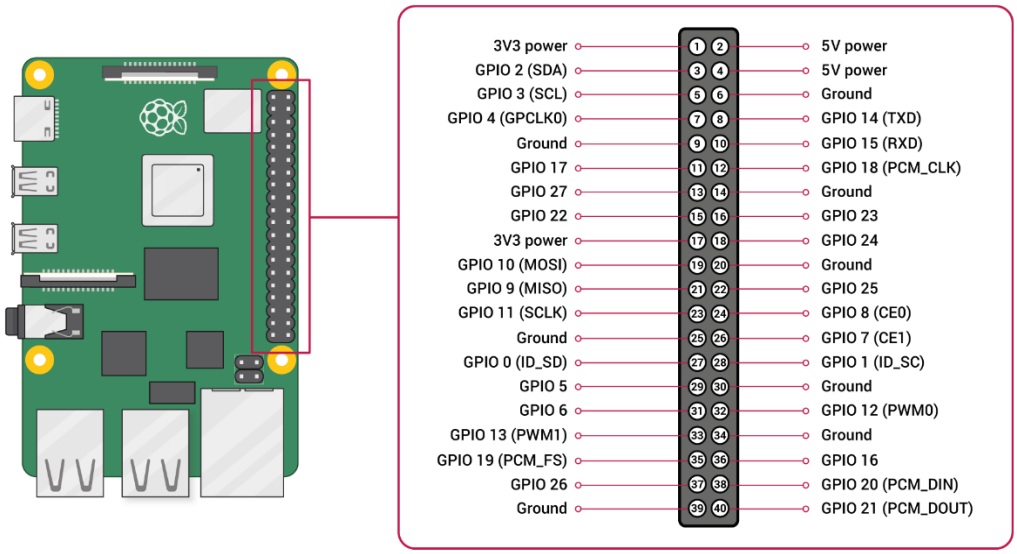
6.2. *Hardware*

Para realizar la conexión entre la Raspberry Pi y el módulo NRF24L01+ se utilizará los pines de propósito general que posee la Raspberry Pi, estos se componen de un conector de 40 pines ubicado en la parte superior de la placa de circuito impreso de cada Raspberry Pi, cada uno de los pines exceptuando a los asignados para la alimentación de 5, 3,3 V y *GND* pueden ser configurados como entrada o salida.

Los pines poseen una lógica de 3,3 V es decir el voltaje máximo que pueden tolerar es de 3,3 voltios, los pines se pueden agrupar dependiendo de su función

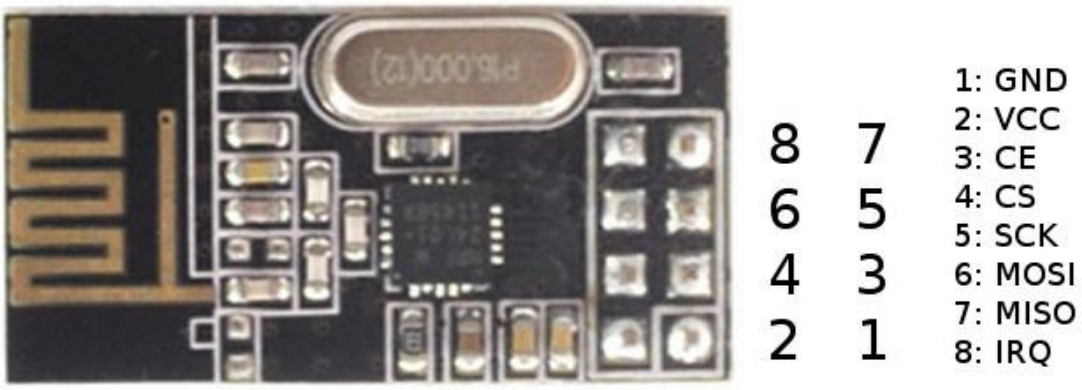
principal, por ejemplo, pines destinados a la comunicación utilizando protocolo SPI, I2C o serial, así como la posibilidad de utilizar modulación de ancho de pulso o PWM por *software* en todos los pines y por *hardware* en los pines GPIO12, GPIO13, GPIO18 y GPIO19.

Figura 63. **Disposición de los pines de propósito general en una Raspberry Pi**



Fuente: Raspberry Pi. *Disposición de pines.* https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#gpio-and-the-40-pin-header. Consulta: 16 febrero 2022.

Figura 64. **Disposición de los pines en el módulo NRF24L01+**



Fuente: https://nrf24.github.io/RF24/. Consulta: 16 febrero 2022.

6.2.1. Diagramas esquemáticos de las conexiones entre los dispositivos

El diagrama completo de conexión se encuentra en el repositorio del proyecto en la carpeta *hardware/PCB* con el nombre Diagrama_conexion_raspberry_nrf24.pdf. La comunicación entre los dispositivos utilizará el protocolo SPI, siguiendo el orden indicado en la tabla XIV.

Tabla XIV. **Asignación de pines en la conexión entre la Raspberry Pi y el módulo NRF24L01+**

Pin	NRF24L01	Raspberry Pi	Pin conector 40 pines
1	GND	Rpi-GND	25
2	VCC	Rpi-3v3	17
3	CE	Rpi-gpio22	15
4	CSN	Rpi-gpio8	24
5	SCK	Rpi-SCKL	23
6	MOSI	Rpi-MOSI	19
7	MISO	Rpi-MISO	21
8	IRQ	-	-

Fuente: elaboración propia, empleando Microsoft Word.

Algunos modelos de los módulos NRF24L01+ poseen un amplificador de señal de bajo ruido y un amplificador de potencia para la sección conectada a la antena, debido a estas funciones extra es necesario agregar una fuente de voltaje que pueda proporcionarle más corriente que la disponible en los pines de propósito general del conector de 40 pines de la Raspberry Pi cuya corriente de salida se limita a 50 mA.

Tabla XV. **Valores de consumo de corriente eléctrica de los módulos NRF24L01+ con amplificador de señal de bajo ruido y amplificador de potencia para la antena.**

Modo de operación	Valor máximo
Modo de transmisión	115 mA
Modo de recepción	45 mA
Modo de bajo consumo	4,2 µA

Fuente: Github.io. *Valores de valores de consumo.*
https://nrf24.github.io/RF24/md_COMMON_ISSUES.html. Consulta: 16 febrero 2022.

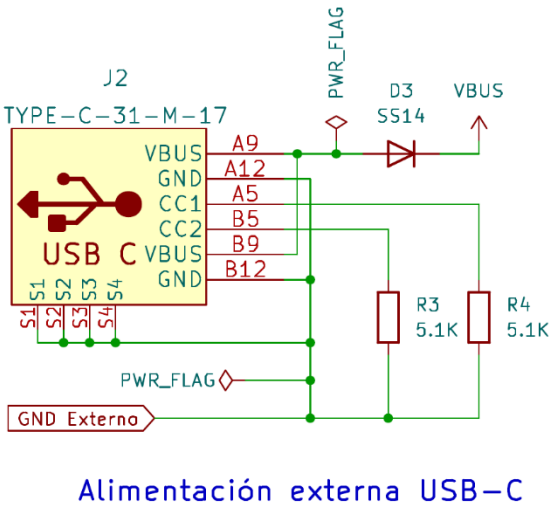
Para solucionar este problema se puede agregar una fuente externa de voltaje para el módulo NRF24L01+ y el segmento de notificación de las estaciones, esto se puede obtener al agregar un conector USB tipo C el cual cuenta con los siguientes modos de voltaje y corriente:

Tabla XVI. **Opciones de voltaje y corriente de USB tipo C**

Modo	Voltaje nominal	Corriente máxima
USB 2.0	5V	500 mA
USB 3.0 / USB 3.1	5V	900 mA
USB BC1.2	5V	1,5 A
USB Tipo C Corriente 1.5 A	5V	1,5 A
USB Tipo C Corriente 2.0 A	5V	3,0 A
USB Power Delivery	Hasta 20 V	Hasta 5,0 A

Fuente: Microchip. *Voltaje y corriente de USB.* https://ww1.microchip.com/downloads/en/appnotes/00001953a.pdf. Consulta: 16 febrero 2022.

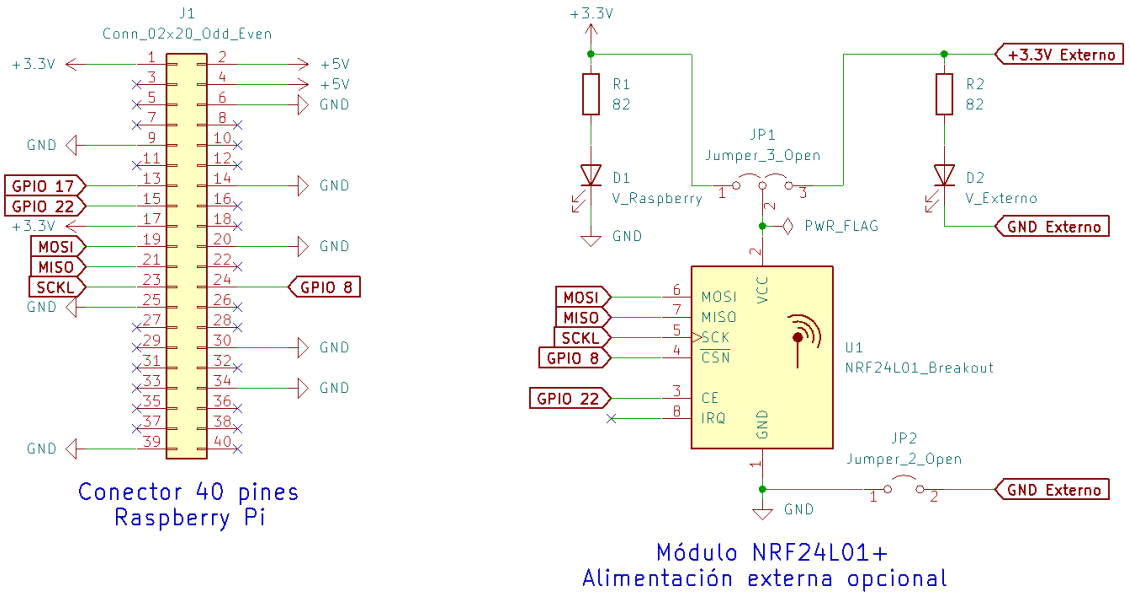
Figura 65. **Diagrama de conexión para obtener la alimentación externa utilizando un conector USB tipo C**



Fuente: elaboración propia, empleando KiCad 6.0.

Para seleccionar la fuente de voltaje para el módulo NRF24L01+ se optó por la inclusión de *jumpers* para realizar el cambio, así como la inclusión de un diodo para proteger contra errores en la conexión y diodos emisores de luz LED para indicar el origen de la fuente de voltaje.

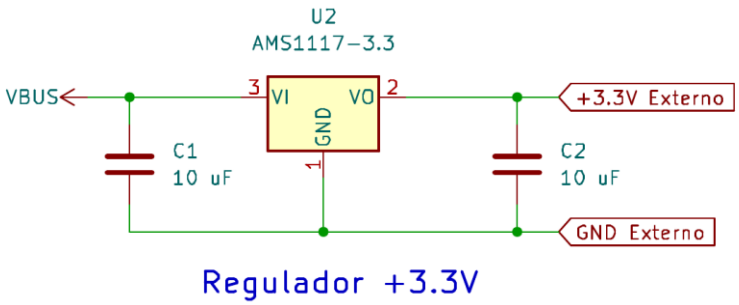
Figura 66. **Diagrama de conexión entre los pines de propósito general de la Raspberry Pi y el módulo NRF24L01+**



Fuente: elaboración propia, empleando KiCad 6.0.

Para proveer de un voltaje estable al módulo NRF24L01+ se utilizó un regulador de baja caída o LDO debido a su tamaño, su ausencia de ruido de conmutación y capacidad para proveer un voltaje de salida cercano a su voltaje de alimentación, esto con el objetivo de evitar interferencias en el funcionamiento del módulo NRF24L01+.

Figura 67. **Diagrama de conexión del regulador de voltaje con salida de 3,3 V**



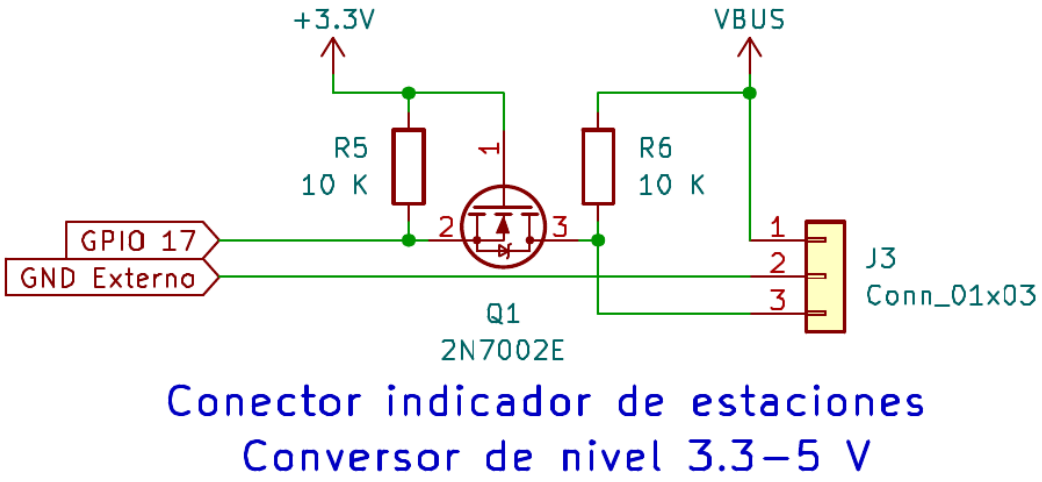
Fuente: elaboración propia, empleando KiCad 6.0.

Para realizar la comunicación entre la placa de circuito impreso y los periféricos utilizados para indicar el estado de las estaciones y demás elementos dentro del sistema se escogieron *LEDS* que utilizan el protocolo de un cable o *sigle-wire control protocol*.

Entre estos dispositivos se encuentran los modelos WS2812, WS2811 o SK6812, debido a que estos dispositivos requieren de un voltaje de operación entre 5 y 7V es necesario utilizar un conversor de niveles para evitar daños en los dispositivos.

El método utilizado para realizar la conversión de niveles es el indicado en el documento *AN10441 Level shifting techniques in I^2C-bus design* publicado el 18 de junio de 2007 por NXP Semiconductors, sugiere el uso transistores tipo MOS-FET para realizar dicha conversión, teniendo como beneficio la protección del lado de baja tensión contra los picos de voltaje que puedan ocurrir en la sección de alta tensión, el soporte de las velocidades de operación que van desde los 100 kbit/s hasta los 400 kbit/s entre otros.

Figura 68. **Diagrama de conexión del conversor de niveles para la conexión de módulos externos**

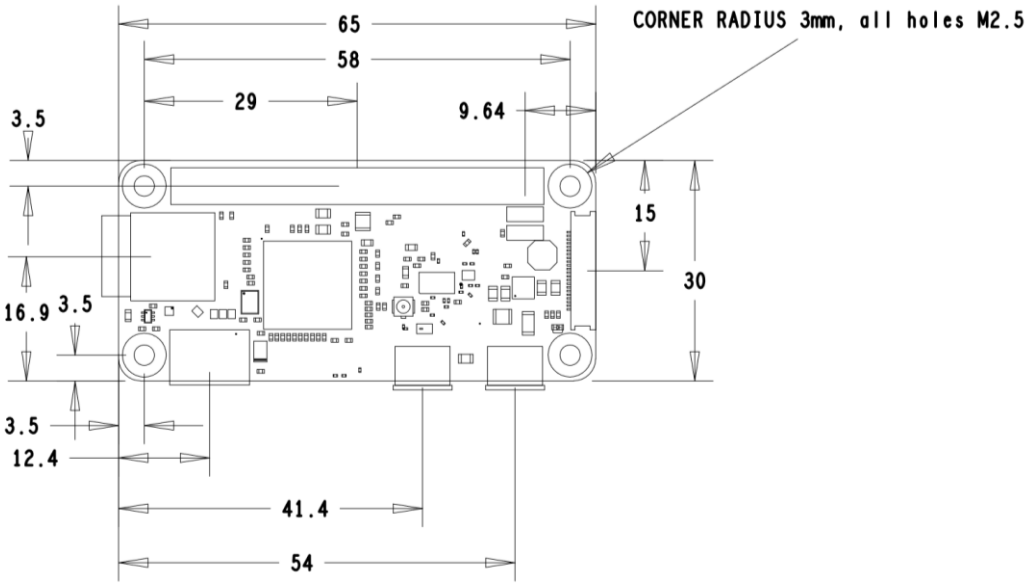


Fuente: elaboración propia, empleando KiCad 6.0.

6.2.2. Diseño de la placa de circuito impreso para el dispositivo

Para el diseño de la placa de circuito impreso se tomó en consideración el tamaño mínimo que puede tener una Raspberry Pi exceptuando la Raspberry Pi Pico, siendo este el tamaño de la Raspberry Pi Zero que posee unas dimensiones de 3,0 x 6,5 cm.

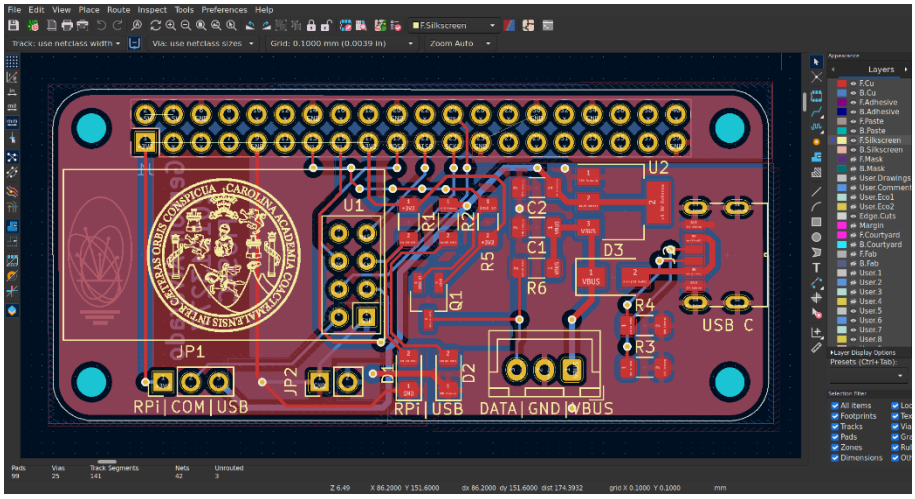
Figura 69. **Dibujo mecánico de la Raspberry Pi Zero W**



Fuente: Raspberry Pi. https://datasheets.raspberrypi.com/rpizero/raspberry-pi-zero-w-mechanical-drawing.pdf. Consulta: 16 febrero 2022.

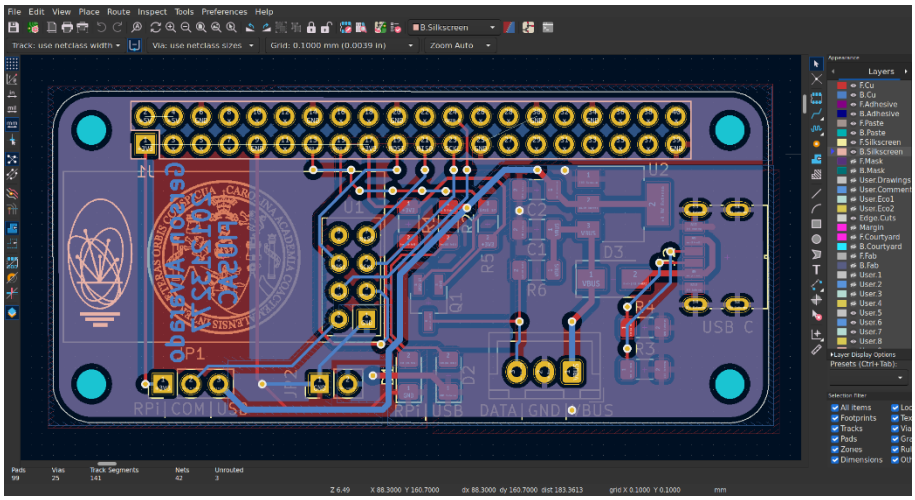
El diseño se realizó utilizando dos capas para realizar las conexiones entre los componentes, así como la conexión de las áreas vacías al punto de referencia *GND* para minimizar las interferencias y mejorar la disipación del calor que puedan generar ciertos componentes.

Figura 70. **Vista previa del diseño de la placa de circuito impreso capa superior**



Fuente: elaboración propia, empleando KiCad 6.0.

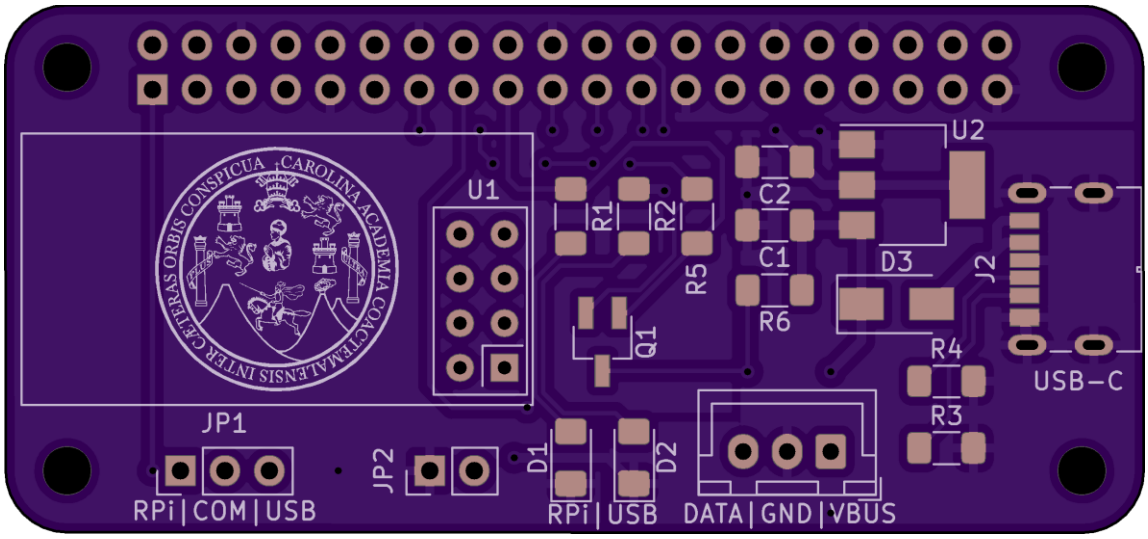
Figura 71. **Vista previa del diseño de la placa de circuito impreso capa inferior**



Fuente: elaboración propia, empleando KiCad 6.0.

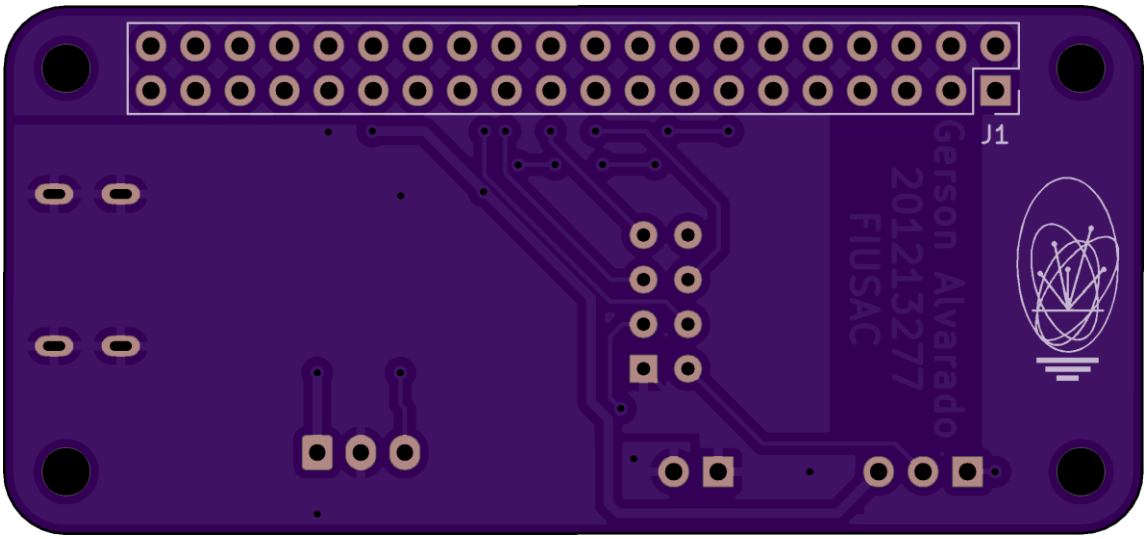
Los archivos *gerber* necesarios para realizar la placa de circuito impreso se encuentran en el repositorio del proyecto en la carpeta *hardware/PCB*. Para la fabricación se decidió utilizar el servicio de OSHPark, el cual ofrece una vista previa del resultado final de la placa de circuito.

Figura 72. **Vista previa superior de fabricación de la placa de circuito impreso**



Fuente: OSHPark. *Placa de circuito.* https://oshpark.com/. Consulta: 16 febrero 2022.

Figura 73. **Vista previa inferior de fabricación de la placa de circuito impreso**

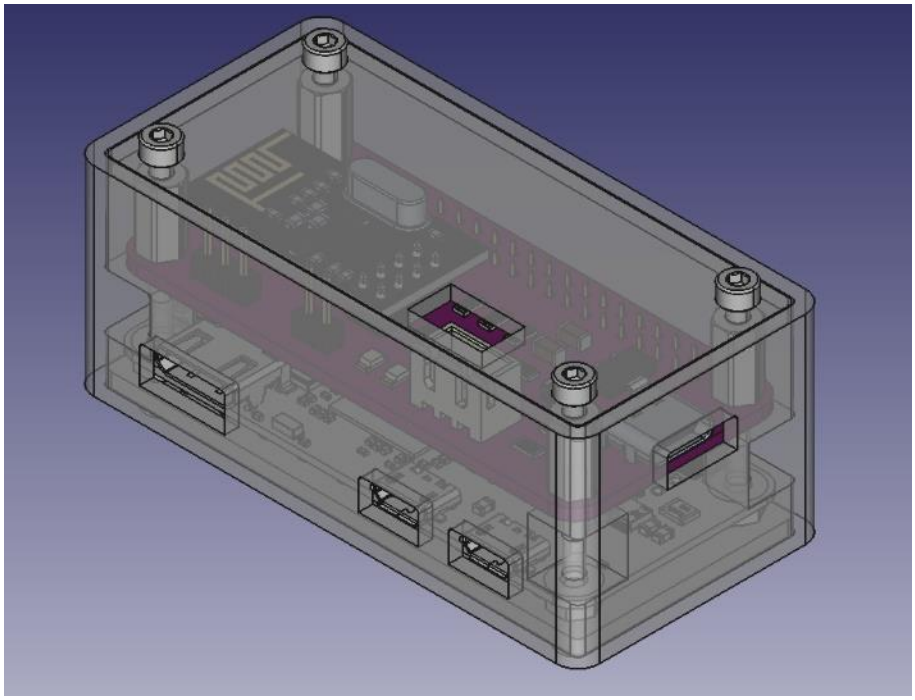


Fuente: OSHPark. *Placa de circuito.* https://oshpark.com/. Consulta 16 febrero 2022.

6.2.3. Diseño de una caja protectora para contener el dispositivo

Para contener el proyecto, protegerlo de agentes externos y montarlo en la infraestructura necesaria se diseñó una caja protectora utilizando FreeCAD, esta caja protectora puede ser impresa en 3D para disminuir costos o realizar iteraciones en el diseño. La caja protectora se compone de 3 partes siendo dos las tapas superior e inferior que pueden ser impresas en 3D o se puede utilizar una cortadora láser para fabricarlas y la tercera el contenedor principal.

Figura 74. **Vista en 3D de la caja protectora con los componentes del proyecto**



Fuente: elaboración propia, empleando FreeCAD 0.19.

CONCLUSIONES

1. Se diseñó un sistema que proporciona información de forma clara y fácil de entender sobre el estado del servicio de transporte público, cuyo uso es simple tanto para los usuarios como para los operadores de los servicios de transporte. En general, el sistema tiene un gran potencial para mejorar el servicio de transporte público.

2. El sistema diseñado facilita el acceso a la información relacionada con el estado del sistema de transporte público proporcionándole a los usuarios información en una plataforma de acceso cómodo, de manera clara y fácil de entender, permitiéndoles planificar sus viajes de manera más eficiente.

3. Al tener la base de datos centralizada, el sistema presentado facilita a los operadores la administración de la información relacionada al estado de sus servicios, debido a que deben realizar los cambios de estado una sola vez y estos se propagan al resto del sistema.

4. Se presentan las distintas opciones disponibles de sistemas embebidos que en su mayoría son de bajo coste, alta disponibilidad y al poseer licencias libres, estás permiten la innovación e invitan a la colaboración colectiva al realizar contribuciones a los proyectos originales utilizando la experiencia adquirida al realizar nuevas implementaciones con dichos sistemas.

5. Se exploran las diferentes opciones de los sistemas de diseño asistido por computadora de licencia libre para la fabricación de los componentes del sistema que son compatibles con los estándares de fabricación actual.

RECOMENDACIONES

1. Realizar una actualización de los equipos y programas informáticos utilizados en el sistema de forma periódica para evitar posibles problemas relacionados con vulnerabilidades encontradas, abastecimiento de repuestos debido a la descontinuación de componentes del sistema o la terminación del soporte y desarrollo de nuevas funciones de los programas informáticos por parte de sus mantenedores principales. Se sugiere evaluar el costo y el tiempo necesario para llevar a cabo esta actualización y asignar los recursos necesarios para llevarla a cabo en el menor tiempo posible.

2. Continuar con la investigación de posibles alternativas a los elementos utilizados en el sistema. La incertidumbre en las cadenas de suministros globales y las posibles restricciones impuestas a los fabricantes de los componentes o a los desarrolladores de programas informáticos pueden generar problemas en la adquisición de los elementos necesarios para el sistema. Se sugiere evaluar alternativas a los elementos actuales y cómo podrían ser beneficiosas para el sistema en términos de costo, disponibilidad y funcionalidad. De esta manera, se puede reducir la vulnerabilidad del sistema ante posibles cambios en el entorno externo

3. Realizar pruebas exhaustivas de transmisión en los lugares donde se instalarán los dispositivos, incluyendo diferentes condiciones climáticas. Las ondas electromagnéticas pueden ser atenuadas al atravesar medios sólidos, lo que puede causar interferencias no deseadas o pérdida de información en el sistema. Al realizar pruebas en diferentes condiciones,

se puede minimizar el riesgo de interferencias y garantizar una transmisión confiable.

4. Contar con sistemas de suministro de energía eléctrica y comunicación de respaldo para los equipos instalados en las estaciones. De esta manera, se garantiza que la información llegue a los usuarios de forma ininterrumpida.

5. Realizar un estudio luego de establecer un funcionamiento estable del sistema para determinar el origen demográfico de los usuarios del sistema y si tienen algún impedimento para obtener la información de manera óptima. El sistema deberá adaptarse para abordar estos problemas y garantizar que todos los usuarios puedan acceder a la información sin obstáculos.

6. Recopilar las opiniones de los usuarios después de cada iteración del sistema, se deben proporcionar múltiples canales para que los usuarios puedan responder. Es recomendable utilizar canales de retroalimentación que sean fáciles de usar y no consuman demasiado tiempo.

BIBLIOGRAFÍA

1. HAYKIN, Simon. Sistemas de comunicación. México. Limusa Wiley, 2006. 836 p.

2. JUBA, S., VANNAHME, A. & VOLKOV, A. *Learning PostgreSQL.* 1a ed. Reino Unido. Packt Publishing. 2015. 464 p.

3. MORÁN, A., HERRERA, A., URBINA, R. & BETHANCOURTH, R. *El transporte colectivo urbano en el área metropolitana; hacia una solución integral: informe final*. Guatemala: USAC, DIGI. 2001. 121 p.

4. Mozilla MDN Web Docs. *An overview of HTTP*. [en línea]. <https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>. [Consulta: 20 de enero 2022].

5. Mozilla MDN Web Docs. *Django Web Framework (Python).* [en línea]. <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django>. [Consulta: 23 de enero 2022].

6. Nordic Semiconductor. *nRF24L01+ Single Chip 2.4GHz Transceiver Product Specification v1.0.* [en línea]. <http://www.sparkfun.com/datasheets/Components/SMD/nRF24L01Pluss_Preliminary_Product_Specification_v1_0.pdf.>. [Consulta: 18 agosto 2020].

7. Raspberry Pi. *Raspberry Pi Documentation.* [en línea]. <https://www.raspberrypi.com/documentation/>. [Consulta: 13 de enero 2022].

8. Raspberry Pi. *Raspberry Pi Fundation Strategy 2018 – 2020.* [en línea]. <https://static.raspberrypi.org/files/about/RaspberryPiFoundationStrategy2018%E2%80%932020.pdf>. [Consulta: 13 de enero 2022].

9. SD Association. *SD Standard Overview.* [en línea]. <https://www.sdcard.org/developers/sd-standard-overview/>. [Consulta: 13 de enero 2022].

10. SHAW, B., BADHWAR, S., BIRD, A., CHANDRA, B. & GUEST, C. *Web development with Django*. 1a ed. Reino Unido. Packt Publishing. 2021. 827 p.

11. SUEHLE, Ruth y CALLAWAY, Tom. *Raspberry Pi Hacks*. O´Relly Media, 2013. 547 p.

12. UNCUYO Universidad Nacional de Cuyo. *Unidad 01: Medios de transporte urbano.* [en línea]. <http://ingenieria.uncuyo.edu.ar/catedras/u1-medios-de-transporte-urbano.pdf>. [Consulta: 13 de agosto 2020].