UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MANTENIMIENTO AUTOMOTRIZ

TEMA:

"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA BASADO EN GPS PARA EL ANUNCIO DE PARADAS EN LAS RUTAS DE LOS AUTOBUSES URBANOS DE LA CIUDAD DE IBARRA"

PLAN DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO
DE INGENIERO EN MANTENIMIENTO AUTOMOTRIZ.

AUTORES:

DARÍO JAVIER QUISHPE ENRÍQUEZ
MARCOS ANDRÉS REYES NAVARRETE

DIRECTOR:

ING. FREDY ROSERO MSC.

IBARRA, 2017

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA.

La Universidad Técnica del Norte dentro del proyecto Repositorio Digital Institucional, determinó la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la Universidad.

Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO 1			
CÉDULA DE IDENTIDAD:	100364164-2		
APELLIDOS Y NOMBRES:	QUISHPE ENRÍQUEZ DARÍO JAVIER		
DIRECCIÓN:	NATABUELA – PUENTE PEATONAL Nro. 9		
E-MAIL:	dar_javier93@hotmail.com		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0939645501

DATOS DE CONTACTO 2			
CÉDULA DE IDENTIDAD:	100359618-4		
APELLIDOS Y NOMBRES:	REYES NAVARRETE MARCOS ANDRÉS		
DIRECCIÓN:	IBARRA – PUGACHO BAJO – CASA 4-129		
E-MAIL:	andres_94_rey@outlook.es		
TELÉFONO FIJO:	062631193	**TELÉFONO MÓVIL:**	0968629653

DATOS DE LA OBRA	
TÍTULO:	"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA BASADO EN GPS PARA EL ANUNCIO DE PARADAS EN LAS RUTAS DE LOS AUTOBUSES URBANOS DE LA CIUDAD DE IBARRA"
AUTORES:	QUISHPE ENRÍQUEZ DARÍO JAVIER REYES NAVARRETE MARCOS ANDRÉS
FECHA: AAAAMMDD	2017/04/10
SOLO PARA TRABAJO DE GRADO	
PROGRAMA:	[X] **PREGRADO** **[ ] POSGRADO**
TÍTULO POR EL QUE OPTA:	INGENIERÍA EN MANTENIMIENTO AUTOMOTRIZ
ASESOR/DIRECTOR:	ING. FREDY ROSERO MSC.

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD.

Nosotros, **QUISHPE ENRÍQUEZ DARÍO JAVIER**, con cédula de identidad Nro. **100364164-2** y **REYES NAVARRETE MARCOS ANDRÉS**, con cédula de identidad Nro. **100359618-4**; en calidad de autores y titulares de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hacemos entrega del ejemplar respectivo en formato digital y autorizamos a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la Biblioteca de la Universidad con fines académicos, para ampliar la disponibilidad del material y como apoyo a la educación, investigación y extensión; en concordancia con la Ley de Educación Superior Artículo 144.

3. CONSTANCIA.

Los autores manifiestan que la obra objeto de la presenta autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y son los titulares de los derechos patrimoniales, por lo que asumen la responsabilidad sobre el contenido de la misma y saldrán en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 10 días del mes de abril del 2017.

LOS AUTORES:

Firma: .....Darío Quishpe.....

Nombre: Quishpe Enríquez Darío Javier

Cédula de Identidad: 100364164-2

Firma: .....

Nombre: Reyes Navarrete Marcos Andrés

Cédula de Identidad: 100359618-4

UNIVERSIDAD TÉCNICA DEL NORTE

CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE GRADO A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

Nosotros, **QUISHPE ENRÍQUEZ DARÍO JAVIER**, con cédula de identidad Nro. **100364164-2** y **REYES NAVARRETE MARCOS ANDRÉS**, con cédula de identidad Nro. **100359618-4**; manifestamos la voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la Ley de Propiedad Intelectual del Ecuador, artículos 4, 5 y 6, en calidad de autores de la obra o trabajo de grado denominado: **"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA BASADO EN GPS PARA EL ANUNCIO DE PARADAS EN LAS RUTAS DE LOS AUTOBUSES URBANOS DE LA CIUDAD DE IBARRA"**, que ha sido desarrollado para optar por el título de Ingeniería en Mantenimiento Automotriz en la Universidad Técnica del Norte, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Técnica del Norte.

Ibarra, a los 10 días del mes de abril del 2017.

Firma: .....Darío Quishpe.....

Nombre: Quishpe Enríquez Darío Javier

Cédula de Identidad: 100364164-2

Firma: .....

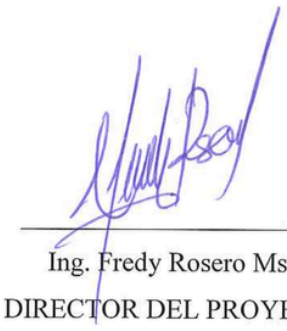
Nombre: Reyes Navarrete Marcos Andrés

Cédula de Identidad: 100359618-4

CERTIFICACIÓN

Certifico que el presente proyecto fue realizado en su totalidad por los señores: Quishpe Enríquez Darío Javier y Reyes Navarrete Marcos Andrés, como requerimiento para la obtención del título de Ingeniería en Mantenimiento Automotriz.

Atentamente,



Ing. Fredy Rosero Msc.
DIRECTOR DEL PROYECTO

AGRADECIMIENTOS

Un sincero agradecimiento a la Universidad Técnica del Norte por brindarnos la oportunidad de formar parte del alma mater y prepararnos como futuros profesionales.

De manera muy especial a la facultad por ser nuestro segundo hogar durante toda nuestra vida universitaria.

De igual forma a nuestro director de trabajo de grado, que más que un docente es considerado un amigo quien compartió sus conocimientos con nosotros.

LOS AUTORES

DEDICATORIAS

El presente trabajo lo dedico a Dios por darme la vida y otorgarme la bendición de mi familia.

A mis padres, quienes me guiaron desde mis primeros pasos hasta la actualidad, con sus consejos y amor incondicional, enseñándome a levantar después de cada tropiezo.

A mis dos hermanas, con quienes he compartido mi infancia, mis anhelos y con quien deseo cumplir mi sueño.

DARÍO JAVIER QUISHPE ENRÍQUEZ

Dedico el presente trabajo investigativo a mis hermanos que son el motor físico, mental, psicológico y espiritual en mi vida.

De igual manera a mis padres por brindarme el apoyo económico y moral para el cumplimiento de este objetivo.

MARCOS ANDRÉS REYES NAVARRETE

ÍNDICE

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

ÍNDICE DE ABREVIATURAS

ARM	Ordenador con Conjunto Reducido de Instrucciones
C.A.	Corriente Alterna
C.C.	Corriente Continua
DOS	Sistema Operativo en Disco
DSI	Controlador de Estilo Identificador
HAT ID	Interruptor Diferencial
HD	Alta definición
HDMI	Interfaz Multimedia de Alta Definición
ITS	Sistema de Transporte Inteligente
GHz	Gigahercio
GNU	No es UNIX
GPIO	Entrada y Salida de Propósito General
GPS	Sistema de Posicionamiento Global
LCD	Pantalla de Cristal Líquido
LED	Diodo Emisor de Luz
MHz	Megahercio
RAM	Memoria de Acceso Aleatorio
RTC	Reloj en Tiempo Real
UNIX	Unidad de Información y Cálculo Uniplexed
USB	Bus Universal en Serie
UTM	Sistema de Coordenadas Universal Transversal de Mercator
SBC	Computadora de Placa Única

RESUMEN

En los últimos años, la capital Imbabureña ha sufrido un gran crecimiento poblacional, lo que ha desencadenado en un aumento del número de personas que se trasladan de un lugar a otro a través del transporte público en su diario vivir. Esta situación ha provocado que las condiciones de movilidad dentro de la ciudad sean precarias conjuntamente con la falta de información que hoy en día brinda el transporte público a sus usuarios al momento de trasladarlos. En consecuencia, una alternativa que implementó el municipio simultáneamente con las cooperativas de buses de la ciudad, es la instalación de paradas inteligentes, las mismas que actualmente ya se encuentran en su fase de prueba, pero que no son suficientes para mejorar la problemática actual. En el presente trabajo de grado se narra el diseño y construcción de un sistema inteligente basado en GPS para el anuncio de rutas y paradas en los autobuses urbanos de la ciudad, el cual permite un fácil acceso y uso del transporte público a través de la ubicación en tiempo real del autobús y mediante el reconocimiento de paradas, brindar información tanto en una forma visual y auditiva a los usuarios. En el desarrollo de este sistema, la adquisición de información en cuanto a rutas de las cooperativas y las paradas de autobuses es esencial, es decir, se necesita de puntos de referencia y coordenadas para trabajar con facilidad. El sistema tangible esta comandado por un minicomputador denominado Raspberry Pi y un módulo GPS, encargados de determinar la posición, almacenar, comparar y reproducir videos sobre una pantalla led, todo esto gracias al sistema operativo Raspbian instalado sobre la terminal y Python como lenguaje de programación sobre el que se desarrolla el sistema y la interfaz en sí. Con esto se brinda una alternativa viable destinada a mejorar la movilidad urbana de los ciudadanos, hacer más eficiente el transporte público de Ibarra y disminuir la problemática actual.

ABSTRACT

In recent years, the capital Imbabureña has undergone a great population growth, which has triggered an increase in the number of people who move from one place to another through public transport in their daily lives. This situation has caused that the conditions of mobility within the city are precarious together with the lack of information that today offers the public transport to its users when transferring them. As a result, an alternative that implemented the municipality simultaneously with the city's bus cooperatives, is the installation of smart stops, which are currently in their test phase, but are not sufficient to improve the current problems. In the present work of degree is narrated the design and construction of an intelligent system based on GPS for the announcement of routes and stops in the urban buses of the city, which allows an easy access and use of the public transport through the location In real time of the bus and through the recognition of stops provide information both in a visual and aural form to the users. In the development of this system, the acquisition of information on cooperative routes and bus stops is essential, that is, reference points and coordinates are needed to work with ease. The tangible system is commanded by a minicomputer called Raspberry Pi and a GPS module, responsible for determining the position, store, compare and play videos on a led screen, all thanks to the Raspbian operating system installed on the terminal and Python as programming language On which the system is developed and the interface itself. With this, a viable alternative is offered to improve the urban mobility of citizens, to make Ibarra's public transport more efficient and to reduce the current problems.

INTRODUCCIÓN

En el presente trabajo de grado se describe el diseño y construcción de un sistema inteligente basado en GPS para el anuncio de paradas en la ciudad de Ibarra, destinado principalmente a la reproducción de contenido audiovisual en cada una de las paradas existentes en las rutas por las que circulan los autobuses. En la actualidad hay falencias en este servicio, a pesar de encontrarse en marcha el desarrollo de algunos proyectos para mejorar esta situación como las paradas inteligentes, no se logra optimizar el tema de movilidad de la ciudad ni hacer del transporte público la primera opción al momento de movilizarse.

Este trabajo de grado está desarrollado por estudiantes de la Carrera de Ingeniería en Mantenimiento Automotriz en la Universidad Técnica del Norte para beneficio de los habitantes, tomando como área de estudio del proyecto las paradas y rutas que se encuentran distribuidas sobre la ciudad de Ibarra. Esta investigación consta de cinco capítulos que se detallan a continuación:

El capítulo I asimila directamente la problemática, desde la situación actual concerniente al tema, el planteamiento, la formulación del problema, así como también el tiempo y el lugar de desarrollo. Los objetivos tanto generales como específicos y la justificación de este proyecto están formulados en dicho capítulo. Así mismo, se observa la metodología a utilizar y las técnicas, métodos e instrumentos que serán útiles durante todo el proceso de construcción de este sistema.

El capítulo siguiente de este trabajo de grado concerniente al marco teórico relata toda la información necesaria que ha sido obtenida de fuentes primarias y secundarias. En este caso, el ítem principal ha sido divido en tres sub-temas: el primero está orientado a todo lo referente a sistemas de transporte, haciendo énfasis al transporte público urbano, rutas y paradas que poseen las cooperativas de buses en la ciudad; el segundo sub-tema extrae normas jurídicas que rigen el país, empezando por la

Constitución de la República del Ecuador hasta el Plan Nacional del Buen Vivir y desde diferentes enfoques como son las personas con discapacidad y el transporte; finalmente se analiza el hardware y software necesario en el sistema, aquí son detallados los SBC encontrados en el medio y la funcionalidad que posee la Raspberry Pi. En relación al software, la recolección de información menciona los sistemas operativos y lenguajes de programación que están en capacidad de correr sobre la terminal antes nombrada.

El tercer capítulo muestra el diseño y construcción del sistema inteligente de anuncio de rutas y paradas, el cual para su desarrollo ha sido dividido en dos etapas: la primera etapa describe el diseño del hardware del sistema, aquí se precisan todos los componentes electrónicos que lo conforman, sus características y la forma en que crean conexiones entre sí para el buen funcionamiento del mismo. La segunda etapa de este capítulo manifiesta el diseño del software propio del sistema inteligente de anuncio de rutas y paradas, este medio está elaborado en Python conjuntamente con la librería Tkinter y cuenta con sub-etapas de trabajo las cuales son: encendido del sistema, inicio de funcionamiento, selección de rutas, fin de funcionamiento y apagado, siendo estas detalladas a través de flujogramas para su fácil comprensión. También se indica la comunicación entre el modulo GPS / Python y la forma de identificar las paradas. Al final los circuitos usados para la conexión de los diferentes dispositivos electrónicos son representados, ya que cada uno de ellos tiene una función específica y en conjunto cumplen con un mismo objetivo.

El capítulo IV presenta las pruebas de funcionamiento del sistema inteligente. En este, la navegación por el sistema es primordial, ya que describe cada sub-etapa de la interfaz y la función que brinda cada una, entre las cuales son: Inicio, Coop. 28 de Septiembre, Coop. San Miguel de Ibarra, Videos. Así mismo, las pruebas de funcionamiento realizadas al dispositivo se clasifican en dos: la primera narra las pruebas de funcionamiento del GPS en relación a la adquisición de datos y al almacenamiento de los mismos, parte fundamental en el desarrollo del software para el reconocimiento de las paradas de la ciudad. La segunda prueba es realizada a todo el conjunto para corroborar el correcto funcionamiento de todos los componentes y subprocesos montados y llevados a cabo en la terminal durante el reconocimiento de las paradas de la ruta especifica por la que se circula mientras trabaja el GPS en tiempo real.

Finalmente, el quinto capítulo expone las conclusiones y recomendaciones resultantes del desarrollo de este trabajo de grado, las fuentes de donde se obtuvo la información para este prototipo y anexos en donde se visualizan ilustraciones de gran importancia, el manual de uso del usuario y el código de programación.

CAPÍTULO I

1. CONCEPTUALIZACIÓN DEL PROBLEMA

1.1 Antecedentes.

Con el mundo cada vez impactado por la globalización, el desarrollo tecnológico y los avances alcanzados, la mejora en cuanto a movilidad urbana es evidente. Debido a esto, se busca optimizar la relación entre el hombre, máquina y tecnología por medio de la sistematización de los mismos, con el fin de mejorar la eficiencia del transporte público. Estos sistemas se han implementado en otras ciudades del mundo tales como: Londres, Paris, Madrid, entre otros, donde se han utilizado ITS (Sistemas de Transporte Inteligente) que básicamente cumplen la función de ofrecer información necesaria al usuario de una forma clara y sencilla, además brindan servicios adicionales como: puntos de acceso wifi, google, publicidad o venta de boletos y hacen del transporte público un medio fácil y accesible para la sociedad. (Pinto, 2012)

El Ecuador al ser un país en vías de desarrollo y con un crecimiento poblacional del 1,6% en los últimos años, busca formas de innovar la movilidad urbana de los ciudadanos a través de la incorporación de ITS como: torniquetes, validadores de tarjetas, cámaras de vigilancia, localizadores GPS, monederos, semaforización adaptiva, placas digitales, tele-peaje, estacionamientos inteligentes y parada de bus inteligente, las mismas que se encuentran o se piensan implementar en las principales ciudades del país. (Ecuador, 2015)

Ibarra no se ha quedado rezagada y actualmente cuenta con un sistema centralizado de semaforización adaptiva, un sistema de control por medio de GPS para los buses urbanos, un proyecto integral de seguridad para el transporte público y comercial implementado por el gobierno nacional y un sistema de parada de bus inteligente en fase de prueba, que en conjunto integran el ITS de la ciudad de Ibarra. La automatización de las paradas forman parte de una serie de soluciones planteadas para

mejorar el transporte dentro de la ciudad y acarrea consigo el desarrollo de un sistema inteligente para el anuncio de rutas y paradas en los autobuses urbanos, este medio es un complemento a las paradas inteligentes y está enfocado directamente a las personas con algún tipo de discapacidad visual o auditiva, turistas, habitantes de sectores rurales e indirectamente a los pobladores que residen en la ciudad. (Castillo, 2012)

1.2 Planteamiento del problema.

El Ecuador posee una población aproximada de 14,48 millones de habitantes según el último censo realizado en el año 2010, de los cuales 181 175 residen en la ciudad de Ibarra. Cabe recalcar que de esta cifra 146 570 se encuentran en rangos entre los 6 y 60 años, los mismos que pueden utilizar el transporte urbano como principal medio de movilización cotidiana dentro del cantón. De este grupo 1 372 personas tienen discapacidad visual, 1 485 poseen discapacidad auditiva, 49 319 pertenecen a parroquias rurales y 43 446 son turistas que visitan los lugares emblemáticos de la capital de Imbabura, es decir, este conjunto tiene dificultad al momento de usar el transporte masivo. (García, 2013)

La población en general tiene dificultad al momento de utilizar el servicio de transporte urbano, debido a la falta de información que existe acerca de las rutas, paradas y horarios. A pesar de la implementación de paradas de bus inteligentes y otros ITS, en la actualidad se observa falencias acerca de la información que brinda cada autobús a los usuarios, por lo que las personas que poseen discapacidad visual o auditiva, no tienen la capacidad de elegir un bus por sí mismas, viéndose obligadas a pedir ayuda a terceros o a su vez, subir al autobús incorrecto, lo mismo sucede con personas de sectores rurales, turistas e incluso propios habitantes de la zona.

1.3 Formulación del problema.

¿Cómo diseñar e implementar un sistema basado en GPS para el anuncio de paradas de las rutas en los autobuses urbanos de la ciudad de Ibarra?

1.4 Delimitación temporal y espacial.

La realización del proyecto tiene una duración de 7 meses, a partir de octubre del 2016 hasta abril del 2017, el cual se desarrolla por estudiantes de la Carrera de Ingeniería en Mantenimiento Automotriz en la Universidad Técnica del Norte para beneficio de los habitantes de la ciudad de Ibarra.

1.5 Objetivos.

1.5.1 Objetivo general.

Diseñar e implementar un sistema basado en GPS para el anuncio de paradas de las rutas de autobuses urbanos de la ciudad de Ibarra.

1.5.2 Objetivos específicos.

- Identificar cada una de las paradas autorizadas en las rutas de los autobuses urbanos de la ciudad de Ibarra.
- Determinar las coordenadas geográficas mediante grados decimales de cada una de las paradas ubicadas en la ciudad mediante el uso del programa informático Google Earth.
- Ensamblar el sistema de anuncio de paradas mediante el uso del ordenador de placa reducida Raspberry Pi un módulo GPS Glonass U-blox 7.
- Diseñar una interfaz que permita la reproducción de videos e imágenes en una pantalla mediante el lenguaje de programación interpretado Python.
- Diseñar los productos audiovisuales para el sistema.
- Efectuar pruebas del sistema de anuncio de paradas en 2 rutas establecidas dentro de la ciudad de Ibarra.

1.6 Justificación.

La ciudad de Ibarra al ser la capital de la provincia de Imbabura posee una gran afluencia de habitantes con discapacidades auditivas o visuales, turistas, pobladores de

sectores rurales y habitantes propios de la urbe. La mayoría desconoce las rutas, horarios o paradas de los buses lo que dificulta el transporte de los mismos. El cantón, al ser el centro del comercio en la zona norte del país debe adaptarse a las necesidades de la ciudad y contribuir con el desarrollo y el plan del buen vivir.

Es necesaria la existencia de un sistema que genere información útil a los usuarios con el fin de terminar con los problemas de comunicación y ayudar a los habitantes de la ciudad con la correcta utilización del autobús. Implementar un medio de comunicación auditiva y visual en el transporte público urbano que informe las paradas es indispensable, ya que la información que transmiten los buses es muy imprecisa.

La realización del trabajo de grado piensa mejorar la información brindada en el servicio de transporte dentro del cantón, cabe recalcar que sin tomar en cuenta las actuales paradas inteligentes, no se han realizado proyectos de este tipo con anterioridad. Este un sistema novedoso y nuevo que necesita de mucho análisis y estudios en la urbe, una vez implementado, este medio va a beneficiar a transportistas y usuarios que utilizan los autobuses como una herramienta de movilidad indispensable en su vida cotidiana.

1.7 Metodología de la investigación.

1.7.1 Tipos de investigación.

El presente proyecto es una investigación bibliográfica y tecnológica.

1.7.2 Investigación bibliográfica.

Al realizar este proyecto se ha de tomar en cuenta la información obtenida mediante libros, con el fin de guiar y direccionar correctamente el tema y los objetivos de este trabajo de grado, motivo por el cual se ha estudiado la Agenda Nacional para la Igualdad de Discapacidades contenida en el Plan Nacional del Buen Vivir vigente desde el año 2013.

1.7.3 Investigación tecnológica.

Para la elaboración de este proyecto, se ha visto necesario el uso de un sistema basado en GPS, que supone la implementación de recursos tecnológicos, los mismos que ameritan el estudio y análisis de su funcionamiento, programación y su correcta utilización, por ende, este proyecto es de carácter tecnológico.

1.7.4 Métodos.

Para la elaboración de este proyecto se cuenta con los métodos prácticos de: programación, diseño, mediciones, optimización y método analítico sintético.

Programación. - Se programa la plataforma de Python para la elaboración de la interfaz y recolección de datos del GPS, además se programa en MysQl para guardar los datos del GPS y compararlos.

Diseño. - Diseño y construcción del sistema basado en GPS mediante la utilización de elementos electrónicos.

Mediciones. - Obtención de datos del GPS.

Método analítico sintético. - Se aplica este método para obtener información de fuentes bibliográficas como: libros, internet, manuales.

1.7.5 Técnicas e instrumentos.

Las técnicas usadas son:

Adaptación. - Adaptación del GPS en el Raspberry Pi para que trabajen en conjunto y así obtener los datos requeridos.

Análisis de datos. - Obtención y análisis de datos almacenados en MysQl para certificar su funcionamiento y utilidad.

Pruebas de funcionamiento. - Se realiza pruebas de funcionamiento del dispositivo construido para diagnosticar la calidad de su trabajo y corregir fallas en caso de ser necesario.

CAPÍTULO II

2. MARCO TEÓRICO

2.1 Sistemas de transporte.

2.1.1 Introducción.

Movilidad cotidiana se puede definir como una serie de actividades consecutivas que se realizan en diferentes tiempos y a través del medio de transporte que sea elegido para tal acción (a pie, en transporte público o transporte privado), este conjunto de desplazamientos efectúan las personas de forma habitual para cumplir ciertas actividades en velocidades distintas como: ir al trabajo, ir de compras, estudiar, entre otras, el mismo que concluye con el retorno de los individuos hacia su lugar de residencia. (Lina Manjarrez, 2011)

La palabra transporte desciende del latín "transportare" cuyo significado es: "trans" (al otro lado) y "portare" (llevar). Entonces puede definirse como la acción de trasladar personas o cosas de un lugar a otro a cambio de una remuneración económica por su servicio. El transporte se lleva a cabo millones de veces al día en todo el mundo y supone una forma de unión entre regiones, países y continentes, esto permite el crecimiento de la economía de un país, su desarrollo y brinda: rapidez, seguridad y supone un bajo costo. (CONCEPTODEFINICION.DE, 2014)

El transporte público está dividido en tres grupos según el modo de realizarse, que son:

- Transporte terrestre
- Transporte marítimo
- Transporte aéreo

Transporte terrestre. - Es aquel que se realiza en la superficie de la tierra o bajo la misma, esto nace de la necesidad del hombre por trasladar alimentos o transportarse el mismo hacia diferentes lugares. En principios se utilizó animales, los cuales realizaban esta acción directamente, después se los manipuló para tirar de carretas, posteriormente se los operaría como transporte de varias personas a cambio de una remuneración y esto permitió un desarrollo de los caminos enfocándose en brindar seguridad y confort.

Ilustración 1 Medios de transporte terrestre
Fuente: (Autores)

Con el desarrollo de la tecnología nace el ferrocarril, un medio de transporte sobre vías que favorece en gran medida al desarrollo de la región. En 1885 aparecen los primeros vehículos que tienen como características lograr trasladar personas y mercancías. El transporte terrestre puede clasificarse según el medio en:

- Coche
- Camión
- Autobús
- Tren

Transporte marítimo. - Se define como la acción de llevar individuos o mercancía en una embarcación por medio de océanos, mares, ríos y lagos, desde un punto terrestre a otro a cambio de un beneficio económico. En la antigüedad era muy utilizado en el transporte de personas y mercancías, pero con el surgir de las aeronaves, su uso se ha limitado al transporte de bienes y constituye un factor importante dentro del comercio exterior. Según su capacidad de carga y tipo, se pueden encontrar los siguientes:

- Graneleros
- Mineraleros
- Frigoríficos
- Portacontenedores

Ilustración 2 Medio de Transporte Marítimo
Fuente: (SENA, 2014)

En la ilustración se observa una embarcación para el transporte de contenedores a través del medio fluvial, el mismo que representa un volumen comercial mundial del 2,2%, generando un desarrollo mediante el aumento de importaciones.

***Transporte aéreo.* -** Medio de transporte para personas y mercancías realizado por medio de una aeronave a cambio de una remuneración económica. Aunque a principios, el avión solo era utilizado para trasladar personas, ya con el desarrollo de la tecnología y avances en el diseño de los aviones, se ha logrado transportar mayores cargas. (Quess.la, 2016)

Ilustración 3 Medio de Transporte Aéreo
Fuente: (FACUA, 2007)

La ventaja de este medio de transporte es la rapidez con que se traslada de un lugar a otro y la seguridad que brinda, pero como desventaja se tiene que su costo de operación es muy elevado y la carga aún es muy limitada. (SENA, 2014) Este medio se lo puede clasificar según su diseño en:

- Avión
- Helicóptero

2.1.2 Transporte público urbano.

El transporte público urbano es un sistema integral compuesto por medios de uso corriente, que busca satisfacer la necesidad de trasladar a las personas que habitan en un sector determinado. El uso colectivo de este servicio, supone una alternativa amigable con el medio ambiente, está dirigido hacia todas las personas que buscan movilizarse a través del área urbana.

Ilustración 4 Transporte público urbano
Fuente: (Autores)

Este sistema logra mantener a sus usuarios desconectados de la vía y mientras se conducen a su lugar de destino proporciona tiempo para que realicen cualquier otro menester o actividad. A raíz de esto, es que se han ido desarrollando herramientas que permitan a los usuarios ver como la principal alternativa de movilidad urbana el transporte público, categóricamente, los datos demuestran que la población en general utiliza este servicio masivo como última alternativa para su desplazamiento. (FACUA, 2007)

El transporte público urbano según el artículo 63 de la Ley Orgánica de Transporte, Tránsito y Seguridad Vial se clasifica de acuerdo a las características vigentes en la norma INEN y son:

Transporte colectivo. **–** Tiene la capacidad de movilizar un número elevado de pasajeros, puede tener una estructura exclusiva o no, además pueden operar según horarios, rutas, niveles de servicio o políticas tarifarias. Está conformado por buses y minibuses, los mismos que se caracterizan por poseer un piso bajo o un diseño convencional. (Tránsito, 2014)

- ***Microbús.*** **–** Vehículo autopropulsado orientado al transporte de pasajeros, posee un corredor central para la circulación de los mismos y tiene una capacidad máxima de hasta 26 usuarios.

- ***Minibús.*** **–** Vehículo autopropulsado orientado al transporte de pasajeros, al igual que el microbús posee un corredor central con la diferencia de que la capacidad máxima es de 60 usuarios.

- ***Bus.*** **-** Vehículo autopropulsado orientado al transporte de pasajeros, posee una gran capacidad que supera la del microbús y minibús, por lo general puede movilizar hasta 90 usuarios.

- ***Bus costa.*** **-** Vehículo autopropulsado orientado al transporte de pasajeros y mercancías, debido a su chasís combinado, no posee puertas y el volumen de pasajeros que soporta no es determinado. (INEN, 2012)

Transporte masivo. **–** Tiene la capacidad de movilizar un número elevado de pasajeros, tomando en cuenta que la cantidad de usuarios transportados es mayor a la del transporte colectivo, cuenta con una infraestructura exclusiva y está creado específica y únicamente para el servicio al usuario, puede operar según horarios, rutas, niveles de servicio y políticas tarifarias. Entre ellos se encuentra a los siguientes vehículos:

- ***Bus de dos pisos.*** **–** Vehículo autopropulsado que sirve para el transporte de pasajeros con una infraestructura especial que consiste en dos plantas, en la cual existe corredores internos para la circulación de los usuarios.

- ***Articulado.*** **-** Vehículo autopropulsado que sirve para el transporte de pasajeros con una infraestructura especial que consiste de dos o más divisiones articuladas entre sí, los pasajeros de cada una de las divisiones están intercomunicados, permitiendo el desplazamiento libre entre ellos.

2.1.3 Transporte público urbano en la ciudad de Ibarra

Ibarra posee como medio de transporte urbano a los autobuses que movilizan a toda la población, cabe recalcar que el ferrocarril que se restauró en la ciudad, hoy en día se utiliza como medio turístico y su uso no es exclusivo para el transporte público urbano dentro de la ciudad. Con referencia a los buses urbanos que circulan, existen dos cooperativas que presiden de sus unidades para brindar el servicio a la ciudadanía en general y son:

- Cooperativa "San Miguel de Ibarra"
- Cooperativa "28 de Septiembre"

Ambas cooperativas están destinadas únicamente al transporte de personas, entra las dos suman un total de 278 unidades, de las cuales 160 buses pertenecen a la cooperativa 28 de Septiembre y 120 unidades forman parte de la cooperativa San Miguel de Ibarra. (Comercio, 2015)

Ruta. **–** Es un camino prestablecido a seguir, que permite la movilidad cotidiana de las personas que precisan de ese recorrido para su traslado hacia un punto en concreto. En el caso de las cooperativas de autobuses urbanos de la ciudad, los recorridos que efectúan estas unidades en la urbe ya se encuentran diseñadas e implementadas. A continuación, se detallan las rutas específicas que cumple cada cooperativa de buses:

Tabla 1 Rutas buses San Miguel de Ibarra

Ruta	***Nombre***	***Longitud (Km)***
1	La Esperanza – Hospital del Seguro	25,24
2	Chugchupungo – La Florida	24,8
3	19 de Enero - Odilas	28,4
4	Colinas del Sur - Aduana	21,6
5	Ejido de Caranqui - Miravalle	28,7
6	Caranqui - Universidades	20,2
7	Santa Lucia – La Victoria	23,2
8	Santo Domingo - Universidades	28,7
9	Santo Domingo (Por la Florida) - Universidades	22,5
10	Santa Isabel – Huertos Familiares	39,9

Fuente: (MOVIDELNOR, 2006)

La cooperativa de buses San Miguel de Ibarra tiene 10 rutas en total, cada una cuenta con una longitud promedio de entre 25 a 30 km, las mismas que cruzan por toda la urbe, sin embargo, las rutas predominantes son aquellas que van a los sectores de: Santo Domingo, Universidades y la Esperanza.

Tabla 2 Rutas buses 28 de Septiembre

Ruta	***Nombre***	***Longitud (Km)***
11	Tanguarín - Aduana	34,5
12	Chorlaví – La Victoria	29,9
13	Milagro - Yahuarcocha	33,7
14	Pugacho – Santa Teresita	18,7
15	Las Palmas – Los Ceibos	19,7
16	San Miguel Arcángel – San Cristóbal de Caranqui	30,4
17	Católica - Alpachaca	15,9
18	Azaya – La Campiña	19,4
19	San Francisco, Longitud	18,8
20	Caranqui - Aduana	26,8
21	Santa Rosa – Los Ceibos	21,2
22	El Carmen - Bellavista	22,4
23	Naranjito, Longitud	20,6
24	Aloburo, Longitud	19,4

Autor: (MOVIDELNOR, 2006)

La cooperativa de buses 28 de Septiembre posee 14 rutas en total, cada una cuenta con una longitud promedio de entre 20 a 28 km, las mismas que cruzan por toda la urbe, sin embargo, las rutas predominantes son aquellas que van a los sectores de: Alpachaca, Tanguarín y Universidades. A continuación, se muestra un plano de la ciudad de Ibarra con todos los recorridos de las rutas anteriormente mencionadas:

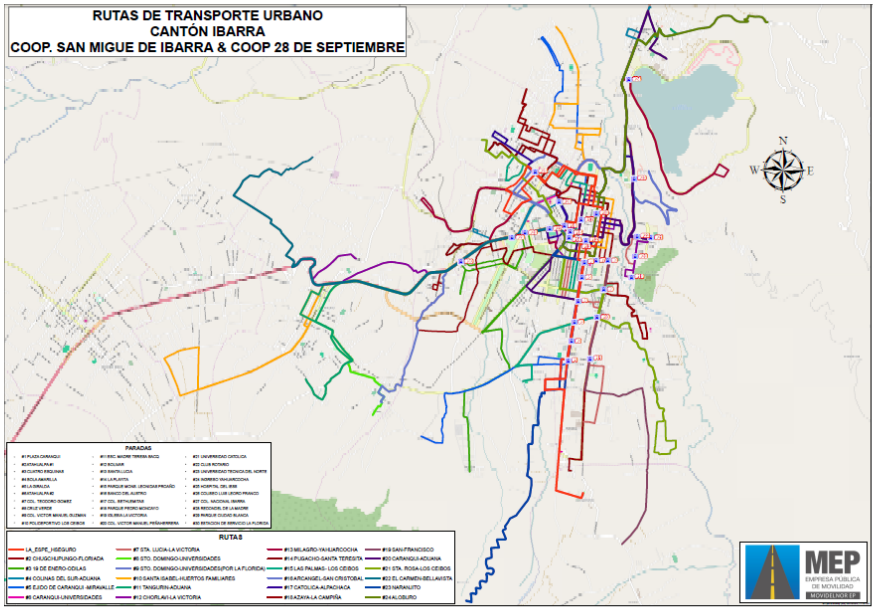


Ilustración 5 Rutas de transporte urbano Ibarra
Autor: (MOVIDELNOR, 2006)

En la ilustración anterior se observan las rutas de transporte urbano del cantón Ibarra, por donde transitan las unidades de la cooperativa San Miguel de Ibarra y 28 de Septiembre respectivamente. Esta información fue brindada por parte de la Empresa Publica del Movilidad del Norte (MOVIDELNOR).

Paradas de los autobuses urbanos. - Con el desarrollo del transporte masivo, desde la aparición del ómnibus, el hombre tubo la necesidad de crear lugares públicos que sirvan para el acopio y desembarque de sus usuarios, es ahí donde nacen las paradas de los autobuses. En la ciudad de Ibarra, según los datos brindados por parte de MOVIDELNOR, existen 245 paradas de autobuses oficiales, las cuales se encuentran distribuidas sobre toda la ciudad, como se logra apreciar en la siguiente imagen:

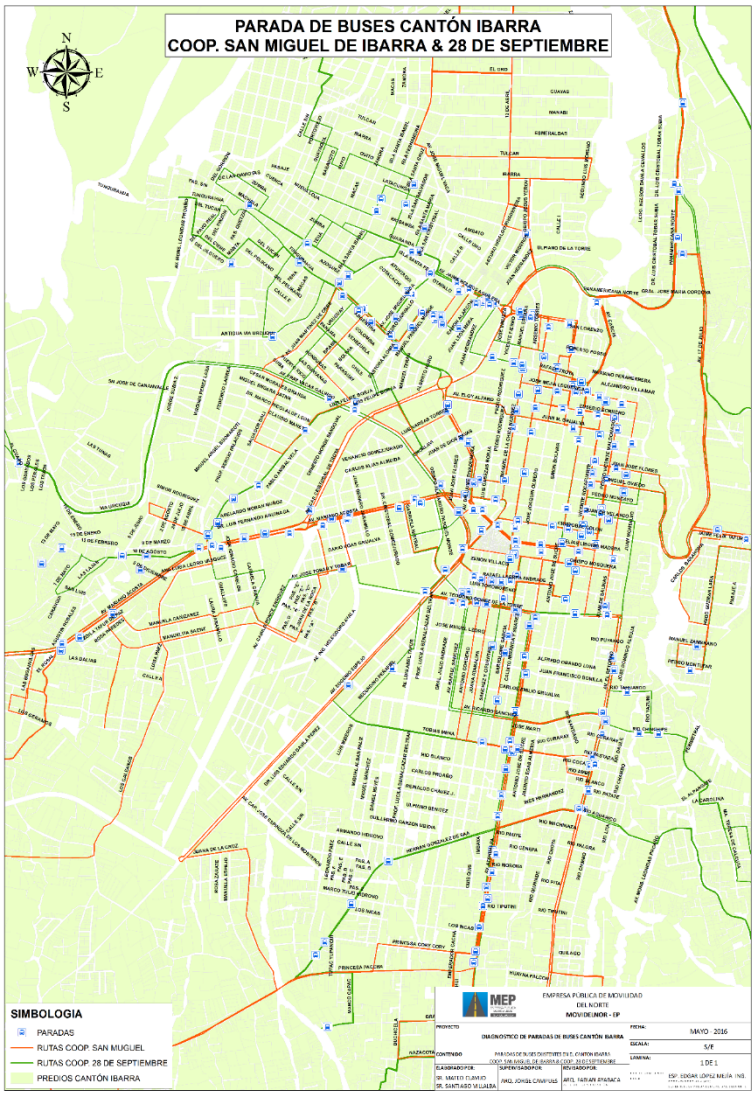


Ilustración 6 Parada de buses Ibarra
Fuente: (MOVIDELNOR, 2006)

En la ilustración anterior, se observa la distribución de las paradas oficiales en el cantón, las mismas que se encuentran con mayor afluencia sobre el centro de la ciudad, adaptándose a las rutas anteriormente citadas por donde transitan las unidades de las cooperativas.

2.2 Marco jurídico.

El Ecuador, al ser un país libre y soberano se rige a la Constitución de la República que gobierna todo el territorio nacional. En el ámbito referente al transporte, discapacidades e igualdad social menciona varios parámetros a tomar en cuenta:

2.2.1 Leyes y reglamentos del transporte terrestre.

Constitución del Ecuador. - En el artículo 394 garantiza la libertad de transporte terrestre, aéreo, marítimo y fluvial dentro del territorio nacional, sin privilegios de ninguna naturaleza. La promisión del transporte público masivo y políticas de tarifas son prioritarias, es decir, el estado regula el transporte terrestre, aéreo y acuático y sus actividades respectivas.

En el artículo 262 y 264 establece que los Gobiernos Autónomos Descentralizados adoptarán políticas de integración y participación, permitiendo el ordenamiento en el ámbito urbano, planificación, regulación y control del tránsito, transporte y seguridad vial en sus correspondientes jurisdicciones.

En el artículo 398 la Asamblea Constituyente da la potestad a la Ley Orgánica de Transporte Terrestre, Tránsito y Seguridad Vial como una normativa que permita adaptarse a las necesidades de los ciudadanos y hacer del transporte un sistema eficiente.

Ley Orgánica de Transporte Terrestre, Tránsito y Seguridad Vial. - Según el artículo 40 el transporte terrestre sea este destinado a personas o bienes, es un servicio esencial y debe responder a las siguientes condiciones:

- ***Responsabilidad.*** – Es responsabilidad del estado generar políticas de regulación y control con el fin de proporcionar a los usuarios y operadores reglamentos y normas.

- ***Comodidad.*** – El nivel de servicio de transporte de pasajeros y bienes debe ser acreditada y cumplir con normas, reglamentos técnicos y homologaciones preestablecidas por la Agencia Nacional de Tránsito.

- ***Continuidad.*** – El transporte conforme a lo establecido en sus respectivos contratos debe operar sin interrupciones.

- ***Seguridad.*** – El estado garantiza la movilidad de pasajeros y bienes eficientemente, mediante una infraestructura y servicios adecuados, garantizando la integridad física de usuarios y bienes transportados.

- ***Calidad. -*** El transporte terrestre debe cumplir con ciertos parámetros de servicios determinados por la ANT, además de valores agregados que ofrezcan los operadores de cada unidad.

- ***Estandarización.*** – A través de procesos de homologación establecidos por la ANT, se verificará que todos los vehículos cumplan con las normas y reglamentos técnicos de seguridad, medio ambiente y comodidad.

- ***Medio ambiente.*** – El estado garantizará que los vehículos a nivel nacional cumplan con normas ambientales, permitiendo que tengan nuevas tecnologías que contribuyan a la disminución de los contaminantes.

En el artículo 52 y 55 el estado garantiza la prestación del servicio del transporte público para personas y bienes dentro del territorio nacional, además se considera este servicio como un bien estratégico, en donde las rutas y frecuencias forman parte del estado y pueden ser comercialmente explotadas por medio de contratos.

Dentro de este ítem, es importante tratar la instalación de televisores en los autobuses, para lo cual se tomo en cuenta algunos párrafos de la Ley Orgánica de Transporte Terrestre, Tránsito y Seguridad Vial, entre ellos se cita al artículo 140 en el que los propietarios de vehículos de servicios públicos serán sancionados con multas equivalentes al diez por ciento de la remuneración básica unificada y con la reducción de 3 puntos en la licencia en caso de instalar equipos de video o televisores en sitios que pueden provocar la distracción del conductor, además, en el artículo 184 se menciona que la publicidad en medios de comunicación no podrá divulgar imágenes, sonidos o mensajes que induzcan a una circulación vehicular imprudente o peligrosa. Es decir, la instalación de televisores en autobuses esta permitida mientras no afecte al conductor o a los pasajeros de ninguna manera posible.

2.2.2 Leyes y normas en torno a la discapacidad e igualdad social.

Constitución del Ecuador. **–** El artículo 16 y 47 hace énfasis a que toda la población debe tener acceso y uso a las formas de comunicación visual, auditiva y sensorial, permitiendo su inclusión, también garantiza un acceso a bienes y servicios, por lo tanto, es necesaria la eliminación de barreras arquitectónicas y comunicacionales.

Ley Orgánica de Transporte Terrestre, Tránsito y Seguridad Vial. **–** En referencia a los artículos 46, 47 y 48 se trata temas como: el transporte automotor es un servicio de carácter público esencial en la economía estratégica del estado, además permite una movilización libre y segura de personas o bienes de un lugar a otro dentro del territorio nacional. El transporte terrestre debe ser responsable, accesible, universal, cómodo, continuo, seguro y con altos índices de calidad y tarifas equitativas, también debe responder a las siguientes condiciones:

- ***Universalidad.*** **–** El estado garantiza el acceso al transporte sin distinción de ninguna naturaleza y cumpliendo con las normas establecidas en la constitución y leyes pertinentes.

- ***Accesibilidad.*** **–** El estado garantiza que el transporte es un derecho que tienen todos los ciudadanos para su movilización y la de sus bienes, por consiguiente, todo el sistema de transporte debe responder con eficacia a este fin.

Normas Jurídicas en Discapacidad Ecuador. **–** En su política pública Nro. 7 asegura el acceso de las personas con discapacidad al medio físico, el transporte, la comunicación, bienes y servicios básicos. Por lo tanto, el país debe proporcionar las condiciones y espacios públicos bajo normas técnicas con el fin de dotar una infraestructura adecuada en óptimas condiciones para su buen acceso generando movilidad sustentable, saludable e incluyente.

Ley Orgánica de Discapacidades. **–** En los artículos 60 y 61 menciona que las personas con discapacidad tienen derecho a acceder y utilizar el transporte público, por lo tanto, los organismos competentes en tránsito, transporte terrestre y seguridad vial en

las diferentes circunscripciones territoriales controlarán el cumplimiento obligatorio de las normas de transporte para personas con discapacidad y establecerán medidas que garanticen el acceso de las personas a las unidades de transporte, asegurando la integridad en la utilización de las mismas.

Se optará por normas técnicas con el fin de adaptar todas las unidades de los medios de transporte público y comercial para que sean libres de barreras, obstáculos y medidas que dificulten el ingreso, los organismos competentes exigirán que al menos un porcentaje de las unidades cuenten con las adecuaciones técnicas necesarias para transportar a personas con discapacidad.

2.2.3 Plan Nacional del Buen Vivir.

El Plan Nacional del Buen Vivir es un documento que reconoce la importancia de la capacidad productiva en el desarrollo económico, además tiene una visión amplia de la naturaleza, cultura y evolución social que permite el crecimiento equilibrado del país. El Buen Vivir es una forma de vida que permite la felicidad, diversidad cultural, diversidad ambiental, armonía, equidad y solidaridad, buscando un incremento económico infinito.

Dentro de las herramientas del Plan Nacional del Buen Vivir en el Objetivo Nro. 3 trata de mejorar la calidad de vida de la población y garantiza el acceso a servicios de transporte y movilidad incluyentes, seguros y sustentables a nivel local e intranacional, lo cual incentiva el uso del transporte público debido a que este presenta características de seguridad, dignidad y sustentabilidad.

El estado tiene como derecho acondicionar adecuadamente los espacios públicos bajo normas técnicas y dotar de infraestructura adecuada en óptimas condiciones para el uso y gestión del transporte público masivo, con esto se garantiza la interconectividad territorial, social, cultural, geográfica y ambiental, además de una calidad e inclusión en el uso del transporte público. (Delgado, 2013)

2.3 Ordenadores y terminales.

2.3.1 Ordenador de placa reducida.

Un ordenador de placa reducida o SBC es una computadora completa en un solo circuito, es decir, posee todas las características de una computadora funcional en una sola tarjeta. Estas surgieron como un medio didáctico para estudiantes de colegios y universidades, con el fin de facilitar el aprendizaje de programación, informática y computación en general.

Las características principales de un SBC es su tamaño reducido, poco peso, confiables, robustos, costos accesibles y mejor manejo de potencia eléctrica que los computadores portátiles tradicionales. Por otro lado, esto implica que actualizar uno de estos sistemas es casi imposible, si hay un fallo o si se necesita una actualización, es normal que toque remplazar la tarjeta completa. (FUNDATION, 2016) Existe una gran variedad de ordenadores de placa reducida como:

- ***Gumstix.* -** SBC de tamaño mínimo creado por Gordon Kruberg en el año 2003, cuenta con conexión mediante bluethooth y Wi-Fi, procesador de 600 MHz o mayor dependiendo del modelo. Su característica principal es su tamaño que rodea el de una goma de mascar. (Medina, 2012)

- ***ECB AT91.* –** Es un ordenador de placa reducida que salió a la venta desde el 05 de diciembre del 2006, posee conexión USB y Ethernet, procesador de 180 MHz desarrollado en Colombia.

- ***Raspberry Pi.* -** Ordenador de placa reducida con múltiples funciones que van desde conexión USB hasta Wi-Fi.

- ***Cubierboard.* –** SBC similar al Raspberry Pi con mayor costo, pero mayores prestaciones. Es una placa para el desarrollo y estudio de electrónica e informática, con grandes rasgos de un ordenador con arquitectura ARM. (Kelly, 2014)

2.3.2 Raspberry Pi.

El Raspberry Pi es un ordenador de placa reducida o SBC desarrollado en Reino Unido con un costo realmente bajo en comparación al de sus competidores u otros ordenadores convencionales, posee grandes prestaciones y es usado en la enseñanza de informática. Su gran éxito radica en la gran comunidad que se ha creado alrededor de este; gracias a ello dispone de una gran variedad de documentos de apoyo.

Tiene por características un procesador central, memoria RAM, puertos USB, salida MIPI CSI, conector HDMI y DSI, conector para salida de audio, ranura MicroSD, puerto Ethernet, puerto GPIO, bus HAT ID y trabaja con sistemas operativos como: Raspbian, Ubuntu, Windows entre otros. (FUNDATION, 2016) Existen varios modelos entre los cuales destacan: el modelo A, B, B+, A+ y 2B, sin embargo, existen alternativas como el Banana Pi y Orange Pi.

Los beneficios de usar el Raspberry Pi a diferencia de otros ordenadores de placa reducida y sistemas embebidos, es la facilidad de obtención de software libres y compatibilidad que posee con dispositivos de todo tipo, además de no necesitar de otros programas como LabView o Matlab para la creación de interfaces gráficas, lectura y depuración de datos. Gracias a los algunos sistemas operativos se puede manipular Raspberry Pi como un ordenador de oficina, pero el propósito principal de este minicomputador gira en torno a la ciencia computacional. (Suárez, 2016)

Ilustración 7 Raspberry Pi
Fuente: (Autores)

2.3.3 Módulo GPS.

Para la ubicación geográfica de un punto, el hombre ha creado dos maneras de determinarlo: la primera es a través de las coordenadas geográficas (latitud/longitud) y la segunda es por medio de coordenadas UTM (x,y). Estas deben cumplir con tres requisitos que son:

- Ser un punto único.
- El sistema a utilizar debe estar perfectamente reconocido.
- Ser fácilmente reconocible la coordenada "z" del punto.

Coordenadas geográficas. **-** Son un medio a través del cual el hombre logra ubicar un punto sobre la superficie de la tierra basándose en el siguiente formato:

0° 21' 28'' N
78° 10' 53'' O

- **Paralelos. -** Son líneas imaginarias horizontales que se encuentran sobre el globo terráqueo en dirección este – oeste, estas son perpendiculares al eje de la tierra y en medida que se acercan a los polos disminuyen en su tamaño. Es así como se puede dividir el planeta en dos mitades equivalentes, ahora denominadas hemisferio norte y hemisferio sur, siendo la línea del Ecuador 0° la encargada de esta división.

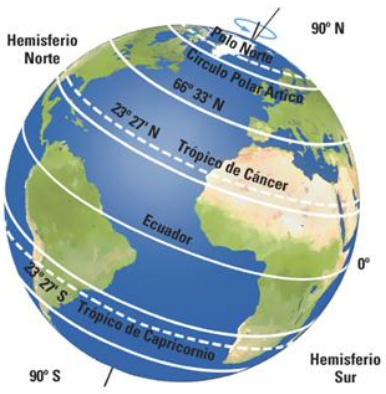


Ilustración 8 Paralelos de la Tierra
Fuente: (Abella, 2014)

La latitud es una coordenada que está determinada por los paralelos, es decir que hace referencia a la ubicación dependiendo sí esta es al norte o al sur del planeta, tomando como línea base la línea Ecuador medida en grados.

- **Meridianos. -** Son líneas verticales imaginarias que parten desde el polo norte al polo sur y dividen el planeta en hemisferio occidental y oriental. Cada línea posee su antimeridiano, que al juntarse forman una circunferencia.

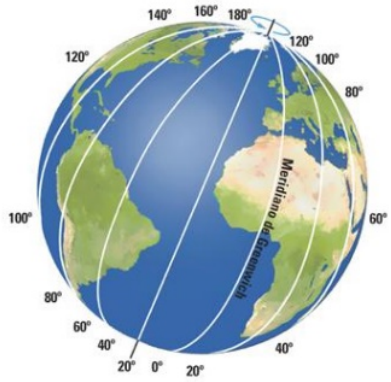


Ilustración 9 Meridianos de la Tierra
Fuente: (Abella, 2014)

El meridiano de referencia es el meridiano 0° o también llamado Greenwich y su antimeridiano el de 180°, los cuales, logran dividir en dos partes iguales al planeta. Los meridianos divididos en todo el planeta forman un total de 360°, estos sirven como referencia para la longitud, partiendo desde el meridiano 0° hacia el este o al oeste.

El sistema de posicionamiento global fue creado en 1960 y dio origen a varios y novedosos proyectos de navegación, a pesar de su pronto origen, recién en 1995 fue plenamente operacional. Su funcionamiento es a base de 24 satélites que circulan alrededor de la tierra cada 12 horas, proporcionando información en cualquier lugar del mundo. (NEO-6, 2015)

El módulo GPS al igual que otros navegadores convencionales recibe información a través de satélites utilizando el principio de trilateración y multilateración, que son métodos matemáticos para determinar posiciones de objetos

usando trigonometría, aproximadamente los dispositivos GPS tienen una precisión de 3 a 15 metros para aplicaciones civiles y de 1 metro para aplicaciones militares, cabe recalcar que cualquier instrumento de medición tiene cierto grado de imprecisión.

Ilustración 10 Módulo GPS
Fuente: (Autores)

En la actualidad existe una gran cantidad de fabricantes de receptores GPS, cada quien formula su propio protocolo de comunicación, sin embargo, la información base que todo GPS brinda es:

- Hora
- Fecha
- Posición
- Dirección
- Velocidad

Las aplicaciones más comunes para los dispositivos GPS van desde la localización de vehículos, verificación de rutas, navegación y orientación, telefonía celular, cartografía, robótica hasta sincronización de relojes. (5Hertz, 2014)

2.3.4 Sistemas operativos.

A un sistema operativo se lo define como el conjunto de órdenes y programas que controlan los procesos básicos de un computador, en el caso de una SBC, los sistemas operativos suelen ser más livianos y mucho más simples, por lo general contienen mayor o menor número de herramientas que un sistema operativo normal dependiendo de la minicomputadora utilizada. Entre estos podemos mencionar los siguientes sistemas operativos para SBC:

- ***Raspbian. -*** Es una distribución del sistema operativo Linux exclusivo para la placa Raspberry Pi, contiene herramientas de desarrollo como IDLE para el lenguaje de programación Python o Scracth, y diferentes ejemplos de juegos usando los módulos Pygame. (FUNDATION, 2016)

- ***Pidora. -*** Es un sistema operativo exclusivo para microcontroladores ARM, mucho más ligero que Raspbian, pero con menor apoyo. Fue creado en centro Séneca y ofrece un modo fácil de utilizar para aquellos usuarios que no disponen de un monitor. (Galnares, 2012)

- ***OSMC. -*** Es una distribución que se apoya en dos pilares básicos, Debian y Kodi. Permite la transformación de una TV sencilla a una Smart TV con acceso a contenidos multimedia. Este sistema operativo ofrece a los usuarios un nuevo concepto en cuanto a centros multimedia, es decir busca ejecutar en primer plano a Kodi, en una interfaz exclusiva para televisores. (Velasco, 2016)

- ***OpenELEC. -*** Es similar a OSMC, más sencilla de usar y pensada exclusivamente para contenidos multimedia. Permite reproducir videos, música, etc. Se puede controlar desde un Smartphone mediante conexión wifi y transforma una televisión en una Smart TV. (Sastre, 2014)

- ***Arch Linux. -*** Es una conocida distribución de Linux que también puede instalarse bajo plataformas ARM, fue creada en el 2002 por Judd Vinet y su enfoque central es la simplicidad, elegancia, coherencia de código y minimalismo. Cuenta con mucho apoyo, sin embargo, no es muy eficiente para Raspberry Pi. (López, 2015)

- ***RISC OS. -*** Es un sistema rápido y compacto, actualmente es obsoleto y fue desarrollado en Cambridge por Acorn Computer, funciona exclusivamente en plataformas ARM. Este sistema no tiene nada que ver con Linux, UNIX o Windows y puede ser usado en el minicomputador Raspberry Pi. (Victor, 2013)

- ***Ubuntu. -*** Es un sistema operativo perfecto para computadores de escritorio, laptops y otros, debido a que contiene aplicaciones desde procesadores de texto hasta herramientas de programación. Está basado en Linux y se distribuye gratuitamente. Pensado en la seguridad, permite arrancar y dejar en funcionamiento el computador rápida y fácilmente. (México, 2011)

- ***Windows 10 Iot core. -*** Es una versión limitada de Windows 10 que cuenta con opciones condicionadas, es un sistema muy simple creado para programadores. Para empezar, se trata de una edición de 32 bits, no cuenta con un entorno para escritorio y no permite la ejecución de aplicaciones o juegos tradicionales de Windows 10. (Pastor, 2015)

- ***Noobs. -*** Es una aplicación propia de Raspberry Pi que permite administrar sistemas operativos contenidos en un mismo paquete desde una sola tarjeta de memoria SD, es una aplicación muy útil ya que da la oportunidad de usar varios sistemas operativos sin realizar cambios drásticos en la MicroSD. En esta aplicación se encuentran Raspbian, OSMC, Pidora, OpenELEC, Arch Linux, RISC OS entre otros. (FUNDATION, 2016)

2.3.5 Lenguaje de programación.

El lenguaje de programación es un lenguaje formal diseñado para realizar procesos que pueden ser llevados a cabo en computadoras, existen varias clases de lenguajes y se usan en la creación de programas que controlan el comportamiento físico o lógico de una máquina, mediante la aplicación de procedimientos lógicos se puede llegar a los siguientes pasos:

- Desarrollo de programas para resolver problemas particulares.
- Codificación de programas mediante un lenguaje especifico.
- Ensamblaje de un programa para convertirlo en un lenguaje de programación.
- Prueba y depuración.
- Desarrollo de documentos.

En los computadores de placa reducida, especialmente en el Raspberry Pi entre los principales lenguajes de programación tenemos:

Mathematica 3.0. **–** Permite la realización de cosas muy simples mediante comandos para efectuar cálculos numéricos como una simple calculadora, pero a su vez cálculos más complejos que las calculadoras normales no pueden realizar. La interfaz de este programa contiene una pantalla, barra de menú, ventana vacía de trabajo y paleta de operaciones. (Informática, 2013)

Greenfoot. **–** Es una herramienta de software para principiantes que permite familiarizarse con la programación orientada a objetos. Se desarrolla mediante Java y fue diseñado entre la Universidad de Kent y la Universidad de Deakin. Puede ser instalado en Windows, Mac OS X y otros sistemas como Raspbian, además contiene dos escenarios denominados WOMBATS y WOMBATS2. (Kolling, 2011)

Python. **-** Es un lenguaje de programación creado por Guido van Rossum a principios de los años 90. Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible. (Duque, 2013) Se trata de un lenguaje interpretado o de script, con tipado dinámico y multiplataforma.

Un lenguaje de script se utiliza ejecutando un programa intermedio llamado intérprete (lenguaje complicado), los lenguajes complicados tienen una ejecución más rápida, pero los lenguajes de script son más flexibles y portables. En el caso de Python, posee características de ambos lenguajes por lo que se podría decir que es semi interpretado.

Python está disponible en una multitud de plataformas tales como: UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS y otros. Por lo tanto, si se usa librerías específicas de cada plataforma, los programas pueden correr en estos sistemas sin grandes cambios. Es decir, este lenguaje de comunicación posee grandes ventajas al ser muy simple y contener variedad de librerías, de esta manera, Python es adecuado para la programación de bajo nivel, programas complicados y orientado a objetos que contiene elementos como:

Funciones. **–** Es un conjunto o una parte de un código reutilizable y cuya función es realizar determinadas tareas. Se la puede representar de diferentes formas, en Python se usan dos y son:

Tabla 3 Tipos de funciones Python

Tipo	***Nota***
def	Define funciones y el valor retorna mediante la instrucción "return".
lambda	Define funciones.

Fuente: (Duque, 2013)

Tipos de datos. **-** Es el valor que una variable puede tomar, las operaciones que puede aplicar y la forma de representarse en un computador. Todos estos valores tienen tipos específicos y los datos usados en Python están representados en la siguiente tabla:

Tabla 4 Tipos de datos Python

Tipo	***Nota***
str	Cadena inmutable.
unicode	Versión Unicode de str.
list	Secuencia mutable, puede contener objetos diversos.
tuple	Secuencia inmutable, puede contener objetos diversos.
set	Conjunto mutable, sin orden ni duplicados.
frozenset	Conjunto inmutable, sin orden ni duplicados.
dict	Grupo de pares.
int	Número entero de precisión fija.
long	Número entero con coma flotante.
complex	Número complejo parte real y parte imaginaria.
bool	Booleano verdadero o falso.

Fuente: (Duque, 2013)

Listas y tuplas. **–** Son un conjunto ordenado de valores, las tuplas pueden ser representadas con cualquier valor mientras que una lista únicamente puede presentarse en forma de una serie de funciones con un amplio manejo de valores internos, en la siguiente tabla se analiza a detalle las características de cada una:

Tabla 5 Tipos de listas y tuplas Python

Tipo	***Nota***
Lista	Se usan corchetes con elementos de la misma cantidad de variables y son mutables.
Tuplas	Se usan paréntesis con elementos distintos en cantidad fija y son inmutables.

Fuente: (Duque, 2013)

Condicionales. - Al igual que en otros lenguajes de programación son expresiones booleanas que indican la rama a ejecutarse, por lo general representan un grupo de sentencias consecutivas y está compuesto por:

Tabla 6 Tipos condicionales Python

Tipo	***Nota***
if	Ejecuta el bloque de código interno solo si se cumple cierta condición.
elif	Condiciones adicionales en caso de existir.
else	Se ejecuta cuando todas las condiciones fueron falsas.

Fuente: (Duque, 2013)

Bucles. - Al igual que en otros lenguajes de programación, estos permiten repetir un parámetro infinitamente mientras no se tenga una condición de parada, en Python se usan los siguientes:

Tabla 7 Tipos bucles Python

Tipo	***Nota***
for	Similar a foreach en otro lenguaje, recorre un objeto como lista o tupla, y por cada elemento ejecuta el bloque de código interno.
while	Evalúa una condición y, si es verdadera, ejecuta el bloque de código interno.

Fuente: (Duque, 2013)

Glosario de términos.

ARM. -Empresa creadora de procesadores de 32 y 64 bits.

Automatización. - Sistema que permite realizar una tarea a una máquina sin intervención del hombre.

Desplazarse. - Acción de moverse de un lugar a otro.

Dispositivo. - Partes que en conjunto forman un componente más complejo.

Ensamblar. - Acción de unir diferentes elementos para crear un elemento único.

Equivalentes. - Número igual o fracciones de un todo que representa a una misma cantidad.

GNU/LINUX. - Combinación de núcleos que funcionan como software libre.

Implementar. - Poner en marcha una cosa pre-establecida.

Ómnibus. - Vehículo desarrollado para el transporte de personas en gran cantidad, generalmente denominado autobús.

Ordenador. - Sistema tecnológico con la capacidad de realizar una o varias funciones a la vez.

Potencia eléctrica. - Paso o flujo de energía por unidad de tiempo.

Preestablecido. - Elemento establecido u ordenado con anterioridad.

SBC. – (Session border controller) siglas en ingles que define al ordenador de placa reducida.

Sistematización. - Organizar un conjunto de partes que conformen un todo, en este caso un sistema.

Sistema integral. - Medio a través del cual se organiza y centraliza información vial.

Smartphone. - teléfono inteligente en capacidad de realizar procesos como un minicomputador.

CAPÍTULO III

3. DESARROLLO DE LA PROPUESTA

3.1 Etapas preliminares.

El presente capítulo consta del diseño y construcción del sistema inteligente de anunció de rutas y paradas en la ciudad de Ibarra, el cual ha sido dividido en 3 etapas; la primera etapa ayuda con la selección de los componentes y son expuestos los diagramas electrónicos de conexión entre los mismos. En la segunda etapa se describe el desarrollo del software, indispensable para el funcionamiento de todo el conjunto y en la última etapa se detalla el armado e implementación del sistema.

A través de un organizador gráfico de procesos se especifican las etapas que componen este capítulo y el orden de desarrollo, el cual se muestra a continuación:

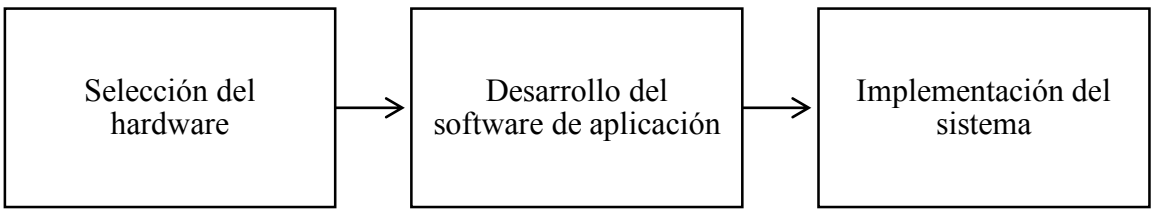


Ilustración 11 Fases del desarrollo de la propuesta
Fuente: (Autores)

3.1.1 Parámetros del sistema.

Es útil recordar los parámetros que han sido trazados para el desarrollo del presente trabajo de grado, debido a que estos son utilizados como bases directrices en el inicio del desarrollo de la propuesta. Las finalidades principales de brindar información útil a los usuarios son: mejorar la movilidad urbana entorno al transporte público, optimizar la información transmitida por los autobuses y contribuir con el desarrollo de la ciudad. Estos objetivos se muestran en la ilustración inferior:

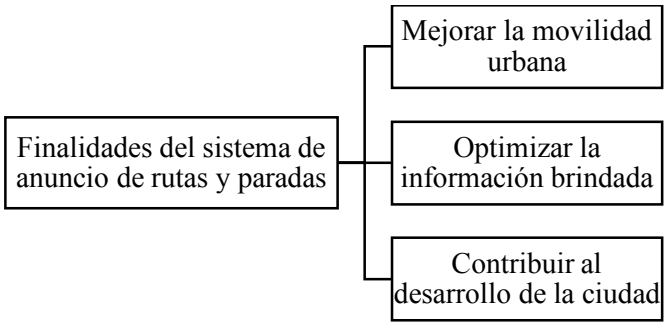


Ilustración 12 Objetivos del sistema
Fuente: (Autores)

3.1.2 Establecimiento de funciones.

Una vez determinados los propósitos, es momento de definir las funciones específicas que debe realizar el indicador de paradas al momento de entrar en marcha, desde la obtención de información hasta la manera en la que se brinda la información hacia los usuarios del autobús. Las funciones específicas que debe cumplir el sistema son las siguientes:

- Posicionamiento geográfico.
- Procesamiento de datos.
- Reproducción de contenido audio/visual.
- Representación de contenido.

3.1.3 Diseño del sistema.

Definidas las funciones e identificadas las características de los componentes principales, se prosigue con el diseño del sistema de anuncio de rutas y paradas basado en GPS. En la ilustración inferior se puede observar a través de un diagrama de bloques los elementos que lo conforman, consiguiendo identificar la estructura base del indicador de paradas y la forma como se encuentran interrelacionados:

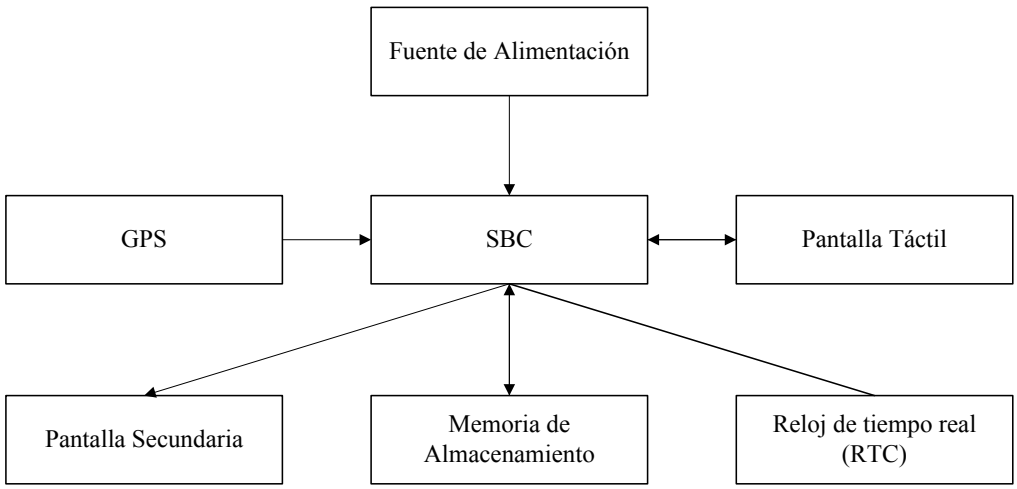


Ilustración 13 Diagrama de bloque del sistema
Fuente: (Autores)

Para la construcción del sistema de anuncio de rutas y paradas, son necesarios los componentes que se describen a continuación:

- Un SBC en capacidad de procesar información proveniente del GPS, que logre almacenar información, comparar datos, goce de características de reproducción de videos en alta definición (HD) y posea un puerto HDMI para una conexión física.

- Un módulo GPS capaz de brindar coordenadas geográficas de su ubicación en tiempo real, para conocer la posición actual del sistema, que sea compatible con la SBC y su enlace con el dispositivo sea sencillo o mediante conector USB.

- Una memoria de almacenamiento masivo, utilizada para el almacenamiento del sistema operativo, necesario para arrancar el ordenador de placa reducida, guardar la información procedente del GPS, así como también del contenido audio/visual.

- Una pantalla principal con tecnología touch destinada como periférico de entrada de órdenes al sistema por parte del operador, en elecciones como: la cooperativa de buses o la ruta a seguir, es decir, para controlar toda la interfaz del sistema.

- Una pantalla secundaria con grandes dimensiones, destinada a mostrar el contenido de audio y video en formato HD procedentes de la SBC mediante una comunicación HDMI entre ambos dispositivos.

- Un RTC (reloj en tiempo real), destinado a conservar la hora y fecha actual en el ordenador de placa reducida cuando este se encuentre apagado o a su vez desconectado de la fuente de alimentación. Debe ser compatible con la SBC, conservar un bajo consumo y una facilidad en el montaje.

- Un inversor de voltaje para vehículo de 12 V C.C. a 110 V C.A. utilizado como fuente de corriente eléctrica para el encendido de la pantalla secundaria, el ordenador de placa reducida y por consiguiente de la pantalla principal y el módulo GPS.

Siendo identificados todos los componentes indispensables para la construcción del sistema de anuncio de paradas y por medio del diagrama de bloques anteriormente diseñado, se realiza un diagrama del sistema con los elementos que posee, de forma que se visualice una estructura más clara. El mismo que se pueden observar en la ilustración inferior:

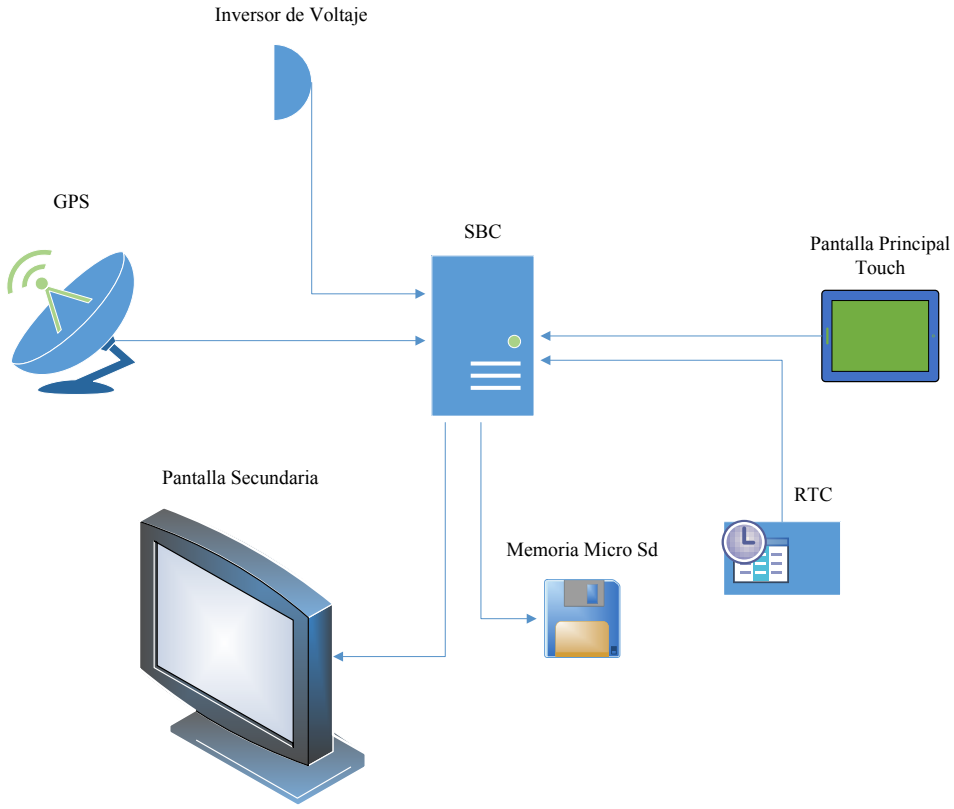


Ilustración 14 Diagrama componentes del sistema
Fuente: (Autores)

3.2 Selección del hardware.

3.2.1 Selección del ordenador de placa reducida.

Para el desarrollo de este sistema, se ha elegido los ordenadores construidos por la Fundación Raspberry Pi, que llevan el mismo nombre. La Raspberry Pi es un ordenador de placa reducida con múltiples funciones, aparte de poseer un potente procesador y una buena memoria RAM, tiene la capacidad de reproducir videos en formato HD, razón por la cual, se ha seleccionado estos computadores como elemento principal del sistema. En la actualidad existen varias versiones de la Raspberry Pi que se van a detallar a continuación conjuntamente con sus especificaciones técnicas:

Tabla 8 Versiones Raspberry Pi

Versión Raspberry Pi (RPI)	***CPU***	***Memoria RAM***	***Puertos USB***	***Puerto GPIO (Pines)***	***Salida de Video***	***Lector de Tarjetas***
RPI Modelo A	700 MHz	256 Mb	1	26	HDMI 1.4	SD
RPI Modelo A+	700 MHz	256 Mb	1	40	HDMI 1.4	Micro SD
RPI Modelo B	700 MHz	512 Mb	2	26	HDMI 1.4	SD
RPI Modelo B+	700 MHz	512 Mb	4	40	HDMI 1.4	Micro SD
RPI 2 Modelo B	900 MHz	1 Gb	4	40	HDMI 1.4	Micro SD
RPI 3	1.2 GHz	1 Gb	4	40	HDMI 1.4	Micro SD

Fuente: (FUNDATION, 2016)

Para este sistema se ha seleccionado la Raspberry Pi 2 modelo B, como dispositivo de proceso central, el cual posee un procesador de 900 MHz Quad-Core modelo ARM Cortex A7 con turbo que le permiten llegar a 1 GHz sin pérdida de garantía, característica principal para el procesamiento de datos a diferentes frecuencias, una memoria RAM de 1 GB que mejora la capacidad de cargar todas las acciones que se ejecuten en el procesador, así como la reproducción de videos en 1080p.

Ilustración 15 Raspberry Pi 2 modelo B
Fuente: (Autores)

Cuenta con 4 puertos USB 2.0 particularidad necesaria al momento de conectar periféricos de entrada o salida (pantalla táctil/módulo GPS), una salida MIPI CSI para instalar cámara; igualmente cuenta con una salida de audio y video por medio de un conector HDMI. Para la instalación del sistema operativo conserva un lector de tarjeta Micro SD y la conexión a internet es mediante puerto Ethernet. En tabla inferior, se detallan todas las características técnicas que posee la Raspberry Pi 2 modelo B:

Tabla 9 Características técnicas Raspberry Pi 2 B

Elemento	***Descripción***	***Característica***
SoC	Sistema en chip	Broadcom BCM2836
CPU	Unidad central de procesamiento	ARM Cortex A7 4 núcleos @ 900MHz
Overclocking	Rapidez del reloj	Hasta 1 GHz
GPU	Unidad de proceso de gráficos	Broadcom Video Core IV 250 MHz. OpenGL ES 2.0
RAM	Memoria de acceso aleatorio	1 GB LPDDR2
USB	Bus universal en serie	4 Puertos USB 2.0
Salida Video	HDMI (Interfaz Multimedia de alta definición)	1.4 @ 1920 x 1200 pixeles
Almacenamiento	Tarjeta de memoria flash	Micro Sd
Ethernet	Red de Área Local	10/100 Mbps
Dimensiones	85,6 x 56,5 mm	
Peso	45 gr.	
Consumo	5 V, 900 mA	

Fuente: (Duotel, 2015)

La Raspberry Pi posee diferentes conectores en su estructura, al igual que en versiones anteriores aún permite montar distintos dispositivos y efectuar un sin número actividades enfocadas en el desarrollo de la programación. A continuación, se detalla los conectores existentes en la Raspberry Pi 2 modelo B y su configuración mínima:

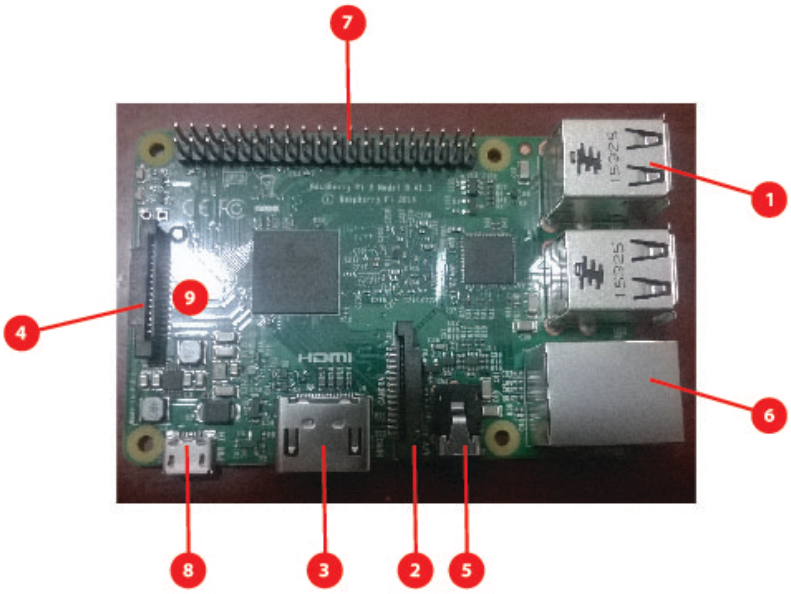


Ilustración 16 Puertos físicos Raspberry Pi 2 modelo B
Fuente: (Autores)

- ***Conector USB 2.0.*** – Este dispositivo tiene 4 puertos USB, lo que le permite la conexión de cualquier elemento que posea un punto de conexión USB como: mouse, teclado, memorias flash, adaptadores de WIFI o Bluetooth e incluso discos duros externos. Se debe recalcar que cualquier elemento que se conecte a estos puertos deben trabajar a menos de 100 mA, ya que, si esta excede se generará un mal funcionamiento. Para solucionar este problema, se puede montar un HUB USB autoalimentado para elementos que requieran de una mayor intensidad.

- ***Conector para cámara MIPI CSI.*** – Esta versión del Raspberry Pi también cuenta con un conector CSI de 15 pines, que le permite montar una cámara desarrollada por la misma fundación, o en su defecto por la empresa ADAFRUIT.

- ***Conector HDMI hembra.*** – Tiene montado un conector HDMI para la conexión con televisores digitales a través de un cable HDMI macho.

- ***Conector para salida de video DSI.*** **–** Este SBC admite el acoplamiento de pantallas LCD y tecnologías de visualización semejantes por medio de este puerto.

- ***Salida de audio (conector 3.5 mm).*** **-** También cuenta con una salida de audio analógica, aunque propiamente la Raspberry Pi maneja una salida de audio digital que se trasfiere por medio del propio conector HDMI.

- ***Puerto Ethernet.*** **–** Este puerto permite acceder a internet a través de la terminal, usando un cable RJ45 para la conexión física.

- ***Puerto GPIO.*** **-** Una de las ventajas que posee este dispositivo es el Bus de expansión GPIO ubicado en la parte superior izquierda de la placa, el cual posee 40 pines, distribuidos en dos columnas, lo que permite que la Raspberry Pi ostente una conexión con el exterior tanto para activar dispositivos como para leer el estado de los mismos. Cabe recalcar que cada pin tiene su función predeterminada dentro del puerto, por lo cual una conexión errónea puede afectar la Raspberry Pi. Estos pines se distribuyen de la siguiente manera dentro del puerto GPIO:

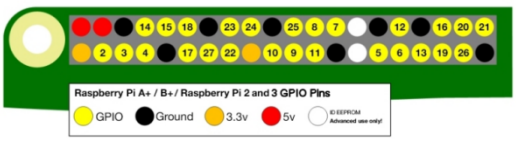


Ilustración 17 Puerto GPIO
Fuente: (FUNDATION, 2016)

- ***Puerto Micro USB.*** **–** Esta terminal sirve como medio de conexión a la fuente de 5V, con un consumo energético que va desde los 700 a 1000 mA en el modelo B y que puede variar a razón de los periféricos que se encuentre conectados.

- ***Puerto para lector de tarjeta Micro SD.*** **–** Este puerto es el encargado de admitir el montaje de una tarjeta Micro SD donde se instala el sistema operativo que permite correr a la Raspberry Pi, por esto, se considera como el disco duro de la

terminal, ya que aquí, es donde se almacena la información y datos del usuario. Su ventaja radica en que con solo cambiar de tarjeta Micro SD podemos cambiar el sistema operativo e incluso realizar alguna configuración sobre la terminal.

En relación a la programación en esta terminal, al adquirirla llega sin ningún sistema operativo, por lo tanto, cuando se habla de programación de la Raspberry Pi, hace referencia a la instalación del mismo, el cual, es necesario descargarlo y ejecutarlo sobre la tarjeta Micro SD para luego ser introducida en el lector de tarjetas. Existen dos formas de realizar este proceso, una es descargando de la página oficial o algún sitio web e instalarlo sobre la tarjeta y la otra es utilizando algún programa como: BerryBoot o Noobs que realizará el trabajo por el propietario.

Entre los sistemas operativos desarrollados para esta placa, una de las más populares es Raspbian, una distribución Linux que permite poseer un entorno tanto para jugador (Gamer) como para desarrollador de programación.

3.2.2 Selección del módulo RCT.

El módulo RTC (reloj en tiempo real) seleccionado para trabajar conjuntamente con la Raspberry Pi es el DS3231, debido a que es totalmente compatible con la terminal, se encuentra fácilmente en los mercados del país y posee un costo accesible. Básicamente, es un reloj muy preciso que posee un oscilador integrado y tiene una entrada para batería, permitiendo que no se modifique la hora actual cuando la alimentación principal de la Raspberry Pi es interrumpida.

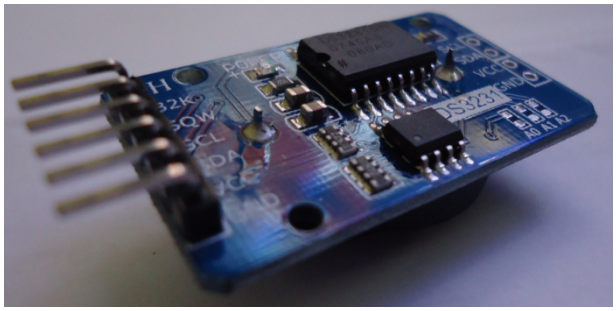


Ilustración 18 Módulo DS3231
Fuente: (Autores)

Este módulo RTC tiene la capacidad de mantener en su sistema desde: segundos, minutos y horas hasta días e incluso años. Puede corregir incluso los meses con menos de 31 días y años bisiestos, su formato de hora puede ser modificable, tanto en 12 horas como en 24 por medio de un indicador en formato "AM" o "PM". Los datos de este dispositivo se transfieren a través de un circuito integrado (I2C) al puerto GPIO de la Raspberry Pi. Seguidamente, se visualiza una tabla de las características que posee este módulo:

Tabla 10 Características módulo DS3231

Parámetro	***Especificación***
Tipo de módulo RTC	DS3231
Tipo oscilador	TCXO
Tipo de ajuste	Digital
Tipo de salida de frecuencia	CMOS Open-Drain
Precisión	± 2ppm de 0 °C a 40 °C
Precisión	± 3,5ppm de -40 °C a 85 °C
Salida digital del sensor de temperatura	± 3 °C Precisión
Registro de envejecimiento	TRIM
Botón de pulsación	Salida RTS
Alarma	2 por día
Señal de salida	Onda cuadrada
Frecuencia interfaz	400 KHz I2C
Fuente de energía en operación de baja potencia.	Batería tipo botón 3V
Temperatura de funcionamiento	Comercial: 0 °C a 70 °C Indistrial: -40 °C a 85 °C

Fuente: (SUNFOUNDER, 2016)

Tiene integrado un circuito de referencia y comparador de voltaje con restitución de temperatura para identificar fallos en la alimentación, lo que le permite proporcionar una salida de restablecimiento y cambiar a la configuración de reserva o batería cuando sea necesario y con ello que la fecha del sistema no se modifique. Las condiciones eléctricas recomendadas para este módulo en relación al suministro son de 2,3 a 5,5 V como se muestra en la tabla inferior:

Tabla 11 Condiciones eléctricas recomendadas DS3231

Parámetro	*Símbolo*	*Mínimo*	*Normal*	*Máximo*	*Unidad*
Voltaje de alimentación	V_{CC}	2,3	3,3	5,5	V
	V_{BAT}	2,3	3,0	5,5	V
Entrada lógica 1 SDA, SCL	V_{IH}	0,7 x V_{CC}		V_{CC} + 0,3	V
Entrada lógica 0 SDA, SCL	V_{IL}	-0,3		0,3 x V_{CC}	V

Fuente: (SUNFOUNDER, 2016)

3.2.3 Selección del módulo GPS.

El módulo GPS seleccionado para este sistema es el GPS/Glonass U-blox 7 debido a que permite conectarse a través de su puerto USB, tiene pequeñas dimensiones y su diseño de bajo consumo lo hacen indicado para este sistema. Este módulo pertenece a la empresa Ublox líder en la creación de chips y dispositivos de posicionamiento global. Dentro de su gama de productos existe una variedad de módulos GPS que pueden ser conectados mediante GPIO o actualmente mediante USB.

Ilustración 19 Módulo GPS
Fuente: (Autores)

Ublox-7 a diferencia de su antecesor Ublox-6 ofrece una mayor precisión, alta sensibilidad, mejor conectividad en tiempos cortos y su enlace a cualquier dispositivo es mediante puerto USB, necesita de un software especializado o en su defecto de librerías dependiendo del dispositivo al cual se desea conectar. A continuación, se observa en la tabla las características de rendimiento:

Tabla 12 Rendimiento U-blox 7

Parámetro	***Especificación***	
Tipo de receptor	56 Canales / Glonass L1OF	
Tiempo al primer arreglo	Inicio en Frío	30 s
	Inicio en Templado	25 s
	Inicio en Caliente	1 s
Sensibilidad	Seguimiento y navegación	-158 dBm
	Readquisición	-156 dBm
	Inicio en Frío	-140 dBm
	Inicio en Templado	-145 dBm
	Inicio en Caliente	-156 dBm
Precisión de posición horizontal	4 metros	
Precisión en la señal de tiempo de pulso	RMS 99%	50ns 100ns
Frecuencia en la señal de tiempo de pulso	0,25 Hz … 10MHz	
Tasa máxima de actualización de navegación	1 Hz	
Precisión de velocidad	0,1 metros/segundo	
Precisión Heading	0,5 grados	
Límites operacionales	Aceleración	< 4 g
	Altitud	50, 000 m
	Velocidad	500 m/s

Fuente: (U-blox, 2014)

GNSS (Sistema Global de Navegación por Satélite). **–** El módulo U-blox 7 posee receptores GNSS, es decir, puede recibir y rastrear por separado señales GPS suministradas a 1575 MHz, señales Glonass usando el mismo hardware y señales Galileo por medio de la actualización del firmware lo que genera una mayor cobertura, fiabilidad y precisión.

Registro de datos. **–** Es una nueva función de registro de datos que posee el U-blox 7, permite un almacenamiento continuo de información como: ubicación, velocidad y tiempo en una memoria interna de 16 Mb, estos datos anteriormente guardados en el receptor se pueden analizar en un lapso de tiempo posterior.

Protocolos de comunicación e interfaces. **–** Todos los protocolos están disponibles según el tipo de acceso a la memoria que se posea, estas interfaces pueden ser: UART utilizado para la comunicación con un Host; USB utilizado como medio de comunicación alternativo al UART; DDC creado para la conexión con un CPU Host externo y SPI diseñado para la conexión con un CPU Host con una frecuencia de reloj mayor al del DDC, los protocolos para estas interfaces son los siguientes:

Tabla 13 Protocolos U-blox 7

Protocolo	***Tipo***
NMEA	Entrada/Salida, ASCII, 0183, 2.3 (compatible con 3.0)
UBX	Entrada/Salida, binario, propiedad de u-blox
RTCM	Entrada, 2.3

Fuente: (U-blox, 2014)

Gestión de la energía. **–** Este dispositivo integra un convertidor DC/DC que permite tener un consumo de energía muy reducido, especialmente cuando el nivel de tensión de alimentación supera los 2.5 V, es así que posee dos modos de funcionamiento: uno que es continuo para un mejor rendimiento y otro denominado "modo ahorro" que optimiza el consumo del recurso energético. Las especificaciones eléctricas de este dispositivo son las siguientes:

Tabla 14 Especificaciones Eléctricas U-blox 7

Parámetro	***Símbolo***	***Mínimo***	***Máximo***	***Unidad***
Tensión de Alimentación	VCC	-0.5	3.6	V
Voltaje de la batería de reserva	V_BCKP	-0.5	3.6	V
Tensión de Alimentación USB	VDD_USB	-0.5	3.6	V
Tensión del pin de entrada	Vin Vin_USB	-0.5	3.6	V
Corriente de Salida RF	ICC_RF	100		mA
Temperatura de Funcionamiento	Tstg	-40	85	°C

Fuente: (U-blox, 2014)

3.2.4 Selección de la pantalla táctil.

Para la selección tanto de rutas como inicialización del sistema, es necesario una pantalla táctil que sirva de medio de comunicación entre la Raspberry Pi y el operador del sistema, para esto, se debe tomar en cuenta que la pantalla debe ser compatible con la terminal, por lo cual las pantallas táctiles compatibles con la Raspberry Pi 2 modelo B son las que se muestran en la siguiente tabla:

Tabla 15 Pantallas táctiles compatibles con Raspberry Pi

Tipo Pantalla	***Touch Screen***	***Tamaño (plg)***	***Dimensiones (largo/ancho)***	***Peso (gr)***	***Conexión Táctil***	***Conexión Video***
LCD – TFT	Resistiva	2.8	69*50 mm	47	Puerto GPIO	Puerto GPIO
LCD – TFT – HDMI	Resistiva	5	121*76 mm	178	Puerto USB	Puerto HDMI
LCD – TFT – HDMI	Capacitiva	7	194*110 mm	259	Puerto GPIO	Puerto DSI

Fuente: (WAVESHARE, 2015)

La pantalla a utilizar en este sistema de anuncio de paradas es la Waveshare 5 plg HDMI LCD (B), compatible con el computador de placa reducida Raspberry PI 2 modelo B, posee una resolución de 800 x 400 pixeles, su conexión se la realiza mediante un cable HDMI para la visualización de la imagen y un cable micro USB para proporcionar energía al componente, así mismo debido a las dimensiones que posee, su peso de 178 gramos y su Touch Screen del tipo resistivo es ideal para su manipulación.

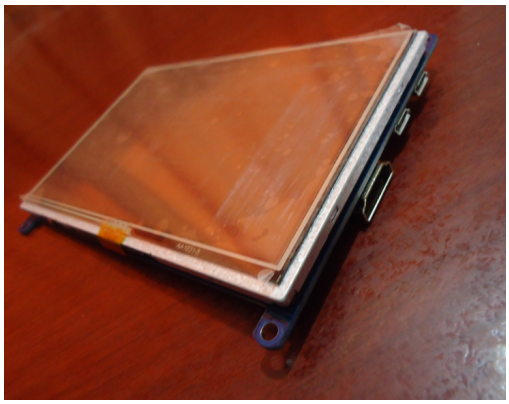


Ilustración 20 Pantalla Waveshare 5 plg HDMI LCD (B)
Fuente: (Autores)

Las medidas de la pantalla táctil conjuntamente con la placa electrónica de la misma poseen un largo de 121 mm y un ancho de 89.55 mm, estas características, así como los puntos de anclaje de la misma, son de suma importancia al momento de diseñar la caja donde ira montado el sistema de anuncio de rutas y paradas. A continuación, se detallan todas las características técnicas de la pantalla conjuntamente con las dimensiones externas de la placa sobre la que se monta la misma:

Tabla 16 Características de la pantalla táctil 5 plg

Parámetro	***Especificación***
Fabricante	Waveshare
Modelo	5 INCH HDMI LCD (b)
Resolución	800 x 480 pixeles
Tamaño Pantalla	5 pulgadas
Soporte	Únicamente Terminales Raspberry Pi
Interfaz LCD	Puerto HDMI
Interfaz Touch	Puerto USB
Tipo de panel táctil	Resistivo
Multi - Touch	Si / soporta hasta 10 puntos
Dimensiones	121 x 76 mm
Característica	Soporta varios sistemas

Fuente: (WAVESHARE, 2015)

3.2.5 Selección de la pantalla secundaria.

El sistema consta de una pantalla secundaria donde se reproducirá la información visual y auditiva procedente de la terminal hacia los usuarios del autobús, en este caso se ha visto conveniente una pantalla de 32 pulgadas. En referencia a su tecnología, actualmente en el mercado se encuentran televisores LCD, Plasma y LED.

Los televisores LCD. - Básicamente constan de múltiples capas, cada una de un distinto material, si se refiere al contraste, estos pueden variar desde 1000:1 hasta un máximo de 5000:1, por esta razón son incapaces de presentar el color negro de forma perfecta, así mismo una gran desventaja son sus ángulos de visión, que disminuyen al

observar la imagen desde sus costados, pero logran mantener sus colores brillantes en lugares iluminados, ya que estos no reflejan la luz del día.

Estos televisores resultan ser más ligeros en comparación al plasma, consumen menor energía (Ahorro del 30% al 50%) y generan menor calor en comparación a un televisor LED o Plasma. Si se refiere a precios, estos televisores son accesibles, pero continúan por encima de los televisores con tecnología plasma.

Los televisores plasma. **–** Basan su funcionamiento en un gas ionizado, lo que le genera mejor calidad de imagen en comparación a la tecnología LCD y posee un mayor contraste (10 000:1), color, brillantes y definición. Como desventaja es que son frágiles, poseen un mayor consumo eléctrico y por ende generan mayor calor, por esta razón la ventilación de las mismas debe ser adecuada e imprescindible. Además, por su superficie de vidrio tienden a reflejar la luz externa o de día, lo que afecta en la brillantes y contraste de sus imágenes. En evaluación al precio de estos televisores, son menores en comparación a los LCD y LED.

Los televisores LED. **–** La popularidad en los últimos años de la tecnología LED ha aumentado considerablemente y a diferencia de las pantallas LCD, se iluminan a través de LEDs o diodos emisores de luz individuales, lo que le permite generar mayores contrastes y una mejor visualización de colores, así mismo se caracterizan por la reducción en el consumo energético si se compara con pantallas convencionales y la vida útil que estas representan es mayor.

Si se menciona el diseño, estas pueden ocupar menor espacio y poseen gran estética, además de los grados de visualización que estas nos ofrecen sin perder la calidad de imagen. Una desventaja de esta tecnología es su precio, el cual es superior a los televisores LCD y también la luminosidad de los LEDs puede variar en función de la temperatura.

Para el desarrollo de este proyecto se ha dado cabida a la tecnología LED, debido a factores como el consumo energético, el espacio que estas ocupan, la vida útil que representan y los grados de visualización que estas ofrecen, siendo ideal para la implementación en los buses urbanos de la ciudad.

Ilustración 21 Pantalla Secundaria LED
Fuente: (Autores)

La televisión a ocupar es la K-LED32HDT2 de la empresa Kalley con sus 81 cm o 32 pulgadas, permite reproducir video en formato HD y sus dos conectores HDMI benefician al momento de conectarlo a la terminal del sistema. A continuación, se muestran sus características técnicas:

Tabla 17 Especificaciones técnicas Kalley LED 32"

Parámetro	***Especificación***
Fabricante	KALLEY
Modelo	LED32HDT2
Tecnología	LED
Resolución	1366 x 768 pixeles
Tamaño Pantalla	32 pulgadas
Entradas	2 Puertos HDMI 1 Entrada USB 1 AV (Video) IN 1 AUDIO IN 1AV OUT 1 Conexión VGA 1 Conexión Antena 1 AC (Entrada Corriente Alterna)
Potencia de audio	6 W + 6 W
Velocidad de Respuesta	60 Hz
Consumo de energía	50 W
Peso	6.2 Kg
Fuente de alimentación	110 – 240 V ~ 50 / 60 Hz
Dimensiones	83 x 52.4 x 13.7 cm
Condiciones Ambientales	Temperatura: 5 ~ 45 °C

Fuente: (Kalley, 2014)

3.2.6 Inversor de voltaje.

Un inversor de voltaje es un aparato electrónico destinado a convertir un voltaje fijo de corriente continua (C.C.) en corriente alterna (C.A.) suministrando un voltaje diferente al que ingresa, utilizado hoy en día para el uso de electrodomésticos en lugares como vehículos, en donde se generan corriente continua. En el caso de este sistema, se busca trabajar con una pantalla LED sobre un autobús que generalmente trabaja entre los 12 V hasta los 24 V en C.C. por ende, es necesario la implementación de un inversor para encender la pantalla LED, así como la Raspberry Pi conjuntamente con la pantalla táctil y el módulo GPS. El inversor escogido para este fin es el Schumacher PI-400 el cual posee la capacidad de convertir la energía eléctrica del vehículo (12 V en C.C.) a energía doméstica (120 V en AC), en la tabla inferior podemos observar las características técnicas que posee este aparato electrónico:

Tabla 18 Especificaciones Técnicas PI-400

Parámetro	***Especificación***
Fabricante	SCHUMACHER
Modelo	PI-400
Máxima energía continua	400 Watts
Capacidad de tensión (Potencia máxima)	800 Watts
Consumo de corriente en vacío	< 0.4 A
Forma de onda	Onda Senoidal Modificada
Ámbito de tensión de entrada	10.5 V a 15.5 V en C.C.
Ámbito de tensión de salida	120 V ± 5% en C.A.
Alarma de batería baja	Audible, de 10.3 V a 10.6 V en C.C.
Cierre por batería baja	10.5 V ± 0.3 V en C.C.
Cierre por batería alta	15.5 V ± 0.5 V en C.C.
Eficiencia óptima	85%
Toma de corriente A.C.	2x NEMA 5 – 15 USA
Fusible	2x 25 A (tipo espada)
Dimensiones	6.5” largo/ 4.13” ancho / 2.2” grosor
Temperatura ideal de trabajo	10 °C y 30 °C
Peso	2 libras

Fuente: (Schumacher, 2014)

Es fundamental, proporcionar una buena ventilación cuando entra en funcionamiento el inversor, así como no utilizarlo en lugares húmedos o asentarlo sobre materiales inflamables como ropa o cartón. Es importante que la capacidad de energía necesaria por el electrodoméstico a utilizar, no sea mayor a la brindada por el inversor, ya que esto puede ocasionar daños sobre ambos equipos.

Una vez seleccionados los componentes que formarán parte del sistema, se procede a realizar un diagrama con dichos elementos, para concebir una mejor idea de la estructura, como se muestra en la ilustración inferior:

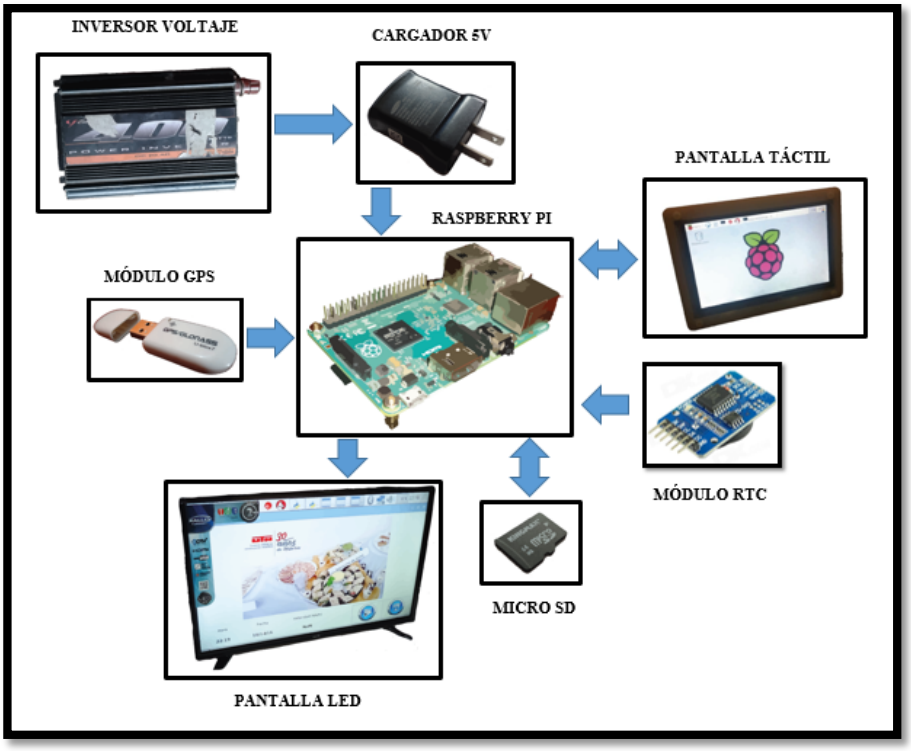


Ilustración 22 Diagrama sistema
Fuente: (Autores)

3.2.7 Diagrama electrónico del sistema.

A continuación, luego de haber seleccionado los componentes a utilizar, se representan los circuitos usados para la conexión de los diferentes dispositivos electrónicos, cada uno de ellos tiene una función específica y en conjunto cumplen con un mismo objetivo.

Adquisición de datos del módulo GPS. - A través de la comunicación entre el módulo GPS Glonass U-Blox 7 y la minicomputadora Raspberry Pi, se obtiene los datos necesarios para el correcto funcionamiento del sistema de anuncio de rutas y paradas. La conexión entre ellos se realiza mediante USB de la siguiente manera.

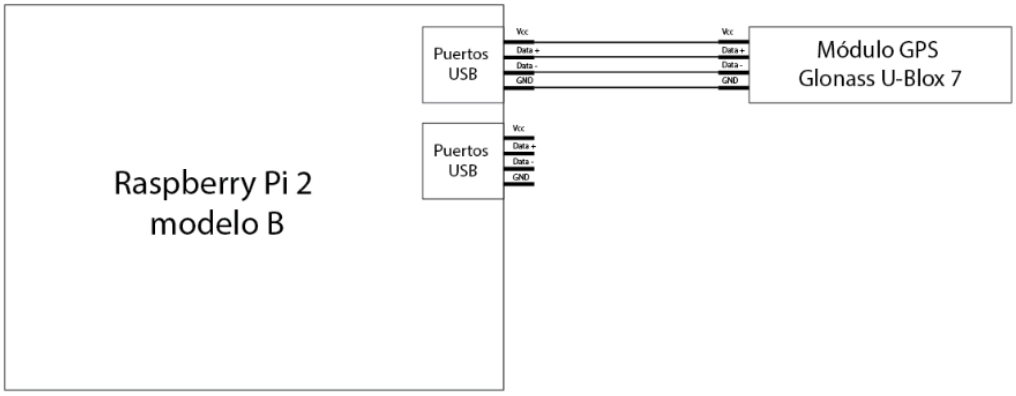


Ilustración 23 Conexión Raspberry Pi/Módulo GPS
Fuente: (Autores)

Establecimiento de la hora actual en el sistema a través del módulo RTC. – Por medio de la conexión entre el módulo RTC y la Raspberry Pi, se logra mantener la hora y fecha actual al momento de inicializar el indicador de rutas y paradas luego de un tiempo prolongado de encontrarse apagado. La conexión entre estos dos dispositivos es a través del puerto GPIO por medio de un bus de datos (I2C), el mismo que se observa en la ilustración inferior:

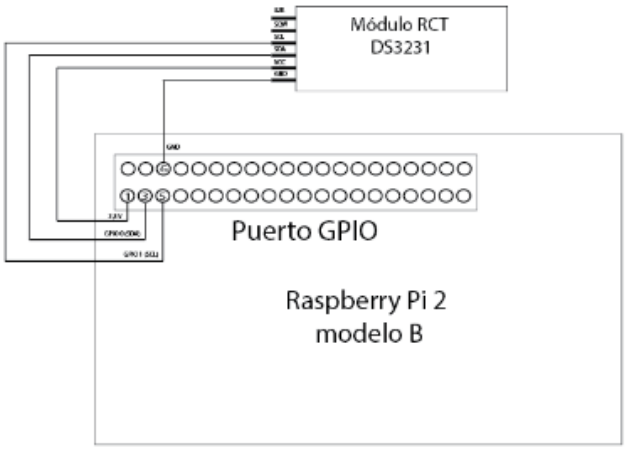


Ilustración 24 Conexión Raspberry Pi/Módulo RTC
Fuente: (Autores)

Pantalla táctil Waveshare 5inch HDMI LCD (B) y Raspberry Pi. - La Raspberry Pi es una minicomputadora que no posee monitor para la reproducción de imágenes, sin embargo, es compatible con todo tipo de pantallas incluyendo táctiles. La pantalla táctil Waveshare de 5 pulgadas permite la relación entre el controlador de autobús y el sistema de anuncio de rutas y paradas. La conexión entre estos dispositivos se realiza mediante USB, micro USB y HDMI como se representa en la figura siguiente.

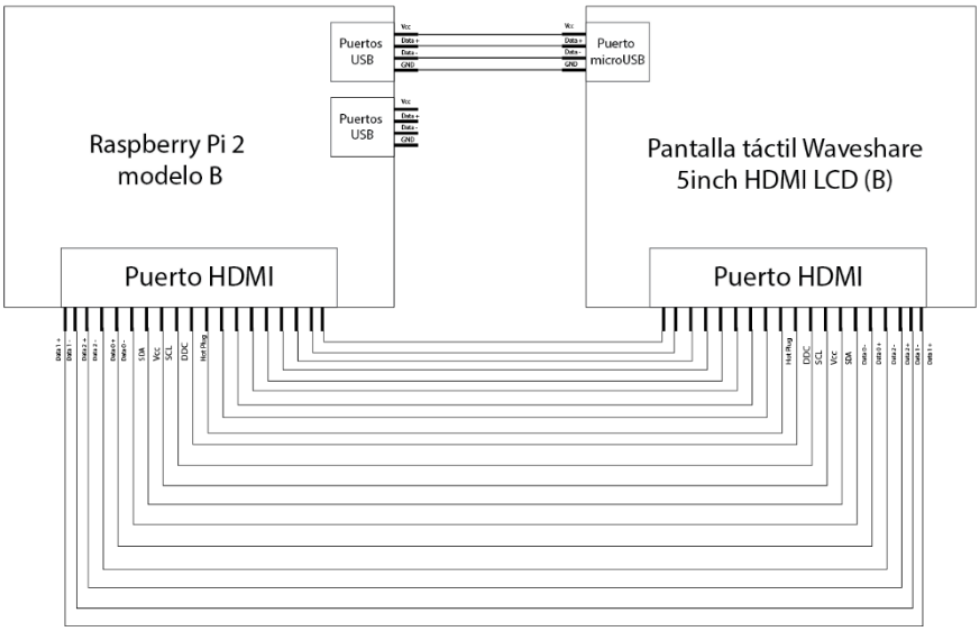


Ilustración 25 Conexión Raspberry Pi/pantalla táctil
Fuente: (Autores)

Pantalla Led de 32 pulgadas y Raspberry Pi. - El sistema de anuncio de rutas y paradas está destinado a brindar ayuda a los usuarios, debido a esto, se utiliza un Pantalla Led de 32 pulgadas para dar información útil a los pasajeros. Los dos instrumentos están conectados mediante HDMI y salida de audio 3,5 mm de la siguiente manera.

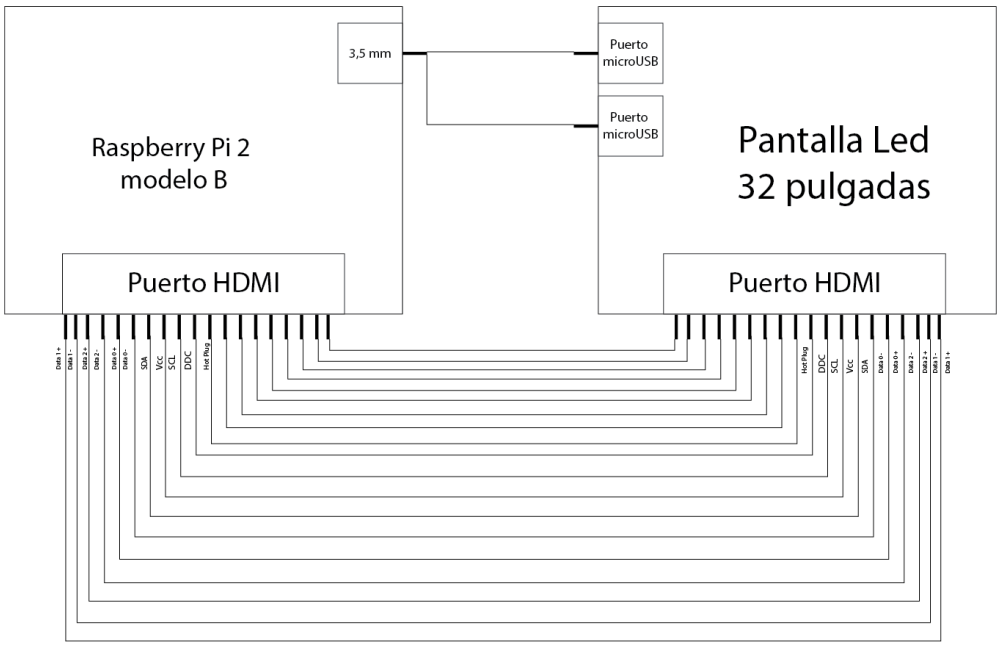


Ilustración 26 Conexión Raspberry Pi/pantalla secundaria
Fuente: (Autores)

3.3 Desarrollo del software de aplicación.

3.3.1 Flujogramas del sistema.

La construcción del sistema inteligente de anuncio de rutas y paradas está elaborada en una sola etapa, esto se debe a la utilización del minicomputador Raspberry Pi que además de ser un sistema muy robusto también posee software de programación preestablecidos, por lo tanto, no se usaron programas adicionales para depuración de datos o visualización de interfaces gráficas tales como LabView o Matlab. El sistema completo está elaborado en Python conjuntamente con la librería Tkinter y posee etapas de trabajo que se desarrollaron y se encuentran en el flujograma principal de funcionamiento del sistema.

Flujograma principal de funcionamiento del sistema. - El sistema de anuncio de rutas y paradas está diseñado con el objetivo de recolectar y analizar datos del GPS durante su funcionamiento, los mismos que luego de ser observados desencadenarán en un conjunto de procesos para la reproducción de videos informativos, reproducción de

imágenes e información sobre hora y fecha. En el siguiente flujograma se representa el funcionamiento general del sistema:

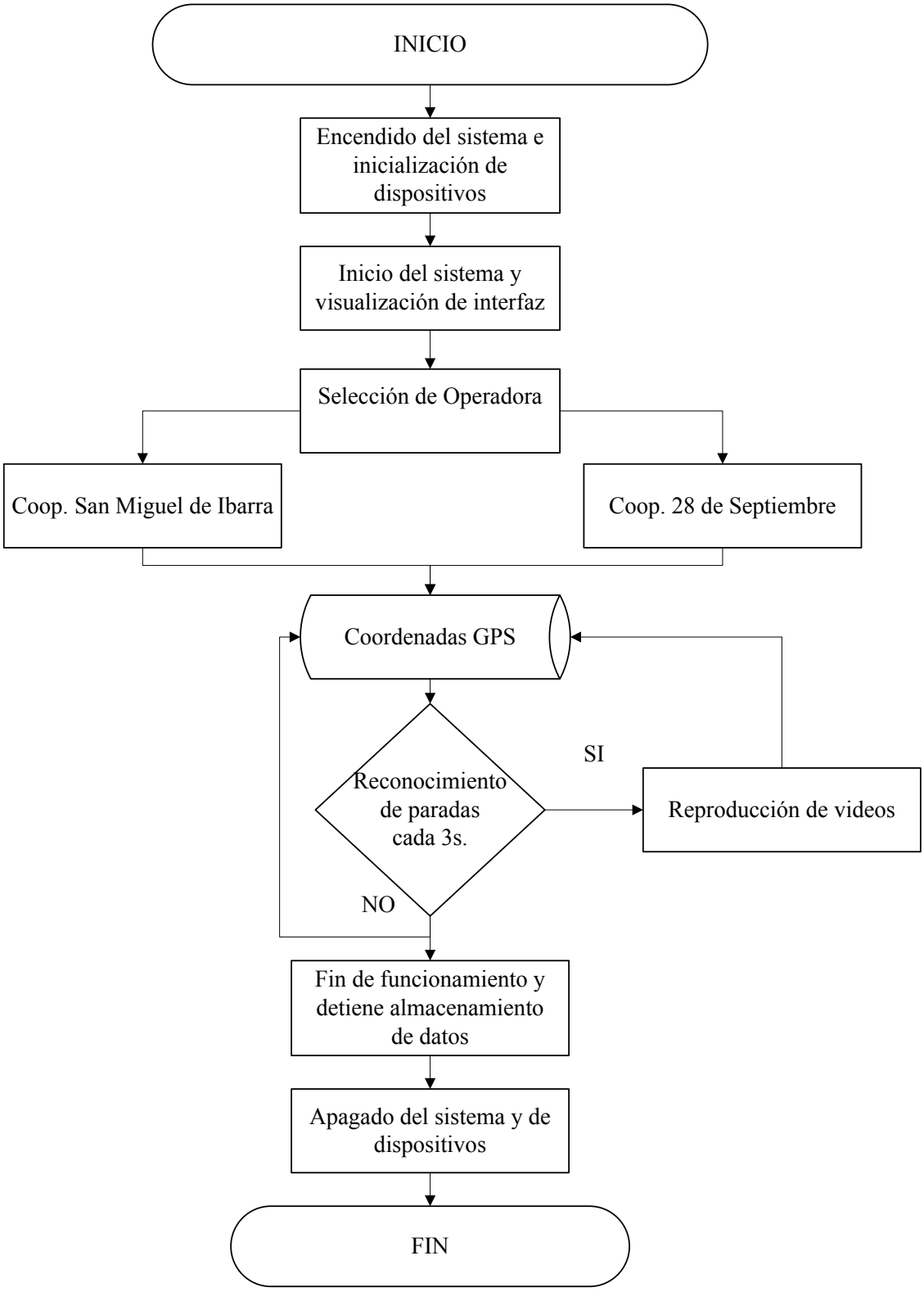


Ilustración 27 Flujograma funcionamiento principal del sistema
Fuente: (Autores)

El programa diseñado consta de etapas que se ejecutan paulatinamente mientras el sistema de anuncio de rutas y paradas está funcionando. Entre ellas se tiene las siguientes:

- Etapa de encendido del sistema.
- Etapa de inicio del sistema.
- Etapa de selección de rutas y almacenamiento de datos.
- Etapa de reconocimiento de paradas y reproducción de videos.
- Etapa de fin de funcionamiento y apagado del sistema.

Flujograma de encendido del sistema. - La Raspberry Pi es una computadora de placa reducida, por lo tanto, está lista para usar cualquier aplicación, software o librería al momento de su encendido. El siguiente flujograma muestra los procesos que realiza el SBC antes de ejecutar las órdenes y selección de rutas del programa principal.

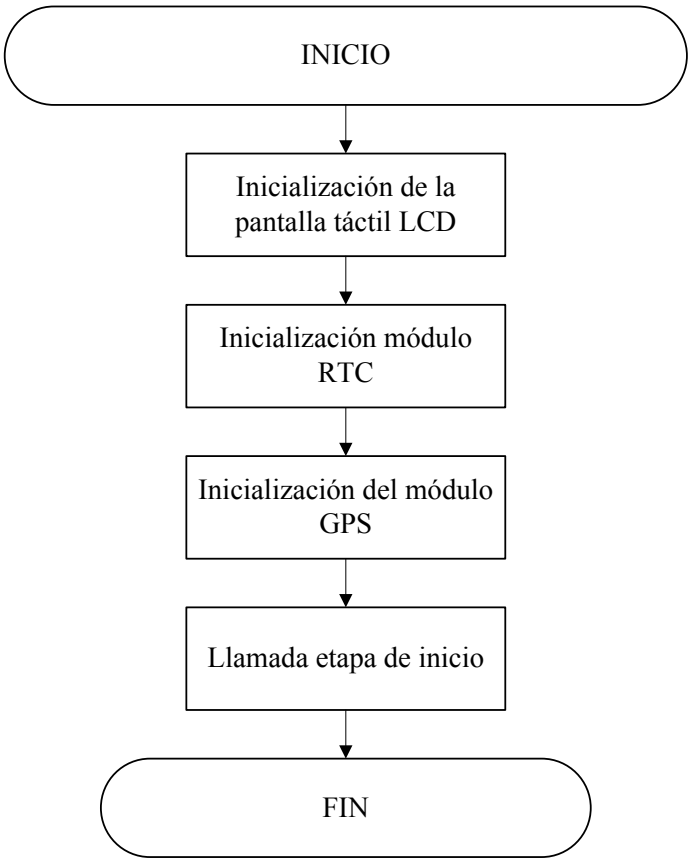


Ilustración 28 Flujograma encendido del sistema
Fuente: (Autores)

Flujograma de inicio del sistema. - Esta subrutina aparece cuando el Raspberry Pi se enciende por completo, en ella se observan tres botones que llevan a la subrutina de rutas Coop. 28 de Septiembre, subrutina de rutas Coop. San Miguel de Ibarra y apagado del sistema, además, contiene el isologo de cada una de las cooperativas, el escudo de la ciudad de Ibarra y el isologo de la carrera de Ingeniería en Mantenimiento Automotriz. A partir de la subrutina de inicio se puede acceder a las funciones del sistema de anuncio de rutas y paradas. A continuación, se observa el flujograma de inicio.

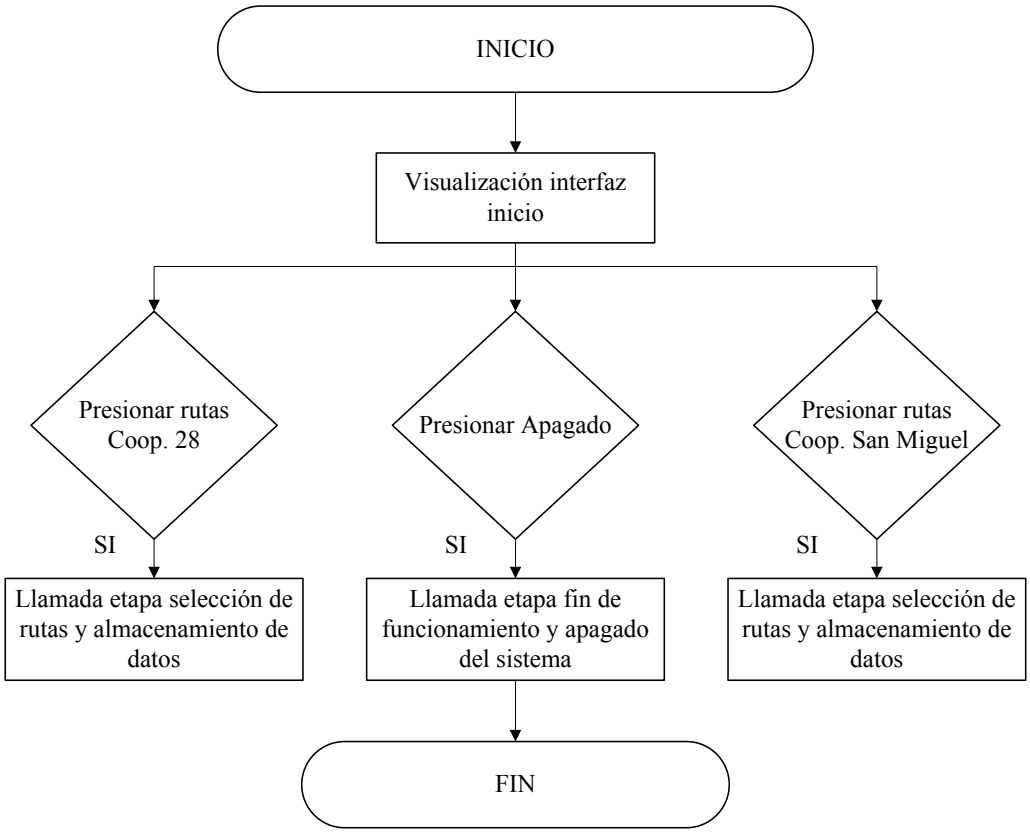


Ilustración 29 Flujograma subrutina de inicio
Fuente: (Autores)

Flujograma de selección de rutas y almacenamiento de datos. - Una vez presionado el botón correspondiente a la Cooperativa 28 de Septiembre o San Miguel de Ibarra se logra ingresar a la selección de rutas, esta consta de botones con el nombre de cada ruta, un botón de "Siguiente", uno de "Home" y un botón con texto variable que en caso de ser presionado permite el ingreso a la etapa de reconocimiento de paradas y reproducción de videos.

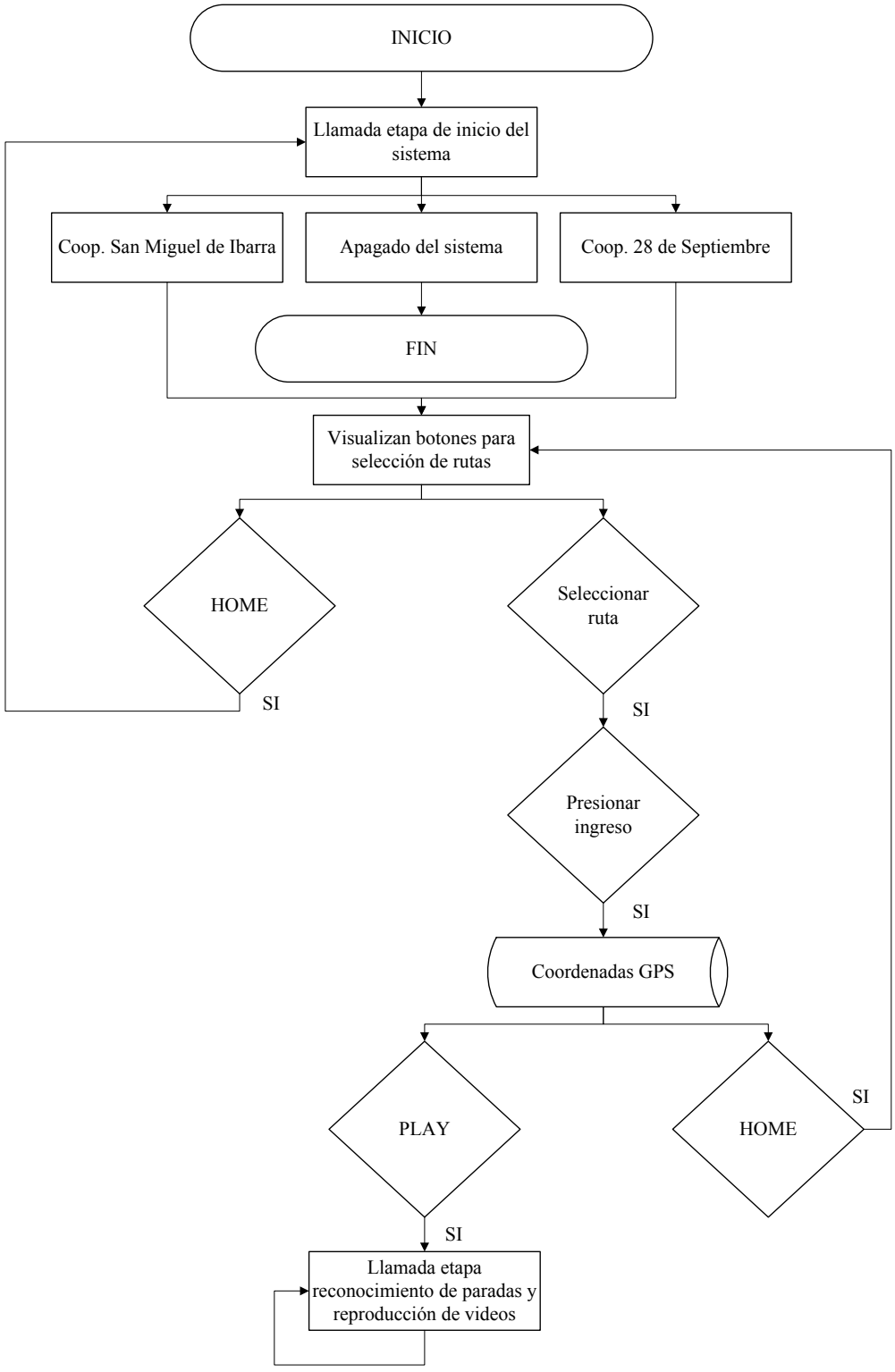


Ilustración 30 Flujograma de selección de rutas y alamcenamiento de datos
Fuente: (Autores)

Flujograma de reconocimiento de paradas y reproducción de videos. **–** En este ítem se analizarán dos aspectos fundamentales del sistema de anuncio de rutas y

paradas, en el primero se hará énfasis a la manera en la que se reconoce cada parada de la ruta y en el segundo la forma como se reproducirá el contenido audiovisual. Para su mejor entendimiento se dividió en dos flujogramas que son:

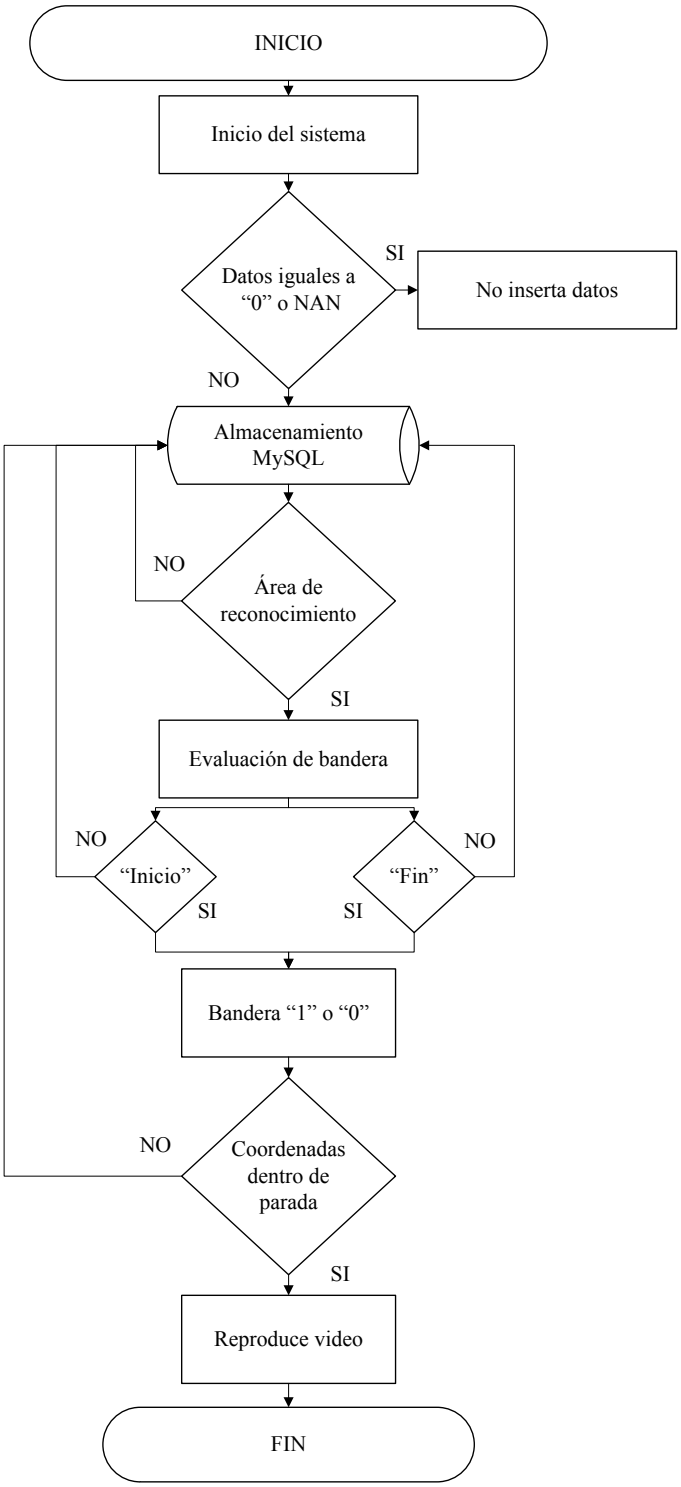


Ilustración 31 Flujograma reconocimiento de paradas
Fuente: (Autores)

El flujograma anterior representa la manera de como reconocer las paradas en el sistema. Este proceso se realiza paralelo a la reproducción de videos ya que actúa al mismo tiempo que la interfaz de reproducción de videos esta activada. Para una mejor comprensión a continuación se representa el flujograma correspondiente a la reproducción de videos:

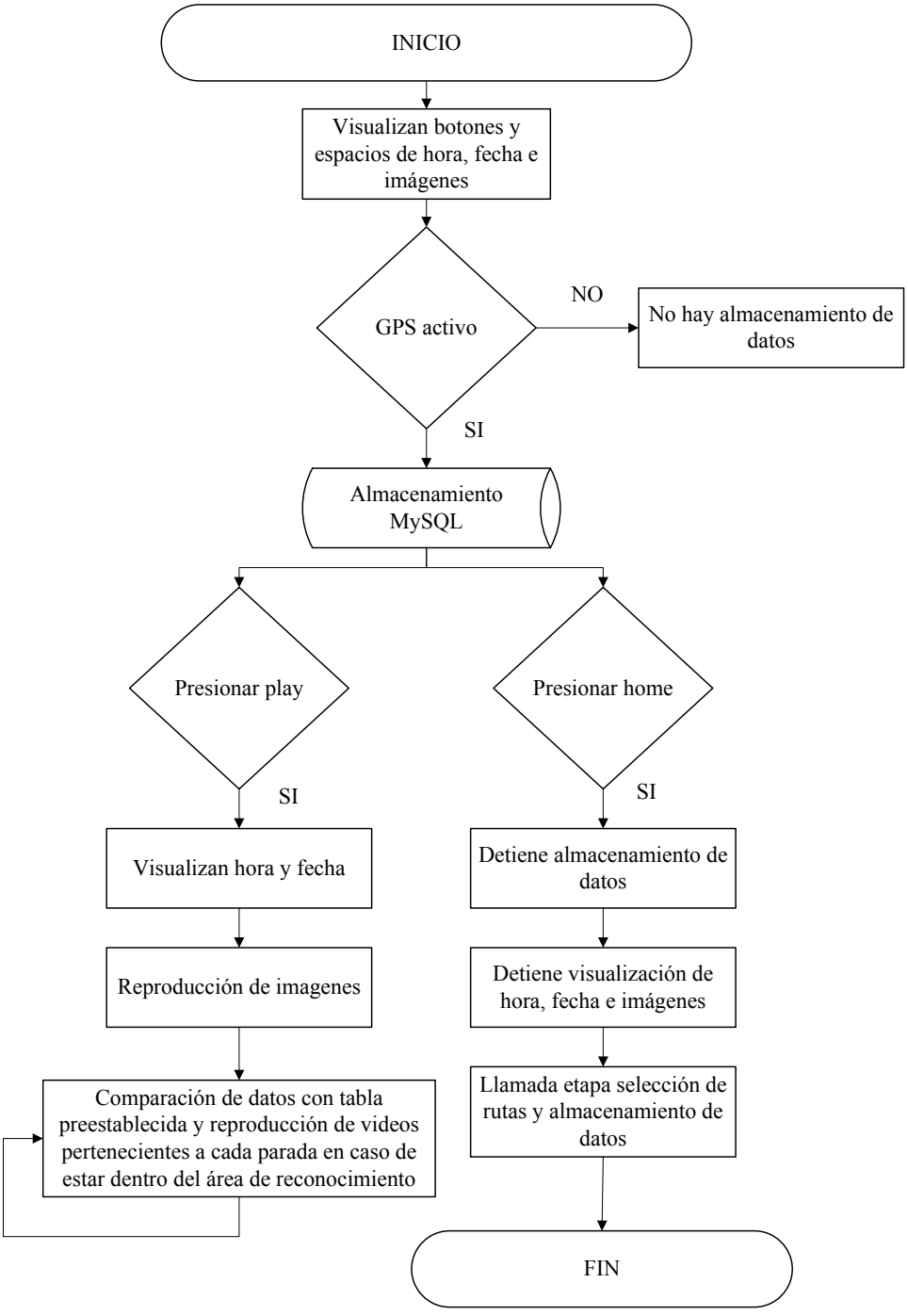


Ilustración 32 Flujograma reproducción de videos
Fuente: (Autores)

***Flujograma fin de funcionamiento y apagado del sistema.* -** Una vez terminados todos los procesos, en la interfaz de inicio se visualiza el botón de apagado, cuando se presiona el mismo da dos opciones, la primera detiene todas las funciones realizadas y apaga completamente la Raspberry Pi, mientras que la segunda retorna a la subrutina de inicio. A continuación, se observa el flujograma de esta subrutina.

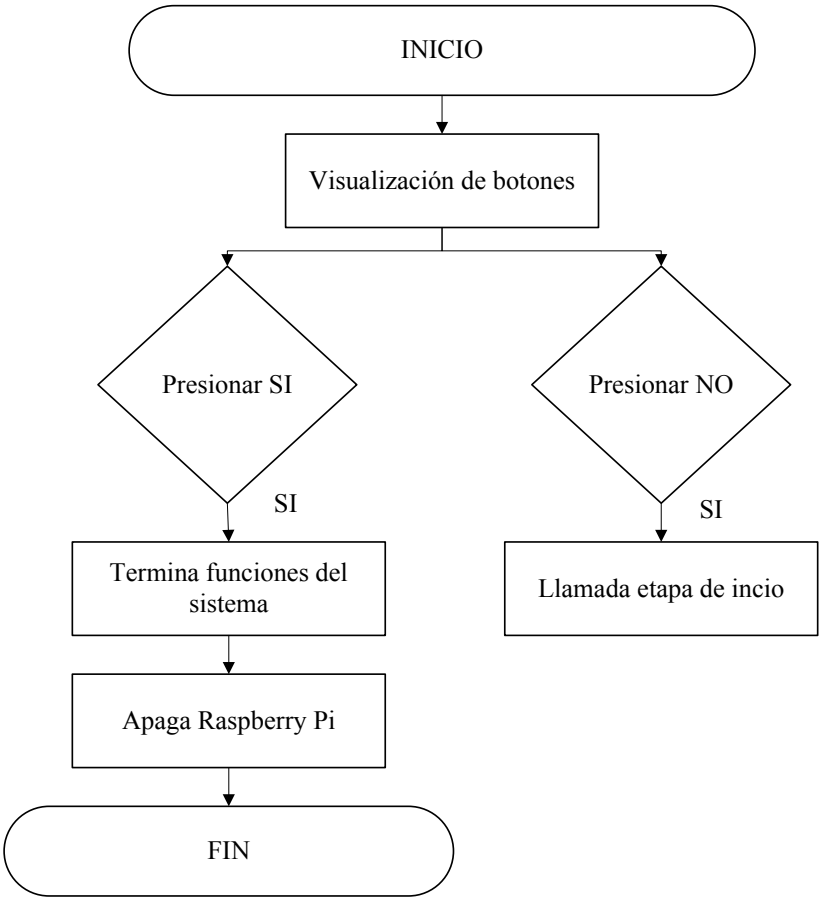


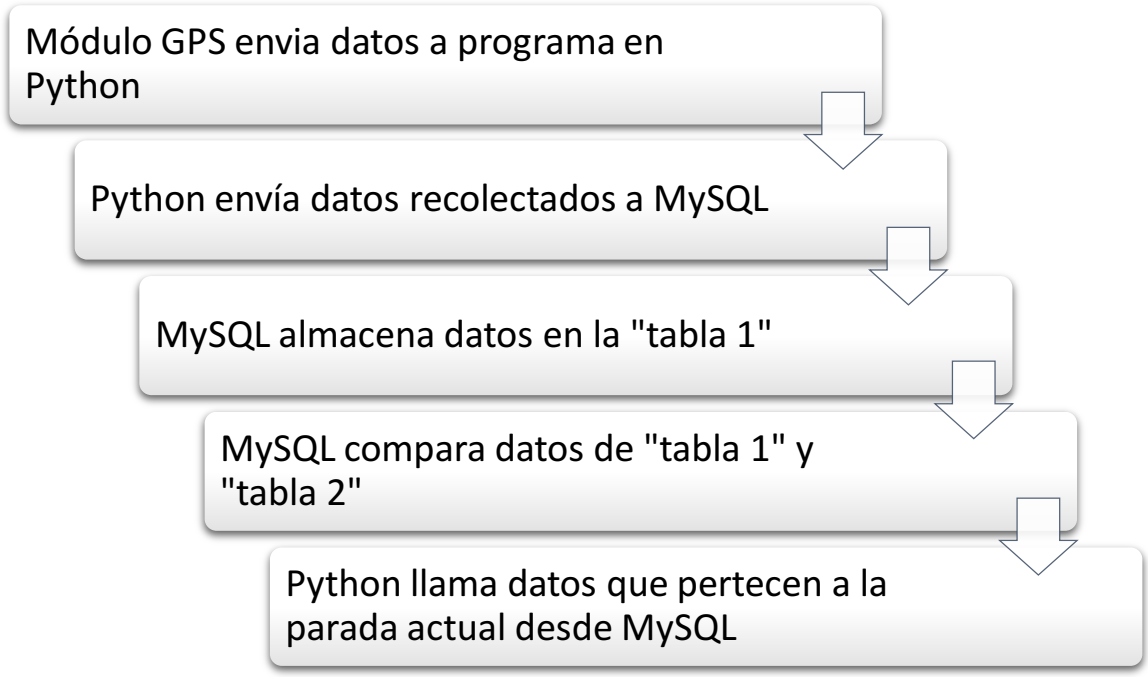
Ilustración 33 Flujograma de fin de funcionamiento y apagado del sistema
Fuente: (Autores)

3.3.2 Comunicación entre módulo GPS y Python.

Los datos del GPS se leen en Python y necesitan ser guardados en una base datos para su correcta utilización, debido a esto se usa MySQL como software adicional. Con la utilización de este programa se creó dos tablas de datos, en la primera se ingresa las coordenadas del GPS a tiempo real y la segunda contiene coordenadas de las paradas que sirven como un medio de comparación con el fin de identificar cuando un autobús se encuentra dentro del área de reconocimiento de una parada.

El funcionamiento en conjunto de estos programas es el siguiente: El módulo GPS envía las coordenadas geográficas, hora, fecha y velocidad a Python mediante la librería GPSD, luego estos parámetros son almacenados y comparados en las tablas de MySQL para finalmente ser llamados en la interfaz de la subrutina de videos. Los pasos ordenados que se cumplen dentro del Raspberry Pi son explicados en la siguiente tabla:

Tabla 19 Pasos de comunicación Raspberry Pi/Módulo GPS



Fuente: (Autores)

3.3.3 Identificación de paradas.

La ciudad de Ibarra se encuentra localizada al norte del país, sus ubicaciones geográficas en grados decimales son: 0,339100 (latitud) y -78,122076 (longitud), se ha empleado las unidades decimales respecto a la ubicación de un punto debido a que el modulo GPS trabaja con los mismos valores. Una vez determinado esto, se procede a realizar una conversión de valores, es decir, establecer cuanto representa un grado, un minuto o un segundo en distancia (metros), tomando en cuenta la zona donde se encuentra situada la ciudad de Ibarra tenemos las siguientes equivalencias para latitud y longitud vs distancia, como se muestra en la siguiente tabla:

Tabla 20 Equivalencias latitud-longitud vs distancia

Grados decimales (Latitud/Longitud)	***Distancia***
1 segundo (1”)	30,92 metros
1 minuto (1’)	1855,42 metros
1 grado (1°)	111,32 kilometros

Fuente: (Autores)

Luego de haber sido reconocidas las paradas pertenecientes a cada ruta, se prosigue por identificar geográficamente cada punto donde se encuentran ubicadas las mismas a través del programa Google Earth como se muestra en la siguiente figura:

Ilustración 34 Paradas ruta Católica – Alpachaca
Fuente: (Autores)

En este caso, se ha tomado como ejemplo la ruta Católica – Alpachaca perteneciente a la Cooperativa de buses “28 de Septiembre”, como se puede observar, los puntos se encuentran identificados sobre el mapa. A continuación, se registran todas estas coordenadas en una tabla para ser ingresadas a la base de datos del sistema, siguiendo el mismo ejemplo. En la siguiente tabla se muestran los valores pertenecientes a las paradas 1-10 de la ruta en mención:

Tabla 21 Coordenadas paradas Católica - Alpachaca

Parada Nro.	*Latitud*	*Longitud*
1	0,363619	-78,133425
2	0,361806	-78,130595
3	0,360089	-78,130399
4	0,358661	-78,128819
5	0,357365	-78,127519
6	0,356148	-78,126056
7	0,353846	-78,124660
8	0,352280	-78,123840
9	0,349652	-78,123175
10	0,347861	-78,122836

Fuente: (Autores)

Al haber sido identificadas geográficamente todas las paradas, se establecieron los puntos denominados “Inicios” y “Finales”, que básicamente son banderas que permiten al sistema reconocer si una parada es de ida o de retorno cuando se encuentran en la misma vía de circulación como una avenida. Estos puntos no representan una parada en sí, pero permiten la creación de tramos, para que, en el caso de que el autobús se desvíe de su trayecto, el sistema pueda reconocer las siguientes paradas de la ruta sin la necesidad de inicializar nuevamente desde un punto único sobre el recorrido. En el caso de ruta mencionada anteriormente, se puede observar estas banderas (Inicio 1 – Fin 1) en la siguiente ilustración:

Ilustración 35 Banderas inicio – fin
Fuente: (Autores)

En la imagen se observa un tramo en la ruta Católica – Alpachaca, este se encuentra definido por un inicio y un final (banderas), para el caso del inicio de ruta, cuando el autobús circule en dirección Sureste, el sistema reconocerá la bandera “Inicio 1” que le permitirá reconocer las paradas 2 – 3 y 4 como se muestran en la ilustración inferior:

Ilustración 36 Trayecto bus sureste
Fuente: (Autores)

En cambio, cuando el autobús se encuentra de retorno y circula por la misma vía en dirección Noroeste, primeramente, se reconocerá el “Fin 1”, que a su vez permite reconocer las paradas: 39 - 40 y 41 en el trayecto de la ruta que se indican en la siguiente ilustración:

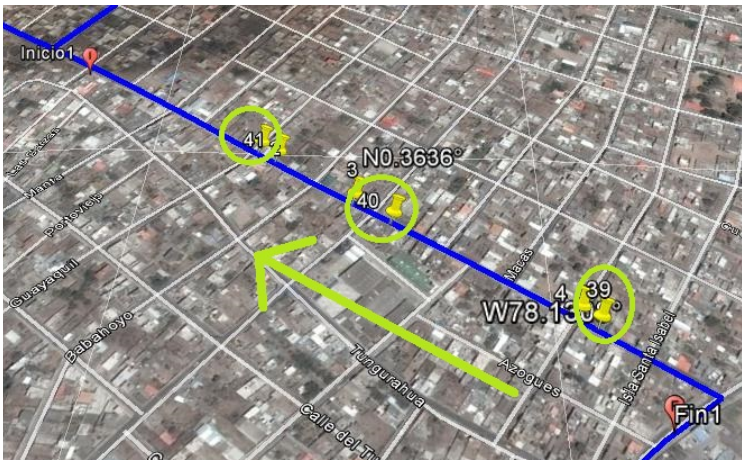


Ilustración 37 Trayecto bus dirección noroeste
Fuente: (Autores)

En relación al radio de las paradas, este se encuadra mediante datos de longitud y latitud, es decir, se toma el punto exacto de cada parada y se lo amplía hasta obtener los 20 metros en coordenadas geográficas. En este sentido, se determina que en programación, para aumentar y disminuir 28 metros a la redonda se usa el valor de 0,0025 grados decimales, como resultado se obtiene valores tales como en la siguiente figura:

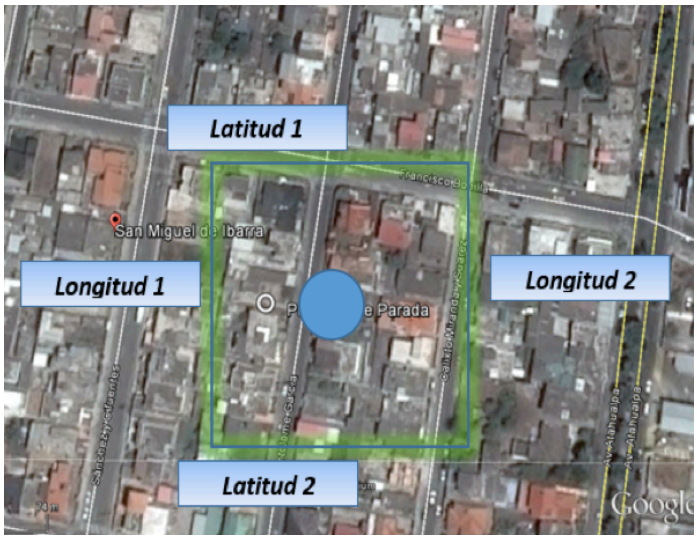


Ilustración 38 Cuadrante Latitud y Longitud
Fuente: (Autores)

3.4 Implementación del sistema.

Para la implementación del sistema de anuncio de rutas y paradas se necesita tomar en cuenta dos aspectos fundamentales, esto se debe a que el conjunto primero debe ser armado por separado para luego ser instalado en el autobús. El trabajo de grado en si está diseñado con el fin de que el montaje en la unidad de transporte público sea lo mas simple y sencilla posible. La implementación del sistema se divide en:

- Armado
- Montaje

3.4.1 Armado.

El armado del conjunto consiste en la conexión, organización y acoplamiento de todos los elementos del sistema de anuncio de rutas y paradas en una sola caja de acrílico diseñada para resistir impactos y otros percances. Además, dicha caja es el centro de operaciones de donde se prende, apaga y manipula todo el sistema y de la cual salen algunos cables que facilitan la conexión y el montaje en el bus.

El paso principal consiste en armar la caja de acrílico que contendrá todos los elementos del trabajo de grado, la misma que tiene por dimensiones 16.1 cm de altura, 19.6 cm de largo y una profundidad de 5.3 cm, estas dimensiones fueron pensadas para que todos los dispositivos que componen el conjunto ingresen sin dificultad.

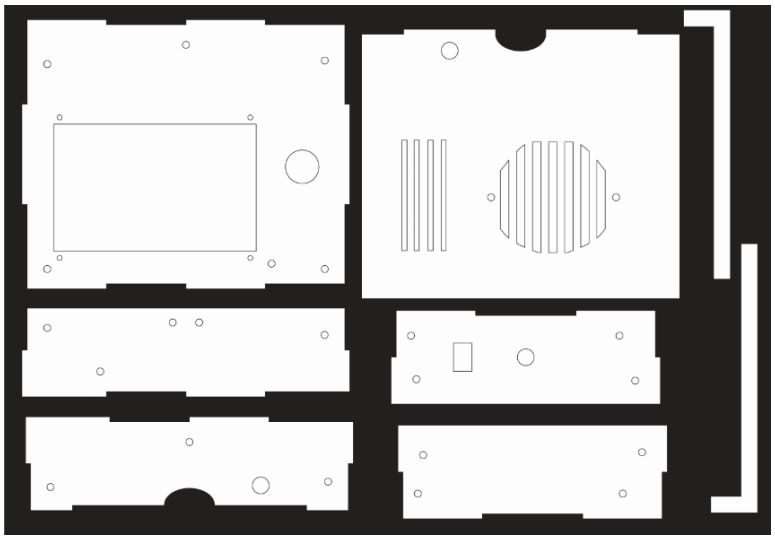


Ilustración 39 Diseño de la caja
Fuente: (Autores)

La caja fue diseñada en el programa Ilustrador en forma de rompecabezas y para no tener margen de error se realizó el corte con laser, el mismo que garantiza que todos los agujeros encajen a la perfección y todas las cavidades sean precisas para los dispositivos respectivos.

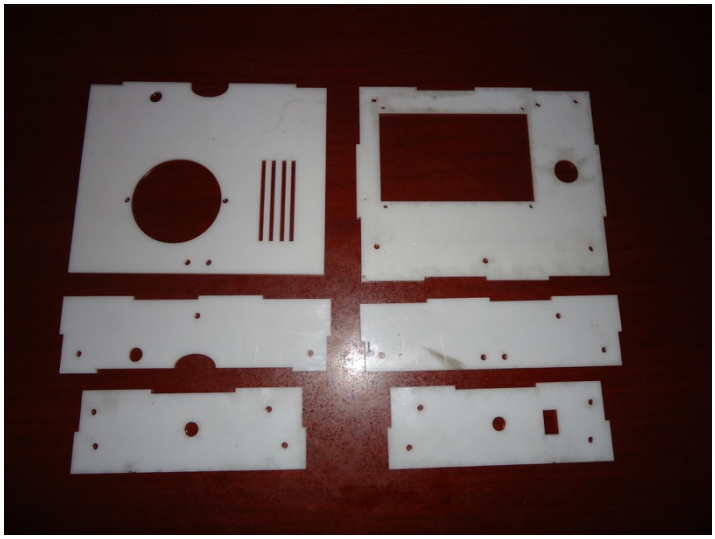


Ilustración 40 Piezas de acrílico
Fuente: (Autores)

Luego de cortar las piezas es necesario quitar la protección plástica transparente que lleva cada pieza y se procede al armado mediante pequeños ángulos acoplados con pernos a cada modulo, esto permite generar una gran resistencia de la caja y un fácil desmontaje de la misma en caso de necesitar mantenimiento.

Ilustración 41 Ángulos metálicos
Fuente: (Autores)

Una vez armada la caja de acrílico se procede a instalar la pantalla táctil, para ello se usan pernos plásticos de gran resistencia que mantienen la pantalla fija a la caja.

También se usa una base de acrílico y tornillos metálicos para adherir la minicomputadora Raspberry Pi en la parte trasera de la pantalla táctil.

Ilustración 42 Armado inicial
Fuente: (Autores)

Cuando se han conectado los dos componentes principales del sistema, se procede a realizar las conexiones respectivas con el fin de que salgan 4 cables principales de la caja: salida HDMI y audio 3.5 para la pantalla Led de 32 pulgadas; salida USB para la comunicación con el GPS y un enchufe para la fuente de energía.

Ilustración 43 Conectores
Fuente: (Autores)

Antes de sellar la caja se realiza pruebas de funcionamiento para verificar el buen comportamiento y comunicación de todos los componentes, también se procede a organizar todos los dispositivos con el fin de generar orden dentro de la caja.

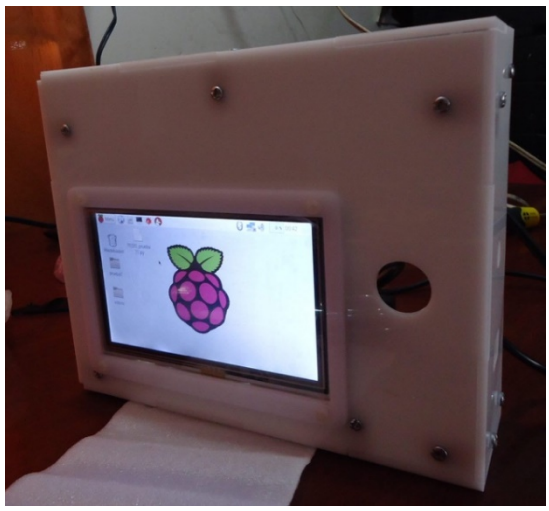


Ilustración 44 Pruebas de funcionamiento
Fuente: (Autores)

Finalmente se instala el botón de encendido y apagado, una salida USB en caso de emergencias y un ventilador que mantiene refrigerado al sistema todo el tiempo. Elaborados estos pasos la caja está lista para el montaje en el autobús.

Ilustración 45 Armado final
Fuente: (Autores)

3.4.2 Montaje del sistema.

En la ciudad de Ibarra existen dos cooperativas de buses urbanos que actualmente transitan las calles de la urbe, en el caso del presente proyecto se ha optado por la implementación en una de las unidades perteneciente a la cooperativa San Miguel de Ibarra. Cabe mencionar que el sistema aún en fase de prueba posee una sola ruta de las totales, por ende, solo podrá entrar en funcionamiento los días en que transite en la ruta designada Esperanza – Hospital del Seguro respectivamente.

Establecimiento de las necesidades de los componentes. - Para el montaje del sistema de anuncio de paradas, se debe tomar en cuenta las especificaciones de funcionamiento que recomienda el fabricante en cada uno de los elementos, los cuales se mencionan a continuación:

- Módulo GPS. - es necesario que la ubicación del mismo sea en un lugar donde la recepción de la señal sea buena ya que mejora su precisión.

- Inversor de voltaje. - el requerimiento para su buen funcionamiento se basa en una buena ventilación, en un lugar seco y libre de humedad.

- Caja controladora. – al albergar la Raspberry Pi y la pantalla táctil en la misma, es necesario que se encuentre en un lugar libre de humedad y que su establecimiento brinde una fácil manipulación al conductor de la unidad.

- Televisión LED 32plg. – su requerimiento básico es una buena ventilación y una excelente visibilidad para los usuarios que estén movilizándose en la unidad.

***Designación de ubicación de los componentes.* -** Una vez determinadas las necesidades que se debe cumplir en cada componente, es hora de designar la ubicación de los mismos dentro del autobús, para lo cual se muestra una imagen a continuación de la cabina:

Ilustración 46 Cabina autobús
Fuente: (Autores)

Tomando en cuenta todas las descripciones citadas anteriormente, se decide que la ubicación de los componentes es la siguiente:

En relación al módulo GPS, es apropiado ubicarlo en el parabrisas frontal del autobús, por la poca interferencia que genera este. Se encuentra sujeto por medio de una ventosa que lo mantiene firme y permite trabajar de la mejor manera, como se muestra a continuación:

Ilustración 47 Ubicación módulo GPS
Fuente: (Autores)

La caja controladora está ubicada en la consola superior izquierda de la cabina, junto al conductor del vehículo, tomando en cuenta que este no sea un distractor y que su manipulación resulte fácil. Está fijada a la consola por medio de dos platinas que la mantienen fija en la unidad como se indica en la imagen inferior:

Ilustración 48 Ubicación caja controladora
Fuente: (Autores)

El inversor de voltaje está colocado en la consola del autobús debido a sus buenas características de ventilación, así como la facilidad de conexión entre la batería del vehículo y los demás elementos, su ubicación se muestra en la ilustración a continuación:

Ilustración 49 Ubicación inversor de voltaje
Fuente: (Autores)

En mención a la televisión LED de 32 plg., es indispensable tomar en cuenta que su ubicación debe ser en un lugar estratégico, el cual sea visible desde cualesquiera de los asientos de pasajeros de la unidad, por esta razón se va a montar en una estructura de tubos que se encuentran detrás del asiento del conductor, tal como se indica en la imagen inferior:

Ilustración 50 Ubicación televisor LED
Fuente: (Autores)

Por último, se muestra una imagen donde se observa el sistema montado completamente sobre el autobús, listo para funcionar y brindar la ayuda necesaria a personas con algún tipo de discapacidad y los 14 000 habitantes que usan diariamente el transporte público en la ciudad de Ibarra.

Ilustración 51 Montaje final sistema
Fuente: (Autores)

CAPÍTULO IV

4. PRUEBAS DE FUNCIONAMIENTO Y ANÁLISIS DE DATOS

4.1 Pruebas de hadware.

Una vez hechas las conexiones respectivas se procede a las pruebas de funcionamiento de cada uno de los dispositivos, es decir se analiza que estos se prendan correctamente y funcionen sin ningun inconveniente, a continuación, se explica detalladamente cada uno de los dispositivos.

4.1.1 Prueba de funcionamiento del Inversor.

El inversor es un dispositivo que permite cambiar los 12 V de corriente continua de la batería de los autos a 110 V de corriente alterna. Una vez conectado dicho dispositivo a la batería del autobús, solo se necesita encender y conectar la caja controladora y la pantalla Led de 32 pulgadas para que todo el sistema de anuncio de rutas y paradas funcione correctamente.

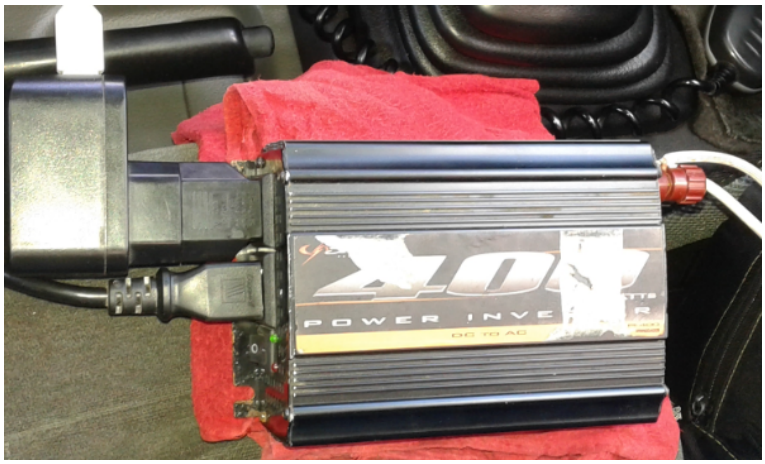


Ilustración 52 Pruebas de funcionamiento del inversor
Fuente: (Autores)

Para conocer que el inversor de voltaje funciona correctamente, fue necesario realizar una comparación entre las medidas que brinde un multímetro automotriz en la corriente eléctrica de una casa y la que proporciona el inversor, en la ilustración inferior se puede observar dicha comparación:

Ilustración 53 Prueba funcionamiento inversor de voltaje
Fuente: (Autores)

La imagen de la derecha representa le valor que se obtuvo al conectar el multímetro en el tomacorriente de la casa y la imagen de la izquierda el valor que se adquirió al conectar el multímetro al inversor cuando este se encuentra en pleno funcionamiento. Como se distingue en la ilustración los valores del voltaje son similares, sacando como conclusión que los dispositivos que se usarán van a tener un funcionamiento correcto.

4.1.2 Prueba de la caja controladora.

En su interior contiene la mini computadora Raspberry Pi, sistema de refrigeración, pantalla táctil y botón de encendido. Una vez conectado al inversor, se inicia el sistema al aplastar el botón de encendido, el mismo que enciende una luz led. Luego se observa el funcionamiento correcto del ventilador y finalmente la pantalla táctil de 5 pulgadas enciende correctamente.

Una vez encendido todo el sistema se debe percatar que el táctil de la pantalla funcione sin problemas y además que todas las interfaces inicien sin dificultad, lo que garantiza que este dispositivo está funcionando a la perfección.

Ilustración 54 Pruebas de funcionamiento de la caja controladora
Fuente: (Autores)

4.1.3 Pruebas de conexión y ubicación del módulo GPS.

Dicho módulo se encuentra fuera de la caja controladora y permite la identificación de una zona determinada mediante coordenadas geográficas. Para conocer que el módulo está funcionando correctamente se realiza una comparación de coordenadas (latitud, longitud) entre el módulo GPS y un Smartphone. A continuación, se muestran las imágenes de los datos obtenidos en cada uno de los dispositivos:

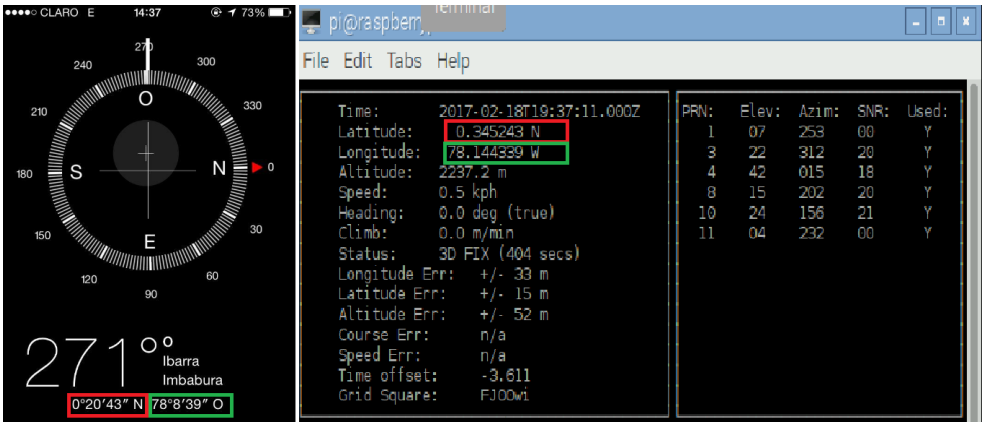


Ilustración 55 Coordenadas Smartphone vs módulo GPS
Fuente: (Autores)

Los valores correspondientes a la imagen de la izquierda son los obtenidos en el Smartphone, mientras que la imagen de la derecha refleja las coordenadas del módulo GPS, haciendo referencia que ambos dispositivos se encontraban en la misma posición al momento de realizar la prueba. Los datos se muestran en la tabla inferior, tomando en

cuenta que el Smartphone provee las coordenadas en grados, minutos y segundos fue necesario transformarlos a grados decimales en dicha comparación:

Tabla 22 Coordenadas Smartphone vs módulo GPS

Smartphone		***Módulo GPS***	
Latitud	***Longitud***	***Latitud***	***Longitud***
0,345240	-78,144340	0,345243	-78,144339

Fuente: (Autores)

Como se aprecia dichos valores, se concluye que el módulo GPS está trabajando correctamente y las coordenadas son similares a las obtenidas en el celular, sin olvidar que el error promedio de exactitud que nos ofrece es de alrededor de ±5 metros en la ubicación.

4.1.4 Pruebas de la pantalla Led de 32 pulgadas.

Pantalla Led de 32 pulgadas es usada para la visualización de videos, es importante recalcar que la conexión de la misma debe ser mediante puerto HDMI. Es te dispositivo no debe presentar problemas y debe mostrar la misma imagen que se muestra en la caja controladora.

Ilustración 56 Pruebas de funcionamiento del televisor LED
Fuente: (Autores)

4.2 Pruebas software de aplicación.

El software de aplicación es un programa que controla directamente a todos los dispositivos que componen el hadware, en el caso del sistema de anuncio de rutas y paradas, el software de aplicación permite ejecutar correctamente al módulo GPS, pantalla táctil, Raspberry PI y demás dispositivos mediante programación, es decir, controla el funcionamiento físico de los componentes usados en el trabajo de grado. En este ítem se analiza el funcionamiento de la programación que controla el hadware en si.

4.2.1 Pruebas de recepción de datos del módulo GPS.

Los módulos GPS tradicionales tienen conexión a través del puerto GPIO, sin embargo, la nueva generación de módulos denominada Ublox-7, realiza esta conexión por medio del Universal Serial Bus (USB). Para la correcta comunicación del GPS con la Raspberry Pi se descargó la librería gratuita GPSD, con ella es posible visualizar los datos a través de una comunicación serial o en forma de consola desde la terminal. En las imágenes siguientes se presenta las dos formas de visualización de datos que la librería GPSD permite.

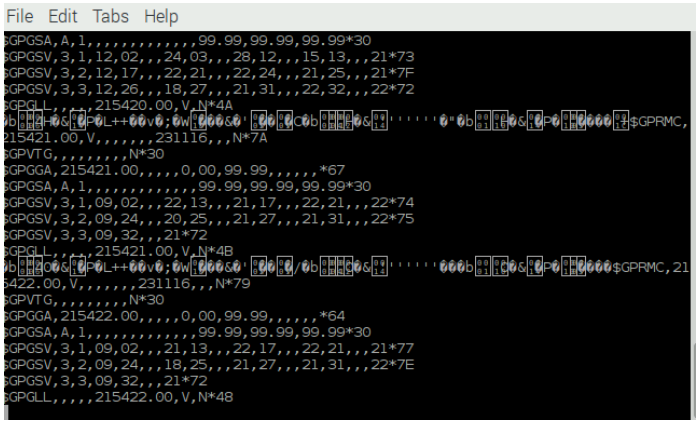


Ilustración 57 Comunicación serial GPS
Fuente: (Autores)

La comunicación serial se realiza inmediatamente y se la puede visualizar con un comando directamente desde la terminal, estos datos son desordenados y necesitan ser procesados antes de su utilización.

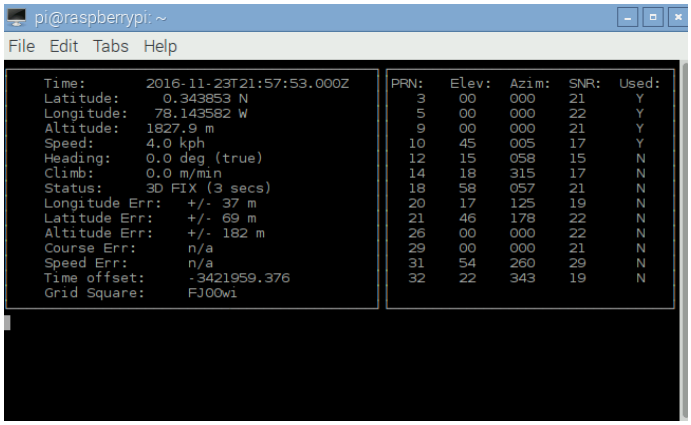


Ilustración 58 Consola GPS
Fuente: (Autores)

Al igual que la comunicación serial, para visualizar la consola de datos del GPS en la terminal es necesario de un comando, sin embargo, los datos son más ordenados y se pueden usar de una forma más sencilla. Los parámetros obtenidos del módulo GPS de esta manera son enviados a Python para el funcionamiento del sistema de anuncio de rutas y paradas como se muestra en la siguiente figura.

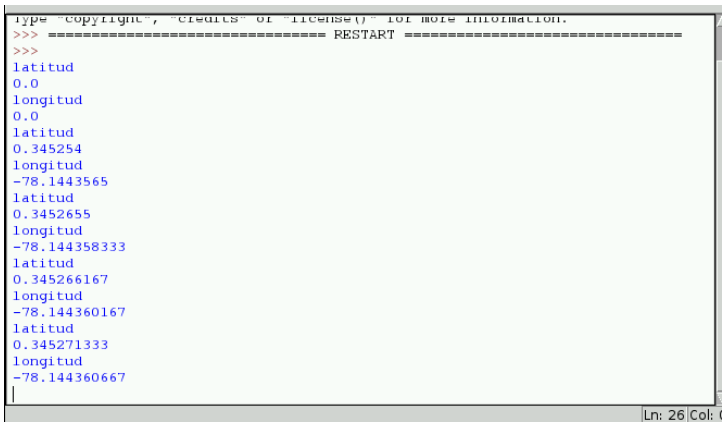


Ilustración 59 Python GPS
Fuente: (Autores)

4.2.2 Prueba de almacenamiento de datos.

Los datos del GPS enviados a Python son momentáneos y en tiempo real, por lo que es necesario usar una base de datos en un sistema de gestión llamado MySQL dentro de la terminal en la Raspberry Pi. Las coordenadas son almacenadas en tres tablas, la primera tabla llamada “bus_ruta” guarda los datos que vienen desde Python y

la segunda y tercera tabla llamadas “bus_esperanza” y “bus_catolica” es un medio de comparación que en caso de ingresar al área de una parada retorna el nombre del video nuevamente a la interfaz de Python para su posterior reproducción. En las siguientes imágenes se observan las dos tablas usadas en el sistema.

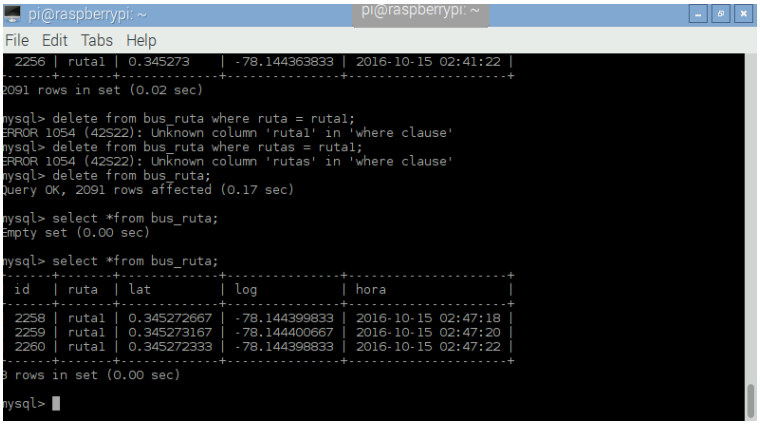


Ilustración 60 Tabla bus_ruta
Fuente: (Autores)

La base de datos se denomina “gpsbus”, en ella están integrados la tabla “bus_ruta” apreciada en la ilustración anterior y contiene: nombre de la ruta seleccionada, latitud a tiempo real, longitud a tiempo real, fecha actual y hora actual

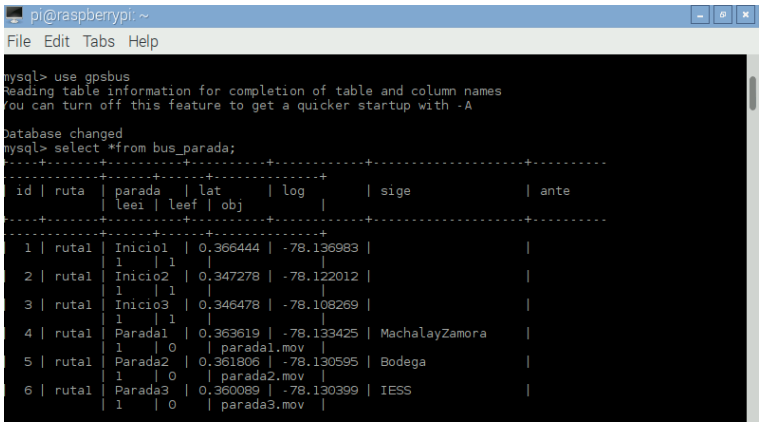


Ilustración 61 Tabla bus_catolica
Fuente: (Autores)

La tabla “bus_catolica” al igual que la tabla anterior esta contenida en la base de datos “gpsbus” y contiene: nombre de la ruta, nombre de la parada, latitud y longitud de una parada específica, parada siguiente, parada anterior, parámetros de inicio,

parámetros de final y el objeto a reproducirse, tal y como se aprecia en la ilustración anterior.

4.2.3 Prueba de la interfaz del sistema.

El sistema de anuncio de rutas y paradas cuenta con una pantalla táctil de 5 pulgadas, en ella se observa la interfaz con la que el operador interactúa. Las funciones de cada uno de los botones permiten navegar por las etapas de programación del prototipo y se explican con claridad en la siguiente tabla:

Tabla 23 Función íconos menú

Icono	***Función***
	Permite elegir las rutas pertenecientes a la Coop. “28 de Septiembre”
	Permite elegir las rutas pertenecientes a la Coop. “San Miguel de Ibarra”
	Muestra la pantalla de apagado del sistema
	Retorna a la pantalla de inicio
	Apaga el sistema
	Botón de texto variable que permite ingresar a la interfaz de videos
	Botones para selección de rutas
	Botón “Siguiente”, ayudan con la visualización de más rutas
	Botón “Home”, regresan al inicio o a la selección de rutas.
	Botón “Play”, inicia la reproducción de videos, hora, fecha, parada actual y parada siguiente.

Fuente: (Autores)

Al tener dos cooperativas, algunos botones tienen diseños diferentes, pero funciones similares. Durante el funcionamiento del sistema de anuncio de rutas y paradas el operador puede manipular y visualizar cinco ventanas que indican las siguientes actividades:

- Inicio.
- Coop. 28 de Septiembre.
- Coop. San Miguel de Ibarra
- Videos.
- Apagado

Inicio.

En esta interfaz se visualiza en la parte superior el escudo de la Universidad Técnica del Norte, el isologo de la carrera de Ingeniería en Mantenimiento Automotriz y el escudo de la ciudad de Ibarra. En la parte inferior se observan tres botones principales, dos de ellos permiten elegir las rutas pertenecientes a cada cooperativa de buses, mientras que el botón central permite ingresar a la interfaz de apagado del sistema.

Ilustración 62 Interfaz inicio
Fuente: (Autores)

Cooperativa 28 de Septiembre.

Luego de seleccionar el botón perteneciente a la Coop. 28 de Septiembre se ingresa a la interfaz secundaria donde se encuentran las 14 rutas de la misma, esta ventana contiene los siguientes botones:

- "Home", permite regresar al inicio.
- "Siguiente", permite ingresar a una nueva ventana con más rutas.
- "Seleccionar Ruta", da comienzo a la ventana de videos.
- Seis botones que permiten cargar información sobre las rutas de la cooperativa.

Para la elaboración del prototipo se encuentra predeterminada una ruta de prueba en la interfaz de esta cooperativa, sin embargo, los demás botones se encuentran habilitados, es decir tienen la capacidad de cargar la información que falta y el sistema en general soporta todas las rutas existentes en la ciudad.

Ilustración 63 Interfaz Coop. 28 de Septiembre
Fuente: (Autores)

Cooperativa San Miguel de Ibarra.

Al aplastar el botón perteneciente a la Coop. San Miguel de Ibarra se ingresa automáticamente a las 10 rutas pertenecientes a la misma, esta ventana contiene funciones similares a la interfaz mencionada anteriormente, diferenciándose tan solo por el color de los iconos. Las rutas de prueba en este caso es una, al igual que la ventana de la Coop. 28 de Septiembre, los botones están habilitados para más rutas. A continuación, se observa la interfaz de este ítem.

Ilustración 64 Interfaz Coop. San Miguel de Ibarra
Fuente: (Autores)

Videos.

Para ingresar a esta interfaz, primero es necesario escoger un botón que cargue la información perteneciente a la ruta que se desea, luego se procede a presionar el botón de texto variable "Seleccionar Ruta" que por default cambia de nombre a la ruta anteriormente seleccionada, una vez cumplido con este procedimiento la ventana que se muestra contiene los siguientes parámetros:

- Ruta, muestra el nombre de la ruta seleccionada.

- Hora, muestra la hora actual.
- Fecha, muestra la fecha actual.
- Imágenes, muestra imágenes informativas, turísticas y publicitarias en intervalos de 7 segundos.
- Parada actual, muestra el nombre de la parada por la que el bus pasó o esta pasando.
- Parada siguiente, muestra el nombre de la parada que sigue en la ruta.
- "Play", permite que arranque el sistema y la reproducción de hora, fecha e imágenes, mientras no se aplaste este icono la interfaz permanecerá vacía.
- "Home", termina todos los procesos y retorna a ventana de la cooperativa.

Dentro de esta interfaz se realiza el proceso de recolección y almacenamiento de datos provenientes del módulo GPS, gracias a esto, cuando el autobús se encuentra en la zona de reconocimiento de una parada, se reproduce un video informativo que indica la parada actual y la parada siguiente en toda la pantalla para su correcta visualización, caso contrario, si la unidad de transporte no ha pisado dicha zona, la interfaz muestra todas las opciones preestablecidas a excepción del producto audiovisual informativo como se observa en la siguiente ilustración:

Ilustración 65 Interfaz videos
Fuente: (Autores)

Apagado.

Al ingresar al botón de apagado se visualiza una interfaz con dos iconos, el de color rojo con una "X" en el centro cancela la operación de apagado y muestra el inicio nuevamente, mientras que el de color verde con un "visto" termina todos los procesos, apaga el sistema y la terminal, a continuación, se observa dicha ventana:

Ilustración 66 Interfaz apagado
Fuente: (Autores)

4.2.4 Pruebas de funcionamiento del sistema.

Luego de seleccionar la cooperativa y la ruta respectiva se ingresa a la interfaz de videos la misma que muestra hora, fecha, ruta, parada actual, parada siguiente e imágenes. En la siguiente ilustración se observa el correcto funcionamiento de estos procesos.

Ilustración 67 Hora y fecha.
Fuente: (Autores)

Los procesos mencionados anteriormente no deben detenerse en ningún punto del trayecto, al inicio y final de la ruta se muestra un video que informa al conductor que el sistema esta en funcionamiento, sin embargo, una vez que la unidad de transporte ingresa al área de reconocimiento de la parada, un video informativo se reproduce sin dificultad. La estructura del video está diseñada para que el usuario no tenga dificultad al momento de usar el transporte público urbano, es decir, genera un contenido visual y auditivo. El producto audiovisual se divide en dos secciones las cuales son:

- La primera sección informa sobre la parada actual por la cual el bus transita, indica una imagen y reproduce un audio de la misma.

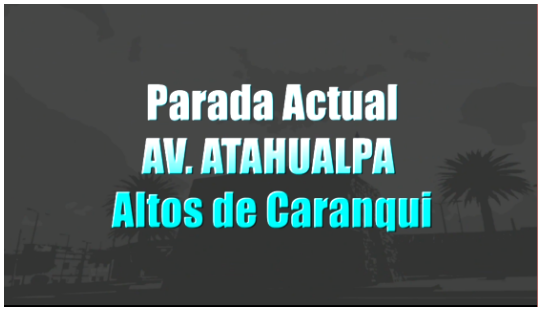


Ilustración 68 Nombre parada actual
Fuente: (Autores)

En cada uno de los videos informativos se aprecia una fotografía actualizada del lugar mencionado, además las animaciones de las letras son personalizadas por tramos, dependiendo del tramo cambia la animacíon generando mayor interés en el usuario.

Ilustración 69 Imagen parada actual
Fuente: (Autores)

- La segunda sección visualiza una animación de la parada altual y siguiente, mientras reproduce un audio de la misma como se muestra a continuación:

Ilustración 70 Animación parada actual
Fuente: (Autores)

Las animaciones están elaboradas según un mapa actualizado del lugar con el fin de que el público e incluso personas desconocidas y turistas se ubiquen con mayor facilidad.

Ilustración 71 Animación parada siguiente
Fuente: (Autores)

Al igual que los videos informativos, los productos audiovisuales de inicio y fin, poseen la misma estructura, el sistema está elaborado para que el conductor únicamente seleccione la cooperativa y la ruta a recorrer, luego de ese proceso el sistema funciona automática e inteligentemente reproduciendo videos y realizando funciones sin necesidad del operador.

4.3 Manual de usuario

El sistema de anuncio de rutas y paradas piensa mejorar la información brindada en el servicio de transporte público con el fin de generar una comunicación útil a los usuarios y terminar con los problemas de comunicación, además de ayudar a los habitantes de la ciudad con la correcta utilización del autobús.

4.3.1 Componentes.

Como ya se observó en las pruebas de funcionamiento de hadware, el sistema de anuncio de rutas y paradas consta de los siguientes componentes:

- Caja controladora: Contiene la mini computadora Raspberry Pi, sistema de refrigeración, pantalla táctil y botón de encendido.

Ilustración 72 Caja controladora
Fuente: (Autores)

- Módulo GPS: Se encuentra fuera de la caja controladora y permite la identificación de una zona determinada mediante coordenadas geográficas.

Ilustración 73 Módulo GPS
Fuente: (Autores)

- Pantalla Led de 32 pulgadas: Pantalla Led de 32 pulgadas para la visualización de videos, es importante recalcar que la conexión de la misma debe ser mediante puerto HDMI.

- Inversor: Permite cambiar los 12 V de las baterías de los autos a 110 V necesario para que funcione todo el sistema.

4.3.2 Conexión.

De la caja controladora salen 3 cables: un cable USB hembra para la conexión con el módulo GPS, un cable HDMI macho para la conexión con la pantalla Led de 32 pulgadas y un enchufe para la conexión con la fuente de energía. En la siguiente ilustración se observa a detalle los conectores necesarios para que el sistema entre en funcionamiento.

Ilustración 74 Conectores del sistema
Fuente: (Autores)

Los pasos a seguir para que la conexión sea exitosa son los siguientes:

- Conectar el módulo GPS con el cable USB que sale de la caja controladora.
- Conectar la televisión Led de 32 pulgadas con el cable HDMI que sale de la caja controladora.
- Conectar el enchufe al inversor de corriente.

4.3.3 Inicio del sistema.

El sistema de anuncio de rutas y paradas posee un proceso ordenado para su encendido y apagado el cual se explica paso a paso a continuación, es necesario seguir con las indicaciones expuestas, caso contrario, se podria causar daño a los componentes del sistema.

Encendido.

Antes de usar el sistema de anuncio de rutas y paradas es necesario verificar que todos los dispositivos estén completos y en buenas condiciones, luego de realizar las conexiones respectivas se procede a prender el sistema, para lo cual es necesario seguir los siguientes pasos:

- Encender el inversor de corriente.
- Encender la pantalla Led de 32 pulgadas.
- Aplastar el botón de encendido de la caja controladora, el sistema está en funcionamiento mientras este botón muestre una luz roja tal como se muestra en la ilustración.

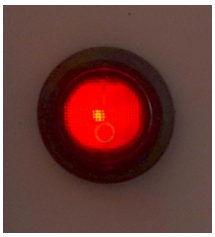


Ilustración 75 Botón encendido
Fuente: (Autores)

Apagado.

Luego de realizar todo el recorrido en el autobús es necesario apagar todo el sistema para evitar fallos, para lo cual es necesario cumplir con los siguientes pasos.

- Apagar el sistema desde la interfaz de la caja controladora.
- Apagar la pantalla Led de 32 pulgadas.
- Apagar el inversor de corriente.

- Aplastar el botón de encendido de la caja controladora, el sistema está apagado por completo cuando este botón ya no muestra una luz roja.

Ilustración 76 Botón apagado
Fuente: (Autores)

4.3.4 Menú funcional.

Como ya se expuso en el item anterior correspondiente a las pruebas de firmware, el sistema de anuncio de rutas y paradas consta de algunas interfaces, las cuales serán analizadas a continuación, con el fin de mejorar el reconocimiento de los elementos que la componen.

Inicio.

Esta interfaz consta del escudo de la Universidad Técnica del Norte, el isologo de la carrera de Ingeniería en Mantenimiento Automotriz y el escudo de la ciudad de Ibarra. Además de tres botones principales tal y como se muestra en la siguiente ilustración.

Ilustración 77 Inicio
Fuente: (Autores)

Selección de rutas.

Luego de seleccionar el botón perteneciente a la Coop. 28 de Septiembre se ingresa a la interfaz secundaria de la misma. Esta ventana contiene los botones expuestos en la ilustración siguiente.

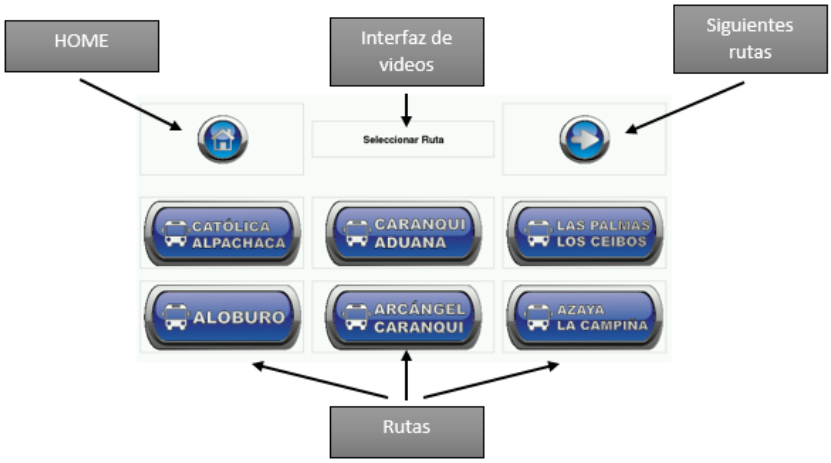


Ilustración 78 Interfaz secundaria
Fuente: (Autores)

Videos.

Para ingresar a esta interfaz, primero es necesario escoger un botón que cargue la información perteneciente a la ruta para luego se procede a presionar el botón de texto variable "Seleccionar Ruta", una vez cumplido con este procedimiento la ventana que se muestra contiene los siguientes parámetros:

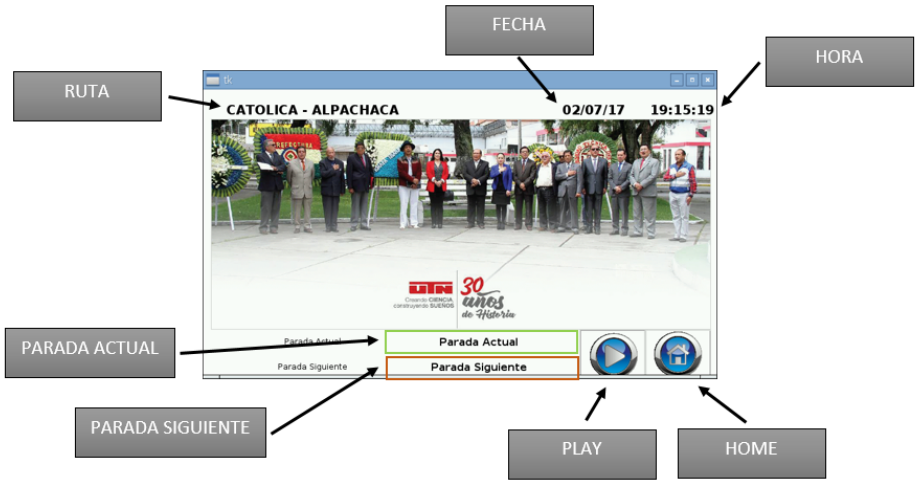


Ilustración 79 Interfaz videos
Fuente: (Autores)

Apagado.

Al ingresar al botón de apagado se visualiza una interfaz con dos iconos, el uno cancela la operación de apagado y muestra el inicio nuevamente, mientras que el segundo termina todos los procesos, apaga el sistema y la terminal. A continuación, se observa dicha ventana:

Ilustración 80 Interfaz apagado
Fuente: (Autores)

4.3.5 Precauciones.

Una vez iniciado el sistema de anuncio de rutas y paradas es necesario tomar en cuenta las áreas de reconocimiento y banderas del sistema. En caso de que el operador no siga con la ruta o inicio establecido, el sistema no leerá un total de una o dos paradas dependiendo de la ruta, es decir, primero se debe reconocer cualquier bandera colocada a lo largo de la ruta para luego leer la parada, sin embargo, el lapso de tiempo que se demora en efectuar este procedimiento es mínimo, además, con esto se garantiza que los autobuses inicien donde es debido y no se salten las paradas. Para el correcto funcionamiento del sistema de anuncio de rutas y paradas se debe seguir las siguientes recomendaciones:

- No apagar ni desconectar el inversor de corriente mientras el sistema de anuncio de rutas y paradas está en funcionamiento.

- No aplastar el botón de encendido mientras el sistema de anuncio de rutas y paradas está en funcionamiento.
- No encender el sistema de anuncio de rutas y paradas mientras la televisión de 32 pulgadas este apagada.
- No acercar o usar elementos filosos para la manipulación de la pantalla táctil instalada en la caja de controladora.
- No abrir la caja controladora sin la ayuda de un técnico especializado.

CAPÍTULO V

5. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Los datos brindados por parte de MOVIDELNOR a cerca de las rutas y paradas son obsoletos, debido a que la información que poseen no se ha actualizado desde el año 2006, por lo cual fue necesario hacer un levantamiento de datos por parte de los autores en cada una de las dos rutas analizadas.

- El dispositivo utilizado como fuente de procesos llamado Raspberry Pi 2 modelo B cumplió con las exigencias que requiere el sistema, con su procesador de 900 MHz al momento de analizar información, así como su capacidad de reproducción de videos en HD. Sus cuatro puertos USB y el puerto GPIO fueron de gran importancia ya que permitieron la conexión de todos los dispositivos necesarios para el anuncio de paradas.

- Para un correcto funcionamiento y una comunicación adecuada entre la Raspberry Pi y la pantalla táctil, fue necesario instalar un sistema operativo denominado RPI_B_5.0.img. Así mismo, para la visualización en ambas pantallas se utilizó una Y (HDMI) que permite conectar dos pantallas simultáneamente al dispositivo sin perder señal en ninguna de las dos.

- Para un correcto funcionamiento de la Raspberry Pi con los demás componentes fue necesario la instalación de algunas librerías o configuraciones que permitieron habilitar puertos, lectura de datos, reproducción de audio, imágenes, gifs y videos, así como actualizar el reloj, las cuales son: Omxplayer, Photoimage, GPSd, MySQLdb entre otras.

- El módulo GPS U-blox 7 tiene un margen de error de ±4 metros, lo cual no afecta en el funcionamiento de este sistema, ya que se trabaja con áreas de 45 metros cuadrados para el reconocimiento de las paradas. Cabe recalcar, que su conexión a través de un puerto USB es de gran ayuda al momento de montar en la Raspberry Pi, porque permite una conexión segura libre de errores.

- Para facilitar la lectura de coordenadas mediante el programa Python, se usó la ayuda de una base de datos en MysQl, en la que se crearon tres tablas base para el prototipo. En la principal de ellas, se almacenan todos los datos provenientes del GPS a excepción del valor: "0,00" y "NAN" cada 2 segundos; en las dos tablas restantes se ingresan los datos de cada parada pertenecientes a una ruta determinada. Mediante este proceso se asegura la lectura correcta de cada parada sin que estas se confundan entre sí.

- El programa Google Earth, fue utilizado para la extracción de las coordenadas de cada parada, sin embargo, estos datos difieren de los que se pueden obtener en un Smartphone o a su vez del sistema en sí mismo, generando un error aproximado de 0.000022 grados decimales dentro de latitud y longitud, lo mismo que en distancia supone un rango de 3 metros, recalcando una vez más que este margen de error es depreciable, ya que no incide en las áreas de reconocimiento de cada parada.

- Una forma de solucionar el reconocimiento de paradas que se encuentran en una misma avenida, pero en distinta dirección, fue la implementación de alrededor de 10 puntos ficticios denominados "Banderas", que básicamente ayudan a determinar si el autobús está avanzando o regresando de su ruta, evitando errores en el reconocimiento de la parada. Con esto se garantiza la lectura de una parada sin importar que el autobús inicie o no en el lugar de descanso.

- La cantidad de paradas que se obtuvo en la ruta Esperanza - Hospital del Seguro es de 58 paradas oficiales y en la ruta Católica – Alpachaca 44 paradas, por lo cual fue necesario desarrollar un video informativo para cada una de ellas, es decir, un total de 102 videos elaborados por parte de los autores del presente

trabajo de grado. Estos videos fueron importados en formato HD (1920 x 1080) con un tamaño de alrededor de 8mb por unidad y con una duración de 16 segundos.

- En relación al costo del sistema, se puede dividir en dos partes. La primera, hace referencia al hardware utilizado que tiene un costo de 550 dólares y la segunda, al programa de aplicación desarrollado; la creación de videos informativos y el levantamiento de datos que tienen un valor de 200 dólares. Con esto da un total de 750 dólares que supone el costo de producción unitario, en caso de ser una producción a gran escala, estos valores disminuirían.

5.2 Recomendaciones

- Debido al desarrollo vial de la ciudad y el aumento del parque automotor, se recomienda realizar estudios de reconocimiento y actualización de paradas de cada ruta pasando cuatro años. Con el objetivo de que la información manejada en el presente trabajo de grado no sea obsoleta en el futuro en caso de no ser actualizada.

- Para futuros proyectos se sugiere la utilización de la Raspberry Pi 3 debido a que en esta vienen incorporadas redes inalámbricas como Wifi y Bluethooth que ayudan en la instalación de nuevos softwares y dispositivos, facilitando el uso de tecnologías móviles y de multimedia, además, este nuevo dispositivo es más robusto y compatible con variados y novedosos elementos tecnológicos.

- Se recomienda definir correctamente las áreas de cada parada y los cuadrantes de reconocimiento de las mismas, de preferencia es necesario delimitar adecuadamente una anticipación con el fin de que el sistema desempeñe su trabajo y ayude al usuario informando sobre la parada a la que llega, de esta manera el pasajero tiene tiempo para analizar la información que brinda el sistema y a su vez realizar una acción previamente al arribo del autobús.

- El sistema de anuncio de rutas y paradas puede ser usado como soporte publicitario en autobuses, ya que dicho prototipo está en la capacidad de reproducir videos promocionales, es decir, la interfaz gráfica está sujeta a variaciones con el fin de generar publicidad, sin embargo, se recomienda mantener la base de programación para que el funcionamiento del sistema no se vea afectado. Cabe recalcar que la publicidad es un medio de ingreso económico rentable, por lo tanto, con la utilización de este servicio, el sistema de anuncio de rutas y paradas sería autosustentable.

- Durante el diseño y armado de un sistema es necesario tomar en cuenta las recomendaciones del fabricante con el fin de alargar la vida útil de los dispositivos y garantizar la eficiencia del sistema en su funcionamiento, es decir, se sugiere la correcta manipulación de cada elemento y del conjunto en sí, evitando el contacto con líquidos u accidentes varios que pueden afectar a su funcionamiento.

- Una vez inicializado el sistema de anuncio de rutas y paradas es recomendable seguir las indicaciones que vienen en el manual de usuario y sobre todo evitar la desconexión de los componentes mientras el sistema se encuentra encendido ya que esto produce errores en su funcionamiento.

- Durante la elaboración de los videos, se propone que los mismos no tengan una duración muy extensa ya que esto impide el reconocimiento adecuado de las paradas cuando el bus transita por su ruta.

- Se sugiere que en proyectos futuros el sistema de anuncio de rutas y paradas actúe como complemento de las actuales paradas inteligentes implementadas en la ciudad de Ibarra, además, se recomienda realizar una comunicación bilateral entre estos con el fin de hacer más eficiente la movilidad actual que presenta la urbe.

Bibliografía

5Hertz. (18 de Febrero de 2014). *Introducción al GPS*. Obtenido de www.5hertz.com

Abella, I. (10 de Octubre de 2014). *Materiales Digitales, Ciencias Sociales*. Obtenido de www.materialescienciassociales.wordpress.com

Acción, E. e. (2011). *Los medios de transporte en la ciudad.* Obtenido de http://www.ecologistasenaccion.org/IMG/pdf_Cuaderno_2_Comparativa_medios.pdf

Castillo, L. (Marzo de 2012). *La caja común busca mejorar el transporte público*. Obtenido de www.elcomercio.com/actualidad/ecuador/la-caja-común-busca-mejorar-el-transporte-público

Codazzi, I. G. (Marzo de 2007). *Sistema de Posicionamiento Global.* Obtenido de www.gisweb.ciat.cgiar.org

Comercio, E. (25 de 05 de 2015). *Los buses urbanos y taxis con nueva tarifa en Ibarra*. Obtenido de http://www.elcomercio.com/actualidad/buses-taxis-tarifas-ibarra.html

CONADIS. (2013). *Personas con discapacidad registradas*. Obtenido de www.consejodiscapacidades.gob.ec

CONCEPTODEFINICION.DE. (08 de 09 de 2014). *Defínción de Transporte*. Obtenido de http://conceptodefinicion.de/transporte/

Delgado, R. C. (2013). *Plan Nacional del Buen Vivir.* Quito: Senplades.

DIVERTEKA. (19 de 11 de 2012). *Trucos Para Raspberry Pi.* Obtenido de http://www.diverteka.com/?p=171

Duotel, F. (10 de Febrero de 2015). *Probamos la nueva Raspberry Pi 2: A fondo*. Obtenido de www.xatakahome.com

Duque, R. G. (2013). *Python para todos.* España: Creative Commons.

Ecuador, T. (Marzo de 2015). *Smart Cities.* Obtenido de www.telefonica.com.ec

FACUA. (2007). *El transporte público*. Obtenido de http://www.facua.org/pdf/web/viewer.php?rutaPdf=guia77.pdf&donde=guias

FUNDATION, R. P. (2016). *Raspberry Pi - Teach, Learn, and Make with Raspberry Pi*. Obtenido de www.raspberrypi.org

Galnares, J. (2012). *Pidora*. Obtenido de www.prototipando.es

García, K. G. (2013). *Mujeres y hombres del Ecuador en cifras*. Obtenido de www.ecuadorencifras.com.ec

Ibarra, G. (2016). *Visitas Turísticas*. Obtenido de www.touribarra.gob.ec

INEC. (2013). *Fasículo Provincial Imbabura.* Imbabura.

INEC. (2013). *Resultados del Censo 2010*. Obtenido de www.ecuadorencifras.gob.ec

INEN. (Noviembre de 2012). *Norma Técnica Ecuatoriana 2656.* Obtenido de www.normaspdf.inen.gob.ec

Informática, E. I. (2013). *Práctica de Análisis Matemático.* Departamento de Análisis Matemático.

Kalley. (2014). *Televisor 32" LED*. Obtenido de www.kalley.com.co

Kelly. (04 de 04 de 2014). *Computadoras de placa reducida*. Obtenido de https://computadorasdeplacareducida.wordpress.com/category/computadoras-de-placa-reducida/

Kolling, M. (01 de Abril de 2011). *Greenfoot Tutorial.* Obtenido de www.greenfoot.org

Lina Manjarrez, P. R. (2011). *Transporte urbano, movilidad cotidiana y ambiente en el modelo de ciudad sostenible: bases conceptuales.* México, D.F.: Plaza y Valdés, S.A.

López, J. (11 de Julio de 2015). *ArchLinux, una distribución simple y ligera*. Obtenido de www.incubaweb.com

Medina, E. (20 de Mayo de 2012). *Gumstix*. Obtenido de www.helpes.eu

México, U. (2011). *¿Qué es Ubuntu?* Obtenido de www.ubuntumx.org

MOVIDELNOR. (05 de 2006). Diagnóstico de Parada de Buses del Cantón Ibarra. Ibarra, Imbabura, Ecuador.

NEO-6. (2015). *u-blox 6 Data Sheet GPS Modules*. Obtenido de www.u-blox.com

Pastor, J. (12 de Agosto de 2015). *Windows 10 Iot Core*. Obtenido de www.xataca.com

Pinto, G. (2012). *Hacia una movilidad inteligente en la ciudad de Quito.* Obtenido de www.epmmop.gob.ec

Quess.la. (2016). *¿Qué es transporte?* Obtenido de http://quees.la/transporte/

Sastre, T. R. (16 de Agosto de 2014). *Openelec mediacenter en Raspberri Py.* Obtenido de www.electroensaimada.com

Schumacher. (01 de Agosto de 2014). *Manual del Propietario (PI-400).* Obtenido de www.allbatterysalesandservice.com

SENA. (2014). *Modos y Medios de Transporte.* Obtenido de https://senaintro.blackboard.com/bbcswebdav/institution/semillas/822203_1_VIRTUAL/Objetos_de_Aprendizaje/Descargables/ADA%207/ADA_7.2.pdf

Suárez, V. (05 de Febrero de 2016). *Introducción a Raspberry Pi.* Obtenido de www.hacklabalmeria.net

Tránsito, A. N. (31 de Diciembre de 2014). *Ley Orgánica de Transporte Terrestre, Tránsito y Seguridad Vial.* Obtenido de www.ant.gob.ec

Turismo, M. d. (2015). *Principales indicadores de turismo.* Obtenido de www.servicios.turismo.gob.ec

U-blox. (11 de Noviembre de 2014). *NEO 7 Data Sheet* . Obtenido de www.u-blox.com

Velasco, R. (21 de Mayo de 2016). *OSMC, un completo centro multimedia para Raspberry Pi.* Obtenido de www.redeszone.net

Victor. (29 de Abril de 2013). *Probando RISC OS.* Obtenido de www.blogspot.com

WAVESHARE. (12 de 06 de 2015). *5inch HDMI LCD (B).* Obtenido de http://www.waveshare.com/w/upload/2/2f/5inch-HDMI-LCD-B-UserManual.pdf

ANEXOS

ANEXO 1.

CÓDIGO DE PROGRAMACIÓN

```
import MySQLdb
import os
from Tkinter import *
from gps import *
from time import *
import time
import threading
from PIL import Image,ImageTk

os.system("sudo gpsd /dev/ttyACM0 -n -F /var/run/gpsd.sock")
db = MySQLdb.connect("localhost", "root", "12345", "gpsbus")
cursor=db.cursor()
paradaactual_data=["---"]
paradasiguiente_data=["---"]
paradaactual_data2=["---"]
paradasiguiente_data2=["---"]
lista_gif=[]
lista_gif2=[]
gpsd = None

class Gpsinicio1(threading.Thread):
   def __init__(self):
      threading.Thread.__init__(self)
      global gpsd
      gpsd = gps(mode=WATCH_ENABLE)
      self.running = True

   def run(self):
      global gpsd
      gpsp1.running=True
      while gpsp1.running:
         gpsd.next()

class GPS1(threading.Thread):
   def __init__(self):
      threading.Thread.__init__(self)
      self.running = True

   def run(self):
      G1.running=True
      bandera= ''
      while G1.running:
         longitud    =   gpsd.fix.longitude
         latitud     =   gpsd.fix.latitude


         if longitud <= 0.0 and latitud >= 0.0:
            log1    =   longitud+0.00018
            lat1    =   latitud+0.00018
            log2    =   longitud-0.00018
            lat2    =   latitud-0.00018

            cursor.execute ("insert into bus_ruta (ruta, lat, log,hora) values ('ruta1','"+str(latitud) +"','"+str(longitud) +"',now())" )
            db.commit ()

            cursor.execute ("select ruta,parada,sige,ante,leei,leef,obj from bus_catolica where cast(lat as decimal(10,8)) <= "+str(lat1)+" and cast(lat as decimal(10,8))>= "+str(lat2)+" and cast(log as decimal(10,8)) <= "+str(log1)+" and cast(log as decimal(10,8))>= "+str(log2))
            en=""
            rut=""

            for r in cursor.fetchall():
               if str(r[0])!="" :
                  rut=    str(r[0])
                  en =    str(r[1])
                  sig =   str(r[2])
                  ant =   str(r[3])
                  li=     str(r[4])
                  lf=     str(r[5])
                  obj=    str(r[6])
```

```
if rut=='ruta1':
    if en=='Inicio1':
        bandera=1
    if en=='Inicio2':
        bandera=1
    if en=='Inicio3':
        bandera=1
    if en=='Inicio4':
        bandera=1
    if en=='Inicio5':
        bandera=1
    if en=='Fin1' :
        bandera=0
    if en=='Fin2':
        bandera=0
    if en=='Fin3':
        bandera=0
    if en=='Fin4' :
        bandera=0
    if en=='Fin5' :
        bandera=0


if en!="" :
    if bandera == 1 :
        if str(li)== "1" :
            os.system("omxplayer /home/pi/Desktop/videos/cato_alpa/"+str(obj))
            paradaactual_data.append(en)
            paradasiguiente_data.append(sig)
            if en == "Inicio1":
                paradaactual_data.append("Calle Machala")
                paradasiguiente_data.append(sig)
            if en == "Inicio2":
                paradaactual_data.append("Av. Victor Manuel Guzman")
                paradasiguiente_data.append(sig)
            if en == "Inicio3":
                paradaactual_data.append("Av. Perez Guerrero")
                paradasiguiente_data.append(sig)
            if en == "Inicio4":
                paradaactual_data.append("Aproximacion Av. Mariano Acosta")
                paradasiguiente_data.append(sig)
            if en == "Inicio5":
                paradaactual_data.append("Av. Aurelio Espinoza Polit")
                paradasiguiente_data.append(sig)
            if en == "Fin1":
                paradaactual_data.append("Juan Martinez de Orbe")
                paradasiguiente_data.append(sig)
            if en == "Fin2":
                paradaactual_data.append("Av. Jaime Rivadeneira")
                paradasiguiente_data.append(sig)
            if en == "Fin3":
                paradaactual_data.append("Calle Vicente Rocafuerte")
                paradasiguiente_data.append(sig)
            if en == "Fin4":
                paradaactual_data.append("Panamericana Norte")
                paradasiguiente_data.append(sig)
            if en == "Fin5":
                paradaactual_data.append("Calle Antonio Jose de Sucre")
                paradasiguiente_data.append(sig)


    if bandera == 0 :
        if str(lf) =="1" :
            os.system("omxplayer /home/pi/Desktop/videos/cato_alpa/"+str(obj))
            paradaactual_data.append(en)
            paradasiguiente_data.append(ant)
            if en == "Inicio1":
                paradaactual_data.append("Calle Machala")
                paradasiguiente_data.append(ant)
            if en == "Inicio2":
                paradaactual_data.append("Av. Victor Manuel Guzman")
                paradasiguiente_data.append(ant)
            if en == "Inicio3":
                paradaactual_data.append("Av. Perez Guerrero")
                paradasiguiente_data.append(ant)
            if en == "Inicio4":
```

```
                    paradaactual_data.append("Aproximacion Av. Mariano Acosta")
                    paradasiguiente_data.append(ant)
                if en == "Inicio5":
                    paradaactual_data.append("Av. Aurelio Espinoza Polit")
                    paradasiguiente_data.append(ant)
                if en == "Fin1":
                    paradaactual_data.append("Juan Martinez de Orbe")
                    paradasiguiente_data.append(ant)
                if en == "Fin2":
                    paradaactual_data.append("Av. Jaime Rivadeneira")
                    paradasiguiente_data.append(ant)
                if en == "Fin3":
                    paradaactual_data.append("Calle Vicente Rocafuerte")
                    paradasiguiente_data.append(ant)
                if en == "Fin4":
                    paradaactual_data.append("Panamericana Norte")
                    paradasiguiente_data.append(ant)
                if en == "Fin5":
                    paradaactual_data.append("Calle Antonio Jose de Sucre")
                    paradasiguiente_data.append(ant)

        time.sleep(2)
        cursor.execute ("delete from  bus_ruta where log='nan'")
        cursor.execute ("delete from  bus_ruta where log='0.0'")

class GPS2(threading.Thread):
    def __init__(self):
        threading.Thread.__init__(self)
        self.running = True

    def run(self):
        G2.running=True
        bandera= ''
        while G2.running:
            longitud    =   gpsd.fix.longitude
            latitud     =   gpsd.fix.latitude


            if longitud <= 0.0 and latitud >= 0.0:
                log1    =   longitud+0.00018
                lat1    =   latitud+0.00018
                log2    =   longitud-0.00018
                lat2    =   latitud-0.00018

                cursor.execute ("insert into bus_ruta (ruta, lat, log,hora) values ('ruta2','"+str(latitud) +"','"+str(longitud) +"',now())" )
                db.commit ()

                cursor.execute ("select ruta,parada,sige,ante,leei,leef,obj from bus_esperanza where cast(lat as decimal(10,8)) <= "+str(lat1)+" and cast(lat as decimal(10,8))>= "+str(lat2)+" and cast(log as decimal(10,8)) <= "+str(log1)+" and cast(log as decimal(10,8))>= "+str(log2))
                en=""
                rut=""

                for r in cursor.fetchall():
                    if str(r[0])!="" :
                        rut=    str(r[0])
                        en =    str(r[1])
                        sig =   str(r[2])
                        ant =   str(r[3])
                        li=     str(r[4])
                        lf=     str(r[5])
                        obj=    str(r[6])


                if rut=='ruta2':
                    if en=='Inicio1':
                        bandera=1
                    if en=='Inicio2':
                        bandera=1
                    if en=='Inicio3':
                        bandera=1
                    if en=='Inicio4':
                        bandera=1
                    if en=='Inicio5':
                        bandera=1
                    if en=='Fin1' :
```

```
            bandera=0
        if en=='Fin2':
            bandera=0
        if en=='Fin3':
            bandera=0
        if en=='Fin4' :
            bandera=0
        if en=='Fin5' :
            bandera=0
        if en=='Fin6' :
            bandera=0

    if en!="" :
        if bandera == 1 :
            if str(li)== "1" :
                os.system("omxplayer /home/pi/Desktop/videos/esperanza/"+str(obj))
                paradaactual_data2.append(en)
                paradasiguiente_data2.append(sig)
                if en == "Inicio1":
                    paradaactual_data2.append("La Esperanza")
                    paradasiguiente_data2.append(sig)
                if en == "Inicio2":
                    paradaactual_data2.append("Av. Atahualpa")
                    paradasiguiente_data2.append(sig)
                if en == "Inicio3":
                    paradaactual_data2.append("Calle Vicente Rocafuerte")
                    paradasiguiente_data2.append(sig)
                if en == "Inicio4":
                    paradaactual_data2.append("Mercado Amazonas")
                    paradasiguiente_data2.append(sig)
                if en == "Inicio5":
                    paradaactual_data2.append("Calle Brasil")
                    paradasiguiente_data2.append(sig)
                if en == "Fin1":
                    paradaactual_data2.append("Calle General Pintag")
                    paradasiguiente_data2.append(sig)
                if en == "Fin2":
                    paradaactual_data2.append("Av. Atahualpa")
                    paradasiguiente_data2.append(sig)
                if en == "Fin3":
                    paradaactual_data2.append("Av. Jaime Roldos Aguilera")
                    paradasiguiente_data2.append(sig)
                if en == "Fin4":
                    paradaactual_data2.append("Estadio las Palmas")
                    paradasiguiente_data2.append(sig)
                if en == "Fin5":
                    paradaactual_data2.append("Calle Simon Bolivar")
                    paradasiguiente_data2.append(sig)
                if en == "Fin6":
                    paradaactual_data2.append("Calle German Grijalva")
                    paradasiguiente_data2.append(sig)

        if bandera == 0 :
            if str(lf) =="1" :
                os.system("omxplayer /home/pi/Desktop/videos/esperanza/"+str(obj))
                paradaactual_data2.append(en)
                paradasiguiente_data2.append(ant)
                if en == "Inicio1":
                    paradaactual_data2.append("La Esperanza")
                    paradasiguiente_data2.append(ant)
                if en == "Inicio2":
                    paradaactual_data2.append("Av. Atahualpa")
                    paradasiguiente_data2.append(ant)
                if en == "Inicio3":
                    paradaactual_data2.append("Calle Vicente Rocafuerte")
                    paradasiguiente_data2.append(ant)
                if en == "Inicio4":
                    paradaactual_data2.append("Mercado Amazonas")
                    paradasiguiente_data2.append(ant)
                if en == "Inicio5":
                    paradaactual_data2.append("Calle Brasil")
                    paradasiguiente_data2.append(ant)
                if en == "Fin1":
                    paradaactual_data2.append("Calle General Pintag")
                    paradasiguiente_data2.append(ant)
                if en == "Fin2":
```

```
                    paradaactual_data2.append("Av. Atahualpa")
                    paradasiguiente_data2.append(ant)
                if en == "Fin3":
                    paradaactual_data2.append("Av. Jaime Roldos Aguilera")
                    paradasiguiente_data2.append(ant)
                if en == "Fin4":
                    paradaactual_data2.append("Estadio las Palmas")
                    paradasiguiente_data2.append(ant)
                if en == "Fin5":
                    paradaactual_data2.append("Calle Simon Bolivar")
                    paradasiguiente_data2.append(ant)
                if en == "Fin6":
                    paradaactual_data2.append("Calle German Grijalva")
                    paradasiguiente_data2.append(ant)

        time.sleep(2)
        cursor.execute ("delete from  bus_ruta where log='nan'")
        cursor.execute ("delete from  bus_ruta where log='0.0'")


class InterfazInicio(object):
    def __init__(self):
        self.InicioI = Tk ()
        self.UTN1=Image.open("/home/pi/Desktop/prueba7/gifs/utn4.png").convert("RGB")
        self.UTN = ImageTk.PhotoImage(self.UTN1)
        self.Ibarra1=Image.open("/home/pi/Desktop/prueba7/gifs/ibarra4.png").convert("RGB")
        self.Ibarra = ImageTk.PhotoImage(self.Ibarra1)
        self.Mec1=Image.open("/home/pi/Desktop/prueba7/gifs/carrera4.png").convert("RGB")
        self.Mec = ImageTk.PhotoImage(self.Mec1)
        self.SEP1=Image.open("/home/pi/Desktop/prueba7/gifs/inicio28.png").convert("RGB")
        self.SEP = ImageTk.PhotoImage(self.SEP1)
        self.San1=Image.open("/home/pi/Desktop/prueba7/gifs/iniciosan.png").convert("RGB")
        self.San = ImageTk.PhotoImage(self.San1)
        self.exit1=Image.open("/home/pi/Desktop/prueba7/gifs/apagar6.png").convert("RGB")
        self.exit = ImageTk.PhotoImage(self.exit1)
        self.lblE1=Label(self.InicioI, bg="white",width = 2, height = 1)
        self.lblE2=Label(self.InicioI, bg="white",width = 1, height = 1)
        self.lblE3=Label(self.InicioI, bg="white",width = 1, height = 3)
        self.lblE4=Label(self.InicioI, bg="white",width = 1, height = 1)
        self.lblE5=Label(self.InicioI, bg="white",width = 1, height = 1)
        self.lblE6=Label(self.InicioI, bg="white",width = 1, height = 1)
        self.lblE7=Label(self.InicioI, bg="white",width = 1, height = 1)
        self.lblUTN=Label(self.InicioI, bg="white", image = self.UTN, width = 245, height = 165)
        self.lblIbarra=Label(self.InicioI, bg="white", image = self.Ibarra, width = 245, height = 165)
        self.lblMec=Label(self.InicioI, bg="white", image = self.Mec, width = 245, height = 165)
        self.btn28 =Button(self.InicioI, activebackground = 'white',bg = "white",relief = 'flat',image = self.SEP, width = 30, height =
160)
        self.btn28.bind('<ButtonRelease-1>', lambda e: self.func28())
        self.btnSAN=Button(self.InicioI, activebackground = 'white',bg = "white",relief = 'flat',image = self.San,width = 30, height =
160)
        self.btnSAN.bind('<ButtonRelease-1>', lambda e: self.funcSAN())
        self.btnSalir=Button(self.InicioI,activebackground = 'white',bg = "white",relief = 'flat', image = self.exit, width = 90, height =
90)
        self.btnSalir.bind('<ButtonRelease-1>', lambda e: self.funcSalir())


    def run(self):
        self.InicioI.geometry("800x480")
        self.InicioI.config(bg="white")
        self.lblE1.grid(column=0, row=0,rowspan=5,sticky=W+E+N+S)
        self.lblE2.grid(column=9, row=0,rowspan=5,sticky=W+E+N+S)
        self.lblE3.grid(column=0, row=0,columnspan = 9,sticky=EW)
        self.lblE4.grid(column=0, row=2,columnspan = 9,sticky=EW)
        self.lblE5.grid(column=0, row=4,columnspan =9,sticky=EW)
        self.lblE6.grid(column=2, row=0,rowspan=5,sticky=W+E+N+S)
        self.lblE7.grid(column=4, row=0,rowspan=5,sticky=W+E+N+S)
        self.lblUTN.grid(column=3, row=1,columnspan = 1,sticky=EW)
        self.lblIbarra.grid(column=1, row=1,columnspan = 1,sticky=EW)
        self.lblMec.grid(column=5, row=1,columnspan = 1,sticky=EW)
        self.btn28.grid(column=1, row=3, columnspan = 1,sticky=EW)
        self.btnSAN.grid(column=5, row=3, columnspan = 1,sticky=EW)
        self.btnSalir.grid(column=3, row=3)
        self.InicioI.mainloop()

    def func28(self):
        Home28().run()
```

```
    def funcSAN(self):
        HomeSAN().run()

    def funcSalir(self):
        InterfazSalir().run()


class Home28(object):

    def __init__(self):
        self.inicioP = Toplevel()
        self.LetraActual = "Seleccionar Ruta"
        self.I11=Image.open("/home/pi/Desktop/prueba7/gifs/catolica.png").convert("RGB")
        self.I1 = ImageTk.PhotoImage(self.I11)
        self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/pugacho.png").convert("RGB")
        self.I2 = ImageTk.PhotoImage(self.I21)
        self.I31=Image.open("/home/pi/Desktop/prueba7/gifs/palmas.png").convert("RGB")
        self.I3 = ImageTk.PhotoImage(self.I31)
        self.I41=Image.open("/home/pi/Desktop/prueba7/gifs/aloburo.png").convert("RGB")
        self.I4 = ImageTk.PhotoImage(self.I41)
        self.I51=Image.open("/home/pi/Desktop/prueba7/gifs/arcangel.png").convert("RGB")
        self.I5 = ImageTk.PhotoImage(self.I51)
        self.I61=Image.open("/home/pi/Desktop/prueba7/gifs/azaya.png").convert("RGB")
        self.I6 = ImageTk.PhotoImage(self.I61)
        self.IC1=Image.open("/home/pi/Desktop/prueba7/gifs/home28.png").convert("RGB")
        self.IC = ImageTk.PhotoImage(self.IC1)
        self.IS1=Image.open("/home/pi/Desktop/prueba7/gifs/siguien28.png").convert("RGB")
        self.IS = ImageTk.PhotoImage(self.IS1)
        self.lblE1=Label(self.inicioP, bg="white",width = 1, height = 1)
        self.lblE2=Label(self.inicioP, bg="white",width = 1, height = 1)
        self.lblE3=Label(self.inicioP, bg="white",width = 1, height = 1)
        self.lblE4=Label(self.inicioP, bg="white",width = 1, height = 1)
        self.lblE5=Label(self.inicioP, bg="white",width = 1, height = 3)
        self.lblE6=Label(self.inicioP, bg="white",width = 1, height = 2)
        self.lblE7=Label(self.inicioP, bg="white",width = 1, height = 1)
        self.lblE8=Label(self.inicioP, bg="white",width = 1, height = 1)
        self.btnPantalla1 = Button(self.inicioP, activebackground = 'white',bg="white", width = 30, height=2, text="",relief = 'flat',font
= 'Arial 11 bold')
        self.btnPantalla1.bind('<ButtonRelease-1>', lambda e:  self.funcVentanaR1())
        self.btn1 = Button(self.inicioP,  bg="white", activebackground = 'white',relief = 'flat',image = "" , width = 235, height = 100)
        self.btn1.bind('<ButtonRelease-1>', lambda e: self.funcLetras1())
        self.btn2 = Button(self.inicioP,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn2.bind('<ButtonRelease-1>', lambda e: self.funcLetras2())
        self.btn3 = Button(self.inicioP,activebackground = 'white', bg="white", relief = 'flat',image = "",width = 235, height = 100)
        self.btn3.bind('<ButtonRelease-1>', lambda e: self.funcLetras3())
        self.btn4 = Button(self.inicioP,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn4.bind('<ButtonRelease-1>', lambda e: self.funcLetras4())
        self.btn5 = Button(self.inicioP,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn5.bind('<ButtonRelease-1>', lambda e: self.funcLetras5())
        self.btn6 = Button(self.inicioP, activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
        self.btn6.bind('<ButtonRelease-1>', lambda e: self.funcLetras6())
        self.btnCerrar= Button(self.inicioP,activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
        self.btnCerrar.bind('<ButtonRelease-1>', lambda e: self.funcCerrar())
        self.btnSigue = Button(self.inicioP, activebackground = 'white',bg="white", relief = 'flat',image = "",width = 235, height = 100)
        self.btnSigue.bind('<ButtonRelease-1>', lambda e: self.funcSigue())

    def run(self):
        self.inicioP.geometry("800x480")
        self.inicioP.config(bg="white")
        self.lblE1.grid(column=0, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE2.grid(column=2, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE3.grid(column=4, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE4.grid(column=6, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE5.grid(column=0, row=0,columnspan = 8,sticky=EW)
        self.lblE6.grid(column=0, row=2,columnspan = 8,sticky=EW)
        self.lblE7.grid(column=0, row=4,columnspan =8,sticky=EW)
        self.lblE8.grid(column=0, row=7,columnspan =8,sticky=W+E+N+S)
        self.btnPantalla1.grid(column=3, row=1, columnspan = 1,sticky=EW)
        self.btnPantalla1["text"]="Seleccionar Ruta"
        self.btn1["image"] = self.I1
        self.btn2["image"] = self.I2
        self.btn3 ["image"]= self.I3
        self.btn4 ["image"]= self.I4
        self.btn5 ["image"] = self.I5
        self.btn6 ["image"]= self.I6
```

```
        self.btnCerrar ["image"]= self.IC
        self.btnSigue ["image"]= self.IS
        self.btnCerrar.grid(column=1, row=1, columnspan = 1,sticky=EW)
        self.btn1.grid(column = 1, row = 3, columnspan = 1,sticky=EW)
        self.btn2.grid(column = 3, row = 3, columnspan = 1,sticky=EW)
        self.btn3.grid(column = 5, row = 3, columnspan = 1,sticky=EW)
        self.btn4.grid(column = 1, row = 5, columnspan = 1,sticky=EW)
        self.btn5.grid(column = 3, row = 5, columnspan = 1,sticky=EW)
        self.btn6.grid(column = 5, row = 5, columnspan = 1,sticky=EW)
        self.btnSigue.grid(column = 5, row = 1, columnspan = 1,sticky=EW)
        self.inicioP.mainloop()


    def funcLetras1(self):
        if self.btn1:
            self.btnPantalla1["text"]="CATOLICA - ALPACHACA"
            self.LetraActual = "CATOLICA - ALPACHACA"

    def funcLetras2(self):
        if self.btn2:
            self.btnPantalla1["text"]="PUGACHO - STA. TERESITA"
            self.LetraActual = "PUGACHO - STA. TERESITA"

    def funcLetras3(self):
        if self.btn3:
            self.btnPantalla1["text"]="LAS PALMAS - LOS CEIBOS"
            self.LetraActual ="LAS PALMAS - LOS CEIBOS"

    def funcLetras4(self):
        if self.btn4:
            self.btnPantalla1["text"]="ALOBURO"
            self.LetraActual = "ALOBURO"

    def funcLetras5(self):
        if self.btn5:
            self.btnPantalla1["text"]="ARCANGEL - CARANQUI"
            self.LetraActual = "ARCANGEL - CARANQUI"

    def funcLetras6(self):
        if self.btn6:
            self.btnPantalla1["text"]="AZAYA"
            self.LetraActual = "AZAYA"

    def funcVentanaR1(self):
        if self.LetraActual == "CATOLICA - ALPACHACA":
            gpsp1 = Gpsinicio1()
            gpsp1.start()
            G1= GPS1()
            G1.start()
            Interfaz1().run()


    def funcCerrar(self):
        self.inicioP.destroy()

    def funcSigue(self):
        Home282().run()


class Home282(object):

    def __init__(self):
        self.inicioP2 = Toplevel()
        self.LetraActual = "Seleccionar Ruta"
        self.I11=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
        self.I1 = ImageTk.PhotoImage(self.I11)
        self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
        self.I2 = ImageTk.PhotoImage(self.I21)
        self.I31=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
        self.I3 = ImageTk.PhotoImage(self.I31)
        self.I41=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
        self.I4 = ImageTk.PhotoImage(self.I41)
        self.I51=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
        self.I5 = ImageTk.PhotoImage(self.I51)
        self.I61=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
        self.I6 = ImageTk.PhotoImage(self.I61)
```

```
        self.IC1=Image.open("/home/pi/Desktop/prueba7/gifs/ante28.png").convert("RGB")
        self.IC = ImageTk.PhotoImage(self.IC1)
        self.IS1=Image.open("/home/pi/Desktop/prueba7/gifs/siguien28.png").convert("RGB")
        self.IS = ImageTk.PhotoImage(self.IS1)
        self.lblE1=Label(self.inicioP2, bg="white",width = 1, height = 1)
        self.lblE2=Label(self.inicioP2, bg="white",width = 1, height = 1)
        self.lblE3=Label(self.inicioP2, bg="white",width = 1, height = 1)
        self.lblE4=Label(self.inicioP2, bg="white",width = 1, height = 1)
        self.lblE5=Label(self.inicioP2, bg="white",width = 1, height = 3)
        self.lblE6=Label(self.inicioP2, bg="white",width = 1, height = 2)
        self.lblE7=Label(self.inicioP2, bg="white",width = 1, height = 1)
        self.lblE8=Label(self.inicioP2, bg="white",width = 1, height = 1)
        self.btnPantalla1 = Button(self.inicioP2, activebackground = 'white',bg="white", width = 30, height=2, text="",relief =
'flat',font = 'Arial 11 bold')
        self.btn1 = Button(self.inicioP2, activebackground = 'white', bg="white", relief = 'flat',image = "" , width = 235, height = 100)
        self.btn1.bind('<ButtonRelease-1>', lambda e: self.funcLetras1())
        self.btn2 = Button(self.inicioP2,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn2.bind('<ButtonRelease-1>', lambda e: self.funcLetras2())
        self.btn3 = Button(self.inicioP2,activebackground = 'white', bg="white", relief = 'flat',image = "",width = 235, height = 100)
        self.btn3.bind('<ButtonRelease-1>', lambda e: self.funcLetras3())
        self.btn4 = Button(self.inicioP2,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn4.bind('<ButtonRelease-1>', lambda e: self.funcLetras4())
        self.btn5 = Button(self.inicioP2,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn5.bind('<ButtonRelease-1>', lambda e: self.funcLetras5())
        self.btn6 = Button(self.inicioP2, activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
        self.btn6.bind('<ButtonRelease-1>', lambda e: self.funcLetras6())
        self.btnAnte= Button(self.inicioP2,activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
        self.btnAnte.bind('<ButtonRelease-1>', lambda e: self.funcAnte())
        self.btnSigue = Button(self.inicioP2, activebackground = 'white',bg="white", relief = 'flat',image = "",width = 235, height =
100)
        self.btnSigue.bind('<ButtonRelease-1>', lambda e: self.funcSigue())

    def run(self):
        self.inicioP2.geometry("800x480")
        self.inicioP2.config(bg="white")
        self.lblE1.grid(column=0, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE2.grid(column=2, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE3.grid(column=4, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE4.grid(column=6, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE5.grid(column=0, row=0,columnspan = 8,sticky=EW)
        self.lblE6.grid(column=0, row=2,columnspan = 8,sticky=EW)
        self.lblE7.grid(column=0, row=4,columnspan =8,sticky=EW)
        self.lblE8.grid(column=0, row=7,columnspan =8,sticky=W+E+N+S)
        self.btnPantalla1.grid(column=3, row=1, columnspan = 1,sticky=EW)
        self.btnPantalla1["text"]="Seleccionar Ruta"
        self.btn1["image"] = self.I1
        self.btn2["image"] = self.I2
        self.btn3 ["image"]= self.I3
        self.btn4 ["image"]= self.I4
        self.btn5 ["image"] = self.I5
        self.btn6 ["image"]= self.I6
        self.btnAnte ["image"]= self.IC
        self.btnSigue ["image"]= self.IS
        self.btnAnte.grid(column=1, row=1, columnspan = 1,sticky=EW)
        self.btn1.grid(column = 1, row = 3, columnspan = 1,sticky=EW)
        self.btn2.grid(column = 3, row = 3, columnspan = 1,sticky=EW)
        self.btn3.grid(column = 5, row = 3, columnspan = 1,sticky=EW)
        self.btn4.grid(column = 1, row = 5, columnspan = 1,sticky=EW)
        self.btn5.grid(column = 3, row = 5, columnspan = 1,sticky=EW)
        self.btn6.grid(column = 5, row = 5, columnspan = 1,sticky=EW)
        self.btnSigue.grid(column = 5, row = 1, columnspan = 1,sticky=EW)
        self.inicioP2.mainloop()


    def funcLetras1(self):
        if self.btn1:
            self.btnPantalla1["text"]="7. Ruta 7"
            self.LetraActual = "7. Ruta 7"

    def funcLetras2(self):
        if self.btn2:
            self.btnPantalla1["text"]="8. Ruta 8"
            self.LetraActual = "8. Ruta 8"

    def funcLetras3(self):
        if self.btn3:
```

```
            self.btnPantalla1["text"]="9. Ruta 9"
            self.LetraActual = "9. Ruta 9"
        def funcLetras4(self):
            if self.btn4:
                self.btnPantalla1["text"]="10. Ruta 10"
                self.LetraActual = "10. Ruta 10"

        def funcLetras5(self):
            if self.btn5:
                self.btnPantalla1["text"]="11. Ruta 11"
                self.LetraActual = "11. Ruta 11"

        def funcLetras6(self):
            if self.btn6:
                self.btnPantalla1["text"]="12. Ruta 12"
                self.LetraActual = "12. Ruta 12"

        def funcAnte(self):
            self.inicioP2.destroy()

        def funcSigue(self):
            Home283().run()

    class Home283(object):

        def __init__(self):
            self.inicioP3 = Toplevel()
            self.LetraActual = "Seleccionar Ruta"
            self.I11=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
            self.I1 = ImageTk.PhotoImage(self.I11)
            self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
            self.I2 = ImageTk.PhotoImage(self.I21)
            self.I31=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
            self.I3 = ImageTk.PhotoImage(self.I31)
            self.I41=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
            self.I4 = ImageTk.PhotoImage(self.I41)
            self.I51=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
            self.I5 = ImageTk.PhotoImage(self.I51)
            self.I61=Image.open("/home/pi/Desktop/prueba7/gifs/ruta.png").convert("RGB")
            self.I6 = ImageTk.PhotoImage(self.I61)
            self.IC1=Image.open("/home/pi/Desktop/prueba7/gifs/ante28.png").convert("RGB")
            self.IC = ImageTk.PhotoImage(self.IC1)
            self.lblE1=Label(self.inicioP3, bg="white",width = 1, height = 1)
            self.lblE2=Label(self.inicioP3, bg="white",width = 1, height = 1)
            self.lblE3=Label(self.inicioP3, bg="white",width = 1, height = 1)
            self.lblE4=Label(self.inicioP3, bg="white",width = 1, height = 1)
            self.lblE5=Label(self.inicioP3, bg="white",width = 1, height = 3)
            self.lblE6=Label(self.inicioP3, bg="white",width = 1, height = 2)
            self.lblE7=Label(self.inicioP3, bg="white",width = 1, height = 1)
            self.lblE8=Label(self.inicioP3, bg="white",width = 1, height = 1)
            self.btnPantalla1 = Button(self.inicioP3, activebackground = 'white',bg="white", width = 30, height=2, text="",relief =
    'flat',font = 'Arial 11 bold')
            self.btn1 = Button(self.inicioP3,  activebackground = 'white',bg="white", relief = 'flat',image = "" , width = 235, height = 100)
            self.btn1.bind('<ButtonRelease-1>', lambda e: self.funcLetras1())
            self.btn2 = Button(self.inicioP3,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
            self.btn2.bind('<ButtonRelease-1>', lambda e: self.funcLetras2())
            self.btn3 = Button(self.inicioP3,activebackground = 'white', bg="white", relief = 'flat',image = "",width = 235, height = 100)
            self.btn3.bind('<ButtonRelease-1>', lambda e: self.funcLetras3())
            self.btn4 = Button(self.inicioP3,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
            self.btn4.bind('<ButtonRelease-1>', lambda e: self.funcLetras4())
            self.btn5 = Button(self.inicioP3,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
            self.btn5.bind('<ButtonRelease-1>', lambda e: self.funcLetras5())
            self.btn6 = Button(self.inicioP3, activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
            self.btn6.bind('<ButtonRelease-1>', lambda e: self.funcLetras6())
            self.btnAnte= Button(self.inicioP3,activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
            self.btnAnte.bind('<ButtonRelease-1>', lambda e: self.funcAnte())

        def run(self):
            self.inicioP3.geometry("800x480")
            self.inicioP3.config(bg="white")
            self.lblE1.grid(column=0, row=0,rowspan=6,sticky=W+E+N+S)
            self.lblE2.grid(column=2, row=0,rowspan=6,sticky=W+E+N+S)
            self.lblE3.grid(column=4, row=0,rowspan=6,sticky=W+E+N+S)
            self.lblE4.grid(column=6, row=0,rowspan=6,sticky=W+E+N+S)
            self.lblE5.grid(column=0, row=0,columnspan = 8,sticky=EW)
            self.lblE6.grid(column=0, row=2,columnspan = 8,sticky=EW)
```

```
        self.lblE7.grid(column=0, row=4,columnspan =8,sticky=EW)
        self.lblE8.grid(column=0, row=7,columnspan =8,sticky=W+E+N+S)
        self.btnPantalla1.grid(column=3, row=1, columnspan = 1,sticky=EW)
        self.btnPantalla1["text"]="Seleccionar Ruta"
        self.btn1["image"] = self.I1
        self.btn2["image"] = self.I2
        self.btn3 ["image"]= self.I3
        self.btn4 ["image"]= self.I4
        self.btn5 ["image"] = self.I5
        self.btn6 ["image"]= self.I6
        self.btnAnte ["image"]= self.IC
        self.btnAnte.grid(column=1, row=1, columnspan = 1,sticky=EW)
        self.btn1.grid(column = 1, row = 3, columnspan = 1,sticky=EW)
        self.btn2.grid(column = 3, row = 3, columnspan = 1,sticky=EW)
        self.btn3.grid(column = 5, row = 3, columnspan = 1,sticky=EW)
        self.btn4.grid(column = 1, row = 5, columnspan = 1,sticky=EW)
        self.btn5.grid(column = 3, row = 5, columnspan = 1,sticky=EW)
        self.btn6.grid(column = 5, row = 5, columnspan = 1,sticky=EW)
        self.inicioP3.mainloop()


    def funcLetras1(self):
        if self.btn1:
            self.btnPantalla1["text"]="13. Ruta 13"
            self.LetraActual = "13. Ruta 13"

    def funcLetras2(self):
        if self.btn2:
            self.btnPantalla1["text"]="14. Ruta 14"
            self.LetraActual = "14. Ruta 14"

    def funcLetras3(self):
        if self.btn3:
            self.btnPantalla1["text"]="15. Ruta 15"
            self.LetraActual = "15. Ruta 15"

    def funcLetras4(self):
        if self.btn4:
            self.btnPantalla1["text"]="16. Ruta 16"
            self.LetraActual = "16. Ruta 16"

    def funcLetras5(self):
        if self.btn5:
            self.btnPantalla1["text"]="17. Ruta 17"
            self.LetraActual = "17. Ruta 17"

    def funcLetras6(self):
        if self.btn6:
            self.btnPantalla1["text"]="18. Ruta 18"
            self.LetraActual = "18. Ruta 18"

    def funcAnte(self):
        self.inicioP3.destroy()

class HomeSAN(object):

    def __init__(self):
        self.inicioP4 = Toplevel()
        self.LetraActual = "Seleccionar Ruta"
        self.I11=Image.open("/home/pi/Desktop/prueba7/gifs/esperanza.png").convert("RGB")
        self.I1 = ImageTk.PhotoImage(self.I11)
        self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/colinas.png").convert("RGB")
        self.I2 = ImageTk.PhotoImage(self.I21)
        self.I31=Image.open("/home/pi/Desktop/prueba7/gifs/santodomingo.png").convert("RGB")
        self.I3 = ImageTk.PhotoImage(self.I31)
        self.I41=Image.open("/home/pi/Desktop/prueba7/gifs/miravalle.png").convert("RGB")
        self.I4 = ImageTk.PhotoImage(self.I41)
        self.I51=Image.open("/home/pi/Desktop/prueba7/gifs/santaisabel2.png").convert("RGB")
        self.I5 = ImageTk.PhotoImage(self.I51)
        self.I61=Image.open("/home/pi/Desktop/prueba7/gifs/victoria.png").convert("RGB")
        self.I6 = ImageTk.PhotoImage(self.I61)
        self.IC1=Image.open("/home/pi/Desktop/prueba7/gifs/homesan.png").convert("RGB")
        self.IC = ImageTk.PhotoImage(self.IC1)
        self.IS1=Image.open("/home/pi/Desktop/prueba7/gifs/siguiensan.png").convert("RGB")
        self.IS = ImageTk.PhotoImage(self.IS1)
        self.lblE1=Label(self.inicioP4, bg="white",width = 1, height = 1)
```

```
        self.lblE2=Label(self.inicioP4, bg="white",width = 1, height = 1)
        self.lblE3=Label(self.inicioP4, bg="white",width = 1, height = 1)
        self.lblE4=Label(self.inicioP4, bg="white",width = 1, height = 1)
        self.lblE5=Label(self.inicioP4, bg="white",width = 1, height = 3)
        self.lblE6=Label(self.inicioP4, bg="white",width = 1, height = 2)
        self.lblE7=Label(self.inicioP4, bg="white",width = 1, height = 1)
        self.lblE8=Label(self.inicioP4, bg="white",width = 1, height = 1)
        self.btnPantalla1 = Button(self.inicioP4, activebackground = 'white',bg="white", width = 30, height=2, text="",relief =
'flat',font = 'Arial 11 bold')
        self.btnPantalla1.bind('<ButtonRelease-1>', lambda e: self.funcVentanaR2() )
        self.btn1 = Button(self.inicioP4,  activebackground = 'white',bg="white", relief = 'flat',image = "" , width = 235, height = 100)
        self.btn1.bind('<ButtonRelease-1>', lambda e: self.funcLetras1())
        self.btn2 = Button(self.inicioP4,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn2.bind('<ButtonRelease-1>', lambda e: self.funcLetras2())
        self.btn3 = Button(self.inicioP4,activebackground = 'white', bg="white", relief = 'flat',image = "",width = 235, height = 100)
        self.btn3.bind('<ButtonRelease-1>', lambda e: self.funcLetras3())
        self.btn4 = Button(self.inicioP4,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn4.bind('<ButtonRelease-1>', lambda e: self.funcLetras4())
        self.btn5 = Button(self.inicioP4,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn5.bind('<ButtonRelease-1>', lambda e: self.funcLetras5())
        self.btn6 = Button(self.inicioP4, activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
        self.btn6.bind('<ButtonRelease-1>', lambda e: self.funcLetras6())
        self.btnCerrar= Button(self.inicioP4,activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height =
100)
        self.btnCerrar.bind('<ButtonRelease-1>', lambda e: self.funcCerrar())
        self.btnSigue = Button(self.inicioP4, activebackground = 'white',bg="white", relief = 'flat',image = "",width = 235, height =
100)
        self.btnSigue.bind('<ButtonRelease-1>', lambda e: self.funcSigue())

    def run(self):
        self.inicioP4.geometry("800x480")
        self.inicioP4.config(bg="white")
        self.lblE1.grid(column=0, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE2.grid(column=2, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE3.grid(column=4, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE4.grid(column=6, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE5.grid(column=0, row=0,columnspan = 8,sticky=EW)
        self.lblE6.grid(column=0, row=2,columnspan = 8,sticky=EW)
        self.lblE7.grid(column=0, row=4,columnspan =8,sticky=EW)
        self.lblE8.grid(column=0, row=7,columnspan =8,sticky=W+E+N+S)
        self.btnPantalla1.grid(column=3, row=1, columnspan = 1,sticky=EW)
        self.btnPantalla1["text"]="Seleccionar Ruta"
        self.btn1["image"] = self.I1
        self.btn2["image"] = self.I2
        self.btn3 ["image"]= self.I3
        self.btn4 ["image"]= self.I4
        self.btn5 ["image"] = self.I5
        self.btn6 ["image"]= self.I6
        self.btnCerrar ["image"]= self.IC
        self.btnSigue ["image"]= self.IS
        self.btnCerrar.grid(column=1, row=1, columnspan = 1,sticky=EW)
        self.btn1.grid(column = 1, row = 3, columnspan = 1,sticky=EW)
        self.btn2.grid(column = 3, row = 3, columnspan = 1,sticky=EW)
        self.btn3.grid(column = 5, row = 3, columnspan = 1,sticky=EW)
        self.btn4.grid(column = 1, row = 5, columnspan = 1,sticky=EW)
        self.btn5.grid(column = 3, row = 5, columnspan = 1,sticky=EW)
        self.btn6.grid(column = 5, row = 5, columnspan = 1,sticky=EW)
        self.btnSigue.grid(column = 5, row = 1, columnspan = 1,sticky=EW)
        self.inicioP4.mainloop()


    def funcLetras1(self):
        if self.btn1:
            self.btnPantalla1["text"]="ESPERANZA - HOSPITAL DEL IESS"
            self.LetraActual ="ESPERANZA - HOSPITAL DEL IESS"

    def funcLetras2(self):
        if self.btn2:
            self.btnPantalla1["text"]="COLINAS DEL SUR - ADUANA"
            self.LetraActual = "COLINAS DEL SUR - ADUANA"

    def funcLetras3(self):
        if self.btn3:
            self.btnPantalla1["text"]="SANTO DOMINGO - UNIVERSIDADES"
            self.LetraActual = "SANTO DOMINGO - UNIVERSIDADES"
```

```
    def funcLetras4(self):
        if self.btn4:
            self.btnPantalla1["text"]="EJIDO DE CARANQUI - MIRAVALLE"
            self.LetraActual = "EJIDO DE CARANQUI - MIRAVALLE"
    def funcLetras5(self):
        if self.btn5:
            self.btnPantalla1["text"]="STA. ISABEL"
            self.LetraActual = "SANTA ISABEL"

    def funcLetras6(self):
        if self.btn6:
            self.btnPantalla1["text"]="SANTA LUCIA - LA VICTORIA"
            self.LetraActual = "SANTA LUCIA - LA VICTORIA"

    def funcVentanaR2(self):
        if self.LetraActual == "ESPERANZA - HOSPITAL DEL IESS":
            gpsp1 = Gpsinicio1()
            gpsp1.start()
            G2= GPS2()
            G2.start()
            Interfaz2().run()

    def funcCerrar(self):
        self.inicioP4.destroy()

    def funcSigue(self):
        HomeSAN2().run()

class HomeSAN2(object):

    def __init__(self):
        self.inicioP5 = Toplevel()
        self.LetraActual = "Seleccionar Ruta"
        self.I11=Image.open("/home/pi/Desktop/prueba7/gifs/ruta2.png").convert("RGB")
        self.I1 = ImageTk.PhotoImage(self.I11)
        self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/ruta2.png").convert("RGB")
        self.I2 = ImageTk.PhotoImage(self.I21)
        self.I31=Image.open("/home/pi/Desktop/prueba7/gifs/ruta2.png").convert("RGB")
        self.I3 = ImageTk.PhotoImage(self.I31)
        self.I41=Image.open("/home/pi/Desktop/prueba7/gifs/ruta2.png").convert("RGB")
        self.I4 = ImageTk.PhotoImage(self.I41)
        self.I51=Image.open("/home/pi/Desktop/prueba7/gifs/ruta2.png").convert("RGB")
        self.I5 = ImageTk.PhotoImage(self.I51)
        self.I61=Image.open("/home/pi/Desktop/prueba7/gifs/ruta2.png").convert("RGB")
        self.I6 = ImageTk.PhotoImage(self.I61)
        self.IC1=Image.open("/home/pi/Desktop/prueba7/gifs/antesan.png").convert("RGB")
        self.IC = ImageTk.PhotoImage(self.IC1)
        self.lblE1=Label(self.inicioP5, bg="white",width = 1, height = 1)
        self.lblE2=Label(self.inicioP5, bg="white",width = 1, height = 1)
        self.lblE3=Label(self.inicioP5, bg="white",width = 1, height = 1)
        self.lblE4=Label(self.inicioP5, bg="white",width = 1, height = 1)
        self.lblE5=Label(self.inicioP5, bg="white",width = 1, height = 3)
        self.lblE6=Label(self.inicioP5, bg="white",width = 1, height = 2)
        self.lblE7=Label(self.inicioP5, bg="white",width = 1, height = 1)
        self.lblE8=Label(self.inicioP5, bg="white",width = 1, height = 1)
        self.btnPantalla1 = Button(self.inicioP5, activebackground = 'white',bg="white", width = 30, height=2, text="",relief =
'flat',font = 'Arial 11 bold')
        self.btn1 = Button(self.inicioP5, activebackground = 'white', bg="white", relief = 'flat',image = "" , width = 235, height = 100)
        self.btn1.bind('<ButtonRelease-1>', lambda e: self.funcLetras1())
        self.btn2 = Button(self.inicioP5,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn2.bind('<ButtonRelease-1>', lambda e: self.funcLetras2())
        self.btn3 = Button(self.inicioP5,activebackground = 'white', bg="white", relief = 'flat',image = "",width = 235, height = 100)
        self.btn3.bind('<ButtonRelease-1>', lambda e: self.funcLetras3())
        self.btn4 = Button(self.inicioP5,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn4.bind('<ButtonRelease-1>', lambda e: self.funcLetras4())
        self.btn5 = Button(self.inicioP5,activebackground = 'white',bg="white", relief = 'flat', image = "",width = 235, height = 100)
        self.btn5.bind('<ButtonRelease-1>', lambda e: self.funcLetras5())
        self.btn6 = Button(self.inicioP5, activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
        self.btn6.bind('<ButtonRelease-1>', lambda e: self.funcLetras6())
        self.btnAnte= Button(self.inicioP5,activebackground = 'white',bg="white", relief = 'flat',image = "", width = 235, height = 100)
        self.btnAnte.bind('<ButtonRelease-1>', lambda e: self.funcAnte())

    def run(self):
        self.inicioP5.geometry("800x480")
        self.inicioP5.config(bg="white")
        self.lblE1.grid(column=0, row=0,rowspan=6,sticky=W+E+N+S)
```

```
        self.lblE2.grid(column=2, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE3.grid(column=4, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE4.grid(column=6, row=0,rowspan=6,sticky=W+E+N+S)
        self.lblE5.grid(column=0, row=0,columnspan = 8,sticky=EW)
        self.lblE6.grid(column=0, row=2,columnspan = 8,sticky=EW)
        self.lblE7.grid(column=0, row=4,columnspan =8,sticky=EW)
        self.lblE8.grid(column=0, row=7,columnspan =8,sticky=W+E+N+S)
        self.btnPantalla1.grid(column=3, row=1, columnspan = 1,sticky=EW)
        self.btnPantalla1["text"]="Seleccionar Ruta"
        self.btn1["image"] = self.I1
        self.btn2["image"] = self.I2
        self.btn3 ["image"]= self.I3
        self.btn4 ["image"]= self.I4
        self.btn5 ["image"] = self.I5
        self.btn6 ["image"]= self.I6
        self.btnAnte ["image"]= self.IC
        self.btnAnte.grid(column=1, row=1, columnspan = 1,sticky=EW)
        self.btn1.grid(column = 1, row = 3, columnspan = 1,sticky=EW)
        self.btn2.grid(column = 3, row = 3, columnspan = 1,sticky=EW)
        self.btn3.grid(column = 5, row = 3, columnspan = 1,sticky=EW)
        self.btn4.grid(column = 1, row = 5, columnspan = 1,sticky=EW)
        self.btn5.grid(column = 3, row = 5, columnspan = 1,sticky=EW)
        self.btn6.grid(column = 5, row = 5, columnspan = 1,sticky=EW)
        self.inicioP5.mainloop()


    def funcLetras1(self):
        if self.btn1:
            self.btnPantalla1["text"]="7. Ruta 7"
            self.LetraActual = "7. Ruta 7"

    def funcLetras2(self):
        if self.btn2:
            self.btnPantalla1["text"]="8. Ruta 8"
            self.LetraActual = "8. Ruta 8"

    def funcLetras3(self):
        if self.btn3:
            self.btnPantalla1["text"]="9. Ruta 9"
            self.LetraActual = "9. Ruta 9"

    def funcLetras4(self):
        if self.btn4:
            self.btnPantalla1["text"]="10. Ruta 10"
            self.LetraActual = "10. Ruta 10"

    def funcLetras5(self):
        if self.btn5:
            self.btnPantalla1["text"]="11. Ruta 11"
            self.LetraActual = "11. Ruta 11"

    def funcLetras6(self):
        if self.btn6:
            self.btnPantalla1["text"]="12. Ruta 12"
            self.LetraActual = "12. Ruta 12"

    def funcAnte(self):
        self.inicioP5.destroy()

class Interfaz1(object):

    def __init__(self):
        self.inicio1 = Toplevel()
        self.lblE1=Label(self.inicio1, bg="white",width = 1, height = 1)
        self.lblE2=Label(self.inicio1, bg="white",width = 1, height = 1)
        self.lblE3=Label(self.inicio1, bg="white",width = 1, height = 1)
        self.lblE4=Label(self.inicio1, bg="white")
        self.lblE5=Label(self.inicio1, bg="white")
        self.lblE7=Label(self.inicio1, bg="white")
        self.I11=Image.open("/home/pi/Desktop/prueba7/gifs/1 (1).png").convert("RGB")
        self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/1 (3).png").convert("RGB")
        self.I31=Image.open("/home/pi/Desktop/prueba7/gifs/1 (4).png").convert("RGB")
        self.I41=Image.open("/home/pi/Desktop/prueba7/gifs/1 (5).png").convert("RGB")
        self.I51=Image.open("/home/pi/Desktop/prueba7/gifs/1 (6).png").convert("RGB")
        self.I61=Image.open("/home/pi/Desktop/prueba7/gifs/1 (7).png").convert("RGB")
        self.imagelist = [self.I11,self.I11,self.I21,self.I21,self.I31,self.I31,self.I41,self.I41,self.I51,self.I51,self.I61,self.I61]
```

```
        self.fotos=ImageTk.PhotoImage(self.imagelist[0])
        self.width=775
        self.height=325
        self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/home28.png").convert("RGB")
        self.I2 = ImageTk.PhotoImage(self.I21)
        self.I31=Image.open("/home/pi/Desktop/prueba7/gifs/play28.png").convert("RGB")
        self.I3 = ImageTk.PhotoImage(self.I31)
        self.lblPantallaR = Label(self.inicio1, bg="white", height=1, text="CATOLICA - ALPACHACA",font = 'bold 13 bold')
        self.lblPantalla1 = Canvas(self.inicio1, bg="white",width = self.width, height = self.height)
        self.lblPantallaPA = Label(self.inicio1, bg="white", height=1, text="Parada Actual")
        self.lblPantallaH1 = Label(self.inicio1, bg="white", height=1, text="---",font = 'bold 13 bold')
        self.lblPantallaPS = Label(self.inicio1, bg="white", height=1, text="Parada Siguiente")
        self.lblPantallaF1 = Label(self.inicio1,bg="white", height=1, text="---",font = 'bold 13 bold')
        self.lblPantallaPA1 = Label(self.inicio1, bg="white", height=1, text="---",font = 'bold 11 bold')
        self.lblPantallaPS1 = Label(self.inicio1,bg="white", height=1, text="---",font = 'bold 11 bold')
        self.btnSalir= Button(self.inicio1, image = "",activebackground = 'white',bg="white",width = 50, height = 72)
        self.btnSalir.bind('<ButtonRelease-1>', lambda e: self.funcVentana1())
        self.btnIngresar= Button(self.inicio1, image = "",activebackground = 'white',bg="white",width = 50, height = 72)
        self.btnIngresar.bind('<ButtonRelease-1>', lambda e: self.funcLetras2() or self.funcLetras() or self.funcImagen())

    def run(self):
        self.inicio1.geometry("800x480")
        self.inicio1.config(bg="white")
        self.lblE1.grid(column=0, row=0,rowspan=7,sticky=W+E+N+S)
        self.lblE2.grid(column=3, row=0,rowspan=7,sticky=W+E+N+S)
        self.lblE3.grid(column=6, row=0,rowspan=7,sticky=W+E+N+S)
        self.lblE4.grid(column=0, row=0,columnspan = 8,sticky=EW)
        self.lblE5.grid(column=0, row=2,columnspan = 8,sticky=EW)
        self.lblE7.grid(column=0, row=6,columnspan = 8,sticky=EW)
        self.btnSalir["image"] = self.I2
        self.btnIngresar["image"] = self.I3
        self.lblPantallaR.grid(column=1, row=1, columnspan = 1)
        self.lblPantalla1.grid(column=1, row=2, columnspan = 5,rowspan=1,sticky=W+E+N+S)
        self.lblPantallaPA.grid(column=1, row=4, columnspan = 1)
        self.lblPantallaH1.grid(column=3, row=1,columnspan = 2)
        self.lblPantallaPS.grid(column=1, row=5, columnspan = 1)
        self.lblPantallaF1.grid(column=5, row=1, columnspan = 2)
        self.lblPantallaPA1.grid(column=2, row=4, columnspan = 1)
        self.lblPantallaPS1.grid(column=2, row=5, columnspan = 1)
        self.btnIngresar.grid(column=4, row=4, columnspan = 1,rowspan=2,sticky=EW)
        self.btnSalir.grid(column=5, row=4, columnspan = 1,rowspan=2,sticky=EW)
        self.inicio1.mainloop()


    def funcImagen(self):
        for imagefile in self.imagelist:
            self.fotos=ImageTk.PhotoImage(imagefile)
            lista_gif.append(self.fotos)
        for k in range (0,1000):
            for gif in lista_gif:
                self.lblPantalla1.delete(ALL)
                self.lblPantalla1.create_image(self.width/2.0,self.height/2.0,image=gif)
                self.lblPantalla1.update()
                time.sleep(2)

    def funcLetras(self):
        self.lblPantallaF1["text"]=time.strftime("%H:%M:%S")
        self.lblPantallaH1["text"]=time.strftime("%D")
        self.inicio1.update()
        self.inicio1.after(1000,self.funcLetras)


    def funcLetras2(self):
        self.lblPantallaPA1["text"]=paradaactual_data[-1]
        self.lblPantallaPS1["text"]=paradasiguiente_data[-1]
        self.inicio1.update()
        self.inicio1.after(1000,self.funcLetras2)

    def funcVentana1(self):
        G1.running = False
        gpsp1.running=False
        self.inicio1.withdraw()

class Interfaz2(object):

    def __init__(self):
```

```
    self.inicio2 = Toplevel()
    self.lblE1=Label(self.inicio2, bg="white",width = 1, height = 1)
    self.lblE2=Label(self.inicio2, bg="white",width = 1, height = 1)
    self.lblE3=Label(self.inicio2, bg="white",width = 1, height = 1)
    self.lblE4=Label(self.inicio2, bg="white")
    self.lblE5=Label(self.inicio2, bg="white")
    self.lblE7=Label(self.inicio2, bg="white")
    self.I12=Image.open("/home/pi/Desktop/prueba7/gifs/2 (1).png").convert("RGB")
    self.I22=Image.open("/home/pi/Desktop/prueba7/gifs/2 (2).png").convert("RGB")
    self.I32=Image.open("/home/pi/Desktop/prueba7/gifs/2 (3).png").convert("RGB")
    self.I42=Image.open("/home/pi/Desktop/prueba7/gifs/2 (4).png").convert("RGB")
    self.I52=Image.open("/home/pi/Desktop/prueba7/gifs/2 (5).png").convert("RGB")
    self.I62=Image.open("/home/pi/Desktop/prueba7/gifs/2 (6).png").convert("RGB")
    self.imagelist = [self.I12,self.I12,self.I22,self.I22,self.I32,self.I32,self.I42,self.I42,self.I52,self.I52,self.I62,self.I62]
    self.fotos=ImageTk.PhotoImage(self.imagelist[0])
    self.width=775
    self.height=325
    self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/homesan.png").convert("RGB")
    self.I2 = ImageTk.PhotoImage(self.I21)
    self.I31=Image.open("/home/pi/Desktop/prueba7/gifs/playsan.png").convert("RGB")
    self.I3 = ImageTk.PhotoImage(self.I31)
    self.lblPantallaR2 = Label(self.inicio2, bg="white", height=1, text="ESPERANZA - IESS",font = 'bold 13 bold')
    self.lblPantalla2 = Canvas(self.inicio2, bg="white",width = self.width, height = self.height)
    self.lblPantallaPA2 = Label(self.inicio2, bg="white", height=1, text="Parada Actual")
    self.lblPantallaH2 = Label(self.inicio2, bg="white", height=1, text="---",font = 'bold 13 bold')
    self.lblPantallaPS2 = Label(self.inicio2, bg="white", height=1, text="Parada Siguiente")
    self.lblPantallaF2 = Label(self.inicio2,bg="white", height=1, text="---",font = 'bold 13 bold')
    self.lblPantallaPA22 = Label(self.inicio2, bg="white", height=1, text="---",font = 'bold 11 bold')
    self.lblPantallaPS22 = Label(self.inicio2,bg="white", height=1, text="---",font = 'bold 11 bold')
    self.btnSalir= Button(self.inicio2, image = "",activebackground = 'white',bg="white",width = 50, height = 72)
    self.btnSalir.bind('<ButtonRelease-1>', lambda e: self.funcVentana2())
    self.btnIngresar= Button(self.inicio2, image = "",activebackground = 'white',bg="white",width = 50, height = 72)
    self.btnIngresar.bind('<ButtonRelease-1>', lambda e: self.funcLetras22() or self.funcLetras2() or self.funcImagen2())

def run(self):
    self.inicio2.geometry("800x480")
    self.inicio2.config(bg="white")
    self.lblE1.grid(column=0, row=0,rowspan=7,sticky=W+E+N+S)
    self.lblE2.grid(column=3, row=0,rowspan=7,sticky=W+E+N+S)
    self.lblE3.grid(column=6, row=0,rowspan=7,sticky=W+E+N+S)
    self.lblE4.grid(column=0, row=0,columnspan = 8,sticky=EW)
    self.lblE5.grid(column=0, row=2,columnspan = 8,sticky=EW)
    self.lblE7.grid(column=0, row=6,columnspan = 8,sticky=EW)
    self.btnSalir["image"] = self.I2
    self.btnIngresar["image"] = self.I3
    self.lblPantallaR2.grid(column=1, row=1, columnspan = 1)
    self.lblPantalla2.grid(column=1, row=2, columnspan = 5,rowspan=1,sticky=W+E+N+S)
    self.lblPantallaPA2.grid(column=1, row=4, columnspan = 1)
    self.lblPantallaH2.grid(column=3, row=1,columnspan = 2)
    self.lblPantallaPS2.grid(column=1, row=5, columnspan = 1)
    self.lblPantallaF2.grid(column=5, row=1, columnspan = 2)
    self.lblPantallaPA22.grid(column=2, row=4, columnspan = 1)
    self.lblPantallaPS22.grid(column=2, row=5, columnspan = 1)
    self.btnIngresar.grid(column=4, row=4, columnspan = 1,rowspan=2,sticky=EW)
    self.btnSalir.grid(column=5, row=4, columnspan = 1,rowspan=2,sticky=EW)
    self.inicio2.mainloop()


def funcImagen2(self):
    for imagefile in self.imagelist:
        self.fotos=ImageTk.PhotoImage(imagefile)
        lista_gif2.append(self.fotos)
    for k in range (0,1000):
        for gif in lista_gif2:
            self.lblPantalla2.delete(ALL)
            self.lblPantalla2.create_image(self.width/2.0,self.height/2.0,image=gif)
            self.lblPantalla2.update()
            time.sleep(2)

def funcLetras2(self):
    self.lblPantallaF2["text"]=time.strftime("%H:%M:%S")
    self.lblPantallaH2["text"]=time.strftime("%D")
    self.inicio2.update()
    self.inicio2.after(1000,self.funcLetras2)

def funcLetras22(self):
```

```
        self.lblPantallaPA22["text"]=paradaactual_data2[-1]
        self.lblPantallaPS22["text"]=paradasiguiente_data2[-1]
        self.inicio2.update()
        self.inicio2.after(1000,self.funcLetras22)

    def funcVentana2(self):
        G2.running = False
        gpsp1.running=False
        self.inicio2.withdraw()


class InterfazSalir(object):

    def __init__(self):
        self.inicio0=Toplevel()
        self.I11=Image.open("/home/pi/Desktop/prueba7/gifs/ok.png").convert("RGB")
        self.I1 = ImageTk.PhotoImage(self.I11)
        self.I21=Image.open("/home/pi/Desktop/prueba7/gifs/cancel.png").convert("RGB")
        self.I2 = ImageTk.PhotoImage(self.I21)
        self.lblE1=Label(self.inicio0, bg="white",width = 1, height = 9)
        self.lblE2=Label(self.inicio0, bg="white",width = 1, height = 6)
        self.lblE3=Label(self.inicio0, bg="white",width = 9, height = 1)
        self.lblE4=Label(self.inicio0, bg="white",width = 16, height = 1)
        self.lblE5=Label(self.inicio0, bg="white",width = 9, height = 1)
        self.btnSi = Button(self.inicio0,bg="white", activebackground = 'white',relief = 'flat',image = "" , width = 250, height = 250)
        self.btnSi.bind('<ButtonRelease-1>', lambda e: self.funcCerrar2())
        self.btnNo = Button(self.inicio0, bg="white", activebackground = 'white',relief = 'flat',image = "" , width = 250, height = 250)
        self.btnNo.bind('<ButtonRelease-1>', lambda e: self.funcCerrar3())

    def run(self):
        self.inicio0.geometry("800x480")
        self.inicio0.config(bg="white")
        self.btnSi["image"] = self.I1
        self.btnNo["image"] = self.I2
        self.lblE1.grid(column=0, row=0,columnspan = 4, sticky = NSEW)
        self.lblE2.grid(column=0, row=2,columnspan = 4, sticky = NSEW)
        self.lblE3.grid(column=0, row=0,rowspan=4,sticky=W+E+N+S)
        self.lblE4.grid(column=2, row=0,rowspan=4,sticky=W+E+N+S)
        self.lblE5.grid(column=4, row=0,rowspan=4,sticky=W+E+N+S)
        self.btnSi.grid(column=1, row=1, columnspan = 1, sticky = NSEW)
        self.btnNo.grid(column=3, row=1, columnspan = 1, sticky = NSEW)
        self.inicio0.mainloop()

    def funcCerrar2(self):
        os.system("sudo shutdown -h now")

    def funcCerrar3(self):
        self.inicio0.destroy()


if __name__=="__main__":
    G1= GPS1()
    G2= GPS2()
    gpsp1 = Gpsinicio1()
    InterfazInicio().run()
```