UNIVERSIDAD NACIONAL DE INGENIERÍA

Facultad de Electrotecnia y Computación

Monografía para optar al título de

Ingeniero en

Telecomunicaciones

Sistema Inteligente de Monitoreo en Tiempo Real, de Arribo por Parada y por Unidad del Transporte Urbano Colectivo (TUC) de Managua, Nicaragua.

Autores:

- David Alonso Cruz Tenorio 2013-44084
- César Augusto Muñoz Horney 2013-44139

Tutor:

Ing. Marco Antonio Munguía Mena.

Managua, marzo 2022

DEDICATORIA

Dedicamos la presente monografía a nuestros padres, hermanas, hermanos y abuelas por su apoyo y consejo en los momentos más difíciles. Por habernos otorgado todos los recursos necesarios para poder estudiar, formando en nosotros valores, principios y carácter para poder alcanzar cualquier objetivo.

AGRADECIMIENTO

Queremos agradecer primeramente a Dios por habernos permitido alcanzar este momento de nuestras vidas.

A nuestros padres por su ayuda incondicional de toda una vida de esfuerzo y sacrificio.

A la Universidad Nacional de Ingeniería (UNI) por permitirnos ser parte de ella.

Agradecemos también a nuestro tutor, Ingeniero Carlos Ortega (q.e.p.d.), por brindarnos el apoyo científico-técnico para poder realizar este proyecto y por su eterna paciencia durante todo el desarrollo de nuestro proyecto monográfico.

De igual manera, agradecemos al Ing. Marcos Munguía por su ayuda oportuna que nos permitió concluir este proyecto.

RESUMEN

El servicio de transporte público urbano de Managua, a lo largo del tiempo, ha tenido muchas deficiencias, diferentes propuestas se han venido implementando con el objetivo de modernizarlo. Pese a los mejores esfuerzos de funcionarios públicos y cooperativas de buses, el servicio del TUC en Managua no ha alcanzado un desarrollo aceptable en todo este tiempo.

¿Cómo podemos mejorar el servicio del TUC? Este trabajo monográfico propone una solución diferente, creando una herramienta que mejore el servicio que se le brinda al usuario sin tener que cambiar el funcionamiento actual. Este proyecto es un primer paso para el desarrollo de un sistema de transporte urbano colectivo moderno.

Con el uso de tecnologías GPS, WifFi, protocolo de enrutamiento, herramienta de prototipado (Rapsberry Pi) entre otros, se diseñó e implementó a escala un sistema de Monitoreo en Tiempo Real (apodado “Término”), de arribo por parada y por unidad del transporte urbano colectivo de Managua, que permitirá al usuario reducir la inquietud con respecto a la llegada de distintas rutas.

ÍNDICE

ÍNDICE DE ILUSTRACIONES

I. Introducción

El transporte urbano colectivo (TUC) es el término aplicado al transporte colectivo de pasajeros. A diferencia del transporte privado, los viajeros de transporte público tienen que adaptarse a los horarios y a las rutas que ofrezca el operador. (HarperCollins Publishers Limited, 2018).

El transporte urbano colectivo es uno de los pilares en los que se sostiene nuestra sociedad. Cuando la nación prospera, las ciudades crecen y con ello las distancias se alargan. Los avances tecnológicos siempre han buscado una forma de reducir el tiempo requerido para cubrir estas distancias; la búsqueda de un sistema eficiente capaz de transportar un sin número de pasajeros en una fracción de tiempo es una de las fantasías que inspiran el desarrollo de nuevos sistemas inteligentes de transporte.

Uno de los grandes problemas que afronta el servicio de transporte urbano en Managua es la falta de un itinerario de llegadas a las paradas de manera tal que los usuarios puedan administrar su tiempo para la utilización del transporte público. Además, de reducir su tiempo de espera en las paradas y por ende su exposición a la delincuencia común.

Para solucionar este problema, se propone desarrollar un sistema que, mediante el uso de tecnologías de telecomunicación inalámbrica, permita a los usuarios del TUC conocer con anticipación el tiempo de llegada de la ruta, mientras esperan en su respectiva parada. Esto contribuirá a mejorar la calidad del servicio del TUC, con lo cual la población podrá administrar mejor su tiempo, movilizándose de forma más ágil y eficiente.

II. Objetivos

Objetivo General

Realizar un prototipo para un sistema de monitoreo que, mediante la observación, análisis e interpretación de variables predeterminadas, permita estimar en tiempo real, el momento exacto de la llegada de las unidades de Transporte Urbano Colectivo de Managua, a cada parada.

Objetivos Específicos

1. Identificar la existencia de los horarios establecidos por las Cooperativas de TUC de Managua, mediante la verificación del itinerario de una ruta representativa de esta ciudad, a través de la observación de campo.

2. Determinar la forma de aplicación de la tecnología GPS y WiFi a un sistema de monitoreo en tiempo real de las unidades del TUC de Managua.

3. Diseñar una red inalámbrica con topología de malla (Mesh) y el protocolo de enrutamiento B.A.T.M.A.M adv que permita la comunicación continua entre paradas.

4. Diseñar un prototipo funcional del sistema de monitoreo del TUC en Managua, mediante el uso de un dispositivo GPS y una microcomputadora Raspberry Pi 3.

5. Diseñar un prototipo funcional del sistema de visualización que muestre los tiempos de arribo a los usuarios del TUC en las paradas de buses.

III. Justificación

En la actualidad, es imposible para un usuario del transporte colectivo de Managua, conocer con exactitud el momento en que una unidad de transporte público arribará a la parada en la que espera.

Para poder mantener control sobre sus unidades de transporte, las cooperativas del TUC hacen uso del sistema GPS integrado en cada unidad. El monitoreo de tiempo se realiza cada vez que la unidad llega a una zona específica de Managua, el monitoreo no se realiza en cada una de las paradas.

Su sistema consiste en hacer uso de balizas (objeto señalizador, utilizado para indicar un lugar geográfico) virtuales, colocadas estratégicamente en zonas específicas de Managua. Estas balizas funcionan como punto de control; al pasar la unidad de transporte a través de ellas, el módulo GPS envía el tiempo de arribo a la cooperativa, lugar en donde se encuentra el equipo que controla y monitorea el sistema.

Hasta este punto el sistema es automático, sin embargo, la persona encargada del sistema debe procesar la información enviada por el GPS de la unidad de transporte y realizar las correcciones pertinentes. En caso de que éste incumpla alguno o todos los tiempos de arribo establecidos, el administrador del sistema deberá evaluar su caso tomando en consideración todos los sucesos ocurridos en ese día que podrían haber contribuido al atraso de la unidad. Existen directrices respecto al tiempo de llegada a cada zona marcada por una baliza que deben ser cumplidas por el conductor de la unidad de transporte.

Este proceso se hace de forma manual y consiste en realizar una simple operación aritmética usando los datos de tiempo para calcular el tiempo real total de arribo al punto de control; este cálculo se obtiene, sumando al tiempo establecido para llegar a cada punto de control, el tiempo de espera provocado por todos los

posibles atrasos ocurridos durante el recorrido. Según estos resultados, se procederá a amonestar al conductor de la unidad, si ha incurrido en el incumplimiento del horario establecido. Las amonestaciones o castigos pueden variar desde un simple llamado de atención, hasta una suspensión laboral de 3, 5 o 7 días sin goce de salario.

Sin embargo, la finalidad de este sistema consiste en mantener un estricto control sobre el cumplimiento de los horarios por parte del conductor de la unidad de transporte, y no toma en consideración la posibilidad de integrar al usuario para eliminar la inquietud causada por desconocer los tiempos de arribo a cada parada.

El sistema que se propone en este trabajo, permite a los usuarios eliminar la inquietud referida anteriormente. Para este efecto, se diseñó una red WiFi descentralizada para la comunicación entre paradas, que consiste en una serie de nodos interconectados mediante una configuración de malla. Esto no solo permite que el intercambio de datos pueda realizarse sin la ayuda de una conexión a Internet, sino que además la comunicación entre paradas se mantiene constante.

Así mismo, el sistema utiliza un algoritmo capaz de predecir la hora de llegada de las distintas unidades de transporte a cada una de sus respectivas paradas. Para lograrlo, el algoritmo realiza diversos cálculos que involucran múltiples variables, como la velocidad y posición de cada unidad (GPS), aplicando la tecnología inalámbrica, cuya efectividad está demostrada con el hecho de que fue utilizado exitosamente en el Sistema de Recaudo Automatizado (SRA) aplicado en Managua hasta el mes de agosto de 2018.

También se diseñó y se ensambló un prototipo físico, que incluye componentes electrónicos como el Raspberry Pi 3. La interfaz gráfica exhibida en un monitor (pantalla led) y la programación del software correspondiente, para comprobar la funcionalidad del sistema a través de una demostración en tiempo real.

En síntesis, el sistema que propone este documento, pretende solucionar la incertidumbre sobre los tiempos de llegada de los buses a las paradas. Los usuarios podrán conocer en tiempo real la hora en que estas unidades de transporte arriben, simplemente observando un tablero digital (display) convenientemente ubicado en cada una de esas paradas.

Finalmente, además de las ventajas que este sistema ofrecería a los usuarios, las autoridades locales o nacionales del transporte podrían, si así lo decidiesen, hacer uso del mismo para mantener un mejor control sobre la flota de unidades, que serían sometidas a continuos procesos de inspección y vigilancia en el cumplimiento de los itinerarios establecidos.

Capítulo 1: Marco Teórico

En este capítulo se describen los conceptos generales necesarios para entender el diseño del sistema “Término”.

1. Conceptos generales:

1.1. Sistema Inteligente de Transporte (ITS)

Se conoce como Sistemas Inteligentes de Transporte, o ITS (Intelligent Transportation Systems), al conjunto de aplicaciones informáticas y sistemas tecnológicos creados con el objetivo de mejorar la seguridad y eficiencia en el transporte terrestre (carreteras y ferrocarriles), facilitando la labor de control, gestión y seguimiento por parte de los responsables.

Estos sistemas obtienen la información de los diferentes elementos de interés de las carreteras, que una vez procesada y analizada, se utiliza para mejorar la seguridad de los conductores, mejorando el tráfico y la comodidad en los desplazamientos. (ITERNOVA, 2011)

Algunos beneficios de sistemas ITS son:

- Facilita una mayor coordinación entre las cooperativas de transporte y la flota de buses, aumentando la eficiencia del sistema al mismo tiempo que reducen los costos económicos y sociales del desplazamiento.

- Reduce considerablemente el tiempo de traslado y combustible utilizado en el transporte de mercadería, reduciendo las emisiones y la congestión vehicular.

- Permite mejorar y transparentar las horas de conducción del chofer, brindar información en tiempo real y asistencia en casos de accidentes o averías

en la ruta, lo cual sin duda tiene beneficios sociales y para la seguridad vial a nivel nacional. (CEPAL, 2012)

1.2. Sistemas de monitoreo de transporte:

Un sistema de monitoreo de transporte es un software online que permite conocer, en todo momento, el estatus de las unidades de transporte a las que se les haga seguimiento. Mediante este sistema, se puede monitorear la fecha y hora de salida, de llegada, la ubicación en tiempo real durante el trayecto, las dificultades que pueda estar teniendo el transportista, entre otras variables asociadas a los distintos tipos de trazabilidad.

Por lo general, esta plataforma digital de rastreo inteligente está vinculada a un dispositivo móvil que sirve de apoyo para los transportistas, supervisores y administradores.

En algunos de estos sistemas, el cliente también es partícipe del proceso al poder supervisarlo de manera online a través de la plataforma del sistema de rastreo satelital. (BEETRACK, 2021)

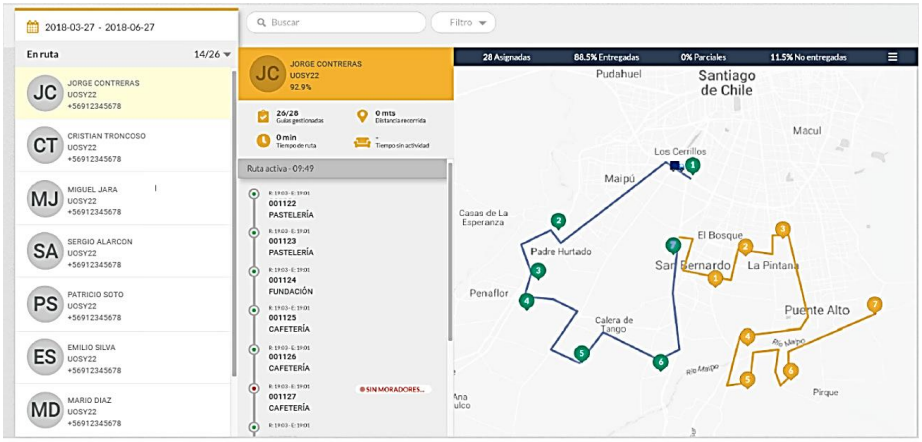

Ilustración 1: Interfaz de BEETRACK, servicio de monitoreo de vehículos

1.3. Sistema “Término”

El prototipo de sistema que se propone en este trabajo se denomina “Término”. Cuyo nombre proviene de la palabra “Terminal” y significa: *un lugar donde la gente viene y va*.

“Término” es un sistema de monitoreo de unidades del TUC offline, descentralizado y semiautomático. Su función principal consiste en calcular los tiempos de arribo por unidad y por parada, para luego proyectar los resultados en una pantalla LED ubicada en cada parada. Esto se consigue mediante el uso de tecnología de redes inalámbricas para la recolección y distribución de datos de posicionamiento de cada unidad del TUC a todas las paradas.

Un sistema de monitoreo de transporte generalmente ofrece a las cooperativas del TUC, herramientas de seguimiento, vigilancia y control de flotas. “Término” por otro lado, está enfocado, únicamente, en mejorar la calidad de servicio para el usuario del TUC. Aunque no descartamos la posibilidad de que “Término” tenga el potencial para beneficiar a las cooperativas del TUC, sentimos la necesidad de aclarar que estas aspiraciones se encuentran fuera del alcance de nuestro proyecto.

El sistema **“**Término” utiliza un dispositivo GPS conectado a una microcomputadora Raspberry Pi 3 colocado en cada bus, que se comunicará, de forma inalámbrica, con todos los nodos de la red formada por cada parada que forme parte del sistema. Esto permitirá observar en una pantalla o “display” led instalado en cada parada, el tiempo exacto faltante para el arribo de la unidad del TUC a la parada.

Capítulo 2: Requerimientos del sistema “Término”.

En este capítulo se describe el estado actual del sistema usado por las Cooperativas del TUC en Managua, posteriormente se definen los requisitos que debe cumplir el sistema “Término”.

2.1. Descripción del estado actual del sistema de Itinerario del TUC en Managua

Para analizar del sistema de control de flotas implementado actualmente en servicio de TUC de Managua se realizó una visita guiada a la Cooperativa de la ruta 165 a cargo del Señor Leslie Nicolas, presidente de la Cooperativa.

El sistema de itinerario del TUC en Managua funciona de la siguiente manera:

- Un módulo GPS se instala en una parte no visible del automotor, esta es la base en la que se sostiene el sistema.
- El dispositivo se encuentra conectado a través de internet, a un centro de control satelital que funciona las 24 horas.
- Marcadores de posición son colocados en una ruta preestablecida que recorrerá la unidad de bus.
- Estos marcadores sirven como punto de control; cada vez que la unidad de transporte transita sobre uno de ellos, el GPS enviara información a la base de la Cooperativa.
- El administrador del sistema usará la información recibida para elaborar un informe de actividad y determinar si el conductor de la unidad ha cumplido con su itinerario.
- En caso de incumplimiento del itinerario por parte del conductor de la unidad, el director aplicará medidas correctivas que pueden involucrar suspensiones de hasta 7 días sin goce de salario.
- El itinerario no se encuentra disponible al público.

2.2. Requerimientos funcionales del sistema ‘’Término’’

Para la correcta distribución de los datos de posicionamiento de cada unidad del TUC, la topología de red del sistema “Término” deberá satisfacer los siguientes requisitos:

- La red debe ser descentralizada
- La red debe ser inalámbrica
- La red debe funcionar sin acceso a internet (offline)
- La red debe ser flexible
- La red debe ser modular

Se ha optado por utilizar el modelo de red malla; este diseño se destaca por su flexibilidad lo que facilita su implementación. Como complemento, se hará uso del protocolo de enrutamiento B.A.T.M.A.N. advanced.

Capítulo 3: Elementos que componen el sistema “Término”

En este capítulo se enumeran y describen los componentes de software y hardware que conforman el sistema “Término”

3.1. Selección de topología de red y protocolo de enrutamiento

En un modelo de red tradicional, cada área de servicio inalámbrico consiste en un grupo cerrado de dispositivos terminales que se forma alrededor de un dispositivo fijo; antes de que un dispositivo pueda participar, este debe anunciar sus capacidades y luego obtener permiso para unirse.

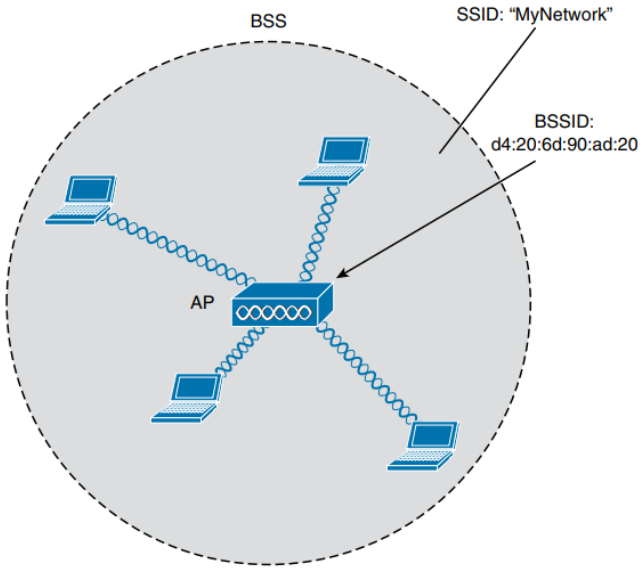

Ilustración 2: Conjunto de servicios básicos (BSS)

El estándar 802.11 llama a esto un conjunto de servicios básicos o BSS (Basic Service Set). En el núcleo de cada BSS hay un punto de acceso inalámbrico o AP (Access Point). El AP opera en modo de infraestructura, lo que significa que ofrece los servicios que son necesarios para formar la infraestructura de una red inalámbrica.

El AP también establece su BSS a través de un solo canal inalámbrico. El AP y los miembros del BSS deben usar el mismo canal para comunicarse correctamente. (Wendell Odom, 2020)

Mientras un cliente inalámbrico permanezca asociado con un BSS, la mayoría de las comunicaciones hacia y desde el cliente deben pasar a través del AP, pero “Término” es capaz de transmitir tramas de datos directamente a cada nodo de la red, que además están formados por terminales fijas (paradas) y móviles (unidades del TUC). Debido a esto es preciso optar por un modelo de red inalámbrico improvisado, en el que puedan formar parte varios dispositivos sin la necesidad de un AP o un BSS.

3.2. Topología de red Mesh

La arquitectura del modelo de red del sistema “Término" se construye sobre la base de la topología de red ad hoc; es un modo de operación de las redes inalámbricas IEEE 802.11, permite que dos o más clientes se comuniquen directamente entre sí de forma inalámbrica, ignorando la necesidad de otros medios de conectividad de red (routers, AP, etc..).

Al configurar una red inalámbrica ad-hoc, todos los adaptadores inalámbricos de las terminales deben usar el mismo Service Set Identifier (SSID) y número de canal. En lugar de usar, por ejemplo, un router donde los datos de la red entran y salen para ser reenviados a los dispositivos destino, cada terminal se convierte en un nodo que forma la estructura de la red ad hoc.

Ilustración 3: Estructura de la red ad hoc

Pero had hoc es solo la base y para solventar la necesidad de hacer de cada parada y cada unidad del TUC integrantes de una misma red, optamos por la topología Mesh, una derivación de ad hoc.

Una red Mesh es un grupo de dispositivos que actúan como una sola red WiFi; por lo que hay múltiples fuentes de WiFi, en lugar de un solo AP. Estas fuentes WiFi adicionales se llaman puntos o nodos.

Todos los nodos están conectados entre sí de forma inalámbrica, siempre que estén dentro del alcance, pueden comunicarse sin la necesidad de un router o AP. Esto permite una mayor cobertura y un enrutamiento de datos rápido y eficiente.

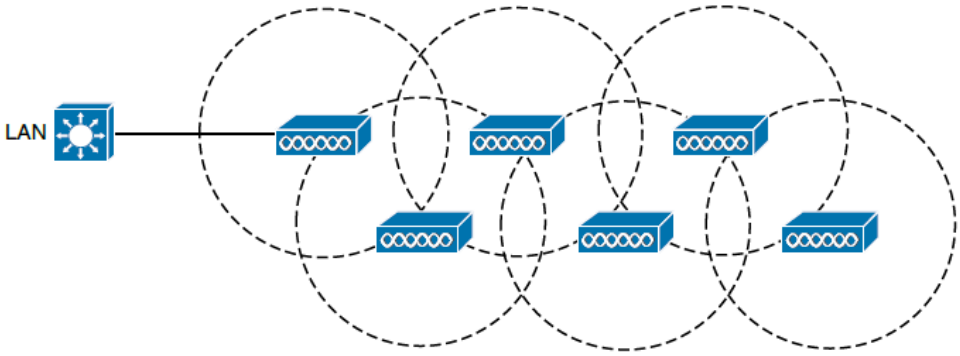

Ilustración 4: Modelo de red Mesh

Una red Mesh tiene las siguientes ventajas:

- Cobertura flexible: se pueden agregar puntos adicionales para obtener una mejor cobertura en áreas difíciles de cubrir como pasillos y cerca de paredes para cobertura al aire libre.
- Regeneración: En una red de malla, si un nodo falla, la comunicación simplemente se desvía a través de otro nodo.
- Ruta directa: Dado que todos los puntos están conectados entre sí, los datos pueden tomar varias rutas hacia su destino y siempre elegirán la mejor ruta desde el punto A hasta el punto B.

3.3. Protocolo de enrutamiento B.A.T.M.A.N adv

B.A.T.M.A.N. adv (Better Approach to Mobile Ad-hoc Networking – advanced por sus siglas en ingles) es una implementación del protocolo de enrutamiento B.A.T.M.A.N. en forma de un módulo del kernel de Linux que opera en la capa 2.

B.A.T.M.A.N. se especializa en la búsqueda de la mejor ruta dentro de un entorno de red descentralizado, no permitiendo que un solo nodo contenga todos los datos. El uso de esta técnica elimina la necesidad de difundir información pertinente a los cambios o alteraciones en la red a todos los nodos de la misma.

Un nodo individual solo guarda información sobre la "dirección" cuando se haya recibido datos de ésta, y envía sus datos en consecuencia. Por este medio los datos se transmiten de un nodo a otro y obtiene paquetes de rutas individuales, creados de forma dinámica. Así se crea una red de inteligencia colectiva. (Freifunk, 2021)

Por lo tanto, al aplicar B.A.T.M.A.N. adv en una topología de red Mesh, podemos aprovechar al máximo sus capacidades y cumplir los requerimientos de la red del sistema ´´Término´´.

Ilustración 5: Logo oficial de B.A.T.M.A.N adv

3.4. Selección de tecnología y dispositivos electrónicos

Una de las mayores problemáticas en el sistema implementado actualmente es que está exclusivamente dirigido hacia las Cooperativas, su enfoque es meramente administrativo y su finalidad es alcanzar un alto grado de coordinación con alto número de unidades de bus que transitan las calles de Managua. Por si fuera poco, el itinerario no existe.

El sistema “Término” tiene como finalidad mejorar la experiencia del usuario sin necesidad de alterar el funcionamiento del sistema actual.

3.5. Herramientas de prototipo: Hardware

A continuación, se describen las tecnologías y dispositivos que conforman la base del sistema y su funcionamiento.

3.5.1. Antena GPS

La antena GPS se encuentra conectada a la tarjeta SIM808, es la encargada de proporcionar servicios fiables de posicionamiento, navegación, y cronometría gratuita e ininterrumpidamente a usuarios civiles en todo el mundo.

Ilustración 6: Antena GPS

3.5.2. Modulo GSM/GPRS/GPS (SIM808)

Este módulo está basado en el chip SIMCOM SIM808 y nos ofrece, las funcionalidades de envío y recepción de datos GSM/GPRS/GPS.

Ilustración 7: Modulo SIM808

El uso de este dispositivo está regido por la norma NMEA 0183, utiliza un protocolo de Comunicación serie simple ASCII que se define como: "Los datos son transmitidos en serie desde un emisor simultáneamente a varios receptores. (National Marine Electronics Association, 2021).

Tanto el módulo SIM808 como la antena GPS están conectados al Raspberry pi 3 ubicado instalado en el vehículo de pruebas por medio del conversor USB a TTL CP2102.

La comunicación con este módulo se realiza a través del programa minicom y vía comandos AT; un lenguaje desarrollado por la compañía Hayes Communications que prácticamente se convirtió en estándar abierto de comandos para configurar y parametrizar módems. Los caracteres «AT», que preceden a todos los comandos, significan «Atención». (Nova, 2013).

3.5.3. USB TTL UART

El conversor CP2102 facilita la comunicación entre una PC y un microcontrolador utilizando el protocolo USB. Es compatible con cualquier microcontrolador como Arduino, PIC, Atmel AVR, ESP8266, ESP32 y más.

Al utilizar el conversor USB se facilita la integración del módulo SIM808 con los programas minicom y Thonny Python, dos pilares importantes para la recolección y procesamiento de los datos de posicionamiento.

Para utilizarlo se debe conectar al módulo SIM808 utilizando los pines: +5V, GND, TXD y RXI.

Ilustración 8: CP2102 Adaptador USB a TTL

3.5.4. Pantalla Led

Cada parada de bus, que pertenezca a la red del sistema “Término”, tiene instalada una pantalla en la que se exhiben los tiempos de arribo de la unidad del TUC. Para la elaboración del prototipo de nuestro sistema, se hace uso de una pantalla led para proyectar la interfaz gráfica del módulo de visualización.

Ilustración 9: Pantalla LED, parte del módulo de visualización.

3.5.5. Vehículo de pruebas

El módulo de posicionamiento se encuentra ubicado en cada unidad del TUC que forme parte de la red del sistema “Término”. Para el montaje de este prototipo, las pruebas de campo se llevan a cabo en un vehículo particular que tomará el papel de unidad del TUC.

Ilustración 10: Vehículo particular en donde se instalará el dispositivo GPS

3.5.6. Fuentes de poder externas (power bank)

Tanto el módulo SIM808 como las micro computadoras Raspberry pi 3, requieren de una fuente de alimentación externa para funcionar. Para el montaje del prototipo se hace uso de baterías portátiles para llevar a cabo las pruebas de campo.

Ilustración 11: Baterías portables

3.5.7. Raspberry PI:

Ilustración 12: Raspberry 3 b+

La Raspberry Pi es una computadora del tamaño de una tarjeta de crédito que se conecta a un televisor y a un teclado. Se puede usar en proyectos de electrónica y para muchas de las cosas que hace una PC de escritorio, como hojas de cálculo, procesamiento de textos, navegación por Internet y juegos. También reproduce video de alta definición (Fundación Raspberry PI, 2018).

Estas microcomputadoras son la base del sistema “Término”. La Raspberry pi 3 instalada en el vehículo de pruebas realizara la recolección y procesamiento y distribución de los datos de posicionamiento mientras que el equipo ubicado en la parada se encargara del cálculo de los tiempos de arribo y su proyección en la pantalla.

3.6. Herramientas de prototipo: Software

A continuación, se listan y describen los programas, bibliotecas y sistema operativo que componen el sistema “Término”.

3.6.1. Raspberry Pi OS

Raspberry Pi OS, previamente conocido como Raspbian, es un sistema operativo gratuito basado en Debian optimizado para el hardware Raspberry Pi. Contiene el conjunto de programas y utilidades básicas que hacen que la Raspberry Pi 3 funcione. Proporciona más de 35,000 paquetes, software precompilado incluido en un formato agradable para una fácil instalación. (Thompson & Green, 2020)

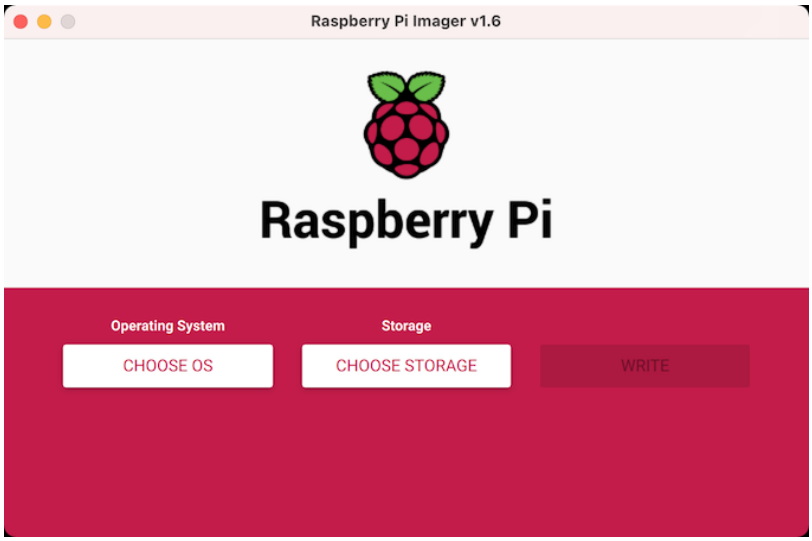

Ilustración 13: Interfaz de instalación de Raspbian

3.6.2. Minicom

Minicom es un programa de comunicaciones de puerto serie basado en texto. Se utiliza para hablar con dispositivos RS-232 externos, como teléfonos móviles, routers y puertos de consola serie. (Penalvch, 2015).

Minicom fue indispensable para establecer una primera comunicación con el módulo SIM808, permitió comunicación directa con el módulo GPS mediante comandos AT y la obtención de datos crudos de geolocalización. Sin embargo, durante el desarrollo del sistema “Término” se diseñó un programa para automatizar este proceso y así obtener los datos de posicionamiento de forma más eficiente (ver sección 6.1 para más detalles).

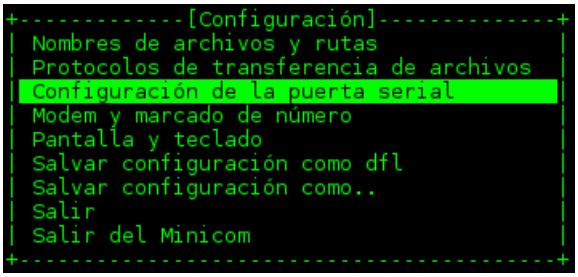

Ilustración 14: Interfaz de configuración de Minicom

3.6.3. Python

Python es un lenguaje de programación de tipo scripting. Un lenguaje de scripting difiere de un verdadero lenguaje de programación en algunos aspectos: (Donat, 2014)

Los lenguajes de programación son compilados, a diferencia de los lenguajes de scripting. Los lenguajes comunes como C, C ++ y Java deben ser compilados por un compilador. El proceso de compilación da como resultado un archivo de código máquina, ilegible para los humanos, que la computadora puede leer y seguir.

Cuando escribe un programa en C y lo compila, el archivo resultante es lo que lee el equipo. Uno de los efectos secundarios / resultados de esto es que los lenguajes de programación pueden producir programas más rápidos, tanto porque la compilación solo ocurre una vez como porque el compilador a menudo optimiza el código durante el proceso de compilación, haciéndolo más rápido de lo que sería como se escribió originalmente.

Los lenguajes de scripting, por otro lado, se leen, interpretan y actúan cada vez que los ejecuta. No producen un archivo compilado, y las instrucciones se siguen exactamente como están escritas. Por esta razón, los lenguajes de scripting pueden resultar en programas más lentos.

Los lenguajes de programación tienden a ser más complejos y difíciles de aprender. Los lenguajes de scripting pueden ser más legibles, son menos estrictos en sintaxis y son menos intimidantes para los no programadores. Solo por esta razón, se escogió este lenguaje de programación para el desarrollo de este proyecto.

3.6.4. Thonny Python

Thonny es un IDE de Python para aprender y enseñar programación a principiantes. Su facilidad de eso fue la razón principal por la que se eligió para este proyecto.

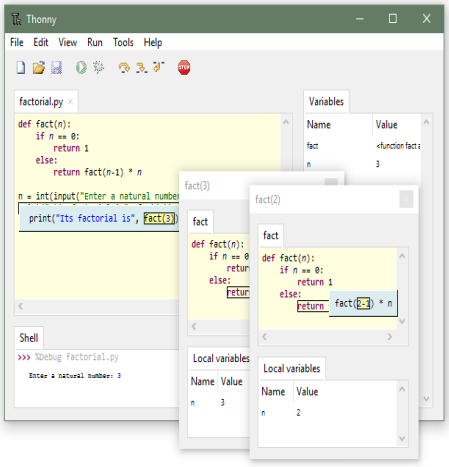

Ilustración 15: Interfaz de Thonny Python

Entre sus características destacadas se encuentran diferentes formas de recorrer el código, la evaluación paso a paso de cada expresión, la visualización intuitiva de la pila de llamadas y un modo para explicar los conceptos de las funciones. Es de uso gratuito. (Aivar Annamaa, 2020).

3.6.5. PyQT

PyQt es un complemento de la biblioteca gráfica QT para el lenguaje de programación de Python. Fue desarrollada por la firma británica Riverbank Computing y está disponible para Windows, GNU/Linux y Mac OS X bajo diferentes licencias.

Incluye abstracciones de sockets de red, hilos, Unicode, expresiones regulares, bases de datos SQL, SVG, OpenGL, XML, un navegador web completamente funcional, un sistema de ayuda, un marco multimedia, así como una rica colección de widgets GUI.

En este proyecto se utilizó para definir las funciones de la interfaz gráfica.

Ilustración 16: Logotipo de Python QT

3.6.6. Qt Designer

Qt Designer es un programa que permite desarrollar interfaces gráficas (GUI) de usuario (multilenguages debido a que genera un archivo XML cuyo contenido es el formato de dicho GUI, pudiéndolo convertir con los programas pertinentes a cada lenguaje).

Qt es utilizada principalmente en Autodesk , Google Earth, KDE, Adobe Photoshop Album, la Agencia Espacial Europea, Opie, Siemens, Volvo, Walt Disney Animation Studios, Skype, Qt Extended, VLC media player, Samsung,Philips, Panasonic, VirtualBox y Mathematica.

Es producido por la división de software Qt de Nokia, que entró en vigor después de la adquisición por parte de Nokia de la empresa noruega Trolltech, el productor original de Qt, el 17 de junio de 2008. (The QT Company, 2021)

Se utilizó para diseñar la apariencia de la interfaz gráfica que muestra los tiempos de arribo en cada parada.

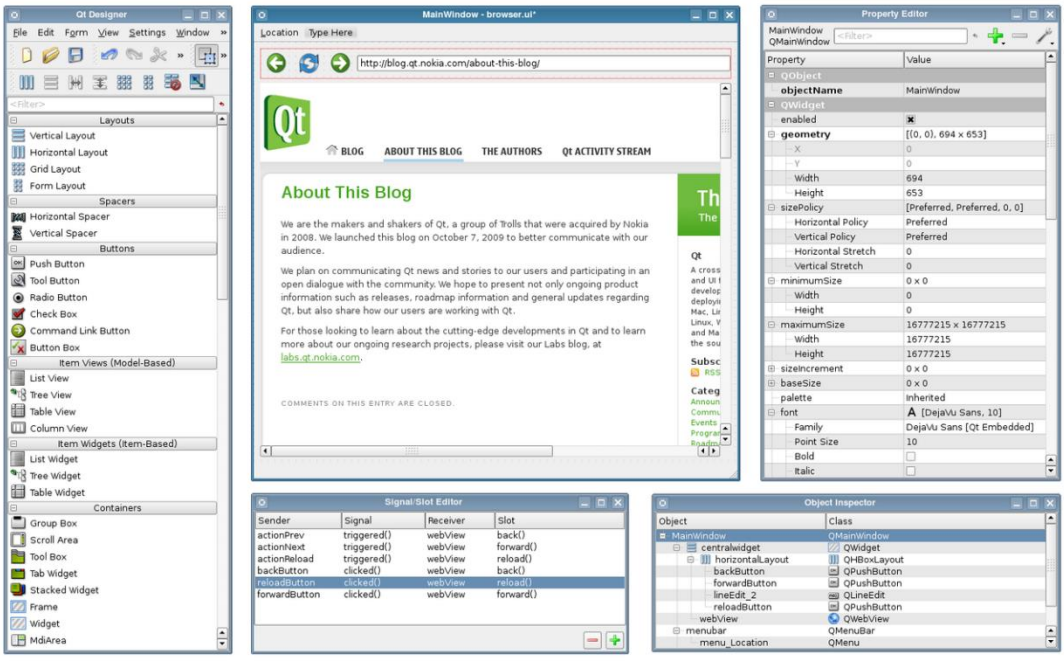

Ilustración 17: Interfaz de QTDesigner

3.6.7. Módulos de Python

A continuación de listan y describen las bibliotecas usadas para el desarrollo del sistema “Término”.

3.6.7.1. Pynmea2

Pynmea2 es una biblioteca compatible con Python 2.7 y 3.0, que permite analizar frases/oraciones del protocolo NMEA 0183 individuales. Recuerda que cada frase contiene diferente información y permite realizar sumas de comprobación para conocer si la información adquirida no contiene errores de estructura. (Pittman, 2021).

Fue utilizada en el proceso de obtención de datos de posicionamiento para filtrar la información recolectada por el módulo GPS y así poder obtener solo los datos requeridos para el cálculo de los tiempos de arribo a cada parada.

3.6.7.2. Datos de Posicionamiento

Se hace uso del término “Datos de posicionamiento” para definir al conjunto de variables obtenidas por el GPS mediante el uso de un script escrito en Python haciendo uso de la biblioteca Pynmnea2.

El módulo GPS usado en este proyecto recolecta una gran cantidad de datos relacionados a la geolocalización del objetivo, pero para calcular los tiempos de arribo solo se preservan tres variables y se descarta todo lo demás. Estas variables son:

- Latitud
- Longitud
- Velocidad

3.6.7.3. Qt Multimedia

El módulo Qt Multimedia proporciona un amplio conjunto de funciones que le permite aprovechar fácilmente las capacidades multimedia de una plataforma, como la reproducción de medios y el uso de dispositivos de cámara y radio. (The QT Company, 2021). En este proyecto se utiliza para la proyección de videos publicitarios en el módulo de visualización.

3.6.7.4. Geopy.distance

GeoPy es una librería de Python para acceder a servicios de geocodificación.

GeoPy facilita a desarrolladores de Python localizar las coordenadas de direcciones, ciudades, países y puntos de referencia en todo el mundo mediante geocodificadores de terceros y otras fuentes de datos. (Aurelio Morales, 2021)

En este proyecto se hace uso de GeoPy para realizar el cálculo de la distancia entre dos puntos utilizando la latitud y longitud de estos. Esta distancia es una de las variables usadas en el algoritmo de cálculos de tiempos de arribo.

3.6.7.5. Pyserial

PySerial es una librería de Python que permite establecer comunicación directa con el módulo GPS por el puerto serial (RS-232). (Chris Liechti, 2015)

3.6.7.6. Socket

Los sockets son los extremos de un canal de comunicación bidireccional. Los sockets se pueden comunicar dentro de un proceso, entre procesos dentro de la misma máquina o entre procesos de máquinas de continentes diferentes.
Los sockets pueden ser implementados a través de un diferente número de canales: sockets de dominio UNIX, TCP, UDP, etc. La librería socket de Python provee clases específicas para manejar el transporte común, así como también una interfaz genérica para controlar todo lo demás. (Unipython, 2021)

El módulo socket de Python es utilizado en este proyecto para enviar los datos de posicionamiento recolectados por el módulo GPS a todos los nodos de red del sistema “Termina”

3.6.7.7. Time

El módulo Time de la biblioteca estándar de Python proporciona un conjunto de funciones para trabajar con fechas y/o horas.

3.6.7.7.1. Características de Time

- En el módulo Time hay funciones para obtener la fecha y/o hora de distintos tipos de relojes (incluido el reloj que ofrece el tiempo civil).
- Para obtener información relacionada con nuestro huso horario.
- Para convertir fechas y/o horas entre distintos tipos.
- Para validar y aplicar formatos de fechas y/o horas.
- Para detener durante un tiempo la ejecución de un programa.

3.6.7.8. B.A.T.M.A.N

El Better Approach To Mobile Adhoc Networking, o B.A.T.M.A.N. es un protocolo de enrutamiento que actualmente se encuentra en fase de desarrollo por la Comunidad "Freifunk" y destinadas a reemplazar el Optimized Link State Routing (OLSR).

Este protocolo de enrutamiento es la base en la que se sostiene el sistema “Término” Las particularidades sobre su implementación de detallan más adelante.

Capítulo 4: Diseño del sistema “Término”

En este capítulo se explica la arquitectura del sistema, se detalla la composición y el funcionamiento de los tres módulos que conforman el sistema “Término”

4.1. Arquitectura del sistema “Término”:

La arquitectura del sistema se describe en el siguiente diagrama de bloques.

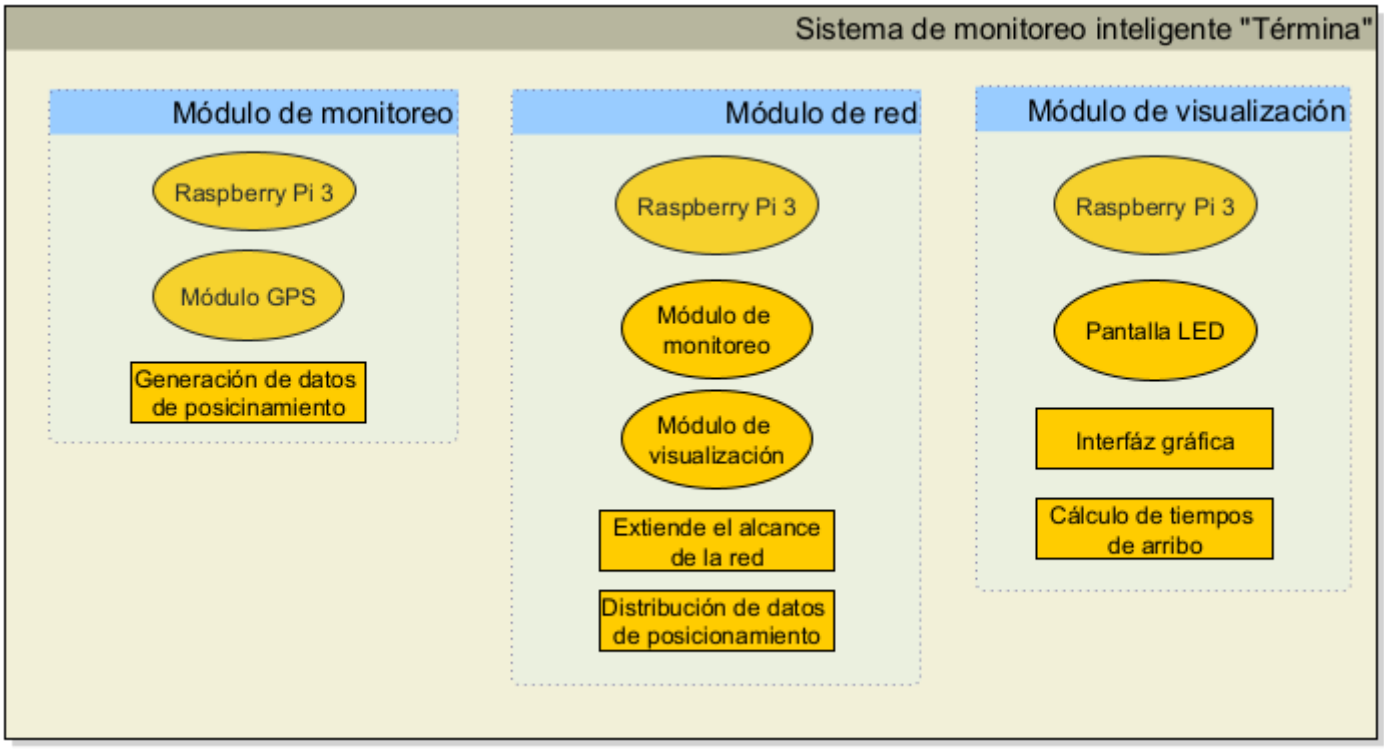

Ilustración 18: Diagrama de bloques del sistema "Término"

La arquitectura del sistema “Término” se encuentra dividido en tres módulos:

4.2. Módulo de monitoreo:

Se encarga de realizar el control de ruta, gestiona la adquisición de información del sensor de posicionamiento conectado al autobús (GPS) y reporta esta información a los nodos receptores ubicados en cada parada mediante un protocolo de comunicación.

El módulo de monitoreo se compone de una microcomputadora (Raspberry pi 3) que tiene los puertos suficientes (PCie x 1, microUSB, USB, RS232, HDMI,) para albergar los dispositivos electrónicos auxiliares (antena WiFi, GPS) necesarios para realizar la conexión inalámbrica a través del módulo de red y enviar los datos de posicionamiento al módulo de visualización.

4.3. Módulo de visualización:

El segundo módulo del sistema es la denominada interfaz gráfica de usuario (GUI), elemento que se convierte en medio de interacción para el usuario con el sistema. Al igual que en el primer módulo, el procesamiento de los datos recibidos a través del módulo de red es realizado por la microcomputadora Raspberry pi 3 y sus componentes auxiliares, mientras que una pantalla LED se encarga de mostrar los tiempos de arribo a los usuarios.

4.4. Módulo de red:

Finalmente, tanto el módulo de monitoreo a bordo de la unidad del TUC, como el módulo de visualización instalado en cada parada tendrán conectividad entre sí

mediante la implementación de una red malla conformada por 4 nodos (Un vehículo de prueba, una parada y dos nodos para extender la red). Este módulo comparte el uso del hardware presente en los dos módulos explicados anteriormente.

Cabe destacar que tanto paradas como unidades del TUC son consideradas como nodos de red y por tanto recibirán y retransmitirán los datos de posicionamiento obtenidos a cualquier otro nodo sin hacer ningún tipo de distinción, en otras palabras; la red considera paradas y unidades del TUC como iguales en todo momento.

Capítulo 5: Diseño de la red Mesh

En este capítulo se explica el diseño de la red Mesh del sistema “Término”. Se describe la topología con la que se construyó el circuito para la prueba de campo. Además, se explica el funcionamiento del protocolo de enrutamiento B.A.T.M.A.N adv y su aplicación en el sistema.

5.1. Diagrama de la Topología

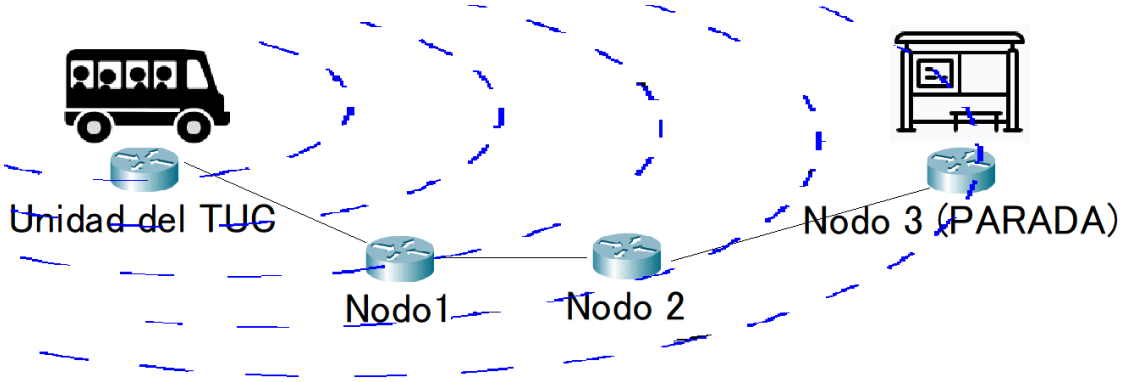

Ilustración 19: Topología de red Mesh para la prueba de campo

Para el montaje del prototipo se hace uso de 4 microcomputadoras Raspberry pi 3; que realiza las siguientes funciones:

- La primera microcomputadora se encuentra instalada junto con el dispositivo GPS dentro del vehículo particular (unidad del TUC) y se encargará de reunir los datos de posicionamiento de la unidad de transporte y transmitirlos al resto de nodos en la red.

- La segunda y tercera corresponden al nodo 1 y nodo 2. Están colocadas a lo largo del camino en donde se realizarán las pruebas de campo, y se encargarán de retransmitir los datos de posicionamiento obtenidos a todo

el resto de la red. Estos nodos no están equipados con el módulo de visualización.

- La cuarta está instalada con la pantalla led ubicada al final del camino (parada). En ella se realizará el cálculo de tiempos de arribo haciendo uso de los datos obtenidos a través de la red Mesh. Posteriormente, los tiempos de arribo y la distancia aproximada de la unidad del TUC a la parada son proyectados en la pantalla led a través del módulo de visualización.

5.2. Configuración de B.A.T.M.A.N adv

Para la elaboración de la red malla se configura el protocolo de enrutamiento B.A.T.M.A.N - ADV (Mejor alternativa a redes ad-hoc móviles - Avanzado) en los dispositivos Raspberry pi 3.

Primero se crea un script llamado start-batman-adv.sh en el directorio personal y se hace ejecutable. Esto se usará para habilitar las interfaces durante el arranque del sistema de otra forma estas configuraciones regresarán a su forma por defecto después de restablecer el Raspberry Pi 3:

En cada nodo de red los apartados de canal, essid y ap deben definirse como iguales. De otro modo no será posible establecer comunicación.

5.3. Funcionamiento de B.A.T.M.A.N adv

La mayoría de las otras implementaciones de protocolo de enrutamiento inalámbrico operan en la capa 3, lo que significa que intercambian información de enrutamiento mediante el envío de paquetes UDP y ponen en práctica su decisión de enrutamiento manipulando la tabla de enrutamiento del kernel. B.A.T.M.A.N - adv por otro lado, opera completamente en la capa 2 del modelo OSI. Debido a esta propiedad, la información de enrutamiento se transporta utilizando tramas Ethernet sin procesar y el tráfico de datos es manejado por B.A.T.M.A.N - adv quien se encarga de encapsular y reenviar todo el tráfico hasta que llega al destino, emulando así un switch de red virtual para todos los nodos participantes.

Por lo tanto, todos los nodos parecen ser enlaces locales y no son conscientes de la topología de la red, así como no se ven afectados por ningún cambio en la red. La forma más fácil de pensar en una red de malla B.A.T.M.A.N - adv es imaginar

a los nodos de la red como una banda de murciélagos; están ciegos, pero gracias a su habilidad natural de eco-localización son capaces de saber dónde están sus presas para capturarlas, orientarse, localizar obstáculos o encontrar más murciélagos. (Ambrosio, 2013).

B.A.T.M.A.N actúa como un switch de distribución. Cada nodo que ejecuta B.A.T.M.A.N - adv es equivalente a un puerto del switch.

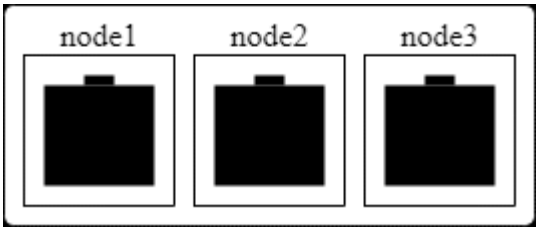

Ilustración 20: Cada nodo es un puerto del switch virtual B.A.T.M.A.N

Capítulo 6: Montaje del prototipo

En este capítulo se detalla el montaje del prototipo. Se describe el proceso de ensamblaje de los tres módulos que componen el sistema “Término”, posteriormente se realiza la prueba de campo para verificar el funcionamiento del prototipo.

6.1. Montaje del módulo de Monitoreo

El GPS se encuentra conectado a un módulo sim808 que responde a comandos AT introducidos en la terminal Minicom. Para optimizar el proceso de comunicación con el módulo sim808 se desarrolló una interfaz de usuario capaz de encender el módulo y solicitar los datos de posicionamiento de forma semiautomática.

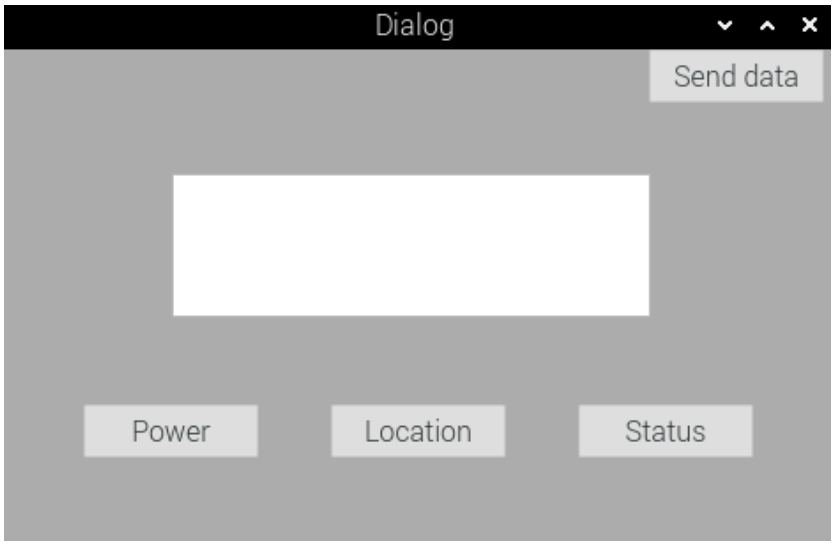

Ilustración 21: Interfaz de control para el módulo GPS

Aunque el módulo sim808 es capaz de obtener una gran variedad de datos de posicionamiento, para esta primera etapa se destacan solamente 3 valores específicos: velocidad, latitud y longitud, asimismo es preciso garantizar que estos

datos sean presentados en formato de grados decimales, esto a fin de facilitar el cálculo de tiempo de arribo que se explicará con más detalle en la etapa 3.

La unidad del TUC usualmente se encontrará en continuo movimiento por lo que el proceso de obtención de datos de posicionamiento, descrito anteriormente, debe ser constante y automático, para cumplir este requerimiento se creó un script en Python que, mediante el uso de la biblioteca Pynmea2, automatizará el proceso de obtención de datos de posicionamiento.

```
import geopy.distance, io, os, re, pynmea2, serial, time, sys
from socket import *

def connect_gps():
    ser = serial.Serial('/dev/ttyUSB0', 9600, timeout=1.0)
    sio = io.TextIOWrapper(io.BufferedRWPair(ser, ser))
    line = sio.readline()
    return line

def location():
    b = 1
    while b:
        line = connect_gps()
        if (line.startswith("$GPRMC")):
            long = pynmea2.parse(line).longitude
            lat = pynmea2.parse(line).latitude
            spd = pynmea2.parse(line).spd_over_grnd
            longitud = (str(long))
            latitude = (str(lat))
            speed = (str(spd))
            coordenadas = (longitud + "\n" + latitude + "\n" + speed)
        if not (line.startswith("$GPRMC")):
            b = 1
```

Ilustración 22: Script para obtener datos de posicionamiento.

El proceso de obtención de datos de posicionamiento es una operación que ocurre en cada unidad del TUC, debido a la enorme cantidad de unidades en servicio y para prevenir errores durante la etapa 3, es preciso catalogar y organizar los datos de posicionamiento obtenidos de cada unidad.

Para cumplir este requerimiento se asigna a cada unidad del TUC carpetas con códigos únicos. En estas carpetas se encuentra un bloc de notas conteniendo la velocidad, latitud y longitud correspondiente a la unidad que ha recolectado estos datos, de esta manera, se obtiene un identificador que permite mostrar el tiempo de arribo de una unidad específica.

```
def save_file():
    while True:
        s=socket(AF_INET, SOCK_DGRAM)
        s.setsockopt(SOL_SOCKET, SO_BROADCAST, 1)
        file = open("/home/pi/Desktop/GPS_DATA.txt", "r+")
        a = location()
        print (a)
        file.write((str(a)) + os.linesep)
        file.seek(0)
        file.write((str(a)) + os.linesep)
        file.truncate()
        file.close()
        file = open("/home/pi/Desktop/GPS_DATA.txt", "r")
        SendData = file.read(1024)
        s.sendto(str.encode(SendData),('169.254.255.255',12345))
        s.close()

while True:
    save_file()
    if False:
```

Ilustración 23: Script para almacenar los datos de posicionamiento en carpetas.

Una vez que se obtienen las variables requeridas para realizar el cálculo de tiempo de arribo, esta información es transmitida al nodo de red más cercano (parada o unidad del TUC) y continuará siendo retransmitida incluso luego de llegar a la

parada más cercana. Este proceso de transmisión masiva de datos de posicionamiento corresponde a la segunda etapa que se explica a continuación.

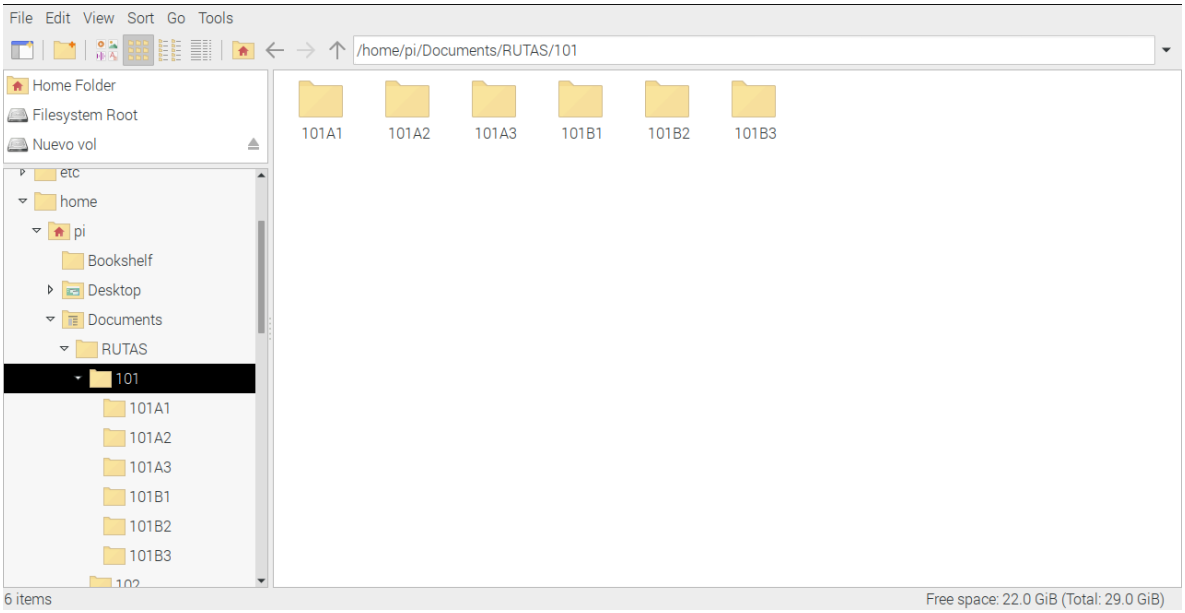

Ilustración 24: Carpeta que contiene los datos de posicionamiento.

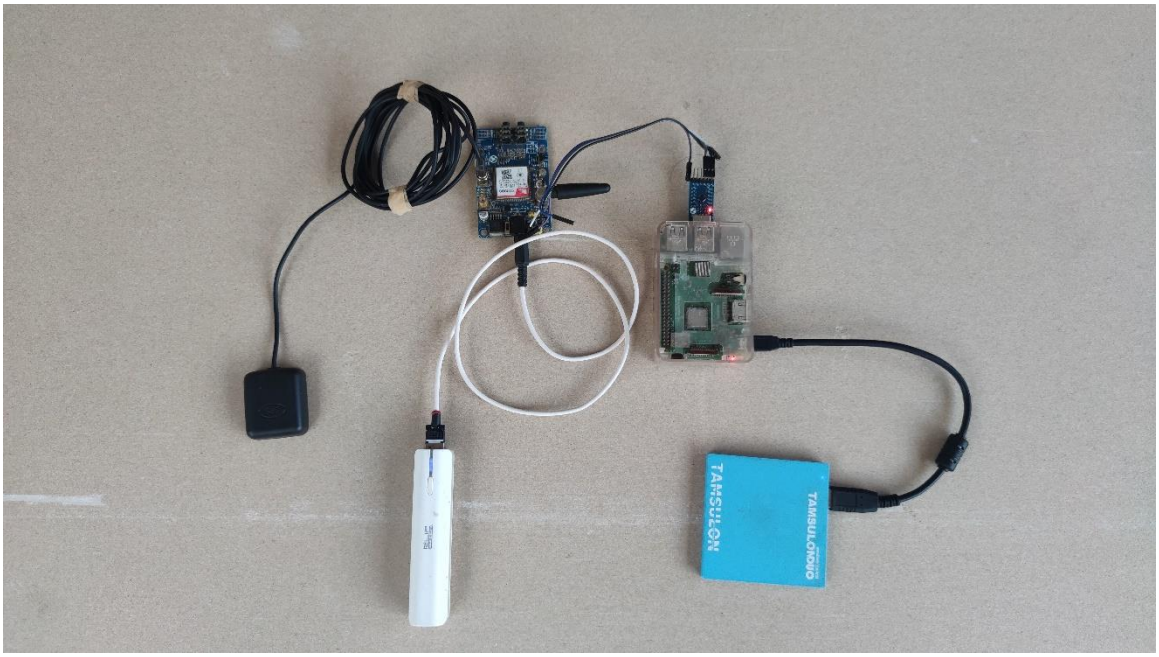

Ilustración 25: Módulo de monitoreo.

6.2. Montaje de Módulo de red

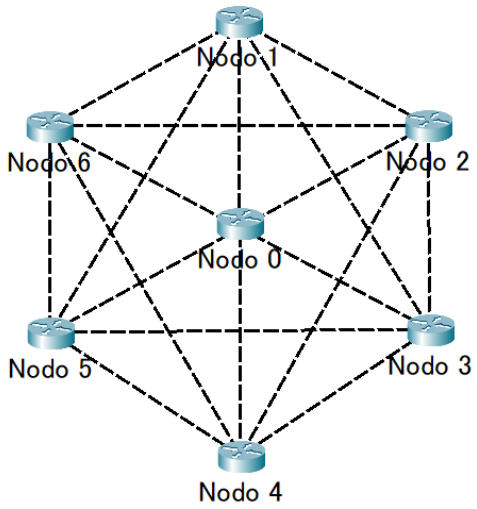

Ilustración 26: Modelo de red Mesh.

Cada parada de buses en la que se encuentre instalado un Módulo de visualización corresponde a un nodo de la red Mesh, de la misma forma, cada unidad del TUC que transmita sus datos de posicionamiento también forma parte de la red. Además, hay otros nodos que se encargan de recibir y retransmitir datos de posicionamiento.

El proceso de cálculo de tiempo de arribo y la visualización de estos se realiza por separado en cada parada; esto significa que debe garantizarse un flujo de datos de posicionamiento constante, en tiempo real por toda la red.

Para cumplir este requerimiento se opta por transmitir los datos de posicionamiento de cada unidad del TUC por medio de una transmisión masiva (broadcast). Gracias a la naturaleza de la red Mesh, cada nodo se encargará de retransmitir los datos de posicionamiento a sus vecinos. Es preciso recordar que cada unidad del TUC es a su vez un nodo de la red Mesh, Gracias a su movilidad la red es incluso más flexible.

La retransmisión masiva (broadcast) de los datos de posicionamiento se lleva a cabo, inicialmente, en el Raspberry pi 3 ubicado en cada unidad del TUC mediante el uso del siguiente script escrito en Python.

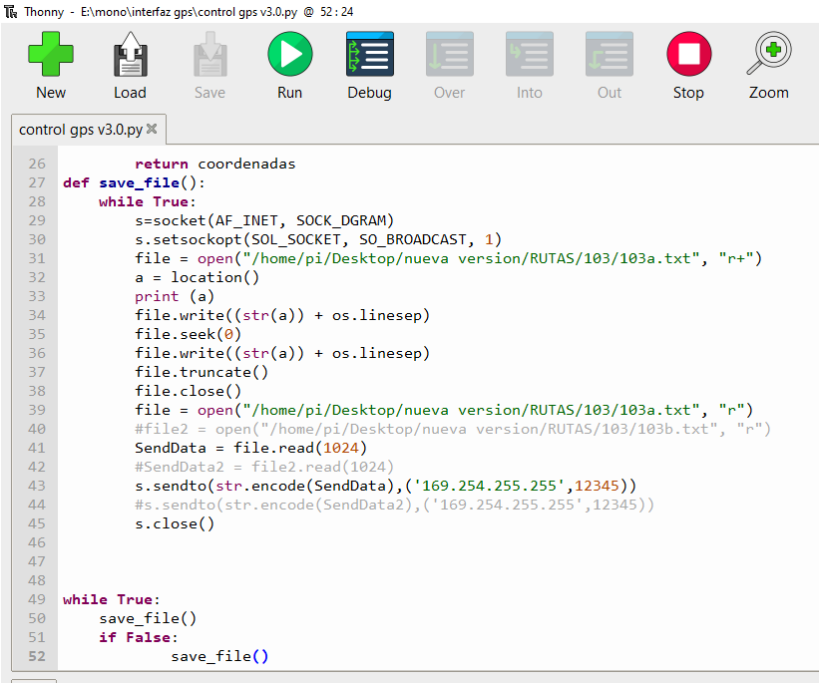

Ilustración 27: Script para transmisión masiva de datos de posicionamiento.

Con esto, la parada más cercana que forme parte de la red obtiene el archivo de texto que contiene los datos de posicionamiento de la unidad del TUC e inmediatamente retransmite estos datos a los nodos más cercanos de forma automática mediante el protocolo de enrutamiento B.A.T.M.A.M adv.

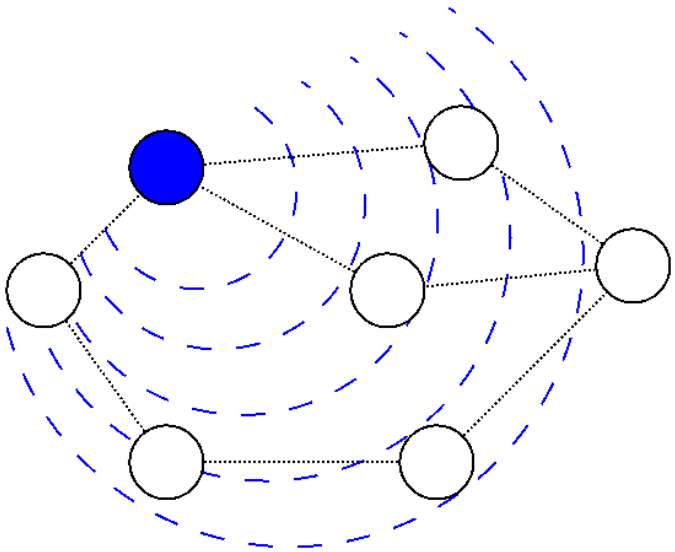

Ilustración 28: Transmisión de paquetes mediante B.A.T.M.A.N adv.

Al mismo tiempo, cada nodo que reciba los datos de posicionamiento y que se encuentre equipado con un módulo de visualización ejecutará el cálculo de tiempos de arribo y finalmente los expondrá en la pantalla ubicada en cada parada. Este proceso corresponde a la tercera etapa de nuestro sistema ¨termina¨ que se detalla en seguida.

Ilustración 29: Nodos de la red.

6.3. Montaje del módulo de visualización

El script encargado del cálculo de tiempos de arribo es ejecutado en el Raspberry pi 3 de cada parada que posea un módulo de visualización. El script, escrito en Python, hace uso de los datos de posicionamiento del TUC contenidos en un bloc de notas y recibidos a través de la red Mesh para determinar los tiempos de arribo. El algoritmo que determina el tiempo de arribo de la unidad del TUC a la parada más cercana es el siguiente:

$$Tiempo = \frac{Distancia}{Velocidad} \qquad (1)$$

- En donde Distancia corresponde a la distancia entre la unidad del TUC y la parada. Este dato se obtiene en la primera etapa mediante el uso de la biblioteca geopy y la biblioteca pynmea2.

- Velocidad corresponde a la velocidad de movimiento de la unidad del TUC y se obtiene durante la primera etapa mediante el uso de la biblioteca pynmea2 y el módulo GPS instalado en la unidad del TUC.

- Finalmente, Tiempo corresponde al tiempo de arribo de la unidad del TUC a la parada y se calcula dividiendo Distancia sobre Velocidad.

- Una vez calculado el tiempo de arribo de la unidad del TUC este es proyectado en la interfaz gráfica del módulo de visualización.

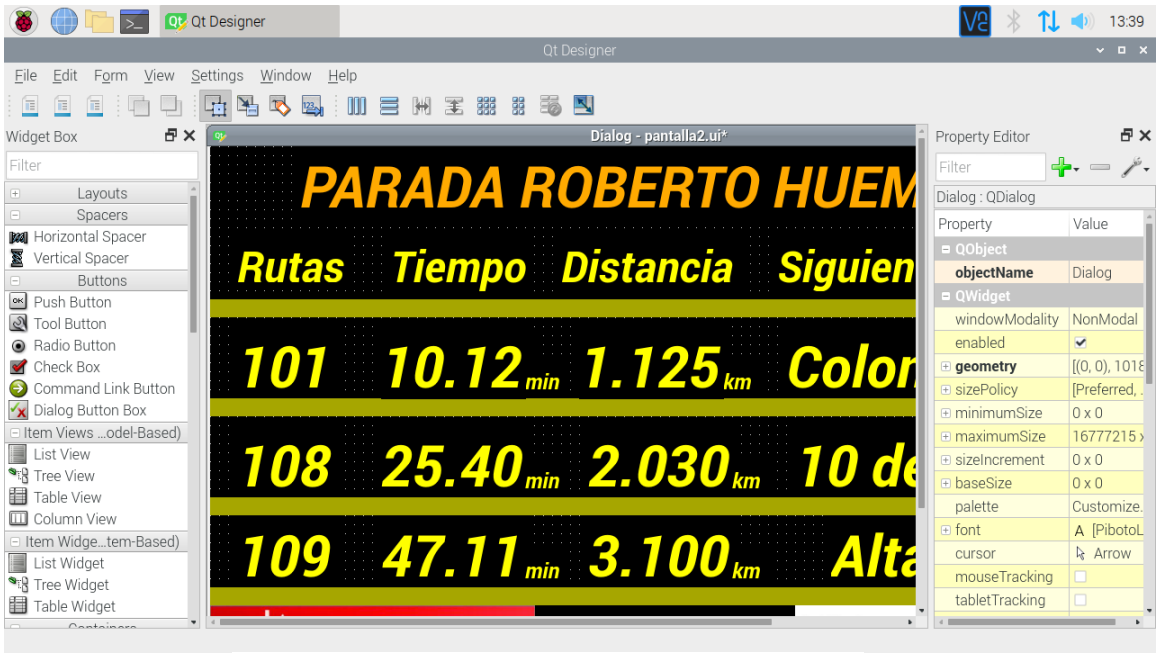

Ilustración 30 Interfaz gráfica elaborado en QTDesigner.

Para facilitar al usuario una correcta interpretación de la información presentada en el módulo de visualización se establecen las siguientes directrices:

- Regla # 1: Si la velocidad de movimiento es inferior a 3 km/h, los datos de posicionamiento obtenidos a través de la red Mesh serán ignorados para el calcula de tiempo de arribo. De lo contrario se ejecutará la regla #3.

- Regla # 2: Si la latitud de la unidad del TUC es mayor o igual a la latitud de la parada, los datos de posicionamiento de esa unidad serán ignorados para el cálculo de tiempo de arribo, De lo contrario se aplica la regla #3.

- Regla # 3: La distancia y el tiempo de arribo mostrados en la interfaz se actualizarán cada 1 minuto.

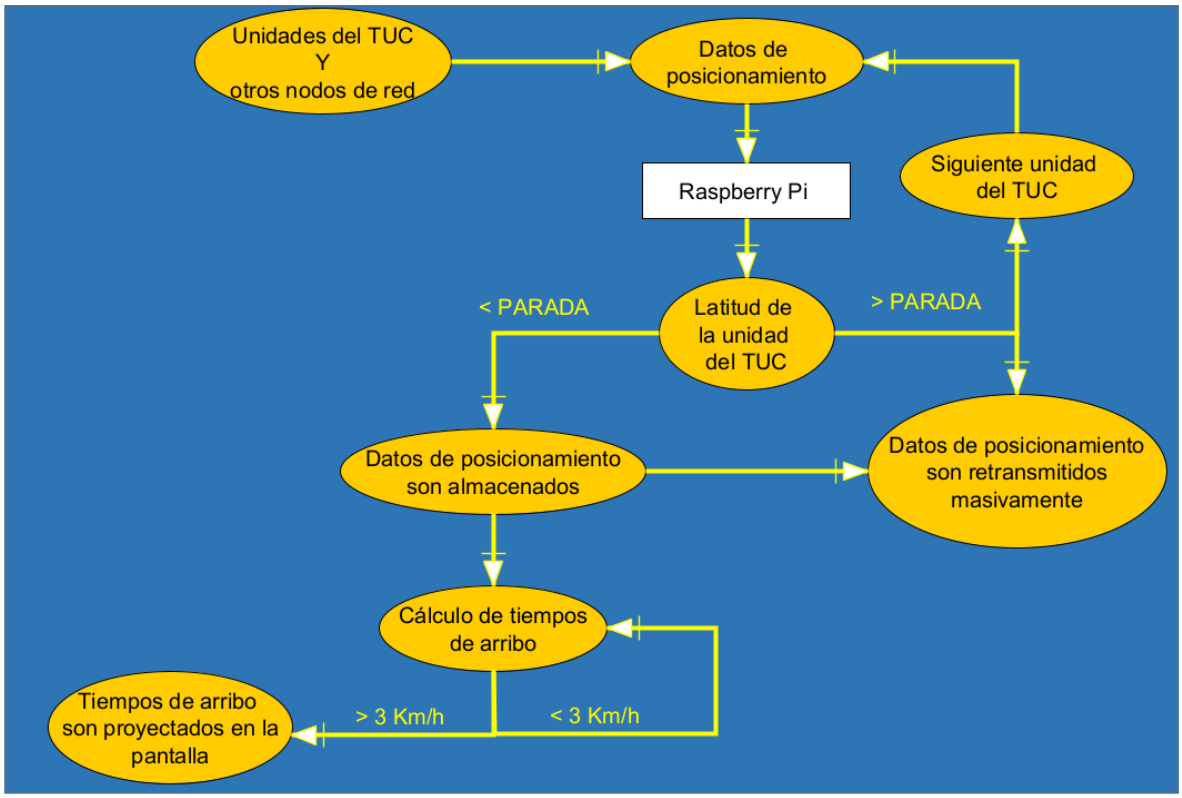

Ilustración 31: Diagrama de flujo del sistema "Término"

Ilustración 32: Prototipo del módulo de visualización.

6.4. Prueba de campo

La prueba de campo del sistema “Término” se realizó en el Residencial Colinas de Santa Cruz; se escogió la calle Isabel como circuito de pruebas.

Ilustración 33 Circuito de pruebas resaltado en amarillo.

La parada se instaló a 123 metros del vehículo de pruebas.

Ilustración 34 Parada; pantalla led mostrando los tiempos de arribo.

El módulo GPS se instaló en el asiento de pasajero del vehículo de pruebas.

Ilustración 35:Módulo GPS instalado en vehículo de pruebas.

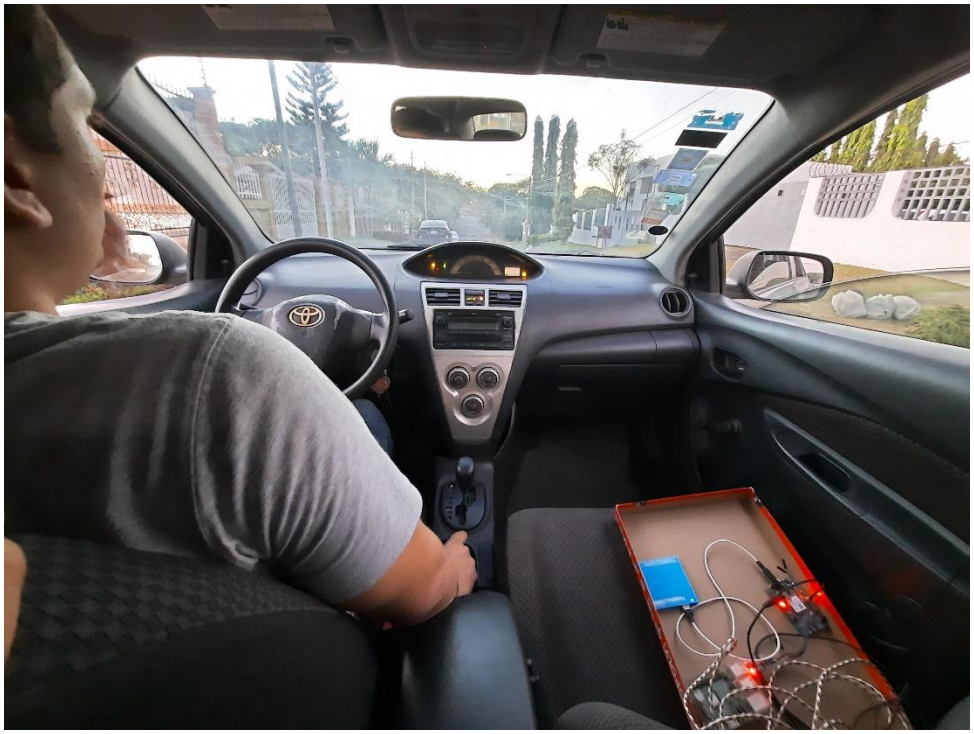

Ilustración 36: Vista desde el interior del vehículo de pruebas.

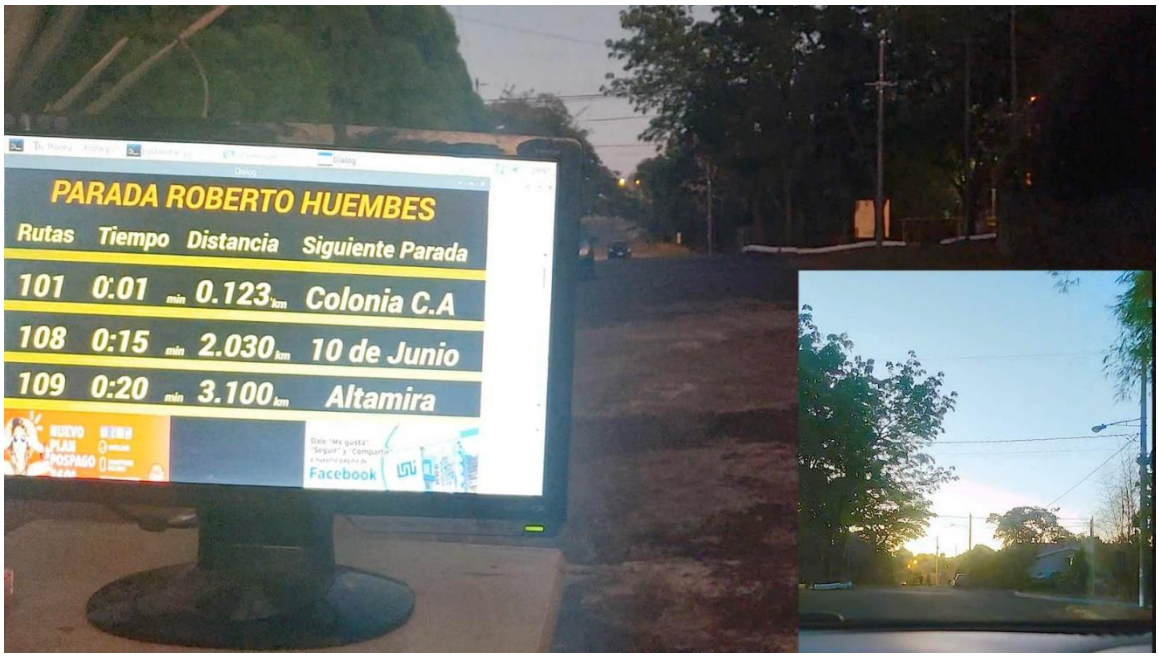

Ilustración 37: Vehículo de pruebas se acerca a la parada.

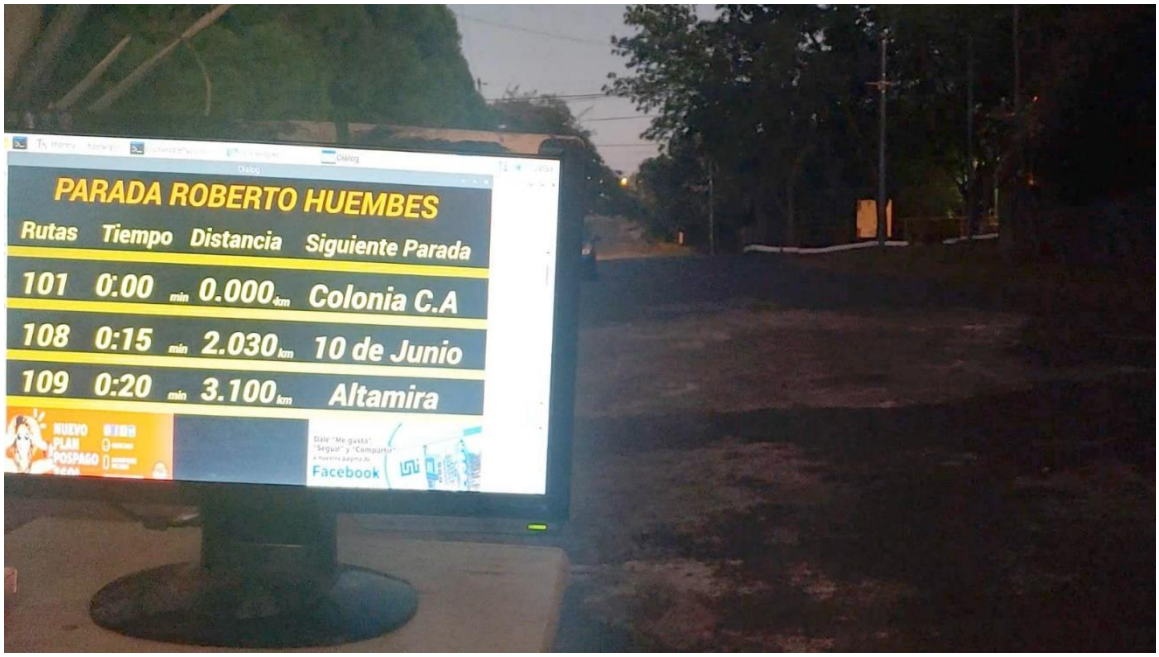

Ilustración 38: Vehículo sobrepasa la parada y tiempo se reduce a cero.

En el siguiente cuadro se detalla el costo de desarrollo del prototipo:

Módulo de monitoreo		
Cantidad	**Artículo**	**Precio**
1	Raspberry pi 3	$30
2	Antena GPS	$20
1	Módulo SIM808	$40
2	Batería portable	$8
40	Cables Jumper	$8
1	Adaptador DC9V para vehículo	$70
	Mano de obra	$440
	Total	$616

Módulo de visualización		
Cantidad	**Artículo**	**Precio**
1	Raspberry pi 3	$30
1	Pantalla LCD	$150
1	Adaptador VGA/HDMI	$18
	Mano de obra	$440
	Total	$638

Módulo de red		
Cantidad	**Artículo**	**Precio**
2	Raspberry pi 3	$30
2	Batería portable	$16
	Mano de obra	$200
	Total	$246

Gran total	**$1,500**

IV. Conclusiones

Este documento hace constatar que se completaron todos los objetivos planteados para el desarrollo del prototipo para un Sistema Inteligente de Monitoreo en Tiempo Real, de Arribo Por Parada y por Unidad del Transporte Urbano Colectivo (TUC) de Managua, Nicaragua.

Primeramente, se realizó una visita guiada a la Cooperativa de buses 165 para entrevistar al presidente y recolectar información sobre el estado del servicio del TUC en managua. Esta información fue útil para determinar los requerimientos del sistema “Término”.

Seguidamente, se diseñó una red inalámbrica de topología tipo Mesh, mediante la implementación del protocolo de enrutamiento B.A.T.M.A.N adv. Así mismo, se completó el desarrollo del prototipo del sistema de monitoreo del TUC, mediante la instalación del módulo GPS en el vehículo de pruebas. Los datos de posicionamiento fueron capturados, analizados y distribuidos con éxito en la red Mesh del sistema “Término”. Además, se completó el diseño de una interfaz gráfica para proyectar los tiempos de arribo de la unidad del TUC (vehículo de pruebas, en la pantalla led ubicada en la parada designada. Se detallaron los costos de producción del prototipo, incluyendo mano de obra, para un total de $1500.

Finalmente, se realizaron pruebas de campo para evaluar el sistema de monitoreo “Término”, obteniéndose resultados satisfactorios.

V. Recomendaciones

Se recomienda hacer pruebas al prototipo en una escala mayor. Para esto se debe ampliar el alcance de la red con más paradas y vehículos de prueba. Esto nos permitirá realizar pruebas de estrés al sistema y analizar situaciones como accidentes de tráfico, congestión de tráfico y su efecto en el desempeño de del protocolo de enrutamiento B.A.T.M.A.N adv y el algoritmo de cálculo de tiempos de arribo.

También se recomienda realizar la documentación oficial para el uso del sistema, lo cual facilitaría su implementación en un entorno a mayor escala.

El sistema “Término” es parte de un servicio público por lo que está expuesto a todo tipo de ataques que afectarán su funcionamiento, se recomienda añadir una capa de seguridad a la red para protegerla de actividad maliciosa que podría llegar a afectar el comportamiento de la red.

Se recomienda actualizar el hardware actual por equipo especializado que ofrezca mejores prestaciones, esto incrementaría la eficiencia y precisión del sistema.

Para finalizar, se recomienda añadir opciones de accesibilidad para que personas con cualquier tipo de discapacidad visual puedan beneficiarse del sistema.

VI. Bibliografía

Aivar Annamaa. (2020). *Thonny Python IDE for beginners*. Obtenido de https://thonny.org/

Ambrosio, L. G. (2013). *Eco-localización, como se orientan los murciélagos*. Obtenido de https://www.fisica.unam.mx/es/noticias.php?id=679#:~:text=Los%20murci%C3%A9lagos%20emiten%20sonidos%20de,trav%C3%A9s%20de%20sus%20membranas%20auditivas.

Arduino project's foundation. (2018). *What is Arduino?* Obtenido de https://www.arduino.cc/en/Guide/Introduction

Aurelio Morales. (2021). *mappingGIS*. Obtenido de https://mappinggis.com/2018/11/geocodificacion-con-geopy/

BEETRACK. (2021). *Sistema de seguimiento, monitoreo y evaluación en logísica*. Obtenido de https://www.beetrack.com/es/blog/sistema-de-seguimiento

CEPAL. (2012). *Serie Seminarios y Conferencias N° 74 .*

Chacón, F. C. (2012). El Sistema de Control del Transporte Urbano Colectivo de Managua.

Chacón, L. F. (2012). El Sitema de Control del Transporte Urbano Colectivo de Managua. Obtenido de http://repositorio.uca.edu.ni/405/1/UCANI3124.PDF

Chris Liechti. (2015). *pySerial 3.0 documentation*. Obtenido de https://pythonhosted.org/pyserial/pyserial.html#overview

Donat, W. (2014). Apress Learn Raspberry Pi Programming with Python.

Ecured. (2018). *Red de computadoras*. Obtenido de EcuRed: https://www.ecured.cu/Red_de_computadoras

Estrella, J. (2018). *Topologia en Malla*. Obtenido de jorge-star.galeon.com: http://jorge-star.galeon.com/MALLA.html

Freifunk. (2021). *B.A.T.M.A.N. Advanced Documentation Overview*. Obtenido de https://www.open-mesh.org/projects/batman-adv/wiki

Fundación Raspberry PI. (2018). *Raspberry PI*. Obtenido de https://www.raspberrypi.org/help/faqs/

HarperCollins Publishers Limited . (2018). *Collins Dictionary*. Obtenido de https://www.collinsdictionary.com/dictionary/english/public-transport

ITERNOVA. (2011). *Qué son los Sistemas Inteligentes de Transporte (ITS)*. Obtenido de Oxford University Press: https://www.tecnocarreteras.es/2011/04/11/que-son-los-sistemas-inteligentes-de-transporte-its/

Lara, R. (29 de Agosto de 2017). *Transporte público de Managua podría cambiar totalmente*. Obtenido de http://diariometro.com.ni/: http://diariometro.com.ni/nacionales/142173-transporte-publico-managua-podria-cambiar-totalmente/

National Coordination Office for Space-Based Positioning, Navigation, and Timing. (2018). *GPS.gov*. (U.S. government) Obtenido de GPS: https://www.gps.gov/systems/gps/

National Marine Electronics Association. (2021). *NMEA 0183 Standard*. Obtenido de https://www.nmea.org/content/STANDARDS/NMEA_0183_Standard

Nova, P. R. (2013). *Comandos Hayes*. Obtenido de https://sites.google.com/site/telecomuniccionesredes/comandos-hayes-1

Ojeda, B. E., & Salmerón, D. A. (Enero de 2013). *Análisis del sistema de monitoreo en posición y tiempo para el transporte urbano colectivo de Managua por medio de GPS.* Obtenido de Repositorio UNAN: http://repositorio.unan.edu.ni/5420/1/93741.pdf

ONU. (Septiembre de 1953). *El transporte en el istmo centroamericano.* Obtenido de Cepal: http://archivo.cepal.org/pdfs/1953/S5300044.pdf

Patiño, Fletscher, & Aedo. (2014). Diseño de un Sistema de Monitoreo, Administración y Recaudo inteligente para el Transporte Público. *III Congreo Internacional de Telecomunicaciones TELCON-UNI 2014*, 6.

Penalvch. (2015). *Minicom*. Obtenido de https://help.ubuntu.com/community/Minicom

Pérez-Salas, G. (2012). CEPAL, División de Recursos Naturales e Infraestructura .

Pittman, R. (2021). *Python library for parsing the NMEA 0183 protocol (GPS)*. Obtenido de https://github.com/Knio/pynmea2

Real Academia Española. (2018). *Definición de WiFi*. (Real Academia Española) Obtenido de http://dle.rae.es/?id=c6ehZd8

Sánchez, V. M. (6 de Septiembre de 2018). *Internet de las Cosas. Horizonte 2050.* Obtenido de http://www.ieee.es/publicaciones-new/documentos-de-investigacion/2018/DIEEEINV17-2018.html

System Analysis and Program Development (SAP). (2018). *¿Qué es el internet de las cosas (IoT)?* Obtenido de Sap: https://www.sap.com/latinamerica/trends/internet-of-things.html

The QT Company. (2021). Obtenido de https://doc.qt.io/qt-5/qtdesigner-manual.html

The QT Company. (2021). *Qt Documentation*. Obtenido de https://doc.qt.io/qt-5/multimediaoverview.html

Thompson , M., & Green, P. (2020). *Raspbian*. Obtenido de https://www.raspbian.org/

Unipython. (2021). *PROGRAMACIÓN DE REDES EN PYTHON: SOCKETS*. Obtenido de https://unipython.com/programacion-de-redes-en-python-sockets/

Wendell Odom. (2020). CCNA 200-301, Volume 1.

VII. Anexos

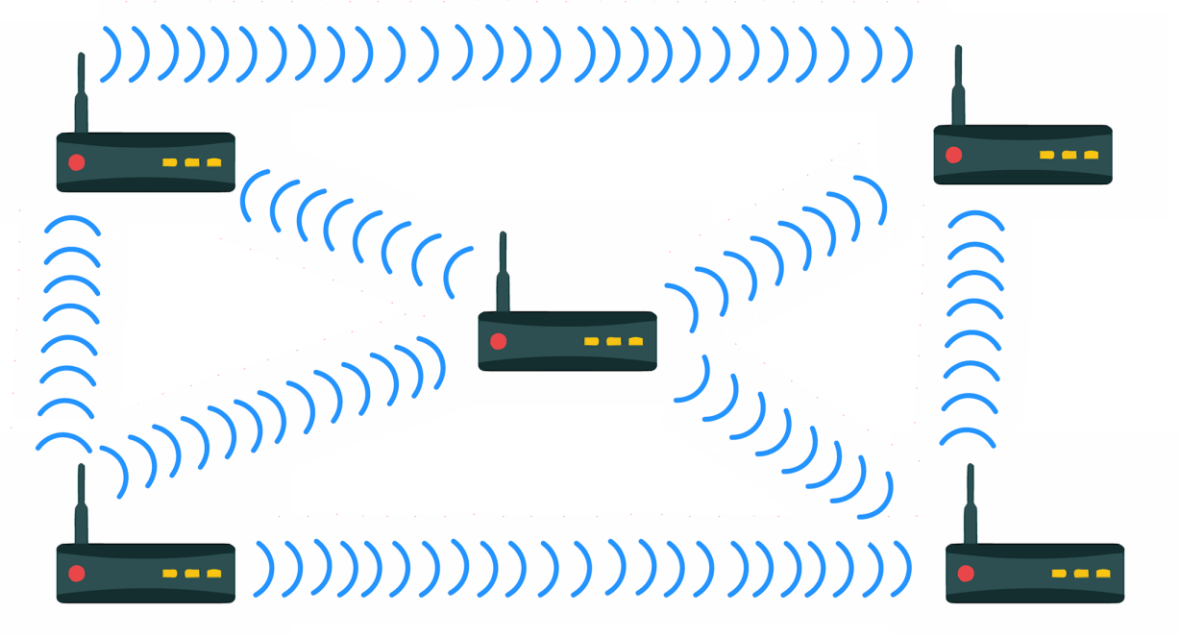

Fig. 3: Red de malla.

Fig. 4: Ejemplo del despliegue de la red.

1. Lista de comandos AT mas usados

Comando	Descripción
AT&F	Reestablecer los parámetros de fábrica
AT&V	Mostrar la configuración actual
AT&W	Guardar los parámetros establecidos en el perfil del usuario
AT+GMI	Solicitar la información del fabricante
AT+GSN	Solicitar el número identificador IMEI del dispositivo (International Mobile Equipment Identity)
AT+GMM	Obtener el modelo del dispositivo
AT+GMR	Obtener la versión del firmware del dispositivo
A/	Repite el último comando
ATA	Responde la llamada entrante
ATD><N>	Llama al número guardado en memoria
ATDL	Llama el último teléfono marcado

ATH	Se desconecta de la conexión actual
ATL	Establecer el volumen de la bocina monitor
ATT	Establecer la llamada de pulsos

2. Comando de instalación de bibliotecas Python

a) Geopy

pip install geopy

b) pynmea 2

pip3 install pinmea2

c) pyqt5 y qt designer

sudo apt-get install python3-pyqt5

luego

sudo apt purge python3pyqt5.qtsql

luego

sudo apt install python3-pyqt5.qtsql

luego

sudo apt-get install qttools5-dev-tools

luego

/usr/lib/x86_64-linux-gnu/qt5/bin/designer

Y listo

d) Minicom

apt-get install minicom