ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE INFORMACIÓN AL USUARIO USANDO ALGORITMOS DE PLANIFICACIÓN DE RUTAS PARA EL TRANSPORTE PÚBLICO DEL DISTRITO METROPOLITANO DE QUITO CASO DE ESTUDIO SECTOR CENTRO NORTE

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA EN "ELECTRÓNICA Y CONTROL"

DENISSE ESTEFANIA SUAREZ JIMENEZ
denisse.suarez@epn.edu.ec

DIRECTOR: JORGE ANDRES ROSALES ACOSTA
andres.rosales@epn.edu.ec
CO-DIRECTOR: ROBERTO OMAR ANDRADE PAREDES
roberto.andrade@epn.edu.ec

Quito, julio 2018

AVAL

Certifico que el presente trabajo fue desarrollado por Denisse Estefanía Suárez Jiménez, bajo mi supervisión.

ANDRES ROSALES ACOSTA
DIRECTOR DEL TRABAJO DE TITULACIÓN

ROBERTO ANDRADE
CODIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo Denisse Estefanía Suarez Jiménez, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

DENISSE ESTEFANIA SUAREZ JIMENEZ

DEDICATORIA

A la pequeña Jen y a mis abuelitos. En memoria de Pablo.

AGRADECIMIENTO

Quiero agradecer a mi familia que siempre me estuvo apoyando y me han ayudado a convertirme en la persona que soy ahora. En especial a mi mamá y hermano que fueron mi fuente de inspiración.

También agradezco mucho a todos mis amigos que conocí durante mi etapa universitaria que me apoyaron, cuidaron y me permitieron disfrutar este viaje.

A todo el equipo de trabajo de Mi Bus, con quienes pude aprender como es el trabajo en equipo y que tenemos el poder de cambiar las cosas y lograr que el mundo sea un lugar mejor.

A Valentín, Sebas, Marylin y Slin por enseñarme a construir un buen proyecto.

Al “Doc”, Andrés Rosales, mi gran amigo y colega, por todo su apoyo e inspiración para ser una mejor profesional.

… Gracias totales …

ÍNDICE DE CONTENIDO

AVAL II
DECLARACIÓN DE AUTORÍA III
DEDICATORIA IV
AGRADECIMIENTO V
ÍNDICE DE CONTENIDO VI
RESUMEN VII
ABSTRACT VIII
1. INTRODUCCIÓN 1
1.1 OBJETIVOS 3
1.2 ALCANCE 3
1.3 MARCO TEÓRICO 4
2. METODOLOGÍA 36
2.1 Creación de GTFS del sector centro norte 37
2.2 Implementación de un servicio REST para calcular el camino más corto 47
2.3 Cliente REST 58
2.4 Pruebas del planificador de rutas 59
2.5 Sistema de información de rutas y paradas de transporte público 60
2.6 Diseño de pruebas de usabilidad 63
3. RESULTADOS Y DISCUSIÓN 66
3.1 Prueba de integración del sistema 66
3.2 Pruebas de usuario 71
3.3 Análisis de los resultados 89
4. CONCLUSIONES 92
5. REFERENCIAS BIBLIOGRÁFICAS 95
ANEXO I 97
ORDEN DE EMPASTADO 103

RESUMEN

En Quito, el 73% de la movilización se realiza en sistema de transporte público. Sin embargo, los usuarios de este sistema notan la falta de información sistematizada sobre las rutas y paradas de transporte. Esta problemática afecta a: ciudadanos quiteños, turistas extranjeros; y, migrantes extranjeros o nacionales.

En este proyecto desarrolla e implementa un sistema de planificación de rutas en formato digital que procesa archivos estandarizados de transporte público, analiza la información en base a un modelo matemático y algoritmo específico; y, entrega resultados que podrían ayudar a los usuarios de transporte público a movilizarse mejor en base a esta información.

Para realizar este sistema de planificación de rutas, se construyeron estos 3 elementos indispensables:

- Front-end: conecta al usuario con el Servicio de Mi Bus y grafica las rutas en un mapa digital usando el resultado entregado por el Sistema de planificación de rutas.
- Servicio de Mi Bus: lee los archivos GTFS (estándar de facto para digitalizar los datos de transporte público) y entrega información procesada de rutas y paradas al servicio de planificación de rutas.
- Servicio de planificación de rutas: arma grafos en base a la información de las paradas, aplica el algoritmo de Dijkstra (resuelve el problema de camino más corto dentro de un grafo).

Además, para comprobar el funcionamiento del sistema se realizaron 10 pruebas con usuarios de distintos perfiles.

La investigación demostró el alto grado de importancia que tienen los sistemas de planificación de rutas digitales en la movilización de los usuarios; además, se reconoció la utilidad de los principios y algoritmos aprendidos en relación a la robótica autónoma móvil en un contexto diferente.

PALABRAS CLAVE: Sistema de planificación de rutas, sistema de transporte público, movilización, front-end, servicios de Mi Bus, Servicio de Planificación de rutas, algoritmo de Dijkstra, grafos, robots móviles autónomos.

ABSTRACT

In Quito, 73% of the urban mobility takes place in the public transport system. However, the users note the lack of systematized information on routes and transport stops. This problem affects: citizens of Quito, foreign tourists; and, foreign or national migrants.

This research develops and implements a route planning system in digital format that processes standardized files of public transportation, analyzes information based on a mathematical model and specific algorithm; and, it delivers useful results for users.

To make this route planning system, these 3 essential elements were built:

- Front-end: connects the user with Mi Bus Service and draws the routes in a digital map using the result received by the Route Planning System.
- My Bus Service: reads the GTFS files based on the information entered by the user and sends processed information of stops and routes to the Route Planning Service.
- Route Planning Service: creates a graph based on the information of the stops, applies the Dijkstra algorithm and returns an array of nodes, which indicates the shortest path.

In addition, to test the usability of the system, 10 tests were conducted with different users' profiles.

The research demonstrated the high degree of importance that digital route planning systems have in the mobility of users. In addition, the utility of the principles and algorithms of autonomous mobile robotics in a different context.

KEYWORDS: Route planning system, public transport system, mobilization, front-end, Mi Bus services, route planning service, Dijkstra algorithm, graphs, autonomous mobile robots.

1. INTRODUCCIÓN

En Quito, aproximadamente el 73% de la movilización de los habitantes se la realiza usando transporte público. Por otro lado, debido a la baja calidad del servicio, el número de vehículos privados en la ciudad incrementa a un ritmo anual del 5.7%. [1]

Esto se traduce en [1]:

- Más un millón de vehículos en circulación para el año 2030, lo implica tener más del doble de los vehículos que circularon en el año 2016.
- Incremento de congestión vehicular, lo que implica gastar más tiempo de movilización, disminuyendo la calidad de vida de las personas.
- Incremento de la contaminación por CO2 dentro de la ciudad.

En este contexto, uno de los mayores problemas que enfrenta el usuario de transporte público en varias ciudades del Ecuador y de Latinoamérica es la cantidad limitada de canales de información digital sobre rutas y paradas.

Las necesidades de los usuarios han cambiado, ahora se demanda precisión en los datos, usabilidad y accesibilidad, pero, debido que existe una ciudad tiene una gran cantidad de información sobre rutas y paradas es imposible esperar que el usuario procese todos los datos por sí solo. Sobre todo, cuando necesita movilizarse a diferentes lugares de la ciudad o es nuevo en la ciudad.

Por ese emotivo nace la necesidad de contar con un sistema digital para que el usuario pueda consultar y seleccionar la ruta más óptima para llegar a su destino [2].

La planificación de rutas consiste en buscar el camino más corto entre dos puntos considerando todos los caminos disponibles; para alcanzar este objetivo se implementan algoritmos que reciban de entrada un mapa que represente:

- El ambiente dónde se puede movilizar.
- La ubicación del robot en el mapa.
- El destino final.

En base a esta información se obtendrá en la salida la ruta más corta a seguir, es decir, la de menor coste.

Estos algoritmos son usados en diversos campos como: la robótica móvil autónoma, el enrutamiento de redes, video juegos, secuencias genéticas, planificación de transporte público y aéreo, entre otros. [3]

Una mejora a los sistemas de información al usuario de transporte público es: la implementación de algoritmos de planificación de rutas en un buscador online de rutas y paradas, donde el usuario ingresa su origen y destino, para recibir información de las rutas disponibles.

Una de las formas de realizar esta mejora es través de la teoría de grafos, donde cada parada representa un nodo en espacio y las rutas de buses representan los vértices. Se debe tomar en cuenta que: para cada nodo se puede tener varios vértices, es decir una parada puede tener varias rutas disponibles. En este grafo se aplica el algoritmo de Dijkstra que identifica la menor distancia entre dos puntos [4]: este algoritmo realiza el cálculo de los pesos (kilómetros a recorrer) que existe entre el nodo inicial y el más cercano que está apuntando al destino. El objetivo de estos cálculos es presentar al usuario las diferentes rutas óptimas (trayectos con menor distancia) para llegar a su destino en transporte público.

Este trabajo estudia e implementa el algoritmo de Dijkstra para encontrar el camino más corto para llegar a un destino dentro de un grafo que representa las rutas y paradas del transporte público. Esta información estará digitalizada en formato GTFS [5]. Se podrá realizar consultas a este servicio desde un cliente web que implemente un servicio REST, protocolo usado para la transferencia de información entre servicios web [6], lo que permitirá integrarlo a un sistema de información de viajes de transporte público. Adicionalmente se diseñará una interfaz gráfica que permita al usuario ingresar el origen y destino para obtener la ruta más corta para llegar a su destino, teniendo como caso de estudio el sector centro norte del Distrito Metropolitano de Quito.

Las rutas y paradas se obtendrán del sistema de mapas abiertos OpenStreetMap. Se usará las herramientas abiertas del emprendimiento social Mi Bus para el manejo de esta información con el objetivo de generar, editar e interpretar los archivos GTFS.

1.1 OBJETIVOS

El objetivo general de este proyecto es realizar el diseño e implementación de un sistema de información al usuario que use un algoritmo de planificación de rutas para el transporte público del Distrito Metropolitano de Quito caso de estudio el sector centro norte.

Los objetivos específicos de este Proyecto Integrador son:

- Analizar herramientas necesarias para la digitalización de rutas en formato estándar GTFS.
- Digitalizar las rutas de sistema de transporte público del sector centro norte del Distrito Metropolitano de Quito para poder representarlas en un grafo.
- Diseñar e implementar un algoritmo de planificación de rutas que permita calcular la ruta más corta que un usuario de transporte público puede tomar para llegar a un destino requerido.
- Implementar un sistema de información al usuario que integre el algoritmo de planificación de rutas y permita ingresar al usuario el origen y destino de la trayectoria a ser calculada.
- Realizar pruebas de usabilidad para validar el funcionamiento del sistema implementado.

1.2 ALCANCE

Se realizará la búsqueda de herramientas para la creación y edición de archivos GTFS. Se procederá a usar estas herramientas en la creación de estos archivos para las rutas que circulan por el sector centro norte del Distrito Metropolitano de Quito y que puedan ser integradas a sistema de planificación de rutas. Esta información se la construirá en base a los datos de las paradas del sector centro norte, que se obtenidos de OpenStreetMaps, sistema de mapas digital construido en base a información abierta.

Se realizará el diseño e implementación de un algoritmo de planificación de rutas usado actualmente en robots móviles en el contexto de optimizar un sistema de información de transporte público, lo que permitirá realizar el cálculo de la ruta más óptima disponible en bus para que un usuario pueda llegar a un destino deseado dentro de la ciudad.

Se creará un sistema digital de información al usuario que permita ingresar fácilmente el origen y destino deseado, y que responda con los pasos de la ruta más corta que debería seguir, en formato de texto y visualizado en un mapa. Este sistema estará alojado en un servidor en la nube lo que permitirá al usuario acceder a través del Internet a este servicio.

Se realizará pruebas con usuarios reales del sistema implementado, donde se realice la comparación del tiempo usado para escoger una ruta usando el planificador de ruta en contraste con medios no digitales o con un sistema que solo indique las rutas de manera estática. Así también se harán pruebas de usabilidad dónde se analizará que tan fácil resulta ingresar origen y destino, así como procesar la información entregada y usarla para movilizarse.

1.3 MARCO TEÓRICO

Transporte público inteligente

El aumento de congestión vehicular es un gran reto que tiene que enfrentar las ciudades actualmente, debido al incremento en la densidad de personas que la habitan. Según un informe de las Naciones Unidades, se tiene previsto que para el 2050 más del 70% de población mundial viva en ciudades, a pesar que el área urbana solo representa el 3% de la superficie de la tierra. [7]

Esto conlleva a diferentes necesidades y problemáticas entorno a la movilidad urbana. Uno de los principales objetivos de la planificación urbana es buscar alternativas más limpias y acortar el tiempo de movilización, reduciendo el tráfico vehicular. Es por eso que la mayoría de ciudades están realizando grandes esfuerzos para mejorar el transporte público, ya que representa un medio eficiente, debido a que ocupa menos combustible y espacio en la vía pública, como se puede apreciar en la Figura 1.1. que es fue un experimento urbano que realizaron para contrastar el espacio usado en la vía pública para transportar el mismo grupo de personas en transporte privado, público y en bicicleta.

Figura 1.1. Experimento urbano sobre el espacio requerido para movilizarse en carro privado, bus y bicicleta. *[8]*

En la actualidad es de gran ayuda utilizar herramientas tecnológicas que permitan mejorar la gestión y servicio de transporte público. Muchas ciudades en el mundo están implementando sistemas inteligentes de transporte público que les permiten: [9]

- Sistema integral de control tarifario.
- Mejorar el servicio y la comunicación con el usuario.
- Mejorar la planificación y gestión de flota
- Mejorar el asesoramiento de viaje para el usuario.

Adicionalmente a esto permite recolectar datos históricos para realizar planificaciones urbana anticipada, mejorar las rutas de transporte público e incentivar a más gente use el transporte público.

En este trabajo se hablará sobre la implementación de un sistema de planificación de rutas que entregue al usuario la información de rutas y paradas disponibles para completar un viaje dentro del sector centro norte del Distrito Metropolitano de Quito.

Sistemas de información de rutas y paradas para el usuario

Situación actual de los sistemas de información al usuario

En la actualidad, una gran mayoría de las ciudades del Ecuador y de otros países en vías de desarrollo prescinden de sistemas digitales de información de transporte público, esto

aumenta las posibilidades del usuario para escoger una ruta óptima y llegar a un destino específico.

Esto se convierte en un grave problema para los turistas, personas nuevas en la ciudad o una persona que con la necesidad de movilizarse a un lugar desconocido en la ciudad.

Dentro de los esfuerzos por brindar información al usuario sobre las rutas disponibles en cada parada, en el caso de Quito, se incorporó letreros que indican el número de ruta del bus, origen y destino de la ruta. Pero este medio de información al usuario, por sí solo, presenta muchas complicaciones, debido a que no todas las personas podrían deducir la ruta completa del bus con estos datos y al estar expuestos en la vía pública presentan un rápido deterioro como se puede ver en la Figura 1.2.

Figura 1.2. Parada deteriorada ubicada en la Av. Patria de la ciudad de Quito.

Este tipo de información en las paradas se debe complementar entregando a los usuarios las rutas completas en un mapa que haga referencia al origen y el destino.

En el caso de la ciudad de Cuenca, esta ciudad tiene disponible dos grandes mapas con diagramaciones de todas las rutas de transporte público, en la Figura 1.3. se puede ver 14 de las 29 líneas de buses que han sido digitalizadas. Aunque este método de información funciona bastante bien para ciudades que tienen dentro de sus sistema de transporte

público líneas de metro, cuando se trata de transporte modal se tiene los siguientes problemas:

- Dificultad para encontrar en el mapa la ubicación de origen y destino. Sobre todo, si es una zona nueva para el usuario.
- Dificultad para determinar la distancia más óptima sin la información de distancia entre paradas.
- Dificultad para seleccionar la combinación de rutas necesarias para llegar a un destino, debido a que toda la información se encuentra en diferentes mapas.
- Costos elevados para la distribución física entre los cuídanos.
- Dificultad por parte del usuario para transportarse por la ciudad usando esta información.
- Dificultad de los teléfonos móviles para abrir y navegar en el mapa en formato pdf.

Figura 1.3. Guía de buses de la ciudad de Cuenca *[10]*

En el último año, la empresa pública Metropolitana de Transporte de Pasajeros de Quito (EPMTPQ), dentro de su plan para modernizar el sistema integrado de transporte, invirtió tiempo y recursos en mejorar la experiencia de los usuarios de transporte público [11]. Esta Institución Pública integró los datos de la ciudad al planificador de rutas de Google Maps, lo que permite a los usuarios consultar los puntos de referencia de paradas, tiempos

estimados de llegada de los buses y obtener las mejores rutas para llegar a un destino específico, como se puede ver en la Figura 1.4.

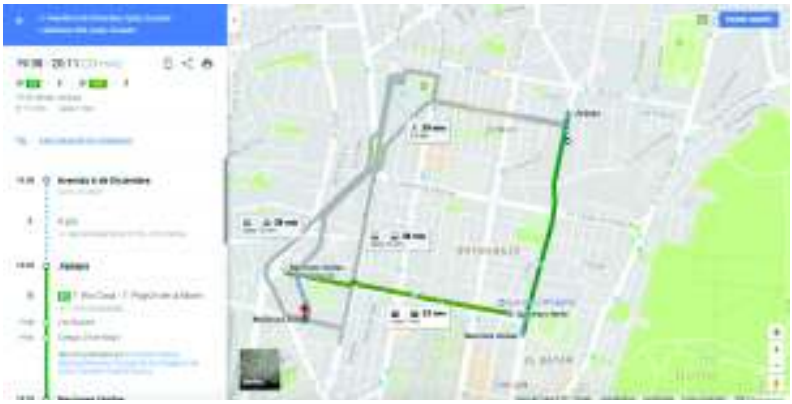

Figura 1.4. Planificador de rutas online del sector centro norte de la ciudad de Quito integrado al mapa de Google.

A pesar de los esfuerzos realizados para mejorar el sistema de información de transporte público, en varias ciudades principales del Ecuador aún no se cuenta con un planificador de rutas online, lo que dificulta la movilidad urbana.

Esto se puede apreciar en la Figura 1.5., que muestra el caso de Manta uno de los puerto del Ecuador.

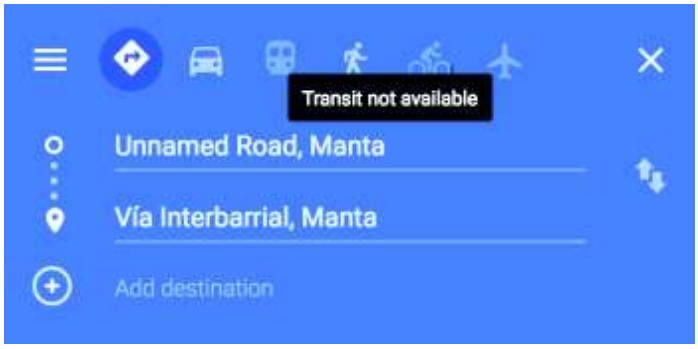

Figura 1.5. Consulta fallida de ruta de transporte público en la ciudad de Manta.

El crecimiento poblacional en las ciudades implica una restructuración y crecimiento del sistema de transporte; es decir, nuevos modos de transporte, nuevas líneas de buses y actualización de las rutas y paradas actuales.

Estos cambios deben ser notificados a los usuarios de la manera más rápida para no afectar su movilidad, además tiene que ser de fácil uso y de alta disponibilidad. Es por eso que durante los últimos años el uso de plataformas online, se ha convertido en una necesidad, porque permiten conocer la manera más eficiente de movilizarse en la ciudad a través del transporte público.

Sistemas digitales

El tener información de transporte público en un formato organizado permite una mejor comunicación con el usuario final. Esto ha dado paso a un auge de diferentes aplicaciones que permiten a los usuarios usar esta información para movilizarse de mejor manera dentro de la ciudad.

Este sistema tiene que ser fácil de usar y con la información suficiente para que el usuario pueda encontrar utilidad a esta herramienta. Además, debe ser un servicio que le permita consultar esta información desde su ordenador o teléfono móvil. Eso mejorará el sistema de información exponencialmente.

Las características principales que valoran los usuarios en un sistema de información de rutas y paradas son: [12]

- Búsqueda de rutas indicando solo origen y destino.
- Mapas con diagramación gráfica de la ruta.
- Cálculos aproximados de la distancia entre el punto de origen hacia la parada más cercana.
- Información clara y específica del bus que opera por esa ruta.
- Hora de salida.
- Duración aproximada del viaje.
- Costo del viaje.

En la Figura 1.6. se encuentra el ejemplo de un planificación de rutas que permite encontrar las posibles rutas de transporte público para llegar al Aeropuerto JFK desde la 5th Avenida, en la ciudad de Nueva York. Este planificador de rutas está disponible en la página https://wego.here.com.

Figura 1.6. Planificador de rutas Here.com.

Tener la información de rutas y paradas de transporte público disponible de manera digital se vuelve fundamental en ciudades grandes que cuentan con un complejo sistema de transporte público. Por ejemplo, Quito cuenta con 361 rutas y más de 3000 paradas, para un usuario sería imposible conocer todos estos puntos, dificultando su movilidad dentro de la ciudad [1].

La digitalización de la información de transporte público puede mejorar el uso de este servicio, porque provee información de más opciones de movilización dentro de la ciudad, además empodera a los usuarios a gestionar su viaje y mejorar sus tiempos de desplazamiento.

En la ciudad de Quito se desarrollaron varias iniciativas para digitalizar esta información y presentarla al usuario. Una de ellas es Mi Bus, un emprendimiento social que implementa herramientas tecnológicas para resolver problemas relacionados con la movilidad en las ciudades empezando por el transporte público [13]. Esta iniciativa ciudadana nace de un grupo de 6 estudiantes de la Escuela Politécnica Nacional de las facultades de Electrónica y Sistemas, que implementaron herramientas para la construcción y manejo de archivos GTFS.

Este trabajo busca analizar los algoritmos de planificación de rutas para mejorar el sistema de información de rutas y paradas de transporte público en las ciudades. También se explicará la metodología usada para abstraer toda esta información en formatos GTFS,

usando las herramientas de Mi Bus, de manera que facilite la incorporación de esta información en un sistema de código abierto para graficar rutas de buses, siendo el caso de estudio las rutas y paradas del sector centro norte del Distrito Metropolitano de Quito.

GTFS

Antes del 2005, no existía un formato de datos, referente al transporte público, que se pueda usar para representar y compartir esta información con el usuario final.

Posterior a esto, Google y Tri-Met trabajaron en conjunto para definir lo que ahora se conoce como "General TransitFeedSpecification (GTFS)", este formato abierto se ha convertido en el estándar de facto adoptado por un gran número de cooperativas a nivel mundial y se lo usa principalmente para entregar información de planificación de ruta para usuario [14].

Funciones principales

Se lo define al formato GTFS cómo un formato que permite a las agencias de transporte público entregar información de manera que pueda ser consumida por programadores que construyen aplicaciones de planificación de ruta en una manera interoperable [15].

Para cumplir con este objetivo, los feeds GTFS tienen que cubrir información estática de horarios, geolocalización de paradas y rutas en el mapa.

Estructura del GTFS

Los "feeds" GTFS están conformados por una serie de 13 archivos en texto plano (*.txt) en un formato donde cada valor está separado por comas y posee una cabecera que identifica cada conjunto de valores, estos archivos son únicos y se encuentran comprimidos en un archivo comprimido .zip.

Éstos archivos deben contener la información de las agencias, rutas, viajes, paradas, tiempo de paradas, entre otros datos, en la Figura 1.7. se indica los archivos de texto que conforman un GTFS generado para pruebas.

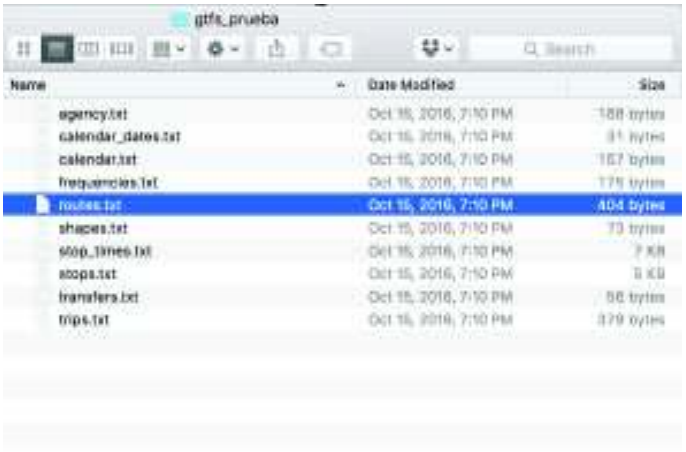

Figura 1.7. Estructura de un "feed" GTFS de prueba.

Las relaciones entre estos archivos, se construyen usando un ID de referencia, como se puede apreciar en la Figura 1.8., por ejemplo, el archivo Stop_Times.txt está relacionada con Stop.txt compartiendo el mismo Stop_id.

Esto facilita el procesamiento de estos archivos como tablas en una base de datos, lo que permite: evitar la duplicación de información, determinar que campos de la tabla son requeridos y qué opciones utilizar según la especificación ver Figura 1.8.

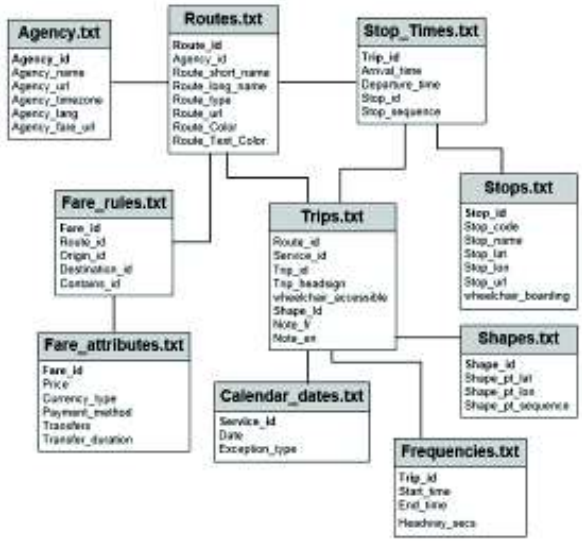

Figura 1.8. Diagrama de relación de archivos de un “feeds” GTFS.

Se debe asegurar la integridad datos y consistencia de las relaciones entre archivos, para que pueda ser consumida fácilmente por cualquier aplicación que entregue información al usuario final.

A continuación, se dará más detalle de lo que representa cada uno de los 6 archivos mínimos necesario para crear un “feed” GTFS y el formato de dato que se almacenarán en cada campo.

- **Agency.txt**

Aquí se encuentra la información relacionada a la agencia o cooperativa de transporte público, está conformada de los siguientes campos obligatorios:

  - **agency_id:** Este ID puede ser opcional en el caso de que exista solo una compañía registrada en el “feed”, caso contrario será obligatorio y representa un número único de identificación de la agencia o cooperativa.
  - **agency_name:** Es el nombre con el cual la agencia será indicada a los usuarios en el planificador de rutas.
  - **agency_url:** Es la dirección web de la agencia, deberá seguir la sintaxis establecida por la iniciativa www, para escribir URIs (UniformResourceIdentifier), no debe contener espacios.
  - **agency_timezone:** Define la zona horaria del sector donde operan las unidades de transporte público, no debe contener espacios.

- **Stops.txt**

Define la localización de cada parada existente dentro de la ruta.

  - **stop_id**: Es un dato único que sirve para identificar cada parada, una parada puede ser compartida por varias rutas.
  - **stop_name:** Es el nombre con el cual la parada será indicada a los usuarios en el planificador de rutas.
  - **stop_lat:** Define la latitud de la parada según el estándar WGS 84, usado en el sistema de posicionamiento global (GPS).
  - **stop_lon:** Define la longitud de la parada según el estándar WGS 84, usado en el sistema de posicionamiento global (GPS).

- **Trips.txt**

Define un viaje en una dirección, que sucede en un horario establecido e implica una relación entre dos o más paradas.

  - **trip_id:** Es un dato único que sirve para identificar cada viaje.
  - **route_id:** Es el ID de la ruta a la cual está asociado este viaje.

- **service_id:** Es el ID con el cual se identifica los días, en el calendario, para los que un viaje ofrece su servicio.

- **Routes.txt**

Define un conjunto de viajes que forman una ruta independiente y completa.

- **route_id:** Es un dato único que sirve para identificar de forma exclusiva cada ruta.
- **route_short_name:** Es el nombre corto con el que los usuarios reconocen la ruta, este dato será visualizado en el planificador de rutas.
- **route_long_name:** Es el nombre completo de la ruta, proporciona más información como destino o paradas de la ruta.
- **route_type:** Define la clase de transporte que opera en una ruta. Se usa valores enteros que van del 0 al 7.

- **Calendar.txt**

Define los días de la semana, en el cual la unidad de transporte público ofrece su servicio.

- **service_id:** Es un dato que permite identificar el conjunto de días de la semana que una ruta está operando.
- **monday:** Representa al día lunes y almacena un valor binario para indicar si la ruta está en operación.
- **tuesday:** Representa al día martes y almacena un valor binario para indicar si la ruta está en operación.
- **wednesday:** Representa al día miércoles y almacena un valor binario para indicar si la ruta está en operación.
- **thursday:** Representa al día jueves y almacena un valor binario para indicar si la ruta está en operación.
- **friday:** Representa al día viernes y almacena un valor binario para indicar si la ruta está en operación.
- **saturday:** Representa al día sábado y almacena un valor binario para indicar si la ruta está en operación.
- **sunday:** Representa al día domingo y almacena un valor binario para indicar si la ruta está en operación.
- **start_date:** Permite identificar el primer día del año en el cual el servicio está disponible, tiene que estar en formato AAAAMMDD.

- **end_date:** Permite identificar el último día del año en el que opera el servicio, tiene que estar en el formato AAAAMMDD.

- **Stop_time.txt**

Tabla de tiempo que determina la llegada o salida de una unidad de transporte público a una parada.

- **trip_id:** Es el ID que permite identificar a que viaje pertenece el horario.
- **arrival_time:** Es la hora, en formato HH:MM:SS, a la cual un bus debería llegar a una determinada parada para un viaje.
- **departure_time:** es la hora, en formato HH:MM:SS, a la cual un bus debería salir de una determinada parada para un viaje.
- **stop_id:** Es el ID que permite identificar a la parada que nos estamos refiriendo.
- **stop_sequence:** Determina la secuencia que deben tener las paradas para poder formar un viaje, se debe de usar número enteros positivos ascendentes.

Editores GTFS

Se debe tener en cuenta que cada vez que exista un cambio en los horarios de funcionamiento, rutas, paradas o cualquier cambio de los “feeds”, se debe generar un nuevo GTFS. Se los recomienda realizar estos cambios periódicamente y mínimo unas 4 a 5 veces por año, aunque se recomienda realizar cambios en un periodo menor de tiempo para reflejar mejor los cambios, haciendo más fácil el proceso de mantenimiento y sostenibilidad de esta información. Entregando siempre la información más actualizada a los usuarios [16].

En el mercado hay varias soluciones comerciales que ofrecen herramientas para generar y mantener archivos GTFS. Estas herramientas pueden ser instaladas dentro de los servidores de la agencia o usar servicios en la nube.

Debido a la popularidad del uso de este formato para representar la información de transporte público, se han desarrollado varios proyectos de software libre que permite la digitalización de información de transporte público en formato GTFS. Uno de ellos es Conveyal, un editor que posee una interfaz gráfica en la cual se puede modificar las paradas en un mapa, ingresando la latitud, longitud e información sobre la parada, además permite crear rutas en base a esta información, añadir horarios de funcionamiento y datos adicionales de la cooperativa de transporte. Para el presente trabajo se usará las

herramientas de Mi Bus, que es una ramificación de este proyecto adaptada a la ciudad de Quito y consta con la información de paradas obtenidas del sistema de mapas online OpenStreetMap, cómo se puede ver en la Figura 1.9. [17]

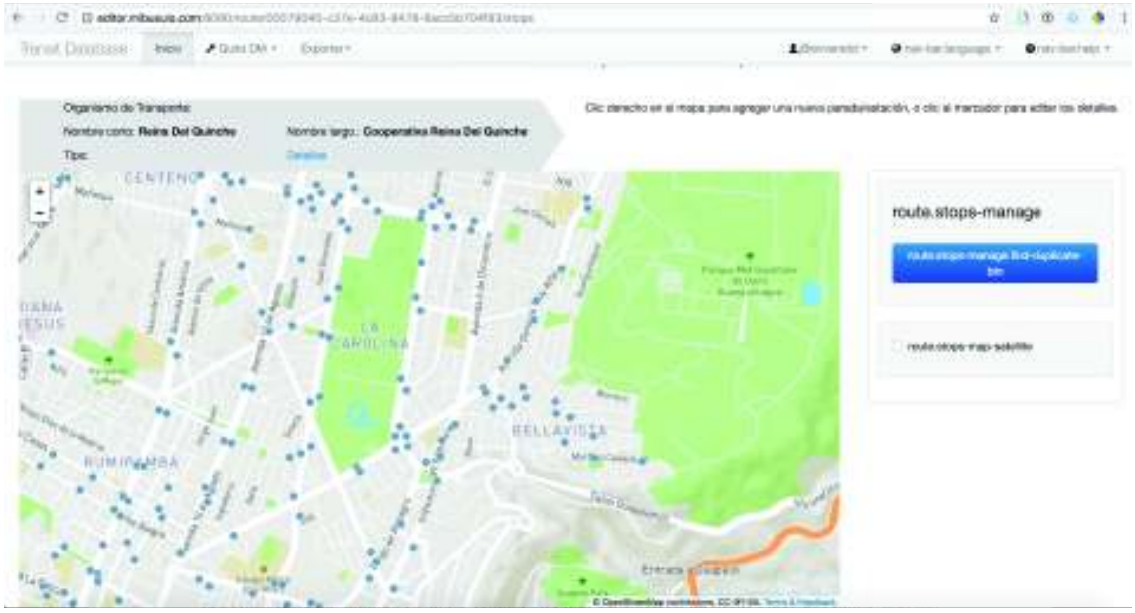

Figura 1.9. Interfaz gráfica para la edición de paradas del editor de GTFS de Mi Bus.

Validadores GTFS

Son herramientas que permite determinar si un "feed" de GTFS tiene problemas, cumple con las especificaciones definidas anteriormente o se lo puede mejorar antes de publicarlo.

Google ofrece su propio validador de archivos GTFS, esta herramienta funciona usando comandos de consola y entrega un informe HTML con datos de errores y advertencias encontradas [5]. Otro servicio, que funciona online, lo ofrece TransitScreen y permite cargar archivos directamente o mediante una URL. [18]

Teoría de grafos

Se define como grafo al modelamiento de un problema matemático discreto, donde se construye una red con nodos para abstraer información que representa el problema y la relación entre estos se lo denomina arista, puede contener datos de peso, valor o costo. Una de las ventajas de modelar un problema usando la teoría de grafos es que permite aplicar algoritmos matemáticos para encontrar la solución, como es el caso de encontrar la ruta más corta entre dos puntos conociendo todos los caminos posibles [19].

También se lo puede definir matemáticamente como $G = (V, E)$ [19].
Donde:
$V = v_1, v_2, \dots, v_n$, y representa un conjunto finito de nodos.
$E = e_1, e_2, \dots, e_m$, y representa un conjunto finito de aristas.

Cada arista mantiene una relación entre dos nodos, $e_1 = (v_1, v_2)$.Dependiendo si identifica que alguna relación mantiene un orden específico, se denomina grafo dirigido o no dirigido [20].

Se define como nodos vecinos o adyacentes a v_1, v_2, cuando existe al menos una arista e_1 que pueda ser presentada por ese par de nodos.

El orden de un grafo se establece dependiendo del número total de vértices en el conjunto V. El grado de un nodo se representa por el subconjunto de aristas vinculadas al nodo.

En la Figura 1.10., se puede ver una representación esquemática de un grafo que posee 5 nodos y 6 aristas. Es decir, es de orden 5 y, por ejemplo, el nodo 1 es de grado 3 debido a que está conectado a las aristas: *a, b, c*.

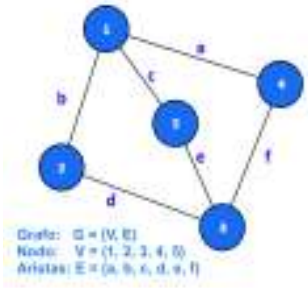

Figura 1.10. Grafo de 5 nodos con 6 aristas.

Un grafo puede contener aristas paralelas, esto sucede cuando están conectando a los mismos nodos. Dependiendo de las carteristas de las aristas se pueden formar un lazo entre esos dos nodos. En caso de que un grafo no posea aristas paralelas se lo denomina un grafo simple.

Dependiendo de la representación de información que se quiera extraer, se puede considerar únicamente ciertos vértices, a estos se les conocerán como un subgrafos y vienen representados matemáticamente como:

$$Grafo: G(V, E)$$

$$Subgrafo: G_1(V_y, E_y)$$

$$V_y \subset V$$

$$E_y \subset E$$

Es decir, que el subgrafo G_1, posee ciertos vértices (V) pertenecientes al grafo G, lo que implica que ciertos nodos son excluidos. Un ejemplo práctico del uso de subgrafos para este proyecto, es construir un grafo en base a las paradas de transporte público de cierta zona que solo incluyan los vértices que van en sentido norte sur, en casos donde un usuario consulte una ruta en este sentido. Y en base al grafo representado se podrá aplicar algoritmos que permitirán calcular la ruta más corta para llegar a un destino.

Grafo no dirigido

Se denomina grafo dirigido a un grafo en el cual todas sus aristas son bidireccionales, es decir si dos nodos $A\ y\ B$ están conectados dentro de un grafo no dirigido, se puede llegar desde A hasta B y regresar sin restricción.

Un grafo no dirigido matemáticamente está conformado por:

$$V \neq \emptyset$$

$$E \subseteq \{x \in P(V): |x| = 2\}$$

Donde $P(V)$es un conjunto de aristas y x representa un par no ordenado de nodos.

En un grafo dirigido el orden de los nodos de la arista no importa, por lo que:

$$\langle 1|2\rangle = \langle 2|1\rangle$$

Es decir, si el nodo 1 está conectado al nodo 2 implica que el nodo 2 está conectado al nodo 1. Su representación gráfica es con una línea como se puede ver en la Figura 1.11.

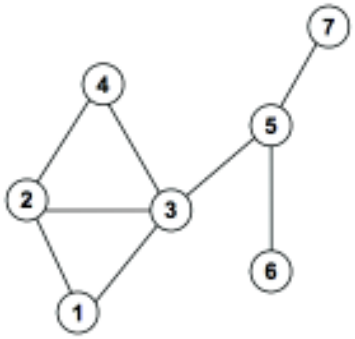

Figura 1.11. Grafo no dirigido *[21]*.

Al grafo de la Figura 1.11. se lo define como:

$$G = (V, E)$$

Donde:

$$V = \{1, 2, 3, 4, 5, 6, 7\}$$

$$E = \{\langle 1|2\rangle, \langle 1|3\rangle, \langle 2|1\rangle, \langle 2|3\rangle, \langle 2|4\rangle, \langle 3|1\rangle, \langle 3|2\rangle, \langle 3|4\rangle, \langle 3|5\rangle, \langle 4|2\rangle, \langle 4|3\rangle, \langle 5|3\rangle, \langle 5|6\rangle, \langle 5|7\rangle, \langle 6|5\rangle, \langle 7|5\rangle\}$$

En un grafo no dirigido el grado de cualquiera de sus nodos es igual al número de nodos adyacentes a éste. Como es el caso del nodo del nodo 3 en la Figura 1.11., que tiene un grado de 4 debido a que sus nodos adyacentes son $\{1, 2, 4, 5\}$

Grafo dirigido

Se denomina grafo dirigido a un grafo en el cual todas sus aristas conectan a los nodos solo en una dirección específica.

Un grafo dirigido o dígrafo matemáticamente está conformado por:

$$V \neq \emptyset$$

$$E \subseteq \{x \in P(V): |x| = 1\}$$

Donde $P(V)$ es un conjunto de aristas y x es un conjunto de pares ordenados.

En un grafo dirigido el orden importa y cada arista tiene un sentido único, por lo que:

$$\langle 1|2\rangle \neq \langle 2|1\rangle$$

Es decir, si el nodo 1 está conectado al nodo 2 esto no necesariamente implica que el nodo 2 está conectado al nodo 1. Su representación gráfica se la realiza mediante flechas indicando la dirección de la arista como se lo puede ver en la Figura 1.12.

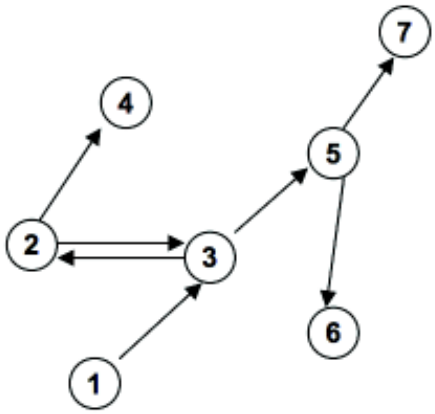

Figura 1.12. Grafo dirigido [21].

Al grafo de la Figura 1.12 se lo define como:

$$G = (V, E)$$

Donde:

$$V = \{1, 2, 3, 4, 5, 6, 7\}$$

$$E = \{\langle 1|3\rangle, \langle 2|3\rangle, \langle 2|4\rangle, \langle 3|2\rangle, \langle 3|5\rangle, \langle 5|6\rangle, \langle 5|7\rangle\}$$

Para los grafos dirigidos los nodos pueden tener grados interiores, que son el número de aristas que llegan al nodo, y grados exteriores, que son el total de aristas que parte desde ese punto. Como es el caso del nodo 2, que su grado interior es de 1 (debido al vértice que parte en 3 y termina en dos) y su grado exterior es de 2 (debido a que existen dos vértices que salen de 2 y terminan en 3 y 4 respectivamente).

Representación computacional de grafos

Existen varias formas de representar un grafo y dependiendo de la cantidad de datos, necesidades del algoritmo se puede seleccionar la más adecuada.

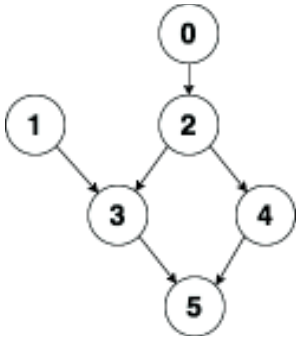

Figura 1.13. Grafo dirigido de orden 5 *[21]*.

Entre las formas más usadas se encuentran:

- **Lista de aristas:**

Un grafo se puede representar a través de una lista de aristas (E). En la que se tiene una lista de objetos que contiene la información de los nodos incidentes en cada vértice del grafo. En caso de que las aristas tengan un peso específico se añadirá un tercer valor.

Este tipo de representación ocupa poco espacio de memoria debido a que el espacio usado es igual al número total de aristas $|E|$, pero dificulta encontrar cuales son los nodos vecinos al nodo que se está analizando, debido a que se debe realizar una búsqueda por toda la lista para encontrar los vértices que contienen el nodo [22].

A continuación, en la Figura 1.14., se presenta un ejemplo de la representación de una lista de aristas para JavaScript del grafo de la Figura 1.13.:

```
var edgeList = [ [0, 2, 4],
                 [1, 3, 8],
                 [2, 3, 2],
                 [2, 4, 7],
                 [3, 5, 3],
                 [4, 5, 9]
];
```

Figura 1.14. Representación de una lista de aristas usando sintaxis JavaScript.

- **Matriz de adyacencia**

Un grafo también puede ser representado a través de una matriz de adyacencia. Esta es una matriz cuadrada de nodos, $|V| \times |V|$, que acepta solo valores de cero (0) y uno (1).

El par ordenado (i, j), representa dos nodos del grafo y toma el valor de uno (1) solo cuando existe una arista que los una.

En caso que se desee encontrar una arista específica solo se deberá usar el par ordenado (i, j) para determinar si existe o no.

Para pocos nodos esta representación facilita la búsqueda de nodos adyacentes al nodo de análisis debido a que solo se debe buscar en la fila correspondiente a este nodo.

En el caso de exista un gran volumen de nodos de un grafo disperso, muchos nodos y pocas aristas, la búsqueda de nodos adyacentes se dificulta debido a que se debe recorrer una fila de muchos ceros antes de encontrar una arista.

El espacio de memoria usado es igual a $|V|^2$ y dependiendo del problema podría resultar ineficiente debido que podría usar una gran cantidad de memoria para representar pocos nodos, como es el caso de los grafos dispersos.

A continuación, en la Figura 1.15, se presenta un ejemplo de la representación de una matriz de adyacencia para JavaScript del grafo de la Figura 1.13:

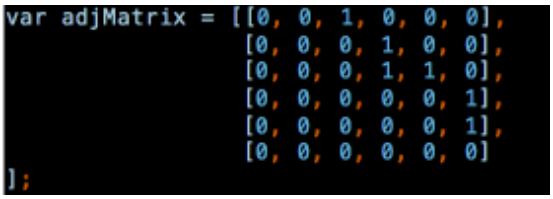

Figura 1.15. Representación de una matriz de adyacencia usando sintaxis JavaScript.

- **Lista de adyacencia**

Se tiene una lista del tamaño de todos los nodos del grafo, $|V|$, donde cada índice representa un nodo contendrá una lista con todos los nodos adyacentes. El orden de los nodos en la lista de adyacencia no debe cumplir ningún orden específico. Esta representación facilita la búsqueda de nodos adyacentes al nodo de análisis.

El tamaño de memoria ocupado en una lista de adyacencia para un grafo no dirigido es igual a dos veces la cantidad de los vértices $2|E|$, y para un grafo dirigido es de $|E|$.

A continuación, en la Figura 1.16., se presenta un ejemplo de la representación de una matriz de adyacencia para JavaScript del grafo de la Figura 1.13.:

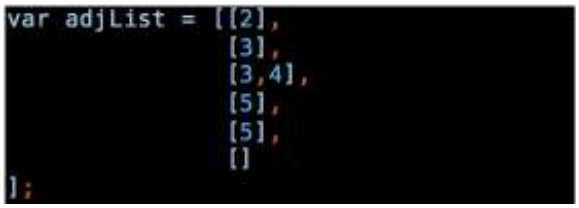

Figura 1.16. Representación de una matriz de adyacencia usando sintaxis JavaScript.

Aplicaciones de los grafos

En la actualidad la teoría de grafos es cada vez más usada, debido a su facilidad para representar un problema real en algo abstracto que se le puede aplicar una serie de algoritmos para encontrar: caminos más cortos, flujos máximos, búsquedas en profundidad, entre otros.

Gracias al auge de procesadores de mayor capacidad y velocidad se ha encontrado soluciones a problemas en diversas áreas como: logística, diseño de redes, genética, desarrollo de sensores, planificación de rutas para robots autónomos y muchos más.

Para el presente trabajo se plantea usar el ID de la parada del archivo GTFS como nombre del nodo y estará representando la ubicación de una parada en el mapa, los vértices representan el flujo de tráfico, además se contará con un peso de cada vértice que es la distancia en kilómetros entre paradas, con el objetivo de crear un grafo dirigido.

Mediante el algoritmo de Dijkstra, algoritmo para encontrar los caminos mínimos, se encontrará el camino más corto para llegar a un destino, estos datos serán enviados a un sistema de información al usuario, que basado en la información de los archivos GTFS, estándar de facto para representar datos de transporte público, representará gráficamente cuál es la ruta en bus es la más corta para llegar a ese punto desde un origen establecido.

Algoritmo

Un algoritmo es una secuencia de pasos preestablecidos que se encarga de transformar una entrada en una salida para cumplir una tarea específica.

Para que un algoritmo sea considerado valido, sus pasos deben cumplir las siguientes características [23]:

- Deben ser finitos: El objetivo de un algoritmo es que cumpla una tarea específica, si el número de pasos es infinito, entonces nunca completara dicha tarea.
- Deben estar definidos: Se debe definir claramente las instrucciones para todos los posibles casos. No debe existir ambigüedad.
- Debe ser efectivo: Tiene que ser capaz de resolver el problema para el que fue diseñado sin importar las veces que se lo realice.
- Debe tener al menos una entrada.
- Debe tener la capacidad de retornar uno o varios resultados.

Algoritmos de búsqueda para grafos

Una de las principales ventajas de representar un problema en forma de grafo es que podemos encontrar una solución mediante un algoritmo, pero para resolverlo se necesita un mecanismo que permita recorrer los nodos de manera sistemática y así encontrar el nodo o camino objetivo [24].

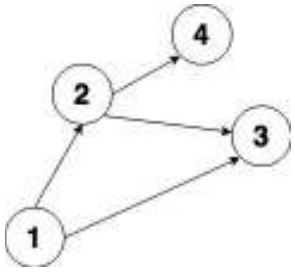

Figura 1.17. Representación de un grafo dirigido de grado 4.

Dado un grafo cualquiera, como el de la Figura 1.17., se usa dos tipos de búsqueda para recorrer los nodos:

- **Búsqueda en profundidad:** Consiste en ir visitando cada nodo una única vez y desde ese punto continuar el camino en profundidad hasta el último nodo en la rama. Se basa en un esquema LIFO (Last In First Out) [24].

  Para el grafo de la Figura 1.17., usando la búsqueda en profundidad se obtiene un árbol como en la Figura 1.18., en el cual se toma el primer nodo, se visita uno de sus nodos adyacente de acuerdo a un criterio pre-establecido (paso 1), desde ese punto se avanza hasta el siguiente nodo adyacente (paso 2).

Una vez que se ha llegado al final de la rama se regresa a la última bifurcación y se visita el siguiente nodo adyacente de ese punto (paso 3), esta operación se repite hasta que todos los nodos hayan sido visitados. Se descartan todos los vértices que conecten a nodos ya visitados.

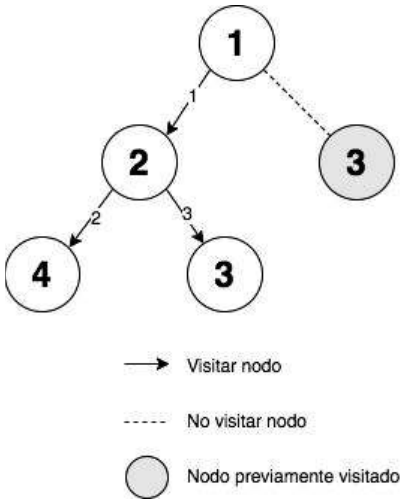

Figura 1.18. Búsqueda en profundidad de un grafo.

- **Búsqueda en amplitud:** Consiste en ir visitando todos los nodos adyacentes al nodo actual (nodo raíz). Luego se asigna a uno de estos nodos, bajo algún criterio, como el nuevo nodo raíz y se repite la operación, hasta llegar a visitar a todos los nodos del grafo. Se basa en el esquema FIFO (First In First Out) [24].

  Para el grafo de la Figura 1.17., usando la búsqueda en anchura se obtiene un árbol como en la Figura 1.19., en el cual se toma el primer nodo, se visita a todos sus nodos adyacentes (paso 1 y paso 2), luego bajo un criterio pre-establecido, se selecciona el nodo 2 como nuevo nodo raíz y se visita a todos sus nodos adyacente, esta operación se repite hasta que todos los nodos hayan sido visitados. Se descartan todos los vértices que conecten a nodos ya visitados.

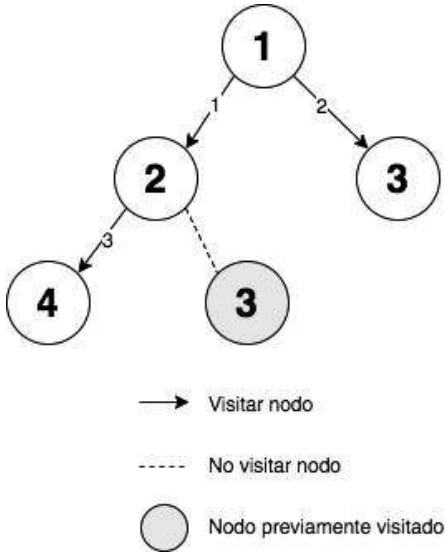

Figura 1.19. Búsqueda en anchura de un grafo.

Algoritmos de planificación de ruta

La tecnología usada para la creación de robot móviles autónomos incluye diversas técnicas que deben ser diseñadas e implementadas como, por ejemplo: técnicas para tomar decisiones, técnicas para planificar rutas, técnica para navegación automática, entre otras. Esto debe estar expresado en un lenguaje que los robots puedan procesar y actuar en base a los resultados obtenidos de los algoritmos implementados.

Una de las técnicas más importante es la planificación de rutas, debido a su amplio uso en diferentes áreas. En el campo de los robots autónomos uno de los más grandes retos es la planificación de rutas. Es decir, el robot tiene que ser capaz de encontrar la ruta más óptima para llegar a un destino específico desde su posición actual.

Esta optimización puede ser: encontrar el camino, con menos curvas, con menos obstáculos, etc., y dependerá de la necesidad del problema. Para esto se deberá representar en una unidad de procesamiento el ambiente donde se encuentra el robot.

Se puede seguir dos aproximaciones en esta representación: continua y discreta.
Los mapas discretos de los ambientes se los pueden representar usando grafos, donde un punto en el mapa se puede simbolizar un nodo y la existencia de un camino que permita que un robot se pueda mover entre dos nodos se representa con una arista, esta

información puede estar almacenada en una lista o una matriz. Los mapas continuos requieren la identificación total de los obstáculos, usualmente se usan polígonos para representarlos [3].

Una vez representado el ambiente donde se movilizará el robot se puede aplicar diferentes algoritmos para obtener un camino óptimo.

Problema de camino más corto

Uno de los problemas que se pueden resolver aplicando la teoría de grafos es encontrar el camino más corto entre dos puntos, formando la trayectoria de menor costo dentro de un grafo.

Matemáticamente se debe repensar el problema en un grafo dirigido:

$$G = (V, E) \qquad [23]$$

La función de pesos (w), tal que, cada arista sea subconjunto de los reales.

$$w : E \subset \{\Re\} \qquad [23]$$

Luego se definen diferentes caminos (p), que están formado por todos los nodos que se visitaron para llegar desde un origen (v_0) hasta el destino (v_k) y viene definido por:

$$p = \langle v_0, v_1, \dots, v_k \rangle \qquad [23]$$

El peso de un camino es la suma del peso de cada vértice que unen los nodos pertenecientes a ese camino (p):

$$w(p) = \sum_{i=1}^{k} w(i) = w(v_{i-1}, v_i) \qquad [23]$$

El camino más corto para llegar de un nodo inicial (v_0) a un final (v_k) se lo puede definir como el camino con el peso mínimo:

$$w_{min}(p) = \delta(v_0, v_k) \quad [23]$$

Donde:

$$\delta(v_0, v_k) = \begin{cases} min\{w(p) : v_0 \to v_k\} & si\ es\ que\ existe\ un\ camino\ entre\ v_0\ y\ v_k, caso\ contrario \\ \infty \end{cases}$$

[23]

Con estas expresiones podemos representar las rutas y paradas sistema de transporte público de la siguiente manera:

- Cada parada representa un nodo.
- Un tramo de una ruta de bus que une dos paradas representa una arista.
- El sentido de desplazamiento representa la dirección de una arista.
- La distancia de ese tramo representa el peso de cada arista para llegar a un siguiente nodo.

La representación del peso de una arista puede cambiar dependiendo de los datos disponibles y de lo que se desea optimizar, pudiendo representar métricas de tiempo, costo, tráfico, entre otras. Debido a que se trabajará con archivos GTFS estáticos, en este proyecto se optimizará en base a la distancia encontrando la ruta que recorra menos kilómetros.

Algoritmo de Dijkstra

Es uno de los primeros algoritmos aplicado en la planificación de rutas dentro de un grafo, fue definido en 1959. Consiste en encontrar el camino más corto desde un nodo inicial hasta un nodo final dentro de un grafo [3].

Para aplicar este algoritmo el grafo deberá cumplir con los siguientes requerimientos:

- Ser un grafo dirigido, el algoritmo necesita saber la dirección del movimiento que tiene que realizar para saltar de un nodo a otro.
- Sus aristas deben tener un peso definido, debido a que esto determinará si se escoge o no ese camino.
- Los pesos de las aristas no deberán ser negativos, debido a que el algoritmo se basa en la premisa de que cada arista añade un peso al camino escogido, es por ese motivo que una nueva arista no debería poner un valor menor al peso actual de un nodo ya visitado.
- Definir el origen, en base a ese dato se establece los pesos iníciales a los nodos, que irán actualizándose en base a los pesos de las aristas.

Funcionamiento

Los pasos definidos para el algoritmo de Dijkstra son los siguientes:

1. Todos los nodos empiezan con un peso infinito, al nodo inicial se le asigna el valor de 0.

2. Se actualizan los pesos de todos los nodos adyacentes al nodo actual con el valor de la arista más el peso acumulado.
3. Se realiza una comparación entre los pesos de los nodos adyacentes y se escoge el de menor valor.
4. Se compara con el valor actual del nodo, si es menor se actualiza el valor y se marca al nodo como visitado.
5. Se repite el paso dos, hasta llegar al nodo final. Si una arista está conectando un nodo visitado se la descarta.
6. Finalmente se obtiene una matriz de nodos con el peso mínimo para llegar a este nodo desde otro nodo vecino. Lo que permite determinar el camino más corto para llegar desde un nodo de origen hacia cualquier otro nodo dentro del grafo.

A continuación, se desarrollará un ejercicio, para verificar el funcionamiento del algoritmo de Dijkstra, se utilizará un grafo dirigido de grado 6 con pesos establecidos para cada arista como se lo puede ver en la Figura 1.20.

El problema a resolver en este ejercicio es encontrar el camino más corto del nodo A hasta el nodo F.

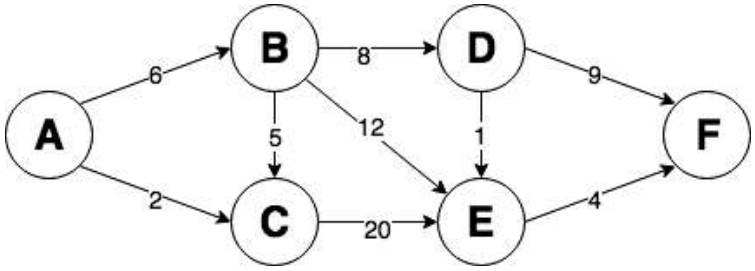

Figura 1.20. Grafo dirigido de grado 6.

Dado que todos los nodos empiezan con un peso de infinito, el primer paso del algoritmo es cambiar a 0 el peso del nodo A, debido a que es el nodo inicial, adicionalmente se lo marcará como visitado. Su etiqueta definitiva será (0, A).

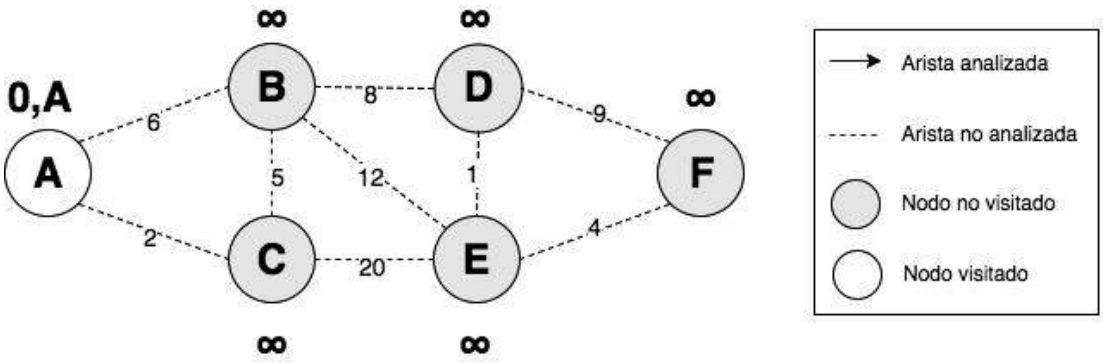

Figura 1.21. Primer paso del algoritmo de Dijkstra para un grafo dirigido de grado 6

El siguiente paso es analizar todos los nodos adyacentes al nodo A. Se actualiza el peso de los nodos usando el valor de las aristas, teniendo las siguientes etiquetas para los nodos B y C:

- Nodo B: (6, A)
- Nodo C: (2, A)

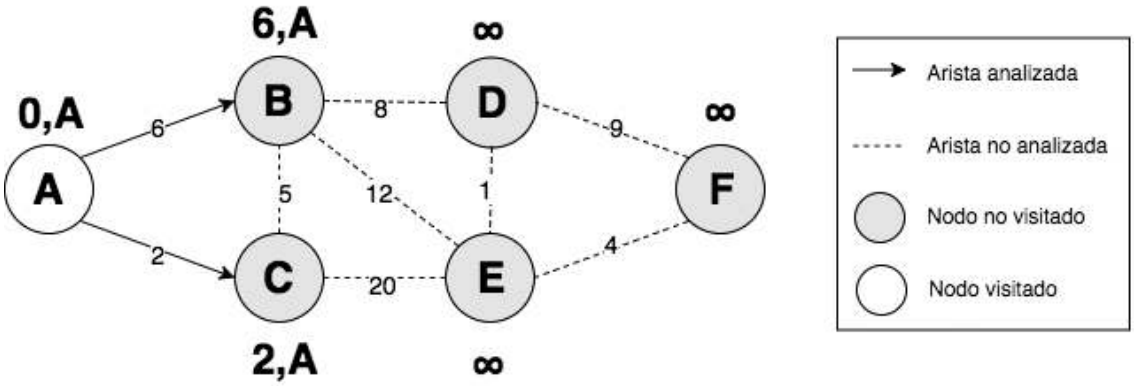

Figura 1.22. Segundo paso del algoritmo de Dijkstra para un grafo dirigido de grado 6

De todos los nodos no visitados se toma el que tenga el menor valor y se lo analiza. En este caso es el nodo C.

A continuación, marcamos al nodo C como visitado, analizamos todos los nodos adyacentes y se procede a etiquetar el siguiente nodo:

- Nodo E: (22, C)

Desde el nodo C hasta el nodo E hay un peso de 20, pero el nodo C ya tiene un peso de 2 desde A, por lo que el valor para llegar a E va a ser la suma del costo de la arista que lo une al nodo anterior y el peso acumulado del nodo anterior.

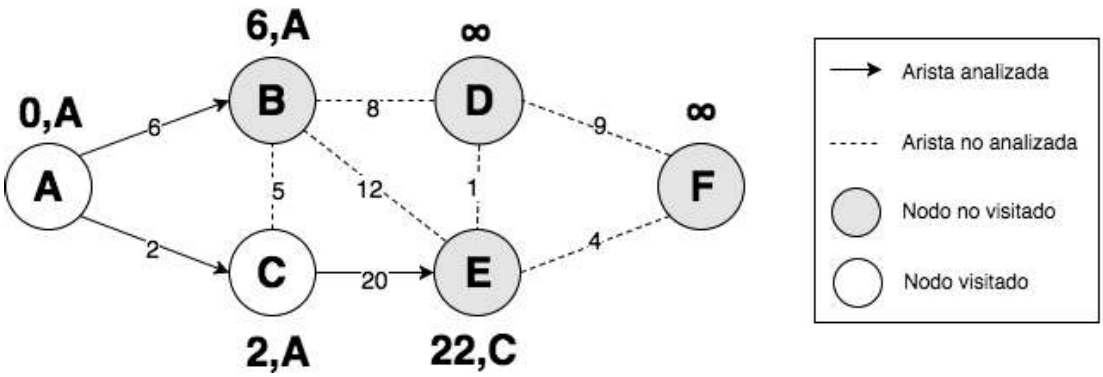

Figura 1.23. Tercer paso del algoritmo de Dijkstra para un grafo dirigido de grado 6

Se selecciona nuevamente el nodo de menor costo entre todos los nodos no visitados, en este caso el nodo B. Se lo marca como visitado, se analizan todos los nodos adyacentes a B y, se procede a etiquetar los siguientes nodos:

- Nodo D: (14, B)
- Nodo E: (18, B)

Debido a que se obtuvo un mejor costo para llegar al nodo E desde B, 4 puntos menos, se actualiza el valor de este nodo con la nueva etiqueta.

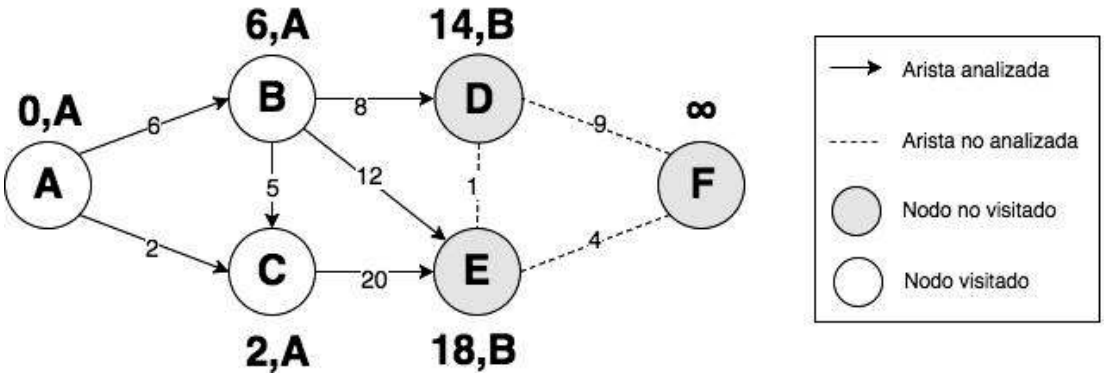

Figura 1.24. Cuarto paso del algoritmo de Dijkstra para un grafo dirigido de grado 6

En esta iteración el nodo con menor costo es el nodo D, se procede marcarlo como visitado y se actuliza las etiquetas de sus nodos adyacentes:

- Nodo E: (15, D)
- Nodo F: (23, D)

Se actualiza la etiqueta del nodo E debido a que desde el nodo D se obtiene un camino con menor costo al actual. Se seleciona el menor costo y se lo marca como visitado, que en esta iteración es el nodo E.

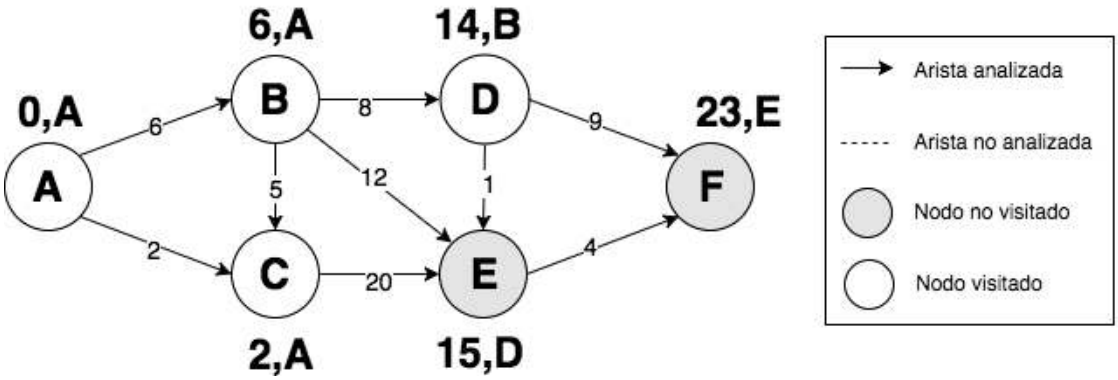

Figura 1.25. Quinto paso del algoritmo de Dijkstra para un grafo dirigido de grado 6

Se analiza las aristas que estén conectadas al nodo E, se actualiza la etiqueta del nodo adyacente, nodo F, debido a que se obtiene un menor costo desde E:

- Nodo F: (19, E)

Se visita el nodo con menor costo, y esta vez coincide con el nodo de destino. Se verifique que no queden nodos sin visitar. Debido a que todos los nodos han sido visitados se termina las iteraciones.

De esta manera se concluye que el camino más corto para llegar del nodo A al nodo F es:

$$A \rightarrow B \rightarrow D \rightarrow E \rightarrow F$$

Con un costo total de 19.

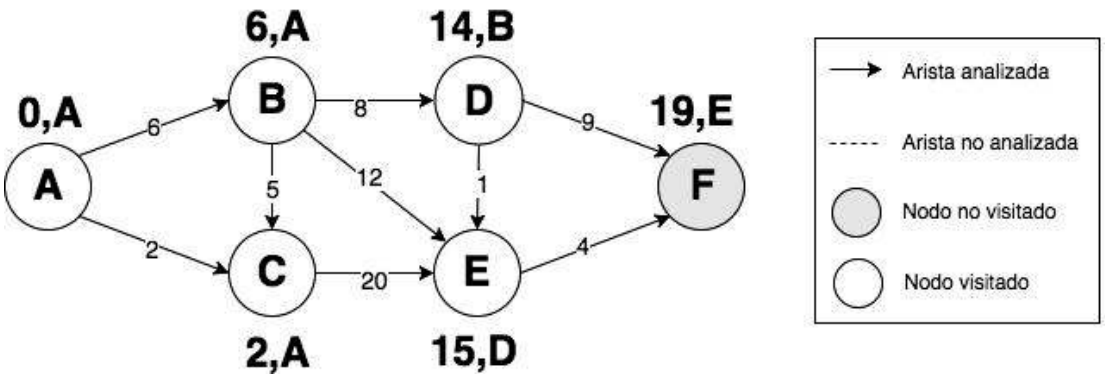

Figura 1.26. Paso final del algoritmo de Dijkstra para un grafo dirigido de grado 6

Aplicaciones

El uso de la información de la ruta más corta dentro de un grafo tiene diversas aplicaciones como:

- La planificación de rutas para robots móviles.
- Enrutamiento de paquetes de datos dentro de una red.
- Cálculo de rutas para sistemas de información de tráfico.
- Mecanismos de toma de decisiones para sistemas de inteligencia artificial.
- Sistemas GPS.
- Enrutamiento de aviones.
- Panificación de tráfico urbano.
- Sistemas de información al usuario de transporte público indicando la ruta más corta para llegar a un destino.

Este trabajo implementa el algoritmo de Dijkstra para encontrar el camino entre un grupo de nodos que representan las paradas del sistema de transporte público. Este algoritmo deberá estar disponible para comunicarse con el planificador de rutas, que procesará los archivos GTFS con el fin de obtener la información de entrada para el algoritmo; y, procesar la salida para entregar un mapa con la ruta óptima que debe seguir el usuario. Este servicio estará disponible en un servidor en la nube y se podrá acceder mediante llamadas REST.

Servicios REST

Es un protocolo que se usa para comunicar servicios web, está basada en una arquitectura cliente/servidor que usa mensajes http para comunicarse con un servicio de servidor en la

red, el nombre del servicio se lo especifica mediante una URI (UniformResourceIdentifier, ejemplo www.google.com/).

En la Figura 1.27. se puede apreciar cómo está distribuida esta arquitectura, en la cual un cliente REST hace una solicitud http a uno de los servicios del servidor REST, que recibe estos datos y dependiendo de la lógica podría almacenar o extraer información de base de datos, para finalizar la interacción el servidor emite una respuesta informando al cliente el estado del proceso.

Figura 1.27. Arquitectura REST

Para realizar una petición REST se debe usar los verbos definidos por el protocolo HTTP: [25]

- GET – Se lo usa principalmente para leer algún dato. El recurso puede responder con un archivo JSON o XML con la información solicitada más un código de respuesta.
- POST – Se lo usa principalmente para crear información. Se envía información al recurso en formato JSON o XML de lo que se desea almacenar, el servidor procesa esta información y responde con un código de respuesta.
- PUT – Se lo usa para actualizar información previamente generada con un POST.
- DELETE - Se lo usa para borrar información existente en el servidor.

Entre los códigos de respuestas más comunes se encuentra: [25]

- 200 – OK: La petición se realizó con éxito.
- 301 – Moved Permanently: La petición se realizó con éxito, pero el cliente deberá redireccionar a un nuevo servicio.
- 400 – BadRequest: La petición falló, el servicio no pudo procesar la petición
- 401 – Unauthorized: La petición requiere algún tipo de autenticación que no fue proporcionada al momento de realizar la llamada.
- 404 – Notfound: El servidor no pudo encontrar el recurso solicitado.

- 500 - Internal Server Error: Problemas internos del servidor, puede ser que el cliente no logre establecer conexión con el servidor.

2. METODOLOGÍA

En este capítulo se presentará la metodología usada para la construcción de un planificador de rutas para el usuario de transporte público. Se realizará una investigación aplicada, para lo cual primero se recolectará y procesará los datos de rutas y paradas de transporte público, posteriormente, se implementará un algoritmo de camino más corto que se comunicará a la interfaz gráfica del servicio de información al usuario. Para validar la utilidad y funcionalidad del sistema se realizará pruebas de usabilidad a potenciales usuarios de la aplicación.

El planificador de rutas usará el algoritmo de Dijkstra para calcular el camino más corto optimizando la ruta en base a la distancia total recorrida, para finalmente presentar la información necesaria para que un usuario pueda completar su viaje.

La ubicación de las paradas y rutas disponibles se obtendrán del sistema de mapas abiertos OpenStreetMap y se transformarán al formato GTFS para graficar una ruta en un mapa digital, adicionalmente el sistema entregará más información sobre el viaje y la cooperativa que presta el servicio.

Para validar los resultados del sistema de información se usará la rutas y paradas del sector Centro Norte del Distrito Metropolitano de Quito, que limita con la parroquia Mariscal Sucre al sur-este, la parroquia Belisario Quevedo al sur-oeste, la parroquia Rumipamba al nor-oeste, la parroquia Iñaquito al nor-este, como se puede ver en el mapa de la Figura 2.1.

Figura 2.1. Mapa de la distribución de Administraciones Zonal del Distrito Metropolitano de Quito *[26]* .

Debido al que el algoritmo implementado puede ser aplicado a cualquier grafo, solo se debe cambiar los datos de entrada para que pueda replicarse en todo el Distrito Metropolitano de Quito y en otras ciudades que presenten la misma problemática.

2.1 Creación de GTFS del sector centro norte

Actualmente no existe un repositorio libre de información de rutas y paradas de transporte público del Distrito Metropolitano de Quito, ni de ninguna ciudad de Ecuador.

El objetivo de esta sección es indicar los pasos necesarios para crear estos archivos a partir de datos de latitud y longitud.

Recopilación de información de rutas y paradas

La información de la geo-localización de paradas de buses y direcciones de los viajes disponibles de transporte público del Distrito Metropolitano de Quito se encuentra disponible de manera gratuita y libre en el sistema de mapas OpenStreetMap.

Se deberá seleccionar la zona que se desea analizar de la siguiente dirección: http://www.openstreetmap.org/export#map=#id_map.

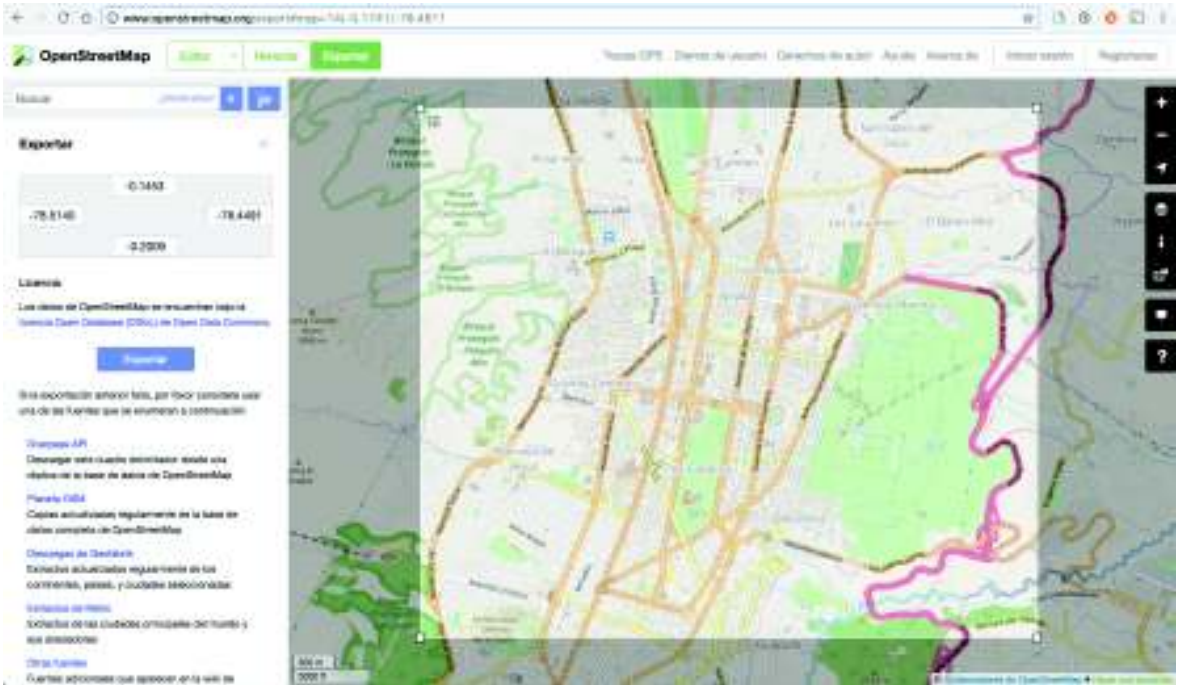

Figura 2.2. Pantalla de selección de selección del área que se dese exportar los datos del mapa.

Para el presente trabajo se tomará en cuenta únicamente los datos del sector centro norte de Distrito Metropolitano de Quito, como se puede visualizar en la Figura 2.2.

La información obtenida se encuentra en formato OSM y dentro de este archivo estarán los datos relacionados a la geo-localización de los diversos nodos que conforman el mapa. Los nodos pueden representar puntos de: intersecciones peatonales, señales de tránsito, paradas de buses, entre otros puntos referentes al sector.

Una vez identificados los nodos correspondientes a las paradas de buses se deberá extraer la latitud, longitud, ID y nombre de la estructura del archivo OSM para poder ingresarlos en el editor de GTFS.

En la Figura 2.3 se encuentra el formato de la información obtenida de OpenStreetMaps para una parada de buses.

```
<node id="269389839" lat="-0.150138" lon="-78.483536" version="13" timestamp="2016-
04-02T00:13:12Z" changeset="38240853" uid="1282514" user="giomaussi">
    <tag k="highway" v="bus_stop"/>
    <tag k="name" v="CAP. Rafael Ramos"/>
    <tag k="public_transport" v="stop_position"/>
</node>
```

Figura 2.3. Ejemplo de formato de datos para una parada obtenida de OpenStreetMaps.

Finalizada la obtención de las paradas, el siguiente paso es obtener las rutas disponibles. Es decir, como estas paradas se conectan entre sí para completar un viaje, esta información es importante porque permite modelar las aristas del grafo sobre el cual se ejecutará el algoritmo de Dijkstra.

Esta información también se encuentra disponible el sistema de mapas OpenStreetMap como relaciones, es decir, una lista ordenadas de nodos que en este caso representan las paradas y cruces de calles.

Actualmente existen 63 rutas de buses, que atraviesan el sector centro norte, la lista completa de rutas y URL de referencia en OpenStreetMap se la puede encontrar en el Anexo 1.

Se debe tener en cuenta que una ruta se compone de dos viajes, uno representa el tramo norte-sur y otro el tramo sur-norte, lo que da un total de 126 viajes a ser analizados. Las rutas en OpenStreetMap se encuentran asociadas únicamente al nodo de origen y final, como se puede ver en la Figura 2.4.

La relación con las paradas intermedias y el viaje se deberá de hacerlo de forma manual al momento de generar los GTFS.

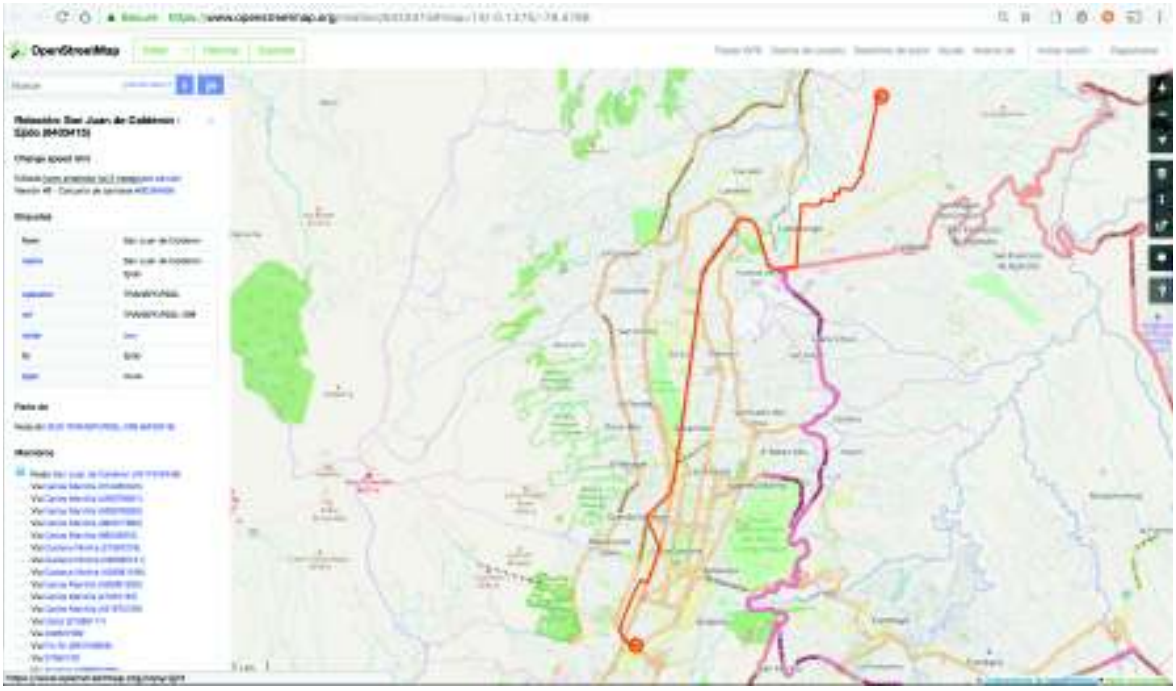

Figura 2.4. Viaje de San Juan de Calderón hacia el Ejido para la una de las rutas de la cooperativa Transporsel.

Uso del GTFS editor

La información de rutas y paradas recolectadas deben ser representadas en un formato que permita contener más información sobre la ruta y la compañía de transporte público que brinda el servicio, con el objetivo de información útil y precisa al usuario.

Esta información deberá contener datos sobre: el horario en el cual la ruta está disponible, frecuencia con la que opera, nombre de la agencia o cooperativa que brinda el servicio, nombre de la parada, precio del viaje, información sobre accesibilidad de la parada y del bus que hace la ruta, entre otros datos que estén disponibles y se consideren necesario para permitir que un usuario pueda movilizarse mejor.

Siguiendo el estándar de facto que se ha implementado en la industria del transporte público se decidió usar el formato GTFS (General Transit Feed Specification) para el presente trabajo.

Actualmente existen muchas herramientas libres que permiten crear y editar archivos GTFS. Para este trabajo se usará el Editor Conveyal que es una de las herramientas implementadas por el emprendimiento social de Mi Bus, al que se le solicitará credenciales de colaborador para la realización del presente trabajo. El editor que encuentra disponible en la siguiente dirección web: http://editor.mibusuio.com:8080/.

Al ingresar a la página web la primera pantalla que se despliega es la página de iniciar sesión, como se puede ven en la Figura 2.5. Luego de iniciar sesión correctamente se podrá acceder a la opción de crear una nueva ruta.

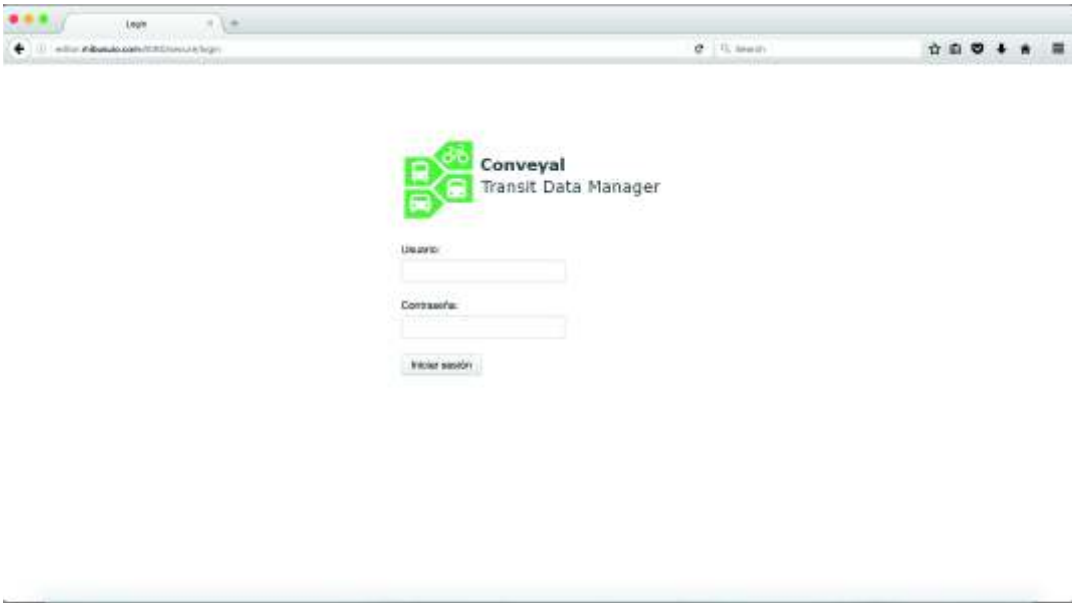

Figura 2.5. Pantalla de Login del editor de GTFS de Mi Bus.

El primer paso para crear una ruta en GTFS es llenar la **información básica** de la cooperativa que presta el servicio. Aquí se incluirá como campos obligatorios: el nombre completo, un nombre corto, el tipo de vehículo que ofrece el servicio (bus, metro, etc.) e información sobre la disponibilidad de acceso para silla de ruedas. Adicionalmente se puede incluir alguna descripción sobre la cooperativa, dirección web, establecer un color específico para graficar la ruta en el mapa y añadir comentarios, este último campo no se incluirá en los archivos GTFS y servirá de información durante la edición.

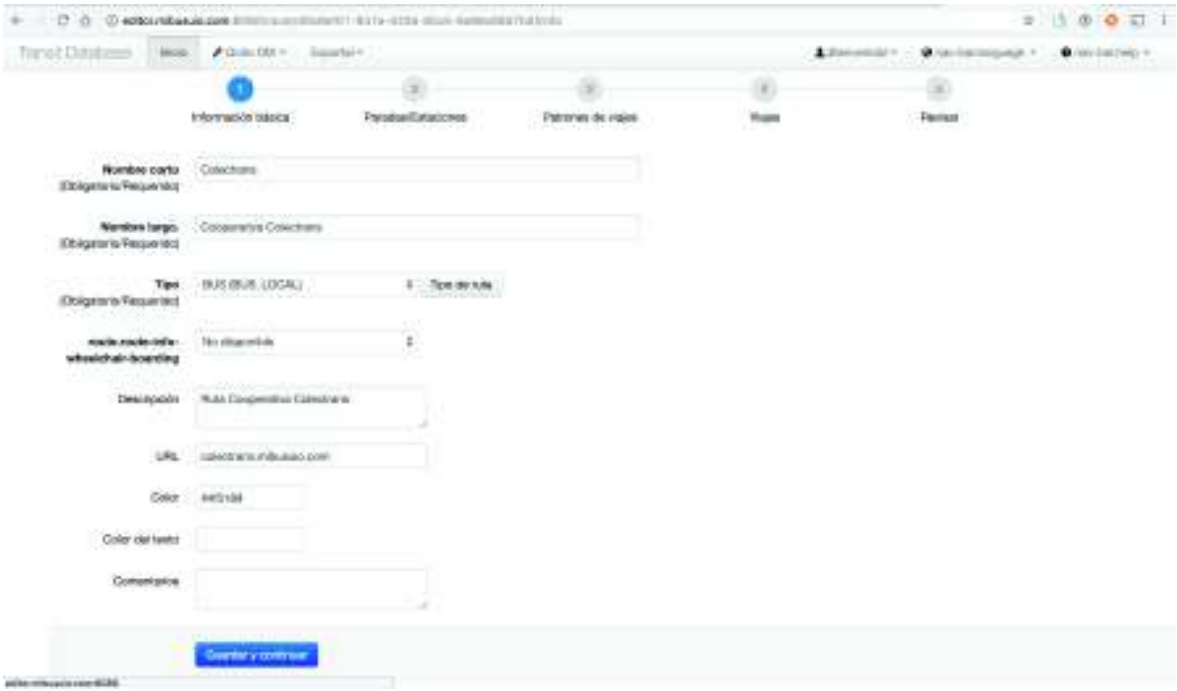

Figura 2.6. Pantalla de edición de información básica de la cooperativa de transporte público.

Esto se deberá hacer para cada una de las cooperativas que posea una ruta que esté disponible en el sector, según la información obtenida y documentada en el Anexo 1.

Luego se deben registrar las paradas en el sistema, ingresar la latitud y longitud de las paradas encontradas, en el caso en el que la información de OpenStreetMaps contenga un "**tagname"** también se deberá incluir el nombre en el editor de GTFS. Además, se deben incluir todos los datos adicionales disponibles sobre la parada.

Como se puede observar en la Figura 2.7., se puede incluir información adicional indicando si existe estacionamientos para bicicletas y/o vehículos privados, si la parada tiene acceso para silla de rutas, el tipo de embarque y desembarque de la parada (con frecuencia regular o si debe coordinar con la agencia), y finalmente si esta parada es una estación principal o una parada regular.

Al guardar una parada, esta quedará registrada en todo el mapa y podrá ser ocupada por cualquier ruta que sea editada. En esta pantalla también se incluye la opción de encontrar paradas duplicadas que resalta únicamente aquellas que están a menos de 100 metros de

distancia en la misma calle, esto permite realizar una depuración los datos más eficientemente y evitar duplicaciones.

Para facilitar el trazo de los viajes, se recomienda primero realizar el ingreso de todas las paradas y luego proceder al paso donde se define los **patrones de viaje**.

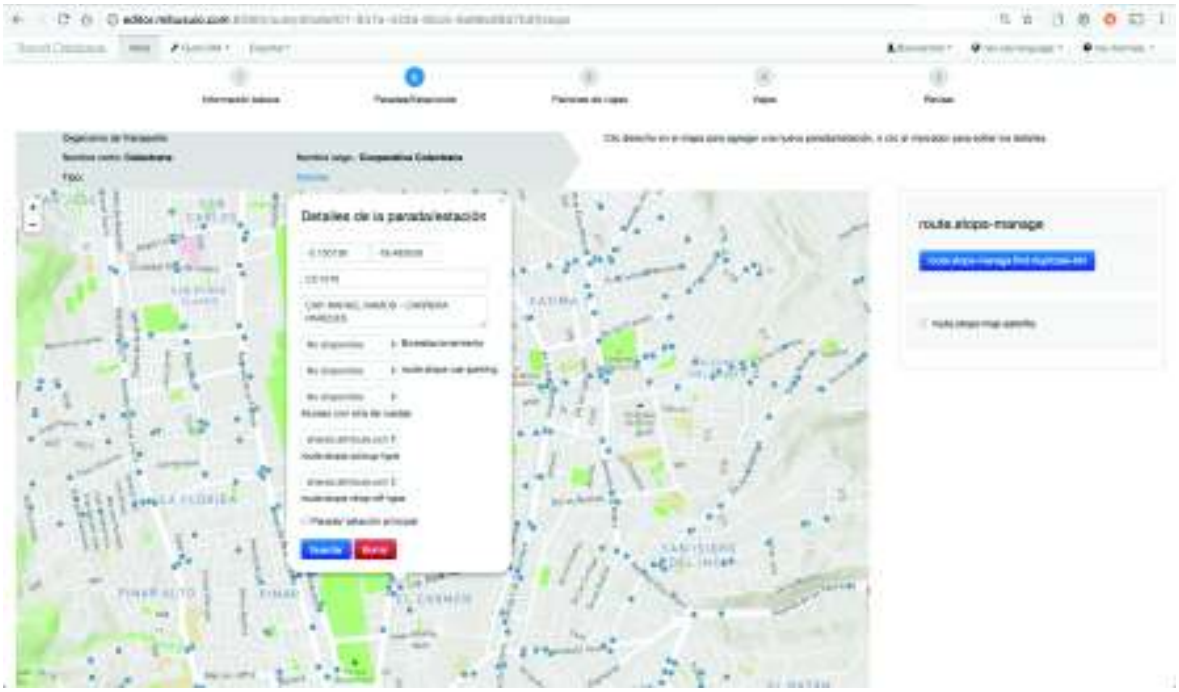

Figura 2.7. Pantalla de edición de paradas y estaciones.

Una vez definidas las paradas se debe proceder a crear los viajes para cada ruta de la cooperativa que se está ingresando, como se muestra en la Figura 2.8., en este paso se creará un nuevo viaje y se tendrá que unir las paradas en el orden deseado, eso asignará automáticamente un ID ascendente a la parada y la relacionará con el viaje.

Debido a que no se tiene información de tráfico, se asumirá una velocidad promedio de 25 km/h para usarlo como referente al momento de indicar la estimado de la periodicidad con la que pasa un bus de ese viaje por una parada.

Se deberá tener en cuenta que la ruta obtenida no seguirá la forma de las calles al momento de graficar en el mapa, debido a que esta información se deberá completar en el archivo “shapes.txt”.

Para generar este archivo se podrá usar herramientas como ArcGIS, especializadas en la creación de mapas, motivo por el cual está fuera del alcance de este trabajo.

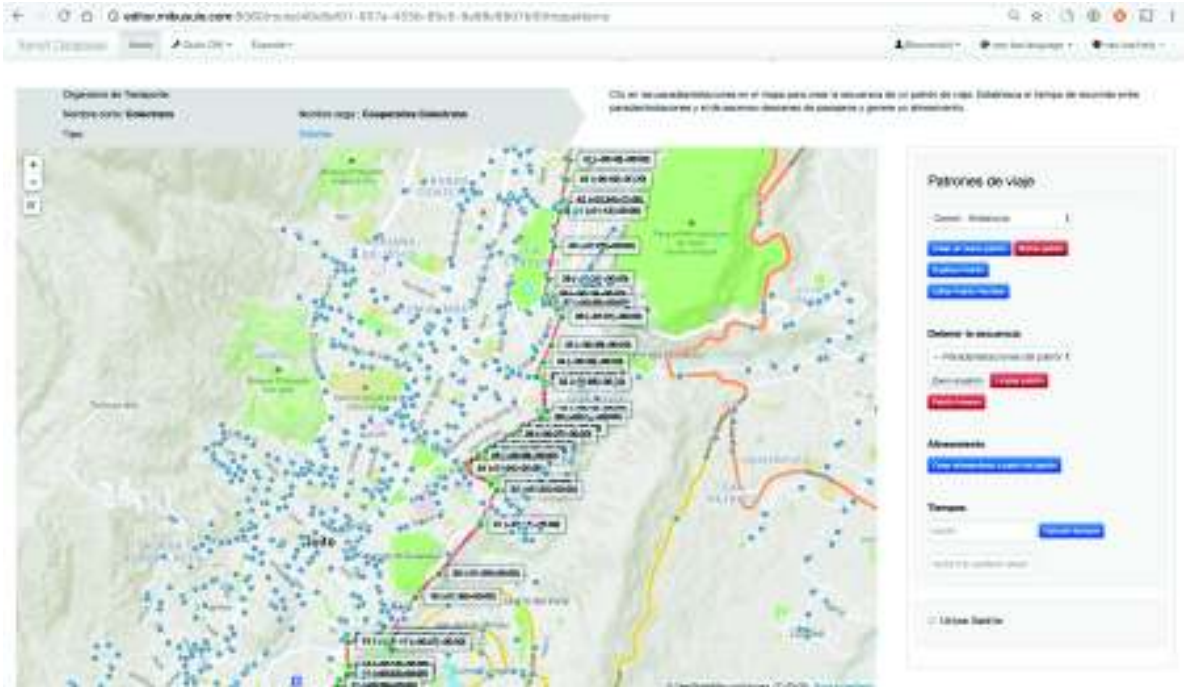

Figura 2.8. Pantalla de edición de patrones de viaje.

Finalizado el trazo de todos los viajes pertenecientes a una cooperativa se debe incluir información del horario en el que opera.

Las cooperativas de transporte público del Distrito Metropolitano de Quito deberían operar en base a una frecuencia definida, pero debido a varios factores como el tráfico, averías de los buses, paros, etc. es difícil poder cumplir con una frecuencia estática, la mejor manera de obtener esta información es mediante un GPS a bordo que entregue estos datos en línea; pero, debido a la falta de esta información es complicado predecir la programación de las rutas.

Para este trabajo, como la optimización de los viajes presentados al usuario es en base a la distancia, se establecerá una frecuencia estática de 10 minutos, que es frecuencia propuesta por el Sistema Integrado de Transporte Masivo [27].

El horario de inicio y fin de la ruta, así como el calendario de servicio juega un papel importante para el sistema de información al usuario, debido a que en base a esta

información se procesarán los datos del GTFS y se generará la información de rutas disponibles con la que se construirá el grafo. Esta información puede ser ingresada en el paso 4 del editor como se muestra en la Figura 2.9.

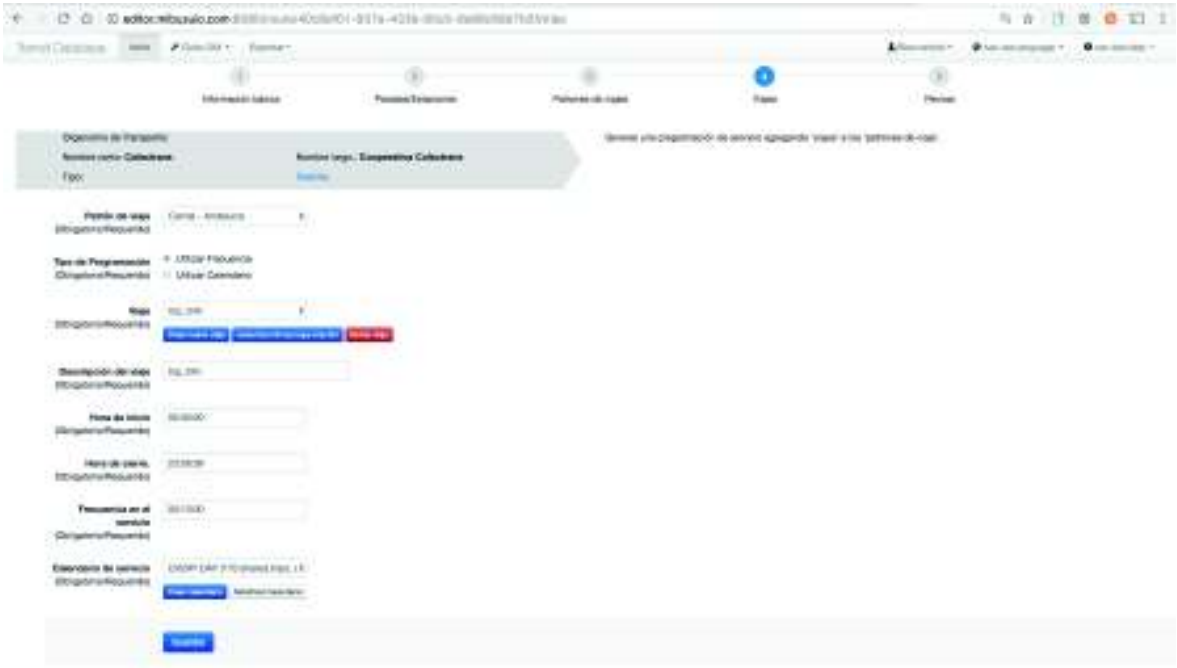

Figura 2.9. Pantalla de edición de horarios de los viajes.

Finalizada la digitalización de la información, se procede a guardar los datos y pasará a una etapa de revisión, en la cual se deberá realizar una verificación para garantizar que todos los datos estén ingresados correctamente, se pueden usar diversos softwares que faciliten el proceso y garantice la calidad de los GTFS, principalmente se debe asegurar el cumplimiento de los estándares establecidos.

Generación y validación de archivos GTFS

Una vez que se finalizó el ingreso de todos los datos al editor de GTFS, se inicia una etapa de revisión, en la que se debe verificar que todos los datos estén ingresados correctamente y se cumpla con el estándar establecido para cada uno de los archivos.

El editor ofrece un panel que refleja el estado de cada una de las rutas, como se puede ver en la Figura 2.10., antes de realizar la exportación de la información, se debe garantizar que el estado de las rutas requeridas este en “APRROVED”.

El momento de seleccionar exportar se debe poner una fecha de validez de los archivos GTFS, esto es debido a que cada cierto tiempo es recomendable hacer un mantenimiento a la información incluyendo los nuevos cambios que ocurren en la ciudad. Para este proyecto se generan GTFS con una validez 6 meses.

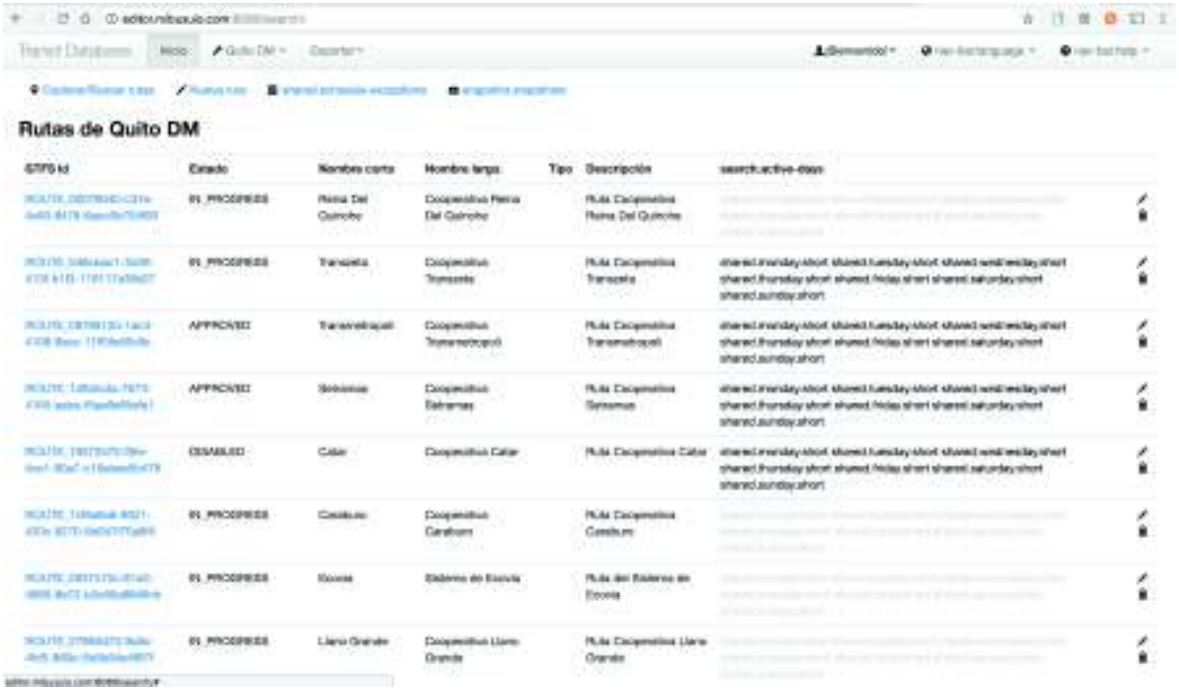

Figura 2.10. Lista de rutas del Distrito Metropolitano de Quito editadas en el GTFS editor.

Finalizada la exportación, se generará en un fichero comprimido en formato .zip con toda la información en organizada en varios archivos .txt, manteniendo la estructura del GTFS.

Para validar que cumple con todas las normas establecidas por el formato GTFS, se usará **FeedValidator**, que es parte de **TransitFeed**, una herramienta de consola desarrolla en Python por Google que permite leer y validar GTFS [5].

Para realizar esta validación es indispensable tener instalada la versión de Python mayor a 2.4 en la máquina local, esto también incorpora **easy_install** que es un módulo que permite descargar, instalar, construir y manejar paquetes de Python.

Para comprobar la versión de Python con la que se trabaja se debe ejecutar este comando en la consola:

```
$ python --version
```

Verificada la versión de Python en la máquina donde se va a realizar la validación, se procede a ejecutar el siguiente comando para instalar **TransitFeed**:

```
$ easy install transitfeed
```

Después de instalar TransitFeed, se debe ingresar al directorio donde se encuentra el archivo GTFS previamente generado y ejecutar el siguiente comando:

```
$ feedvalidator.py <feed file>
```

Esto generará, en el mismo directorio, un reporte con datos sobre el total de rutas y viajes, la cantidad de paradas, la fecha de valides del GTFS y también una lista de los errores y advertencias encontrados, como se indica en la Figura 2.11.

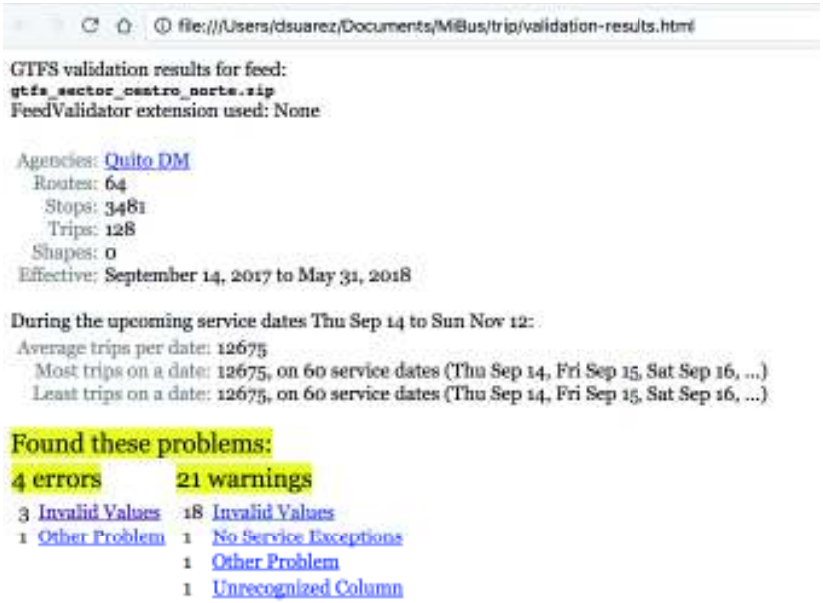

Figura 2.11. Reporte de verificación del archivo gtfs_sector_centro_norte.zip

Finalizada la corrección de los problemas y advertencias se dispone de un GTFS que puede ser procesado por sistemas estándares de información de rutas y paradas para el usuario de transporte público, actualmente existen varios sistemas que lo utilizan como: Google Maps, Moovit, Allytransport, Mi bus UIO, entre otros.

Para el presente proyecto se usará la base de código del emprendimiento social de Mi Bus UIO para realizar la lectura de GTFS y para dibujar el mapa de las rutas. Este sistema estará conectado vía servicios REST a la aplicación donde se implementa el algoritmo de Dijkstra, que calculará y procesará datos para encontrar el camino más corto para llegar a un destino.

2.2 Implementación de un servicio REST para calcular el camino más corto

El problema más común durante el trabajo con grafos es: poder encontrar el camino más corto entre dos nodos, es decir, buscar la combinación de aristas para llegar a un punto específico del grafo desde un origen establecido, en el que la suma total de sus pesos sea siempre el mínimo posible.

Si queremos usar los beneficios de un sistema computacional para realizar los cálculos necesarios para resolver el problema de encontrar el camino más corto desde un punto A de la ciudad hasta un punto B usando transporte público, primero debemos modelar las rutas y paradas en un grafo, posteriormente usar un algoritmo que permita que una computadora entregue el resultado en el menor tiempo posible y finalmente presentar este resultado al usuario.

El lenguaje usado para representar este problema es JAVA 8 y los datos GTFS obtenidos serán el valor de entrada al sistema de información al usuario, que los procesará y luego realizará una llamada REST mediante un POST al servicio web donde se encuentra implementado el algoritmo de Dijkstra, este servicio responderá un 200 OK e incluirá la información del camino más corto, en el caso de que las rutas existan y los valores de entrada estén correctos.

A continuación, se explicará el proceso para la creación de un grafo en base a la información de rutas y paradas obtenidas, la implementación del algoritmo de Dijkstra para encontrar este camino y la implementación de un servidor REST para realizar la consulta.

Representación de las paradas y rutas en un grafo

El primer paso para crear un planificador de rutas para usuarios de transporte público es procesar la información con la que se está llamando al servicio y construir una representación digital de las paradas y rutas disponible para los usuarios.

La entrada del sistema del sistema se la indica en la Figura 2.12.

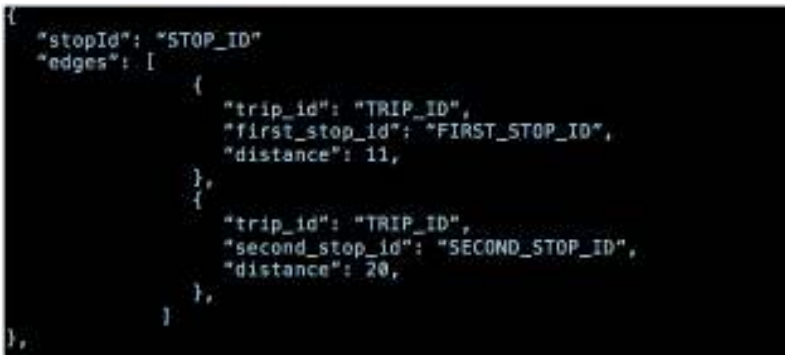

Figura 2.12. Entrada de datos para el sistema de planificación de rutas usando Dijkstra

Los datos que lleguen al servicio de planificador de rutas estarán en formato JSON y cada objeto representará una parada que conforma uno de los posibles viajes que unen los puntos de origen y destino que el usuario escogió.

Estos objetos poseen dos atributos: el número de identificación de la parada de bus y un arreglo que contiene información de la siguiente parada que pertenece a los viajes que se está analizando para cierta búsqueda, en el que se especifica la distancia, el ID de la parada próxima y el ID del viaje al que pertenece.

El procesamiento de GTFS a JSON lo realiza el sistema de información al usuario, para este caso se usará como base el proyecto Open Source del emprendimiento social de Mi Bus y se modificará para que realice llamadas al servicio del algoritmo de Dijkstra implementado en el presente trabajo.

Se usará el paradigma de programación orientada a objetos para realizar la representación del grafo y el cálculo del algoritmo. El primer paso es crear el objeto que representa la arista, para lo que se construye la clase ***Edge*** con los atributos: primer nodo (firstNode),

segundo nodo (secondNode) y costo (cost). Esta clase tendrá un constructor en la que recibirá los siguientes datos:

- ID de la primera parada como una cadena de caracteres
- ID de la segunda parada como una cadena de caracteres
- Distancia que las separa en metros como un entero.

Adicionalmente al constructor, se exponen 3 métodos públicos para acceder a cada uno de estos atributos, como se puede observar en el diagrama UML (Lenguaje Unificado de Modelado) en la Figura 2.13.

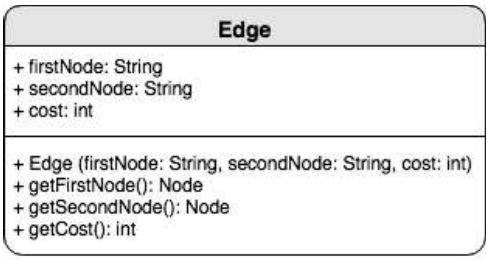

Figura 2.13. Representación de la clase **Edge** del servicio de planificación de rutas.

El nodo es otro objeto necesario para representar un grafo, para lo que se construye la clase ***Node***, como se puede ver en el diagrama UML de la Figura 2.14. y contiene los siguientes atributos:

- Identificador de la parada ***(ID)***.
- Costo total para llegar a este nodo ***(totalCost)***.
- Nodo anterior visitado ***(previousNode)***.
- Estado que indique si el nodo fue visitado ***(visitedNode)***.
- Aristas que tienen de origen este nodo ***(edges)***.

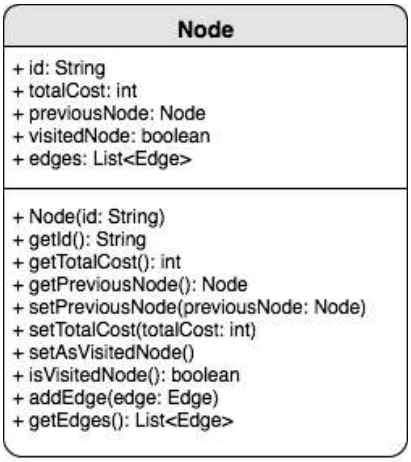

Figura 2.14. Representación de la clase **Node** del servicio de planificación de rutas.

Esta clase tiene un constructor que le permite crear un objeto ***Node*** solo con un identificador y pondrá valores por defecto para los demás atributos:

- **Costo total:** Será el máximo valor entero posible en JAVA8, que es 2147483647.
- **Estado de nodo visitado:** Se pondrá el valor inicial de falso.
- **Lista de aristas:** Será una lista de ***Edges*** vacía.
- **Nodo anterior visitado:** Tendrá un valor inicial de nulo.

Se dispone de varios métodos que le permitirán al objeto cambiar estos estados durante la ejecución del programa, tales como:

- **addEgdes():** Se usan para construir el grafo y permite añadir una arista a la propiedad ***edges*** del nodo.
- **setPreviousNode:** Se usa para registrar cuál es el nodo anterior de la trayectoria que estableció el costo del nodo actual.
- **setTotalCost:** Se usa para cambiar el costo total que se requiere para llegar a este nodo desde el nodo origen, se deberá analizar su valor en cada iteración del algoritmo de Dijkstra mientras el nodo no haya sido visitado.
- **setAsVisitedNode:** Se lo usa durante la ejecución del algoritmo de Dijkstra y permite marcar el nodo como visitado, cuando se ha encontrado el menor costo para llegar a este punto, cuando el estado pasa a ***verdadero*** se marcan los atributos ***previousNode*** y ***totalCost*** como valores definitivos y se los descarta del análisis.

También expone 5 métodos públicos para extraer la información de cada uno de los atributos que se usarán durante la ejecución del algoritmo para encontrar el camino más corto y para presentar la información al usuario.

Finalmente se necesita un objeto que represente el grafo y sobre el cual se pueda aplicar el algoritmo de Dijkstra. Para eso se construye la clase ***Graph*** que posee un único atributo que es la lista de nodos, como se puede ver en el diagrama UML de la Figura 2.15.

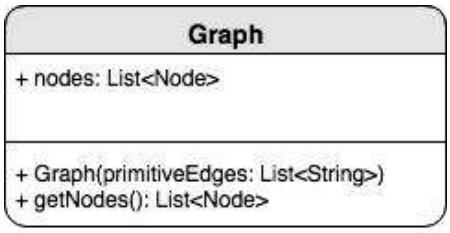

Figura 2.15. Representación de la clase ***Graph*** del servicio de planificación de rutas.

Para construir un nuevo grafo se necesitan tener una lista de cadena de caracteres que cumpla con el formato de la Figura 2.16.

```
List<String> primitiveEdges = Arrays.asList(
        "ID_STOP_1::ID_STOP_2::200",
        "ID_STOP_1::ID_STOP_3::800",
        "ID_STOP_2::ID_STOP_4::900",
        "ID_STOP_3::ID_STOP_5::900"
);
```

Figura 2.16. Formato de entrada de datos para para la clase Graph.

Este formato se construirá en la capa de servicios usando la información JSON recibida en la llamada POST y será enviada al objeto Dijkstra para que ejecute el algoritmo y responda con la información -camino más corto-. Esta respuesta será enviada al sistema de información al usuario para combinarla con los datos adicionales que se encuentran en los archivos GTFS; y, dibujar en un mapa digital la ruta más corta para el origen y destino que un usuario solicitó.

Debido a que el algoritmo devuelve el peso de cada uno de los nodos y el nodo anterior para obtener ese peso, el sistema de información al usuario puede priorizar más de una opción y presentarle al usuario otras alternativas de viajes para llegar al destino.

La lógica que se sigue para la creación del grafo se encuentra documentada en el diagrama de flujo de la Figura 2.17.

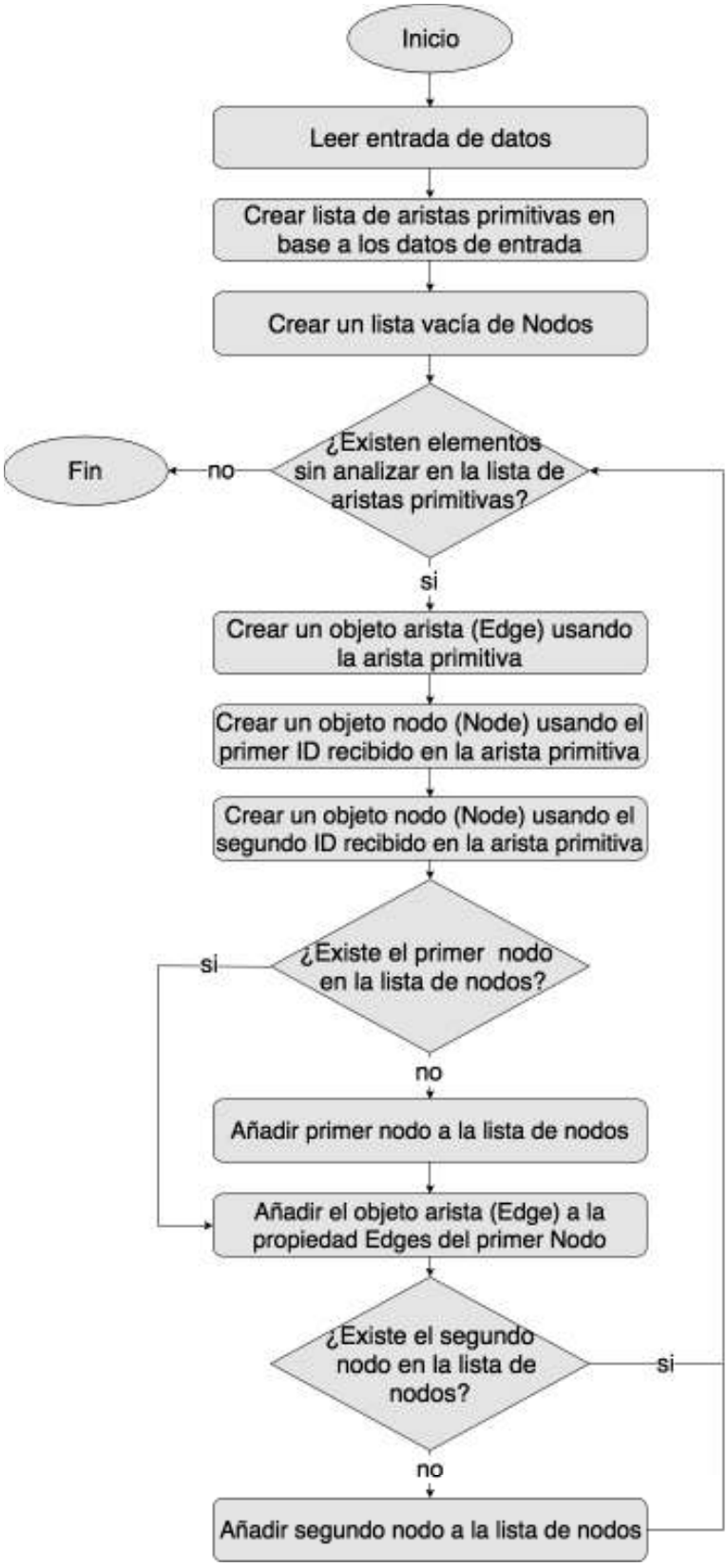

Figura 2.17. Diagrama de flujo de la construcción de un grafo en base a los datos de las aristas.

Se decidió usar la representación de lista de adyacencia debido a las ventajas que presenta respecto al espacio de memoria usado y facilidad para buscar los nodos adyacentes, estos factores habilitan la opción de preguntar al mismo nodo analizado cuáles serían los siguientes nodos a los que se puede llegar y el costo de dar ese paso, sin tener que iterar por toda una lista o matriz.

Al final se obtiene un objeto ***Graph*** que contiene una lista de todos nodos ***(Node1, Node2, etc.)*** en la cual se encuentra la información sobre cada ***arista (edge1, edge2, etc.)***, como se puede observar en la Figura 2.18.

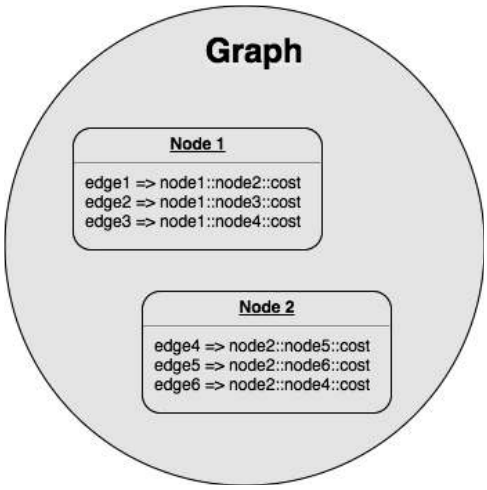

Figura 2.18. Estructura del modelamiento de un grafo en objetos usando la representación de lista de adyacencia.

Implementación del algoritmo de Dijkstra

Dado un grafo direccionado con pesos positivos establecidos para cada una de sus aristas se plantea implementar el algoritmo de Dijkstra para encontrar el camino más corto desde un nodo de origen hacia otro nodo dentro del grafo.

El algoritmo de Dijkstra usa el principio de reclusión, en cada iteración se toma la mejor la solución sin tener en cuenta los resultados posteriores, en el caso que se encuentre un mejor resultado en una próxima iteración el resultado anterior se remplaza. Entregando así al final de la iteración el cálculo de los pesos mínimos de cada nodo desde un origen. En el caso de que no exista un camino posible el peso será de infinito y al nodo inicial se le asigna un peso de 0.

En base al pseudo-código presentado en la Figura 2.19. se hará la implementación del algoritmo de Dijkstra usando el lenguaje de programación Java versión 8.

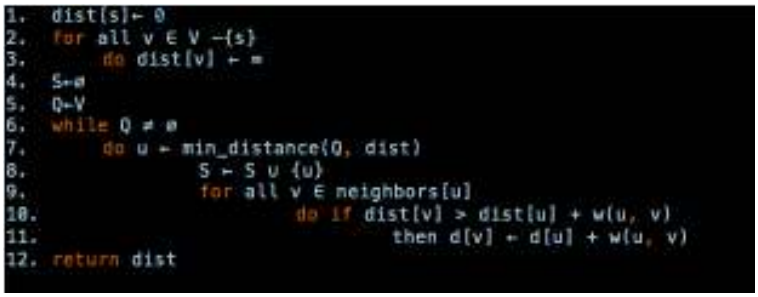

Figura 2.19. Pseudo-código del algoritmo de Dijkstra.

Cabe señalar que el pseudocódigo es un lenguaje simplificado entre el programador y la máquina, que se utiliza en programación para describir un algoritmo y facilitar el paso del programa a un lenguaje de programación. Las características principales del pseudocódigo son:

- Independencia: El pseudocódigo es independiente del lenguaje de programación que se implemente.
- Flexibilidad: El pseudocódigo permite lenguaje natural y funciones básicas de ejecución de un sistema. Los pseudocódigos pueden ser ejecutados en ordenadores.

Para lo que se necesitará previamente construir una representación de grafo (revisar sección 2.2.1), usando la información de rutas y paradas obtenidas del procesador de GTFS. Se debe tener en cuenta que este grafo deberá ser dirigido y debe tener pesos positivos.

El sistema necesitará: recibir el origen desde donde se desea calcular las distancias más cortas hacia cada nodo del grafo. Para realizar eso se construye la clase Dijkstra, como se puede ver en el diagrama UML de la Figura 2.20. Esta clase tiene un constructor que necesita de entrada un objeto ***Graph***.

Contiene los atributos:

- **visitedNodes:** Es una lista de nodos que ya fueron asignados el camino más corto y por lo tanto se debe marcar como visitados para excluirlos del análisis.
- **nodes:** Es una lista de todos los nodos que se obtiene del grafo que es enviando al momento de construir este objeto.

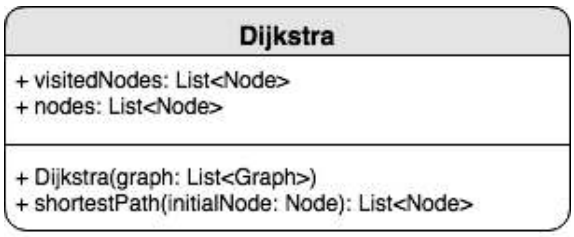

Figura 2.20. Representación de la clase ***Dijkstra*** del servicio de planificación de rutas.

Una vez que se ha construido el objeto, se puede usar el método ***shortestPath***, donde se implementa toda la lógica que para encontrar el camino más corto para cada nodo del grafo desde el nodo inicial. El siguiente paso es construir un objeto ***Graph*** y pasarlo al objeto ***Dijkstra***, una vez que se obtenga el grafo se procede a marcar el nodo inicial (s) asignándole un peso de cero. El grupo restante de nodos $(V - \{s\})$ vendrán asignados con un costo infinito desde el momento que se creó el grafo.

Se usará una lista (S) de elementos para almacenar todos los nodos visitados y en otra lista (Q) estarán todos los nodos no visitados del grafo.

Mientras existan nodos no visitados, se analizará la lista (Q) para encontrar el camino más corto para los nodos no visitados.

De la lista (Q) se obtiene el nodo de menor costo, para el primer paso es el nodo inicial, se lo marca como visitado y se analiza sus nodos vecinos. Si el costo actual del nodo vecino es mayor al nuevo costo que viene dado por el valor del vértice se actualiza estos valores.

Para el primer paso el costo actual de los nodos vecinos es infinito, por lo tanto, se actualizan todos los nodos con el valor del peso de los vértices que los unen al nodo inicial. Este proceso se lo realiza hasta que ya no existan más nodos en la lista (Q), es decir que todos los nodos hayan sido visitados.

Al final se entregará una lista de todos los nodos, paradas de buses, con el menor coste para llegar a ese punto desde un nodo previo que se conecta con el nodo inicial.

Esta lista de nodos se convertirá en formato JSON y se incluirá en el cuerpo de la respuesta que entregue el servicio Dijkstra cuando reciba una petición POST.

El sistema de información al usuario realizará esta llamada; que le permitirá, en base a: los costos de las paradas, la secuencia de paradas obtenida y los GTFS**,** dibujar en el mapa digital las rutas de buses que un usuario puede tomar para llegar a su destino.

Dado que se entrega una lista con los caminos más cortos de todos los nodos, el procesador podrá presentar al usuario más de una opción de ruta para llegar a su destino y datos de horarios e información sobre costos del viaje.

En la Figura 2.21. se representa el diagrama de flujo usado para implementar el algoritmo.

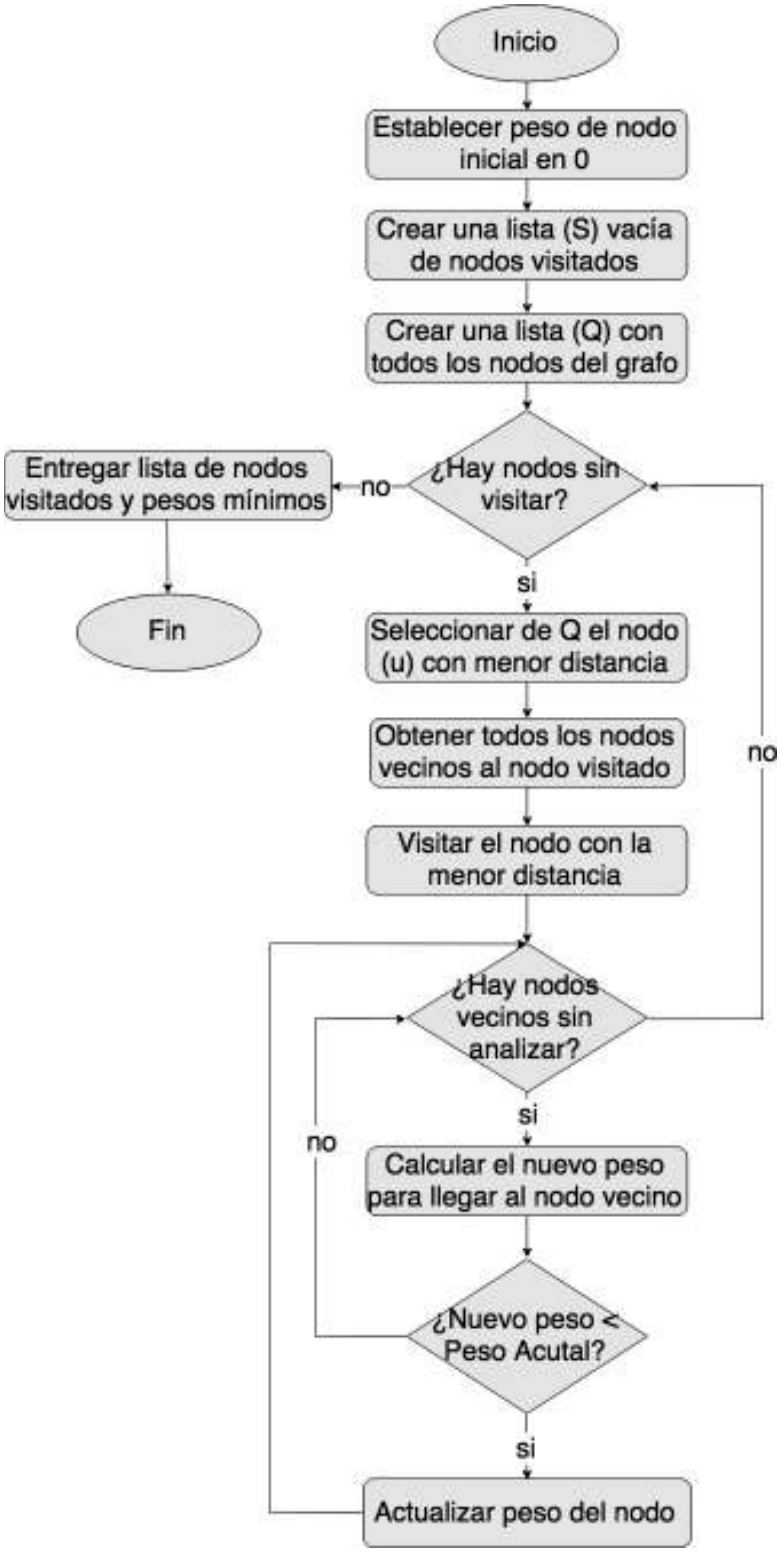

Figura 2.21. Diagrama del flujo del algoritmo de Dijkstra.

2.3 Cliente REST

Finalizada la implementación del algoritmo, este sistema debe tener la capacidad de comunicarse con otros servicios.

Para el caso del presente proyecto este sistema expondrá un servicio REST para recibir los datos de las aristas del grafo sobre el cual se quiere realizar el análisis.

Si cumple con el formato establecido y existe un camino posible devuelve una respuesta 200 OK que incluirá en el cuerpo de la respuesta un objeto JSON con: el ID de cada parada, el menor costo para llegar desde el origen y el ID de la parada anterior en el viaje.

Esto permitirá que el sistema de información al usuario, que tiene acceso a los datos de rutas y parada del sistema de transporte público del sector norte del Distrito Metropolitano de Quito en formato GTFS, pueda procesarlos y realizar una llamada http al sistema de planificación de rutas para obtener los pesos que tienen cada una de las paradas y en base a esta información graficar en un mapa digital la ruta más corta para llegar a un destino.

Se construirá la capa **Controlador** para manejar las llamadas REST y proveer acceso web a la aplicación, adicionalmente a esto se tendrá una capa de **Servicio** que permitirá manipular la entrada de datos para construir el Grafo y llamar a la implementación de **Dijkstra** para realizar el cálculo de la ruta más corta, luego este **Servicio** recibirá la repuesta, armar el objeto JSON y pasarle al **Controlador** para que pueda entregar la respuesta a la llamada, como se puede ver en la Figura 2.22.

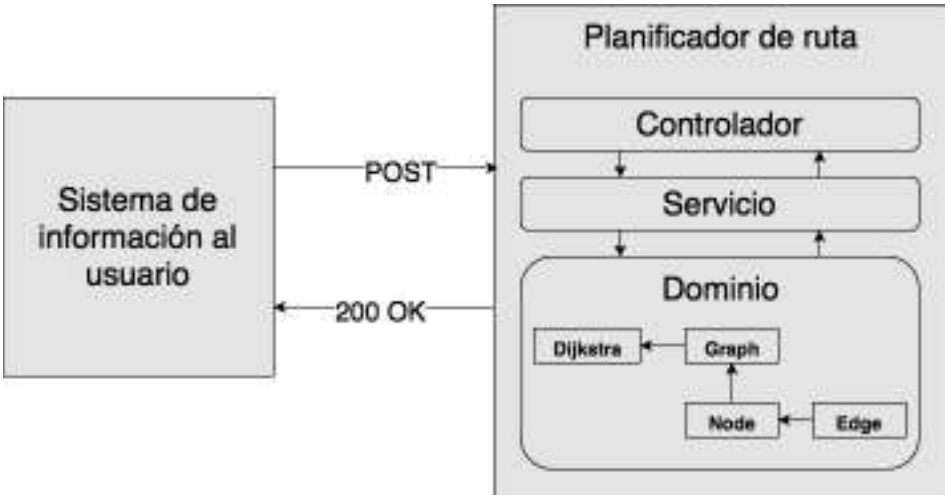

Figura 2.22. Arquitectura del servicio REST de planificación de rutas

2.4 Pruebas del planificador de rutas

Para verificar el funcionamiento del algoritmo de Dijkstra, se ingresará los mismos datos del grafo de la Figura 1.20. con el objetivo de validar el resultado del sistema implementado.

Después de realizar de manera manual del ejercicio, se valida el procedimiento con el sistema implementado para comparar los resultados.

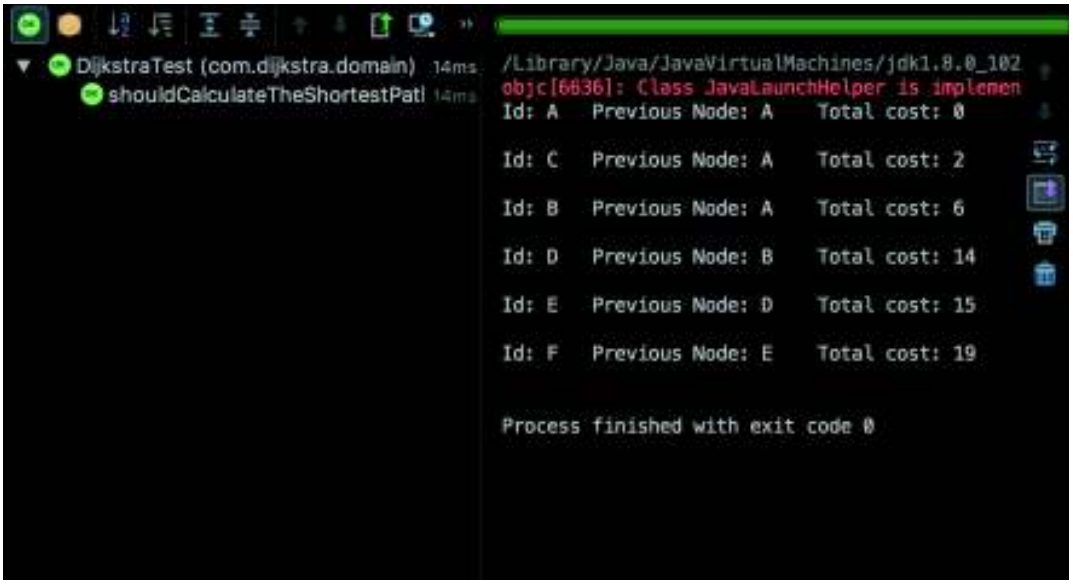

Figura 2.23. Resultados obtenidos de la implementación del algoritmo de Dijkstra usando un ejemplo de un grado de grado 6.

Como se puede apreciar en la Figura 2.23., el resultado obtenido por el proceso manual y el resultado entregado por el sistema implementado es el mismo. En ambos casos el camino más corto para llegar del nodo A al nodo F es:

$$A \rightarrow B \rightarrow D \rightarrow E \rightarrow F$$

Cabe señalar la importancia de entregar una matriz en función de los pesos de todos los nodos para que el sistema de información al usuario entregue más de una opción de ruta.

Después de calcular las posibles rutas de movilización el servicio se encarga de generar un JSON (formato ligero de interpretación y generación de datos) con los resultados, estos datos pasan a un controlador para que se puedan incluir en la respuesta del servicio REST.

2.5 Sistema de información de rutas y paradas de transporte público

Después de desarrollar el algoritmo en un sistema de programación, lo siguiente es diseñar una interfaz gráfica de fácil acceso para los usuarios. Esta interfaz facilitará el ingreso de datos y el entendimiento de los resultados debido a que se los presentará de manera gráfica en un mapa y con indicaciones resumidos para llegar al destino. Cabe aclarar que el diseño de esta interfaz está desarrollado para navegadores web, aunque no es totalmente responsiva se puede usar esta interfaz de manera fácil en teléfonos móviles, para proyectos futuros se podría construir una interfaz nativa para teléfonos móviles.

Diseño de una interfaz gráfica para ingresar la búsqueda

La interfaz es un dispositivo que permite comunicar dos sistemas que no hablan el mismo lenguaje. También se utiliza este término para definir el juego de conexiones y dispositivos que posibilitan la comunicación entre dos sistemas.

En este caso, se define como interfaz gráfica, al programa informático que interactúa con el usuario y se comunica con el sistema de procesamiento, esta interfaz está constituida por una serie de menús e íconos que representan las opciones que el usuario puede tomar dentro del sistema. Su principal objetivo es proporcionar un entorno visual sencillo que permita la comunicación con el sistema operativo de una máquina o computador.

Diseñar una interfaz gráfica que sea intuitiva con los usuarios facilita la interacción del usuario con el sistema. Ejemplo de esto son las interfaces gráficas de sistemas como:

- Google Maps: Esta aplicación de servicio de Web Mapping con tecnología proporcionada por Google, alberga una amplia gama de funciones como: ubicación de calles, planificador de rutas para viajar a pie (beta), viajes en automóvil, bicicleta o en transporte público. Además, localiza geográficamente lugares, edificios o comercios en diversos lugares del mundo y permite visualizar imágenes satelitales de la geografía. El usuario puede acceder a todas estas funciones a través de un menú bien diseñado, intuitivo que permite navegar a través de todas las funciones.
- Heremaps: Es la aplicación de Web Maping y Map Data que permite a los usuarios: conocer datos del tráfico en tiempo real, localizar direcciones, planificar rutas y otras funcionales. Su sistema cuneta con una interfaz gráfica de fácil movilidad para el usuario.

La interfaz desarrollada para el presente sistema captura las mejores características de interfaces como: Google Maps y Heremaps. Lo que permitió crear una interfaz gráfica intuitiva y con alta usabilidad; que deja todo el procesamiento de datos al backend del sistema.

En la Figura 2.24. se puede visualizar la interfaz gráfica del menú principal desarrollado para el presente sistema y cuenta con estos componentes:

- **Campo de texto para ingresar el origen:** Este campo tiene el nombre de "Inicio". Dentro de este campo el usuario deberá ingresar su punto actual de ubicación o el punto desde el cual desea iniciar el cálculo de la distancia.
- **Campo de texto para ingresar el destino:** Este campo tiene el nombre de "Fin". En este campo el usuario debe ingresar el punto al cual desea llegar, este será el punto final en el que concluirá el cálculo de la distancia. Cabe explicar que para mejorar la experiencia del usuario se realizó la integración del sistema con la API de Google Maps para autocompletar las posiciones de inicio y fin.
- **Botón para obtener resultados:** Este botón tiene el nombre de "Planear viaje" y su función es arrojar los resultados del cálculo en función de los datos ingresados en los campos de "Inicio" y "Fin".

- **Cuadro de resultados:** Este cuadro es el resultado inmediato del botón "Planear viaje". Dentro de este cuadro se indica al usuario:
    - Hasta tres opciones de rutas que el usuario puede escoger para realizar su viaje. Cada opción indica:
        - Los IDs de las paradas en las que el usuario debe iniciar y terminar el viaje respectivamente.
        - Información completa sobre el número de transbordos que el usuario debe realizar en caso de ser necesario.
        - La duración aproximada del viaje bajo la suposición de que el bus viaja a 25 km/h.
        - Información sobre la línea de bus en la que debe realizar el viaje.
        - La ruta por la cual el usuario debe realizar el viaje a pie para llegar a su destino.

  Todos estos datos son posibles gracias al procesamiento de los GTFS y los cálculos de los algoritmos de Dijkstra.

- Botón para limpiar campos: Este botón tiene el nombre de "Limpiar campos" y su función es volver a cero el sistema para iniciar un nuevo cálculo.

Figura 2.24. Componentes principales del sistema

Además, el sistema cuenta con estas bondades que elevan la experiencia del usuario y aumentan las probabilidades de uso del sistema:

- Mapa digital: En la Figura 2.25. se puede visualizar el mapa del Distrito Metropolitano de Quito, en esta área se permite a los usuarios:
    - Acercar el mapa para mejorar la visualización de calles e intersecciones.
    - Alejar el mapa para mejor visualización de la zona.
    - Geo-localizar su origen de manera automática.
    - Establecer su origen y destino con clic derecho.
    - Establecer un punto en el mapa movilizando marcador con clic derecho.
    - Visualizar las rutas disponibles para realizar el viaje.

Figura 2.25. Mapa digital del sistema.

2.6 Diseño de pruebas de usabilidad

La usabilidad es una característica obligatoria con la que debe cumplir un producto con usuario final, ya sea en el área de: la tecnología, robótica, genética, etc. Estas pruebas de usabilidad tienen la capacidad de demostrar la facilidad del producto probándolo sobre el grupo de usuarios al que está dirigido.

En el área de la tecnología donde se desarrollan productos dirigidos a usuarios conectados a través de un ordenador o un teléfono inteligente, se les pide a los usuarios completar tareas específicas dentro del sistema mientras son observados por un investigador. La misión de los usuarios es localizar los problemas o confusiones que pueda presentar el sistema. [28]

A partir de esto los desarrolladores tienen la retroalimentación para realizar cambios dentro del sistema con el fin de mejorar la experiencia de usuario.

Cabe aclarar que la prueba de usabilidad se diferencia de las pruebas de error, de aceptación y otras; porque, esta es realizada con los usuarios finales del sistema. Mientras las otras pruebas son desarrolladas con el equipo interno de desarrollo.

El primer paso para realizar la prueba de usabilidad es definir las actividades que desarrollaran dentro del sistema. Estas "actividades" deben estar situadas dentro de un escenario de trabajo en el cual los usuarios puedan desarrollar todas las actividades solicitadas.

En este caso los usuarios deben ser capaces de desarrollar estas actividades:

1. Abrir la página web
2. Seleccionar origen y destino
3. Encontrar la ruta optima en bus para llegar al destino

El segundo paso para realizar una prueba de usabilidad es definir el "escenario de trabajo" en el que se le pedirá al usuario desarrollar las actividades. En este caso el escenario de trabajo puede ser planteado de la siguiente manera:

"Usted debe movilizarse de la manera más rápida desde un punto A hacia un punto B; y, dos horas después debe movilizarse de un punto C hacia un punto D.
Toda su movilización ocurre en el sector Centro Norte de Quito, un día lunes a partir de las 15h00. Usted es nuevo en la ciudad y desconoce todas las paradas al igual que las líneas de transporte público. Usted conoce una herramienta digital que le permitirá planear sus rutas"

En este escenario de trabajo el usuario debe realizar estas actividades:

1. Ingresar a la aplicación.
2. Ingresar el inicio o el punto de origen.
3. Ingresar el destino.
4. Obtener respuestas a través del botón planear viaje.
5. El usuario deberá seleccionar la opción limpiar campos para planear otro viaje.

Cabe señalar la importancia de estos escenarios de trabajo, ya que permiten a los diferentes usuarios del sistema relacionarse de mejor manera con las funciones del sistema en otros contextos. Además, debemos tomar en cuenta los diferentes tipos de usuario que harán uso de la plataforma. Estos pueden ser:

- **Turistas:** Quito al ser posicionado como destino turístico líder en Sudamérica, atrae gran cantidad de turistas nacionales e internacionales. Ellos necesitan un sistema de planificación de rutas que les permita relacionarse con la cultura y los lugares turísticos durante su estadía en la ciudad.
- **Migrantes Nacionales y Extranjeros:** Quito es un centro político, económico y social que atrae gran cantidad de personas que deben movilizarse a través del sistema de transporte público, el objetivo de este sistema es brindar una opción de movilidad sostenible con todos los usuarios independientemente de su nacionalidad u origen.
- **Ciudadanos Quiteños:** Quito es una ciudad con 324 de km², en la que el 73% de sus habitantes utiliza el transporte público para movilizarse de un lugar a otro. El problema a parte del servicio ineficiente, es la falta de herramientas digitales que permita conocer a los quiteños las líneas de transporte disponible, su horario de funcionamiento y la ruta que recorre.

3. RESULTADOS Y DISCUSIÓN

3.1 Prueba de integración del sistema

Para entregar resultados útiles y comprensibles para el usuario el sistema de planificación de rutas está conformado por tres componentes que son:

- **Front-end:** se encarga de interactuar con el usuario y entregar información procesada que pueda ser interpretada. En el proceso, cada vez que el usuario planifica sus trayectos e interactúa con en el Front-end, este realiza peticiones Rest que son conectadas con el Servicio de Mi Bus.
- **Servicio de Mi Bus:** es el encargado de procesar los archivos GTFs en base a la información de inicio y fin ingresada por el usuario. Localiza la ubicación de las paradas y entrega la información en forma de aristas para que pueda ser analizada por el Servicio de Planificación de Rutas.
- **Servicio de Planificación de Rutas:** Se encarga de recibir la información en forma de aristas (IDs y la distancia de las paradas) entregada por el Servicio de Mi Bus. El Servicio de Planificación de Rutas arma los grafos, aplica el algoritmo de Dijkstra y devuelve arreglos en forma de nodos (ID de la parada actual, el ID de la parada anterior y el costo entre ellas). Este Servicio se encarga además de graficar las rutas en el mapa que visualiza el usuario en la plataforma.

Para comprobar la usabilidad del sistema se realizará una prueba manual, que será contrastada con la información proporcionada por cada Servicio del Sistema y validada con el Planificador de Rutas de Google Maps.

Las pruebas tienen como punto de origen la calle Buorgeois y como destino la intersección de las calles Juan de Ascaray y Mariano Jimbo.

Para realizar la prueba manual se localizó todas las paradas entre los puntos seleccionados y se analizó sus IDs y sus costos. Como se indica en las Figuras 3.1 y 3.2.

Figura 3.1. Localización de paradas

ID Parada 1	ID Parada 2	Costo
1	2	550
1	4	1000
1	5	1200
2	6	260
4	12	800
5	8	350
6	10	450
8	9	250
9	13	350
10	14	550
12	18	1000
12	21	800
13	17	350
13	16	400
14	15	280
15	19	400
15	21	450
16	21	750
17	21	500
18	21	400
19	21	190
20	21	
21	21	0

Figura 3.2. Tabla con IDs de paradas y costos

Después de analizar la información se estableció el camino más corto de manera gráfica y analítica. Como se muestra en las Figuras 3.3 y 3.4.

Figura 3.3. Grafo manual

Resultados		
Id: 1	Previous Node: 1	Total cost: 0
Id: 2	Previous Node: 1	Total cost: 550
Id: 6	Previous Node: 2	Total cost: 810
Id: 4	Previous Node: 1	Total cost: 1000
Id: 5	Previous Node: 1	Total cost: 1200
Id: 10	Previous Node: 6	Total cost: 1260
Id: 8	Previous Node: 5	Total cost: 1550
Id: 12	Previous Node: 4	Total cost: 1800
Id: 9	Previous Node: 8	Total cost: 1800
Id: 14	Previous Node: 10	Total cost: 1810
Id: 15	Previous Node: 14	Total cost: 2090
Id: 13	Previous Node: 9	Total cost: 2150
Id: 19	Previous Node: 15	Total cost: 2490
Id: 17	Previous Node: 13	Total cost: 2500
Id: 20	Previous Node: 15	Total cost: 2540
	Destino 20:	20,15, 14, 10, 6, 2, 1

Figura 3.4. Tabla de Resultados

El siguiente paso se realizó para:

- Validar los datos del Servicio de Planificación de Rutas implementado.
- Contrastar los IDs de las paradas y sus costos con los datos obtenidos de manera manual.

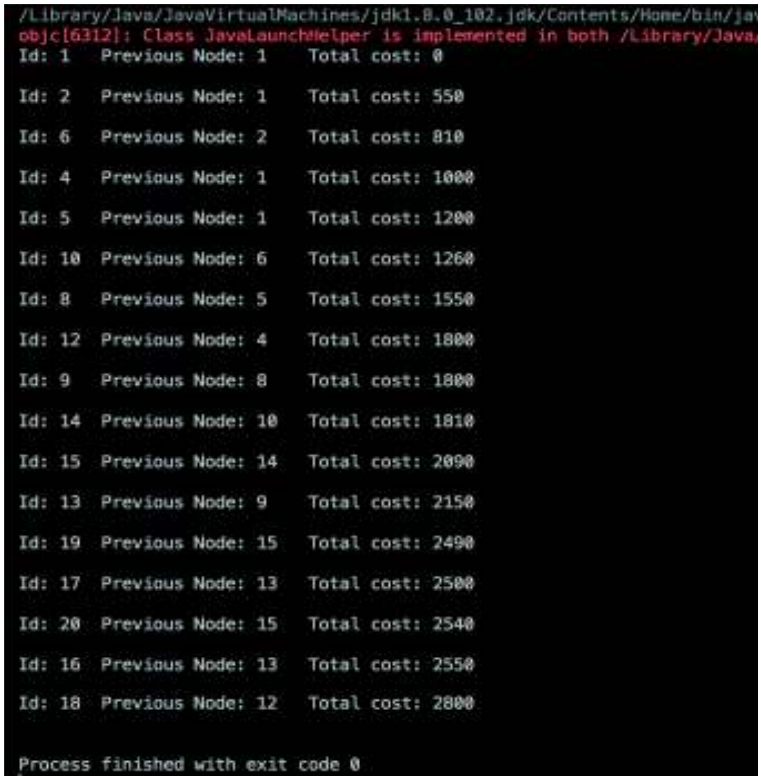

Figura 3.5. Tabla de resultados entregada por el Sistema de Mi Bus

Se puede apreciar que los IDs de las paradas y los costos entregados por el Servicio de Planificación de Rutas implementado coinciden con los datos de la prueba manual.

El siguiente paso será validar el programa completo para comprobar el flujo de datos y la comunicación entre los tres componentes del sistema: Front-end, Servicio de Mi Bus y Servicio de Planificación de Rutas.

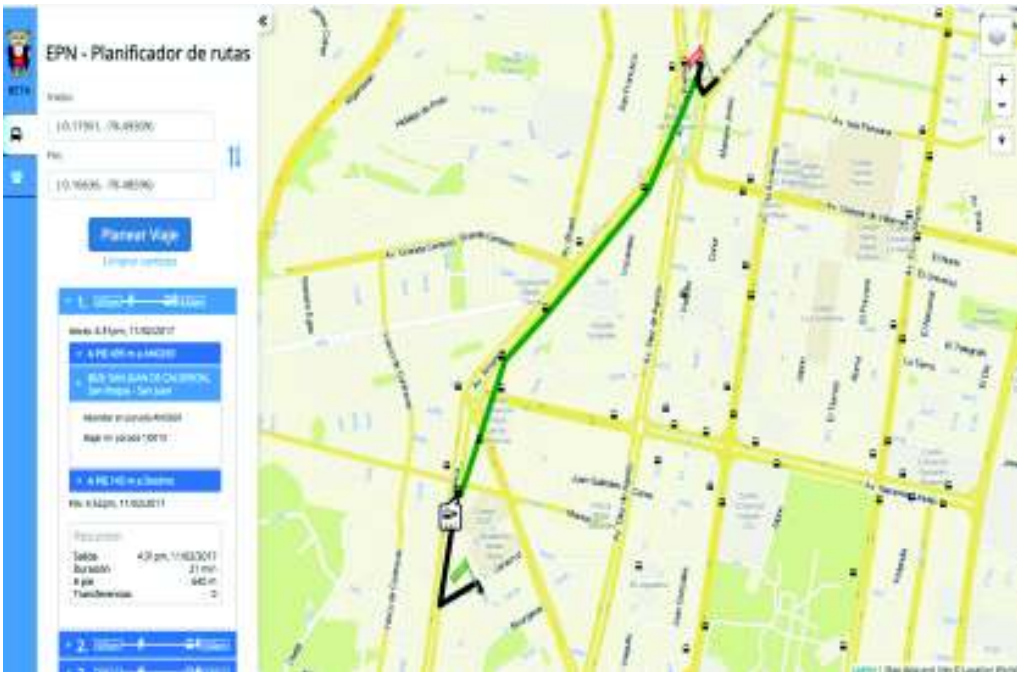

Figura 3.6. Resultado de la prueba de usabilidad del sistema

Como se aprecia en la Figuras 3.6. el sistema y el flujo de datos están funcionando correctamente. Esto comprueba que todos los componentes se están comunicando adecuadamente.

Para finalizar, se contrastará la información proporcionada por el Sistema de Mi Bus con el Sistema de Planificación de Rutas de Google Maps.

Figura 3.7. Resultados en Google Maps

Como se observa en la Figuras 3.7. Los resultados obtenidos con el Planificador de Rutas de Google Maps son idénticos a los resultados del Sistema de Mi Bus. Esta validación es indispensable para iniciar las pruebas de usabilidad con otros usuarios.

3.2 Pruebas de usuario

Estas pruebas están orientadas hacia personas que encajen con los perfiles de usuario planteados en el capítulo anterior (Turistas, Migrantes y Ciudadanos quiteños).

Las pruebas permitirán: validar la usabilidad del sistema (que tan fácil le resulto al usuario encontrar información de transporte público para movilizarse a su destino) y receptar información sobre las características que deben ser mejoradas.

Descripción de la metodología de la prueba de usabilidad

Para realizar las pruebas de usabilidad se seleccionaron diferentes perfiles de usuario, con cada grupo se realizará diferentes pruebas con las siguientes características:

- Cada prueba de usabilidad tendrá una duración total de aproximadamente: 10 a 15 minutos.
- Cada prueba de usabilidad estará compuesta de tres fases:

    - **Primera Fase:** En esta fase se le preguntará al usuario las rutas de transporte público que conoce para llegar del punto A hacia el punto B. El usuario deberá contestar de manera oral en un tiempo máximo de 2 minutos.
    - **Segunda Fase:** Durante esta fase el usuario debe identificar las rutas existentes del punto A hacia el punto B con el apoyo de una hoja de rutas de Street Maps; y, comentar su experiencia durante la prueba. El usuario tendrá un tiempo máximo de 5 minutos frente al computador.
    - **Tercera Fase:** Para esta fase el usuario debe ingresar los puntos A y B al sistema de Mi Bus; y, navegar por la interfaz para interpretar los resultados, al final el usuario deberá comentar su experiencia. Esta fase tendrá una duración máxima de 5 minutos.

Descripción de los perfiles de los usuarios

Para realizar las pruebas de usuario es imperativo construir y definir:

- Los perfiles de los usuarios ideales (Users Personas): Es la representación semi-ficticia de las personas a las que va orientado el sistema, esta representación define las cualidades, comportamientos, aficiones y necesidades de los usuarios.
- Viaje del usuario (User's Journey): Es el proceso de búsqueda activa que realiza una persona hasta convertirse en usuario permanente de un sistema.

En base a los tres tipos de usuario a los que se dirige el sistema, el primer paso es representar sus cualidades, comportamientos, características demográficas, relación con la tecnología y necesidades.

Turistas:

Figura 3.8. Perfil semi-ficticio de turistas

- Jóvenes/adultos ecuatorianos o extranjeros entre los 20 y 40 años de edad.
- Estadía aproximada en Quito menor a tres meses.
- Son intrépidos, aventureros, prefieren gastar su dinero en experiencias antes que en movilización.
- Presupuesto mensual igual o menor a 500$.
- Suelen alojarse en el Centro Norte o en el Centro de la ciudad, puesto que son las zonas hoteleras de Quito.
- Por lo general cuentan con una computadora portátil.
- En su totalidad disponen de un teléfono inteligente.
- Poseen alta adaptación y relación con la tecnología.
- Poseen nulo conocimiento sobre el funcionamiento del transporte público.

- Necesidad: Requieren conocer las rutas de transporte público para llegar a los lugares turísticos.
- Ganancia: Podrán relacionarse con la cultura de la ciudad de manera económica.

Migrantes nacionales y extranjeros:

Figura 3.9. Perfil semi-ficticio de migrantes.

- Migrantes ecuatorianos o extranjeros entre los 18 y 40 años de edad.
- Estadía permanente en Quito.
- Presupuesto mensual igual o menor a 500$.
- Son personas que tienden al ahorro y la eficiencia.
- Arriendan viviendas en zonas periféricas de la ciudad, puesto que en esas zonas los arriendos suelen ser más económicos.
- Sus actividades laborales se desarrollan en las zonas: Centro Norte o Norte.
- Por lo general cuentan con una computadora portátil y/o un teléfono inteligente.
- Poseen entre alta y mediana adaptación y relación con la tecnología.
- Poseen escaso o nulo conocimiento sobre el funcionamiento del transporte público.
- Necesidad: Requieren conocer la mejor manera más eficiente para movilizarse dentro de una ciudad nueva, con el menor coste posible.
- Ganancia: Utilizar el sistema permitirá a estos usuarios llegar a sus destinos en el menor tiempo posible, relacionarse mejor con la estructura de transporte de la ciudad y economizar dinero por concepto de movilización.

Ciudadanos Quiteños:

Figura 3.10. Perfil semi-ficticio de ciudadanos quiteños.

- Ciudadanos quiteños entre los 18 y 40 años de edad.
- Residentes en Quito.
- Ingreso mensual igual o menor a 500$.
- Son personas que desean agilizar su movilización por la ciudad.
- Sus actividades laborales o estudiantiles se desarrollan en las zonas: Centro Norte o Norte.
- Por lo general cuentan con una computadora portátil y/o un teléfono inteligente.
- Poseen entre alta y mediana adaptación y relación con la tecnología.
- Poseen alto o mediano conocimiento sobre el funcionamiento del transporte público.
- Necesidad: Requieren conocer la mejor manera más eficiente para movilizarse dentro su ciudad.
- Ganancia: Utilizar el sistema permitirá a estos usuarios llegar a sus destinos en el menor tiempo posible.

En base a toda esta información se define un viaje del usuario (User's journey) que podría ser implementado en proyectos futuros en combinación con acciones de Marketing y publicidad.

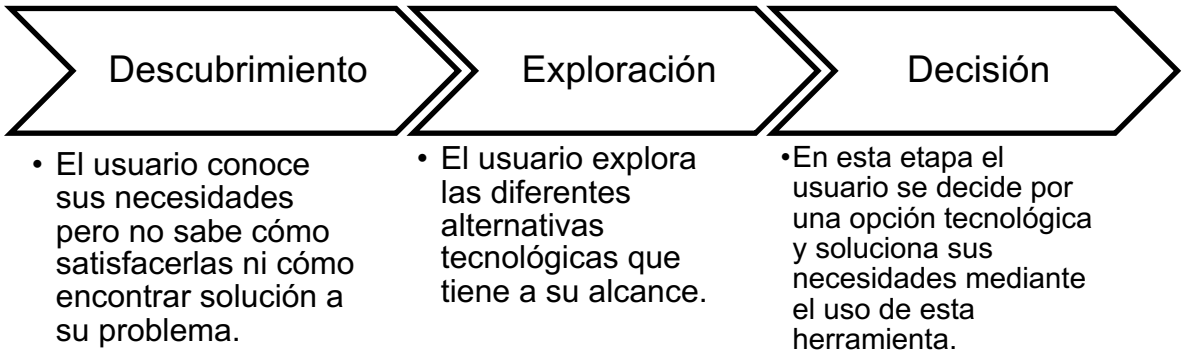

Figura 3.11. Descripción del Viaje del usuario por el sistema

Descripción del problema

En base a la descripción de los perfiles, el siguiente paso es definir los destinos del Centro Norte de Quito y plantear los problemas con los que se realizarán las pruebas de usabilidad en cada tipo de usuario.

- Prueba de usabilidad para Turistas:

Usted es un Turista Extranjero que debe movilizarse de la manera más rápida desde el Parque Botánico de Quito hacia el Teleférico; y, dos horas más tarde debe movilizarse desde el Teleférico hacia el Museo del Agua “Yaku”. En un intervalo de dos horas usted decide movilizarse desde el Museo del Agua “Yaku” de regreso al Parque Botánico de Quito. ¿Cómo realizaría su movilización única y exclusivamente en transporte público?

Recuerde toda su movilización ocurre en el sector Centro Norte de Quito, un día lunes a partir de las 10h00. Usted desconoce las paradas al igual que las líneas de transporte público.

- Prueba de usabilidad para Migrantes:

Usted es un Migrante Extranjero con menos de 3 meses de residencia en la ciudad de Quito, que debe movilizarse de la manera más rápida desde su domicilio en la Mariana de Jesús y Amazonas hacia la Corte Nacional de Justicia; y, una hora más tarde debe movilizarse desde la Corte Nacional de Justicia hacia el Parque el Ejido. En un intervalo de dos horas usted decide movilizarse desde el Parque el Ejido hacia la Mariana de Jesús y Amazonas para regresar a su hogar. ¿Cómo realizaría su movilización única y exclusivamente en transporte público?

Recuerde toda su movilización ocurre en el sector Centro Norte de Quito, un día martes a partir de las 8h00. Usted desconoce las paradas al igual que las líneas de transporte público.

- Prueba de usabilidad para Ciudadanos Quiteños:

Usted es un Ciudadano Quiteño que está cursando sus años de Universidad, usted debe movilizarse de la manera más rápida desde la Escuela Politécnica Nacional hacia el Centro Comercial Quicentro Shopping; y, 45 minutos después debe movilizarse desde el Quicentro Shopping hacia la Capilla del Hombre. En un intervalo de dos horas usted decide movilizarse desde la Capilla del Hombre hacia la Escuela Politécnica Nacional. ¿Cómo realizaría su movilización única y exclusivamente en transporte público?

Toda su movilización ocurre en el sector Centro Norte de Quito, un día viernes a partir de las 14h00. Usted no tiene total conocimiento acerca de las paradas al igual que las líneas de transporte público.

Resultados y opiniones

Durante esta sección se analizarán los resultados de las pruebas de usabilidad realizadas con usuarios de distintos perfiles.

Debido a que las pruebas se realizaron con una cantidad pequeña de usuarios, no se pueden sacar datos estadísticos pero se indicará la tendencia para cada uno de los casos establecidos.

Pero se puede observar que el 100% de los usuarios pudieron completar la tarea asignada usando el sistema implementado en menos tiempo del asignado. También se puede ver la tendencia de las rutas encontradas que oscila de 1 a 0 para los usuarios que viven poco tiempo en la ciudad.

A continuación se indican los resultados de las pruebas para cada uno de los perfiles escogidos de potenciales usuarios, en los resultados se desea contrastar el número de

rutas encontradas para los diferentes escenarios: usando su conocimiento actual, usando los datos de Open Street Maps y usando el sistema implementado en este proyecto.

Prueba de usabilidad 1

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad ecuatoriana oriunda de la provincia de Imbabura. El hombre:

- Tiene 4 años de residencia en la ciudad de Quito.
- Está medianamente familiarizado con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Migrante Nacional.

	Rutas conocidas por el usuario	Rutas identificadas en Street Maps	Rutas proporcionadas por Mi Bus
Número de Rutas encontradas	4	4	9
Tiempo para encontrar rutas	2	5	3

Tabla 3.1. Resultados de la Prueba de usabilidad 1

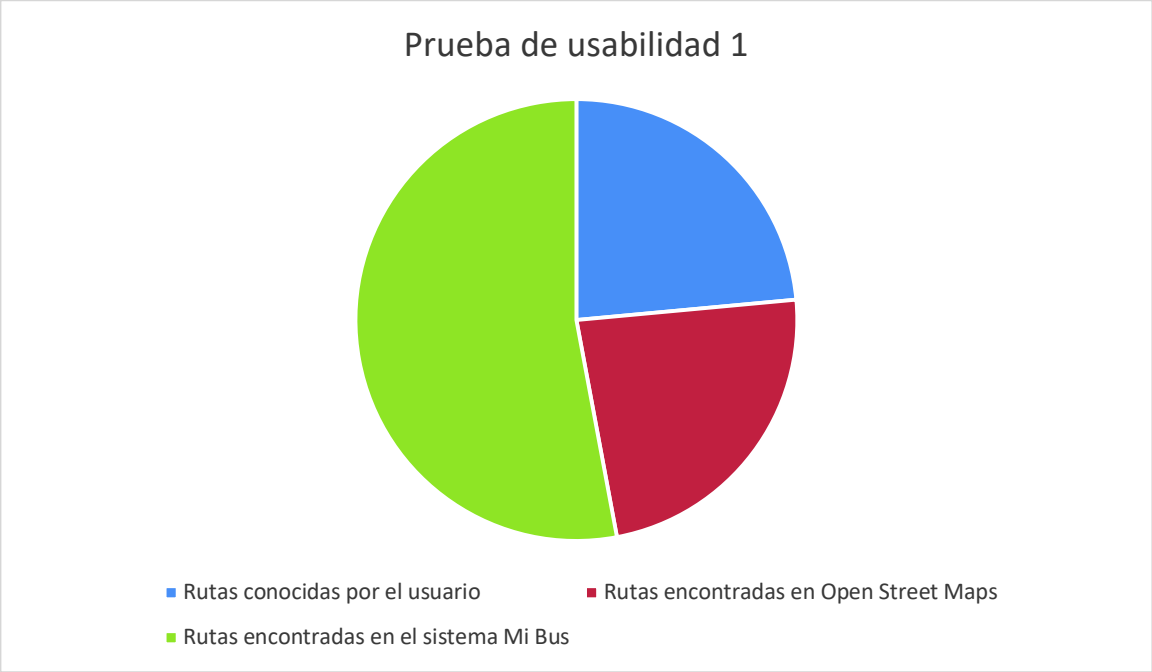

Figura 3.12. Gráfico de Resultados – Prueba de usabilidad 1

Comentarios del usuario:

Según el usuario, la información proporcionada por Open Street Maps es de difícil lectura y no es lo suficientemente ágil. Por otro lado, al realizar la prueba en el Sistema de Mi Bus el usuario notó mejoría en la entrega de resultados; para el usuario, el sistema es fácil da comprender, se despliega de manera rápida y puede ayudarle a las personas a mejorar su movilidad.

El usuario expresó que se podría mejorar el sistema debería con una mejor señalética y ubicación de las paradas y barrios dentro del mapa.

Prueba de usabilidad 2

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad ecuatoriana oriunda de la provincia de Pichincha, cantón Quito. La mujer:

- Está totalmente familiarizada con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Ciudadana Quiteña.

	Rutas conocidas por el usuario	**Rutas identificadas en Street Maps**	**Rutas proporcionadas por Mi Bus**
Número de Rutas encontradas	5	5	9
Tiempo para encontrar rutas	2	5	4

Tabla 3.2. Resultados de la Prueba de usabilidad 2

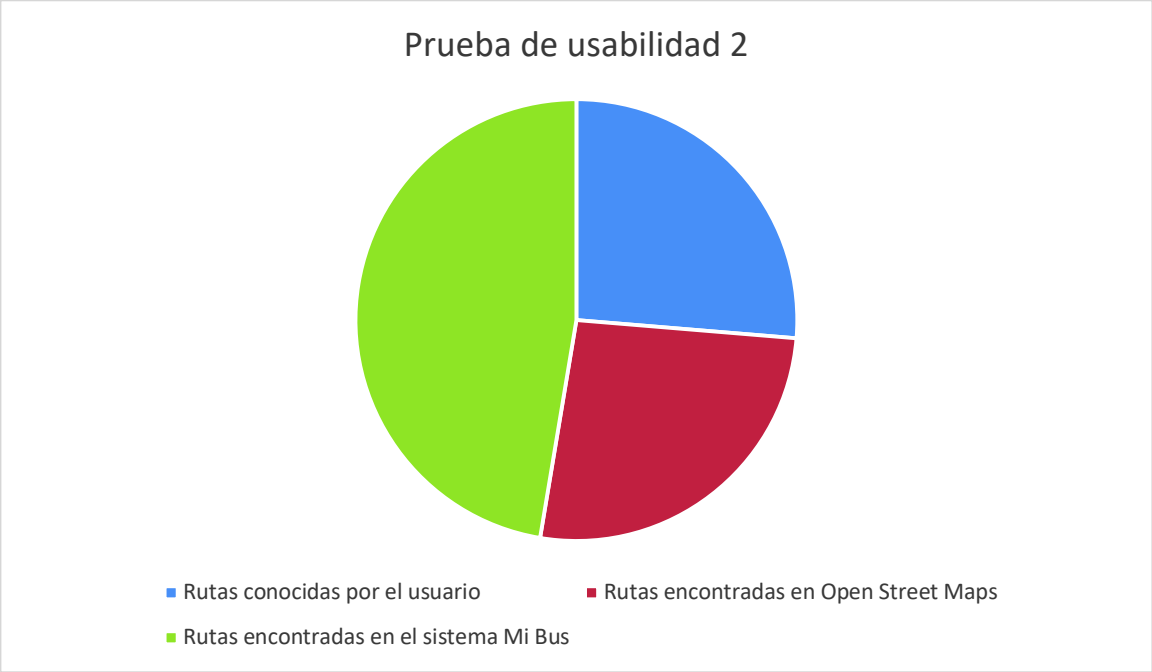

Figura 3.13. Gráfico de Resultados – Prueba de usabilidad 2

Comentarios del usuario:

Según la usuaria, la información proporcionada por Open Street Maps es intuitiva y fácil de utilizar. Sin embargo, este método es más efectivo para las personas que conocen la ciudad.

Por otro lado, al realizar la prueba en el Sistema de Mi Bus la usuaria comentó que la información de las paradas es difícil de entender. En cuanto a la interfaz gráfica expresó que se despliega correctamente y es comprensible para todo usuario.

La usuaria cree que se podría mejorar la experiencia con el sistema colocando puntos de referencia para ubicar las paradas.

Prueba de usabilidad 3

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad ecuatoriana oriunda de la provincia de Pichincha, cantón Quito. La mujer:

- Está totalmente familiarizada con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Ciudadana Quiteña.

	Rutas conocidas por el usuario	Rutas identificadas en Street Maps	Rutas proporcionadas por Mi Bus
Número de Rutas encontradas	5	4	9
Tiempo para encontrar rutas	1	5	3

Tabla 3.3. Resultados de la Prueba de usabilidad 3

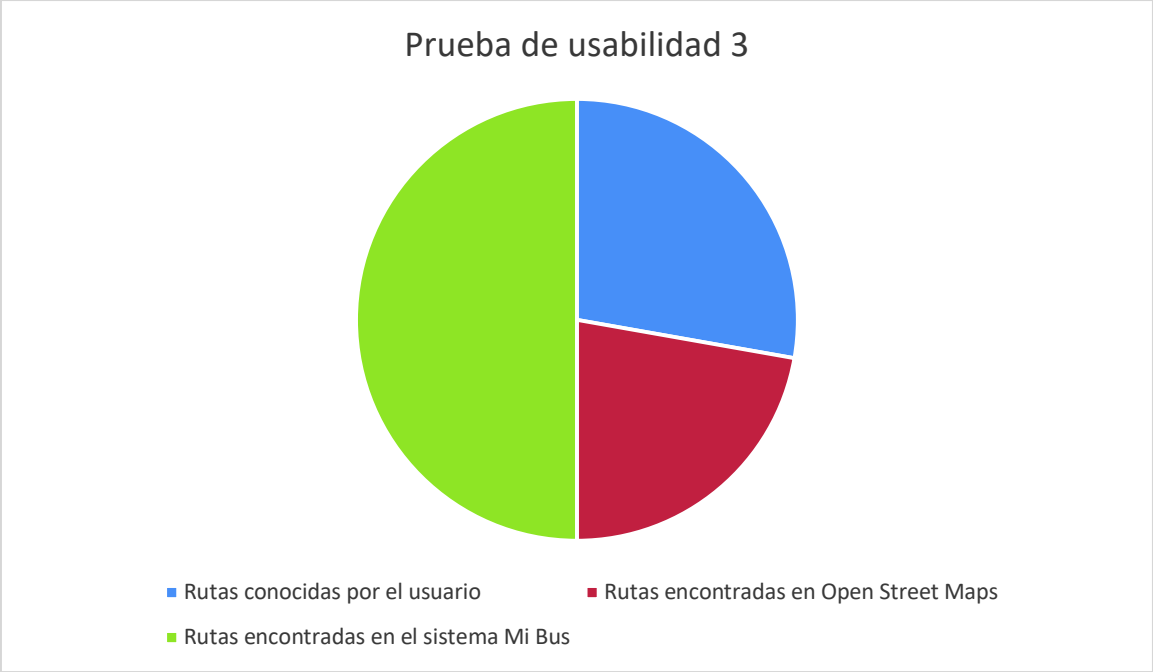

Figura 3.14. Gráfico de Resultados – Prueba de usabilidad 3

Comentarios del usuario:

Para la usuaria la planificación de rutas con la hoja de Open Street Maps disminuye la agilidad y no brinda una buena experiencia al usuario. Es poco intuitivo y confuso para la persona.

Por otro lado, ella expresó que la información proporcionada por el Sistema de Mi Bus es: clara, comprensible y amigable con las personas. Cualquiera puede usar el sistema y llegar con facilidad a su destino. Para la usuaria es una opción viable para mejorar la movilización de las personas residentes en Quito.

Prueba de usabilidad 4

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad ecuatoriana oriunda de la provincia de Pichincha, cantón Quito. El hombre:

- Está totalmente familiarizado con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Ciudadano Quiteño.

	Rutas conocidas por el usuario	**Rutas identificadas en Street Maps**	**Rutas proporcionadas por Mi Bus**
Número de Rutas encontradas	3	4	9
Tiempo para encontrar rutas	2	5	4

Tabla 3.4. Resultados de la Prueba de usabilidad 4

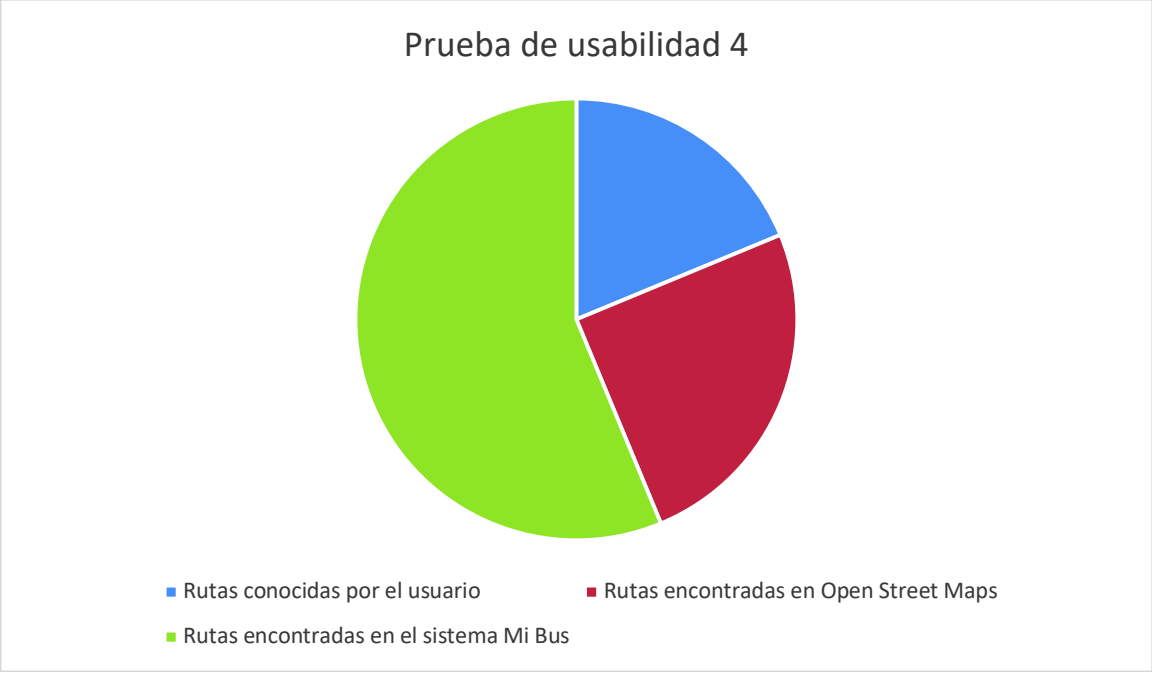

Figura 3.15. Gráfico de Resultados – Prueba de usabilidad 4

Comentarios del usuario:

Para el usuario la hoja de rutas de Open Street Maps se limita a personas que conocen la ciudad, él expresó que: para una persona que no conoce la distribución de la ciudad y quiere un lugar movilizarse a un lugar desconocido, este método no es óptimo.

Por otro lado, el comentó que con la aplicación de Mi Bus la movilización es inmediata y la interpretación de los resultados es sencilla; pues, la información se muestra de manera clara y las indicaciones son específicas. Además, señaló que: el sistema es simple de manejar y amigable para los usuarios; y, puede mejorar la vida de extranjeros, migrantes, turistas y quiteños.

Prueba de usabilidad 5

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad ecuatoriana oriunda de la provincia de Pichincha, cantón Quito. El hombre:

- Está medianamente familiarizado con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Ciudadano Quiteño.

	Rutas conocidas por el usuario	Rutas identificadas en Street Maps	Rutas proporcionadas por Mi Bus
Número de Rutas encontradas	2	0	9
Tiempo para encontrar rutas	1	4	4

Tabla 3.5. Resultados de la Prueba de usabilidad 5

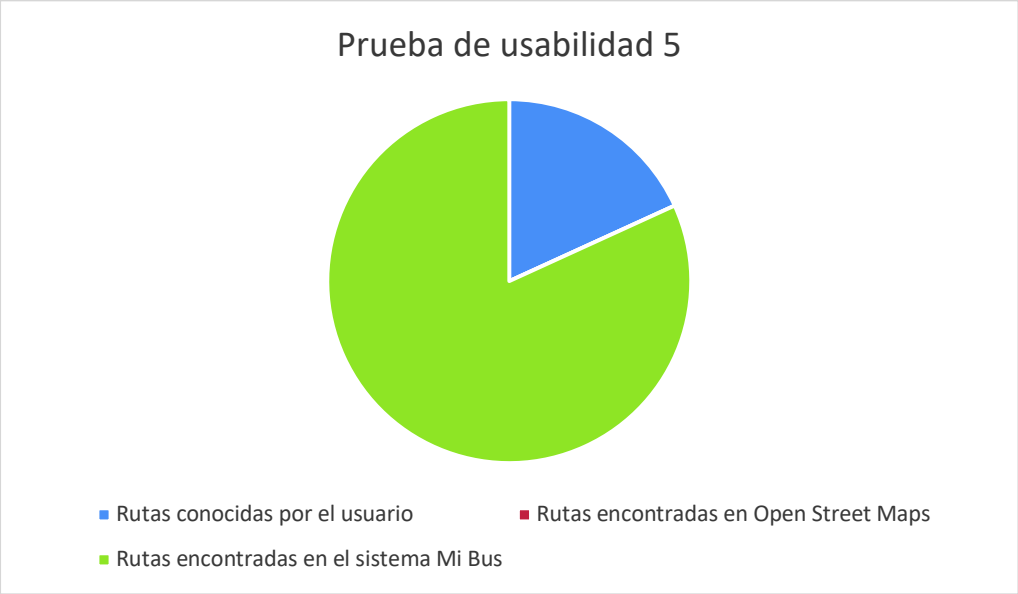

Figura 3.16. Gráfico de Resultados – Prueba de usabilidad 5

Comentarios del usuario:

Para el usuario fue extremadamente complicado encontrar las rutas de buses que cumplían con el recorrido de las pruebas en Open Street Maps. El usuario expresó que el nivel de dificultad era similar para un quiteño que para un migrante o un turista.

Por otro lado, con el Sistema de Mi Bus, el usuario señaló que es la manera más fácil de planificar un recorrido, porque la información es completa, confiable, comprensible y útil. Además, la interfaz presenta los datos de manera clara y amigable.

Prueba de usabilidad 6

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad ecuatoriana oriunda de la provincia de Imbabura. La mujer:

- Tiene 3 años de residencia en la ciudad de Quito.
- No está familiarizada con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Migrante Nacional.

	Rutas conocidas por el usuario	**Rutas identificadas en Street Maps**	**Rutas proporcionadas por Mi Bus**
Número de Rutas encontradas	0	0	9
Tiempo para encontrar rutas	2	2	4

Tabla 3.6. Resultados de la Prueba de usabilidad 6

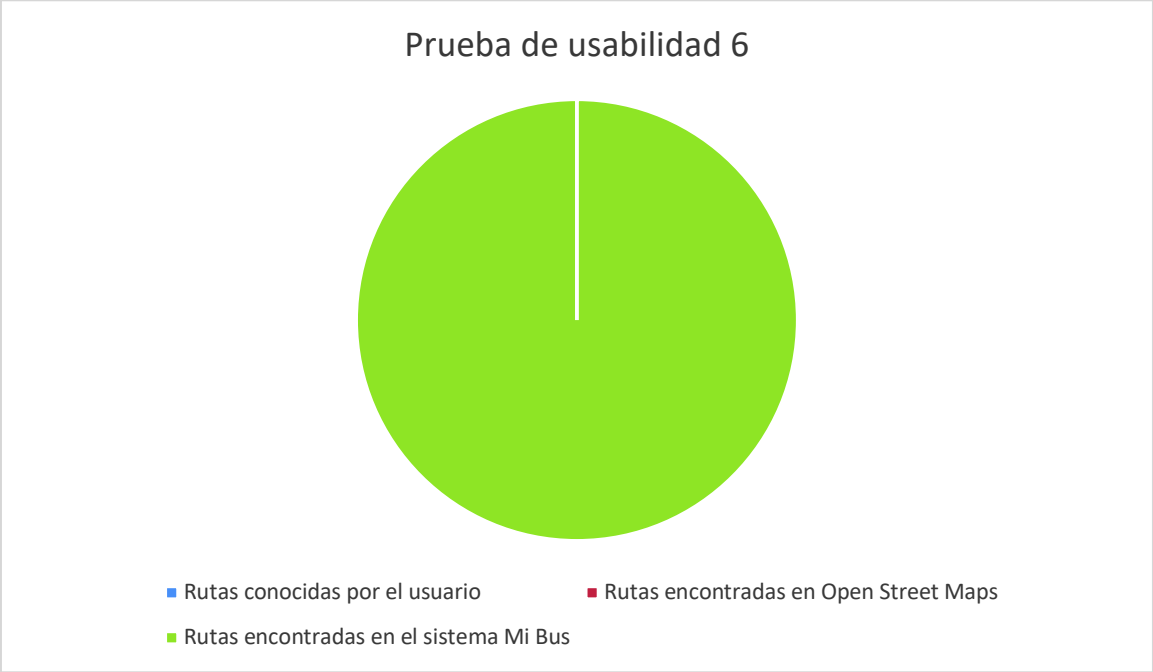

Figura 3.17. Gráfico de Resultados – Prueba de usabilidad 6

Comentarios del usuario:

Para la usuaria la información proporcionada por Open Street Maps es de difícil lectura y no es útil para migrantes o turistas. Por otro lado, la usuaria notó una gran mejora en la información proporcionada en el Servicio de Mi Bus, ella expresó que los datos proporcionados por Mi Bus son confiables y ayudan a mejorar la movilidad de las personas que llegan por primera vez a la ciudad.

La usuaria recomendó que el sistema podría reemplazar el nombre del botón “Limpiar campos” por el nombre "Iniciar nueva ruta” o “Iniciar nuevo viaje".

Prueba de usabilidad 7

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad ecuatoriana oriunda de la provincia de Los Ríos. El hombre:

- Tiene 3 años de residencia en la ciudad de Quito.
- Está medianamente familiarizada con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Migrante Nacional.

	Rutas conocidas por el usuario	Rutas identificadas en Street Maps	Rutas proporcionadas por Mi Bus
Número de Rutas encontradas	1	0	9
Tiempo para encontrar rutas	2	5	4

Tabla 3.7. Resultados de la Prueba de usabilidad 7

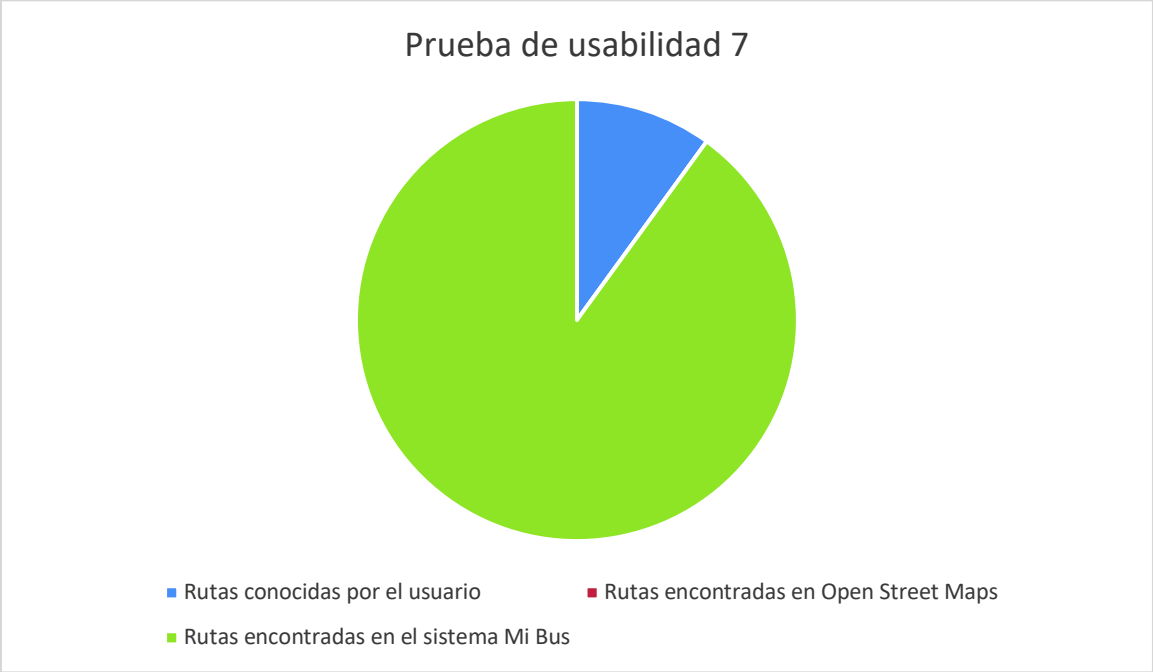

Figura 3.18. Gráfico de Resultados – Prueba de usabilidad 7

Comentarios del usuario:

Para el usuario la información proporcionada por Open Street Maps es de difícil lectura; y, no es intuitiva con los usuarios que desconocen la estructura del sistema público de transporte y la ciudad en general.

Por otro lado, el usuario logró interpretar mejor los resultados brindados por el Sistema de Mi Bus. El usuario sugirió que se podría mejorar el sistema colocando los nombres de las calles transversales dentro de la navegación del mapa y el ingreso de datos en los campos.

Para el usuario el sistema de Mi Bus es compresible y apoyaría en la movilidad de las personas en especial de los que no conocen la ciudad.

Prueba de usabilidad 8

Descripción del Perfil:
Esta prueba de usabilidad se realizó con una persona de nacionalidad hondureña. El hombre:

- Tiene 2 años de residencia en la ciudad de Quito.
- No está totalmente familiarizado con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Migrante Extranjero.

	Rutas conocidas por el usuario	Rutas identificadas en Street Maps	Rutas proporcionadas por Mi Bus
Número de Rutas encontradas	1	0	9
Tiempo para encontrar rutas	2	3	4

Tabla 3.8. Resultados de la Prueba de usabilidad 8

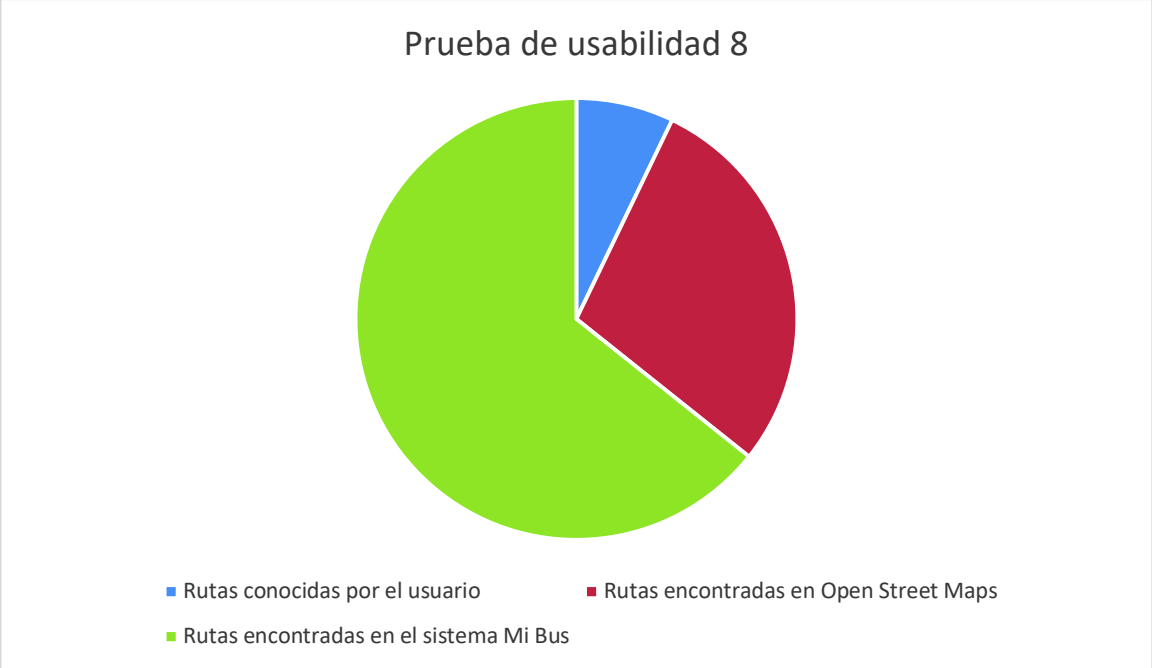

Figura 3.19. Gráfico de Resultados – Prueba de usabilidad 8

Comentarios del usuario:

Para el usuario la búsqueda en Open Street Maps fue complicada porque los mapas no cuentan con puntos de referencia que ayuden a las personas a ubicarse y planificar su ruta.

Por otro lado, el usuario cree que esta herramienta puede facilitar la movilidad de los usuarios que deben transportarse en sistema público.

Prueba de usabilidad 9

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad venezolana. La mujer:

- Tiene 1 año de residencia en la ciudad de Quito.
- Está medianamente familiarizada con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Migrante Extranjero.

	Rutas conocidas por el usuario	Rutas identificadas en Street Maps	Rutas proporcionadas por Mi Bus
Número de Rutas encontradas	0	0	9
Tiempo para encontrar rutas	2	5	4

Tabla 3.9. Resultados de la Prueba de usabilidad 9

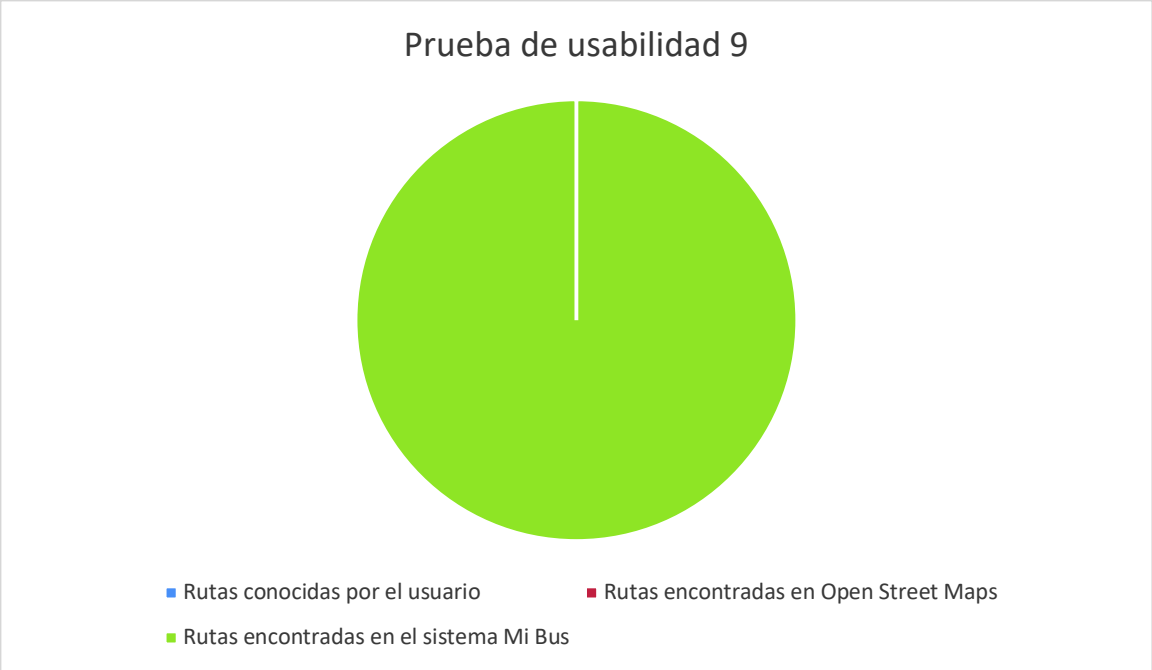

Figura 3.20. Gráfico de Resultados – Prueba de usabilidad 9

Comentarios del usuario:

La usuaria no pudo encontrar los destinos con las rutas de Open Street Maps, para ella ese método no brinda usabilidad al usuario y es poco útil.

Por otro lado, la usuaria cree que el Sistema de Mi Bus ubica de manera sencilla los puntos de origen y destino y la interpretación de resultados es ágil. Para ella esa información muy valiosa, porque ayuda a planificar de mejor manera los resultados ya que se cuenta con la ubicación de las paradas y el tiempo estimado de trayecto.

Prueba de usabilidad 10

Descripción del Perfil:

Esta prueba de usabilidad se realizó con una persona de nacionalidad venezolana. La mujer:

- Tiene 1 mes y media de residencia en la ciudad de Quito.
- No tiene ninguna familiaridad con la estructura de transporte público.
- Prueba de usabilidad en base al perfil de Turista.

	Rutas conocidas por el usuario	**Rutas identificadas en Street Maps**	**Rutas proporcionadas por Mi Bus**
Número de Rutas encontradas	0	0	9
Tiempo para encontrar rutas	2	3	5

Tabla 3.10. Resultados de la Prueba de usabilidad 10

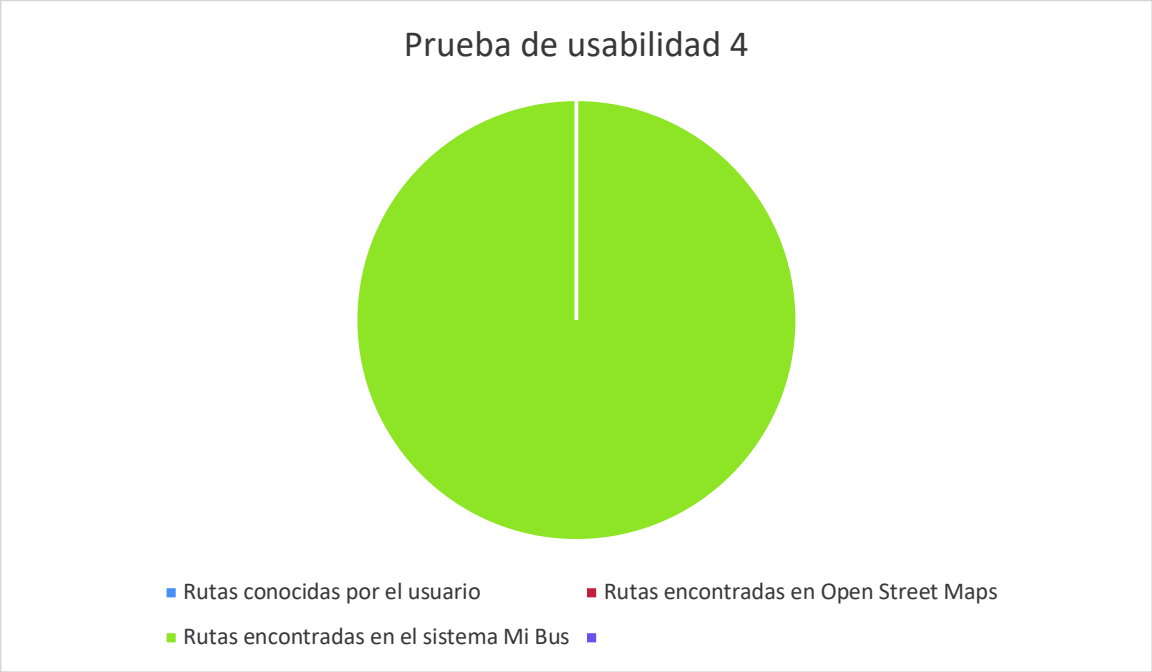

Figura 3.21. Gráfico de Resultados – Prueba de usabilidad 10

Comentarios del usuario:

La usuaria no pudo reconocer ninguno de los puntos planteados de manera oral, ni con los datos de Open Street Maps. Para la usuaria la búsqueda en Open Street Maps es poco intuitiva y no se adapta a las necesidades de los usuarios.

Por otro lado, para la usuaria fue más fácil encontrar la información de los destinos en el Sistema de Mi Bus, ella cree que el sistema es práctico de usar e interpretar.

3.3 Análisis de los resultados

En base a los resultados obtenidos se puedo observar que el sistema de información al usuario de rutas y paradas de transporte público si ayudó a los usuarios a encontrar diferentes opciones para movilizarse en bus dentro a los diferentes puntos dentro del sector centro norte de Quito.

Durante la primera etapa de la prueba, en la que el usuario daba información de cómo movilizarse en bus usando su conocimiento actual, se pudo notar que las personas que han vivido menos de 2 años en la ciudad les resulto muy difícil poder dar nombres de rutas para movilizarse a los puntos de la ciudad especificados en la prueba. Las personas que han vivido en la ciudad de Quito, debido a que estaban más familiarizadas con el sistema

de transporte público, les resulto un poco más fácil poder dar instrucciones para llegar al destino, aunque no conocían todas las rutas disponibles, ni las podían organizar en base a un criterio de optimización.

Se pudo notar una gran dificultad para encontrar manualmente las rutas, es decir usando la lista de rutas de Open Street Maps. Se identificó los siguientes problemas en los usuarios:

- Las personas no conocían la ciudad y/o no estaban familiarizadas con las rutas de transporte público, por eso el tener la referencia de inicio y fin de la ruta ayudaba en la búsqueda.
- Se requiere invertir mucho tiempo para realizar la búsqueda manual en una lista de 126 rutas.
- La primera ruta encontrada en la búsqueda manual, no necesariamente es la más óptima.
- Dificultad para realizar intersecciones entre diferentes rutas.

A pesar de que los usuarios tenían toda la información disponible el tiempo de procesamiento manual es muy grande y requiere conocimiento previo de la ciudad.
En la tercera parte de las pruebas, cuando los usuarios usan el sistema de Mi Bus, se puede ver una gran mejora en el número de rutas que el usuario que el usuario descubre para llegar a los diferentes puntos de la ciudad establecidos en la prueba.

El usuario emplea un tiempo aproximado de 4 minutos para realizar la búsqueda de rutas y paradas, esto se debe a que:

- El usuario se está familiarizando con el sistema.
- El usuario explora la ciudad en el mapa digital.
- Interpretar todos los datos entregados.

Este método les resulto más cómodo a todos los usuarios, debido a que no requería conocimiento previo de transporte público o de la ciudad para realizar la búsqueda de rutas, solo necesitaban tener claro el origen y destino.

En base a la retroalimentación de la experiencia de usuario, se notó una potencial mejora al momento de presentar los nombres de las paradas, indicando una referencia y nombre de las calles en lugar de un código de identificación que se está usando actualmente.

4. CONCLUSIONES

Esta sección expondrá las conclusiones más relevantes del trabajo de investigación; según, los objetivos planteados inicialmente en la Tesis. Acorde a esto:

De manera general este proyecto concluye que:

- El algoritmo de Dijkstra (algoritmo para encontrar la ruta más óptima) constituye una pieza fundamental en la construcción de sistemas automáticos para brindar información de rutas y paradas; lo que facilita la movilización de los usuarios de transporte público dentro de un perímetro específico.
- Algunos algoritmos usados en la robótica móvil autónoma se emplean también en la construcción de sistemas digitales que solucionen problemáticas de diversos contextos, como: encontrar la ruta más corto dentro del Sistema de Transporte Público, para llegar de un punto X a un punto Y en la ciudad.
- Contar con un sistema de planificación de rutas como Mi Bus facilita el acceso a información sobre transporte público. El tener estos datos en un formato de fácil acceso ayuda a la movilidad urbana de: ciudadanos turistas y migrantes.

Por otro lado, de manera específica este proyecto permite concluir que:

Es útil contar con herramientas digitales que permitan procesar la información del transporte público en formato GTFS de manera eficiente. Además, se debe reconocer la importancia de centralizar y estandarizar la información de las diferentes rutas de transporte en este formato (GTFS); ya que, permite integrar datos, reducir costos de investigación e implementación; y, construir sistemas con mayores beneficios para los usuarios.

De igual manera este proyecto reconoce la importancia de los modelos matemáticos para la resolución de problemas. En el caso de esta investigación se pudo resolver un problema de movilización urbana, a través de grafos y la aplicación del algoritmo de Dijkstra.

Este proyecto también demuestra la importancia de la interfaz gráfica, que conecta al usuario con el sistema y los datos procesados. Esta interfaz gráfica es parte fundamental del proyecto porque permite a los usuarios:

- Ingresar datos como: Punto de inicio y punto de destino. De manera rápida y sin complicación.
- Obtener resultados ágiles, eficientes y confiables sobre su movilización.
- Interpretar los resultados de manera gráfica y sistemática con la información de las diferentes rutas de transporte en las que debía realizar su movilización en el sector Centro Norte de Quito.

Por otro lado, este proyecto comprueba la importancia y la necesidad de validar el funcionamiento del sistema implementado a través de pruebas usabilidad con distintos perfiles de usuario. Estas pruebas de usabilidad permitieron recabar datos importantes sobre:

- La utilidad del sistema y sus componentes interactivos.
- La interacción del sistema con diferentes tipos de usuario.
- La claridad y calidad de los datos entregados por el sistema.
- Mejoras que se podrían implementar en proyectos futuros.

Los usuarios que probaron la utilidad del sistema reconocieron la importancia de contar con un sistema de planificación de rutas inclusivo que permita a toda persona ejercer la libre movilidad en la ciudad. A partir de este antecedente es necesario señalar la influencia de los sistemas digitales de planificación de rutas en la mejora de la calidad de vida de las personas frente a la sabiduría popular y datos dispersos, que resultan de difícil interpretación para los usuarios.

Finalmente, como conclusiones adicionales, se agrega que:

- El sistema implementado es totalmente escalable y puede alimentarse de un mayor número de datos para ampliar la cobertura del servicio a toda la ciudad de Quito y sus parroquias aledañas.
- El sistema implementado sirve como paradigma; y, puede ser replicado en cualquier ciudad alrededor del mundo que presente necesidades similares, independientemente del contexto de urbanización; puesto, que el algoritmo de Dijkstra se lo puede aplicar a cualquier grafo con "n" número de nodos.
- También es indispensable señalar que el algoritmo de Dijkstra está en la capacidad de entregar más de una alternativa de viaje al usuario, puesto que recorre todo el grafo en busca de las rutas más óptimas.

- Para finalizar, se debe recalcar en la importancia de mantener el estándar de calidad en los archivos GTFS; ya que, de ellos depende el buen funcionamiento del sistema y la información proporcionada al usuario.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] «Diagnóstico de la movilidad en el distrito metropolitano de Quito para el plan metropolitano de desrrollo territorial,» Municipio del Distrito Metropolitano de Quito, Quito, 2014.

[2] E. Sanchez, D. Chacón y S. Garrica, «Ordenanza Metropolitana que regula la implementación de los sitemas inteligentes de transporte, en el sistema metropolitano de transporte público de pasajeros del Distrito Metropolitano de Quito,» Secretaria de Movilidad, Quito, 2015.

[3] N. Correll, Introduction to Autonomous Robots, CreateSpace Independent Publishing Platform, 2015, pp. 73-77.

[4] M. Yan, «MIT MATHEMATICS,» 08 01 2014. [En línea]. Available: http://math.mit.edu/~rothvoss/18.304.3PM/Presentations/1-Melissa.pdf. [Último acceso: 05 09 2017].

[5] Google, «FeedValidator,» Transit Feed, 8 Octubre 2014. [En línea]. Available: https://github.com/google/transitfeed/wiki/FeedValidator. [Último acceso: 30 Julio 2017].

[6] M. Masse, REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces, Sebastopol, California: O'Reilly Media, Inc., 2011.

[7] B. Handwerk, «National Geographic News,» National Geographic, 13 03 2008. [En línea]. Available: https://news.nationalgeographic.com/news/2008/03/080313-cities.html. [Último acceso: 20 08 2017].

[8] *Espacio requerido en la ciudad para transportar el mismo número de personas por tres medios diferentes: coche, bus y bicicleta.* [Art]. Departamento de planificación urbana de la ciudad de Münster, 2001.

[9] J. Houghton, J. Reiners y C. Lim, «Transporte inteligente. Cómo mejorar la movilidad en las ciudades,» IBM Global Business Services, Somers, 2009.

[10] Gobierno Autónomo Descentralizado Municipal Del Cantón Cuenca, «Guía de Buses,» [En línea]. Available: http://www.cuenca.gov.ec/?q=node/576. [Último acceso: 25 07 2017].

[11] Empresa pública Metropolitana de Transporte de Pasajeros de Quito (EPMTPQ), «Plan estratégico EPMTPQ 2013-2017,» 2013. [En línea]. Available: http://www.trolebus.gob.ec/lotaipadjuntos/2013/Plan%20Estrategico%20EPMTPQ%202013%20ver2013.pdf. [Último acceso: 2017].

[12] U. D. o. T. «Multimodal Trip Planner System,» Economic and Industry Analysis Division, Cambridge, 2011.

[13] Agencia de promoción económica CONQUITO, «Mi Bus UIO una alternativa emprendedora que promueve el uso del transporte público,» 13 Abril 2016. [En línea]. Available: http://www.conquito.org.ec/mi-bus-uio-una-alternativa-emprendedora-que-promueve-el-uso-del-transporte-publico/. [Último acceso: 17 Septiembre 2017].

[14] J. Wong, «Use of the General Transit Feed Specification (GTFS) in transtir performance measurement,» Georgia Institute of Technology, Georgia, 2013.

[15] Google, «Referencia de la Especificación general de feeds de transporte público,» Google Transit, 20 junio 2012. [En línea]. Available: https://developers.google.com/transit/gtfs/reference. [Último acceso: 28 junio 2017].

[16] A. Antrim y S. J. Barbeau, «The many uses of GTFS data–opening the door to transit and multimodal applications,» *Location-Aware Information Systems Laboratory at the University of South Florida,* p. 4, 2013.

[17] Mi Bus, «Editor GTFS,» Mi Bus, 05 2016. [En línea]. Available: http://editor.mibusuio.com:8080/. [Último acceso: 30 06 2017].

[18] Transitscreen, «Transitscreen real-time display of transportation,» [En línea]. Available: https://transitscreen.com/.

[19] K. Thulasiraman y M. N. S. Swamy, Graphs: Theory and Algorithms, Wiley, 2011.

[20] J. L. Gross y J. Yellen, Handbook of Graph Theory, CRC Press, 2004.

[21] Universidad de Valladolid, Apuntes de Grafos, Valladolid: Departamento de informática, 2006.

[22] T. Cormen y D. Balkcom, «Khan Academy,» Dartmouth Computer Science , 2017. [En línea]. Available: https://www.khanacademy.org/computing/computer-science/algorithms/graph-representation/a/representing-graphs. [Último acceso: 2017].

[23] T. Cormen, C. Leiserson y R. Rivest, Introduction to Algorithms, 3ra Edición ed., Massachusetts: The MIT Press, 2009.

[24] A. García, Inteligencia artificial : fundamentos, práctica y aplicaciones, RC Libros, 2012, p. 296.

[25] RestApiTutorial.com, «HTTP Status Codes,» 30 01 2017. [En línea]. Available: http://www.restapitutorial.com/httpstatuscodes.html. [Último acceso: 3 08 2017].

[26] Telegrafo, «Sistema identificacion de calles y predios,» Diario El Telegrafo, 2013. [En línea]. Available: http://www.eltelegrafo.com.ec/images/eltelegrafo/Quito/2013/28-2-13-sistema-identificacion.jpg. [Último acceso: 10 09 2017].

[27] P. W. Pimploi Tirastittam, «Public Transport Planning System by Dijkstra Algorithm: Case Study Bangkok Metropolitan Area,» *Engineering and Technology International Journal of Social,* vol. 8, nº 1, pp. 54-59, 2014.

[28] C. McCracken, «UX Mastery,» 20 Noviembre 2016. [En línea]. Available: https://uxmastery.com/beginners-guide-to-usability-testing/. [Último acceso: 13 Octubre 2017].

ANEXO I

Links con la información sobre rutas del sector centro Norte del Distrito Metropolitano de Quito, obtenidas del OpenStreetMaps.

Routename	OSM Route ID	OSM Link
Atacazo - Mariana de Jesus	6425203	https://www.openstreetmap.org/relation/6425203
Mariana de Jesus - Atacazo	6425214	https://www.openstreetmap.org/relation/6425214
San Juan - San Roque	6425663	https://www.openstreetmap.org/relation/6425663
San Roque - San Juan	6425714	https://www.openstreetmap.org/relation/6425714
Estadio Olimpico - San Fernando - Peralta	6427312	https://www.openstreetmap.org/relation/6427312
Peralta - San Fernando - Estadio Olimpico	6427455	https://www.openstreetmap.org/relation/6427455
Rocio de Guamani - UCE - Estadio Olimpico	6427514	https://www.openstreetmap.org/relation/6427514
Estadio Olimpico - UCE - Rocio de Guamani	6427560	https://www.openstreetmap.org/relation/6427560
Andalucia - Camal	6429615	https://www.openstreetmap.org/relation/6429615
Camal - Andalucia	6429934	https://www.openstreetmap.org/relation/6429934
Quitus Colonial - Universidad Central	6422912	https://www.openstreetmap.org/relation/6422912
Universidad Central - Quitus Colonial	6423404	https://www.openstreetmap.org/relation/6423404
Jardin del Valle - Monjas - Las Casas	6423418	https://www.openstreetmap.org/relation/6423418
Las Casas - Monjas - Jardin del Valle	6424084	https://www.openstreetmap.org/relation/6424084
San Juan de Calderon - Ejido	6433415	https://www.openstreetmap.org/relation/6433415
Ejido - San Juan de Calderon	6433451	https://www.openstreetmap.org/relation/6433451
Jardin - Nuevo Amanecer - Luz y Vida	6433493	https://www.openstreetmap.org/relation/6433493
Marin - Planada	6423347	https://www.openstreetmap.org/relation/6423347
Planada - Marin	6425068	https://www.openstreetmap.org/relation/6425068
Ejido - Carapungo - Ecuador - Bicentenario	6425209	https://www.openstreetmap.org/relation/6425209

Bicentenario - Ecuador - Carapungo - Ejido	6425303	https://www.openstreetmap.org/relation/6425303
Comuna - Obrero Independiente	6432815	https://www.openstreetmap.org/relation/6432815
Obrero Independiente - Comuna	6432894	https://www.openstreetmap.org/relation/6432894
Eloy Alfaro - Obrero Independiente - Rosapamba	6432901	https://www.openstreetmap.org/relation/6432901
Rosapamba - Obrero Independiente - Eloy Alfaro	6433442	https://www.openstreetmap.org/relation/6433442
Primavera - Balcon del Valle	6433495	https://www.openstreetmap.org/relation/6433495
Balcon del Valle - Primavera	6433609	https://www.openstreetmap.org/relation/6433609
San Vicente - Ejido	6433683	https://www.openstreetmap.org/relation/6433683
Ejido - San Vicente	6433691	https://www.openstreetmap.org/relation/6433691
Pululahua - Miraflores	6433688	https://www.openstreetmap.org/relation/6433688
Miraflores - Pululahua	6433717	https://www.openstreetmap.org/relation/6433717
Pululahua - Panecillo	6422871	https://www.openstreetmap.org/relation/6422871
Panecillo - Pululahua	6422591	https://www.openstreetmap.org/relation/6422591
Marin - Carcelen Bajo	6422603	https://www.openstreetmap.org/relation/6422603
Carcelen Bajo - Marin	6422819	https://www.openstreetmap.org/relation/6422819
Ejido - Brasilia - Carcelen - CarcelenBajo	6422901	https://www.openstreetmap.org/relation/6422901
CarcelenBajo - Carcelen - Brasilia - Ejido	6423092	https://www.openstreetmap.org/relation/6423092
Ejido - La Josefina	6423263	https://www.openstreetmap.org/relation/6423263
La Josefina - Ejido	6423301	https://www.openstreetmap.org/relation/6423301
Cochapamba Sur - Cochapamba Norte - Don Bosco	6423407	https://www.openstreetmap.org/relation/6423407
Don Bosco - Cochapamba Norte - Cochapamba Sur	6423463	https://www.openstreetmap.org/relation/6423463
Carcelen - Marin	6423480	https://www.openstreetmap.org/relation/6423480
Marin - Carcelen	6423664	https://www.openstreetmap.org/relation/6423664
La Pulida - Ejido	6423675	https://www.openstreetmap.org/relation/6423675
Ejido - La Pulida	6423686	https://www.openstreetmap.org/relation/6423686
Ejido - 23 de Junio	6423695	https://www.openstreetmap.org/relation/6423695

23 de Junio - Ejido	6423748	https://www.openstreetmap.org/relation/6423748
Condado - Marin	6425174	https://www.openstreetmap.org/relation/6425174
Marin - Condado	6425254	https://www.openstreetmap.org/relation/6425254
Parlamento - Ciudadela Alegria - Pueblo Blanco	6425290	https://www.openstreetmap.org/relation/6425290
Pueblo Blanco - Ciudadela Alegria - Parlamento	6425381	https://www.openstreetmap.org/relation/6425381
Trebol - Bueno Aires	6425385	https://www.openstreetmap.org/relation/6425385
Bueno Aires - Trebol	6425448	https://www.openstreetmap.org/relation/6425448
Eden - San Pablo	6425501	https://www.openstreetmap.org/relation/6425501
San Pablo - Eden	6425536	https://www.openstreetmap.org/relation/6425536
Orquideas - Hospital Metropolitano	6425744	https://www.openstreetmap.org/relation/6425744
Hospital Metropolitano - Orquideas	6425777	https://www.openstreetmap.org/relation/6425777
La Gasca - Barrionuevo	6426018	https://www.openstreetmap.org/relation/6426018
Barrionuevo - La Gasca	6426036	https://www.openstreetmap.org/relation/6426036
Marin - Quintana	6427285	https://www.openstreetmap.org/relation/6427285
Quintana - Marin	6427329	https://www.openstreetmap.org/relation/6427329
Comite del Pueblo (Zona 11) - Marin	6422605	https://www.openstreetmap.org/relation/6422605
Marin - Comite del Pueblo (Zona 11)	6422825	https://www.openstreetmap.org/relation/6422825
Camal - Hipodromo - E. Olimpico	6422948	https://www.openstreetmap.org/relation/6422948
E. Olimpico - Hipodromo - Camal	6423281	https://www.openstreetmap.org/relation/6423281
Ejido - Velasco	6423298	https://www.openstreetmap.org/relation/6423298
Velasco - Ejido	6423444	https://www.openstreetmap.org/relation/6423444
Roldos - Jardin	6423457	https://www.openstreetmap.org/relation/6423457
Jardin - Roldos	6423501	https://www.openstreetmap.org/relation/6423501
Camal - El Inca	6425263	https://www.openstreetmap.org/relation/6425263
El Inca - Camal	6425323	https://www.openstreetmap.org/relation/6425323
Luz y Vida - Nuevo Amanecer - Jardin	6425504	https://www.openstreetmap.org/relation/6425504
Marin - Comite del Pueblo	6425971	https://www.openstreetmap.org/relation/6425971
Comite del Pueblo - Marin	6426000	https://www.openstreetmap.org/relation/6426000
Llano Grande - Terminal Rio Coca	6426052	https://www.openstreetmap.org/relation/6426052

Terminal Rio Coca - Llano Grande	6426058	https://www.openstreetmap.org/relation/6426058
Atucucho - Dos Puentes - Terminal La Magdalena	6427390	https://www.openstreetmap.org/relation/6427390
Terminal La Magdalena - Dos Puentes - Atucucho	6427434	https://www.openstreetmap.org/relation/6427434
Curiquingue - Roldos - Dos Puentes - La Magdalena	6427499	https://www.openstreetmap.org/relation/6427499
La Magdalena - Dos Puentes - Roldos - Curiquingue	6427517	https://www.openstreetmap.org/relation/6427517
Magdalena - 2 Puentes - Estadio de Liga - Ofelia	6427555	https://www.openstreetmap.org/relation/6427555
Ofelia - Estadio de Liga - 2 Puentes - Magdalena	6427581	https://www.openstreetmap.org/relation/6427581
Cima de la Libertad - Bellavista	6427651	https://www.openstreetmap.org/relation/6427651
Bellavista - Cima de la Libertad	6427676	https://www.openstreetmap.org/relation/6427676
San Gabriel - Quitumbe	6428397	https://www.openstreetmap.org/relation/6428397
Quitumbe - San Gabriel	6429416	https://www.openstreetmap.org/relation/6429416
San Vicente de las Casas - Quitumbe	6429446	https://www.openstreetmap.org/relation/6429446
Quitumbe - San Vicente de las Casas	6429477	https://www.openstreetmap.org/relation/6429477
La Dolorosa - Estadio Olimpico	6425186	https://www.openstreetmap.org/relation/6425186
Estadio Olimpico - La Dolorosa	6425213	https://www.openstreetmap.org/relation/6425213
San Francisco de Asis - Floresta	6425292	https://www.openstreetmap.org/relation/6425292
Floresta - San Francisco de Asis	6425319	https://www.openstreetmap.org/relation/6425319
Hospital Carolo - Registro Civil - Floresta	6425351	https://www.openstreetmap.org/relation/6425351
Floresta - Registro Civil - Hospital Carolo	6425551	https://www.openstreetmap.org/relation/6425551
La Isla - Las Casas	6425515	https://www.openstreetmap.org/relation/6425515
Las Casas - La Isla	6425595	https://www.openstreetmap.org/relation/6425595
Estadio Olimpico - Chillogallo - Cristo Rey	6425657	https://www.openstreetmap.org/relation/6425657
Cristo Rey - Chillogallo - Estadio Olimpico	6427669	https://www.openstreetmap.org/relation/6427669

Mariana de Jesus - Chillogallo - La Esperanza	6427287	https://www.openstreetmap.org/relation/6427287
La Esperanza - Chillogallo - Mariana de Jesus	6427287	https://www.openstreetmap.org/relation/6427287
Buenaventura de Ch. - Libertad - P. Artigas	6427673	https://www.openstreetmap.org/relation/6427673
P. Artigas - Libertad - Buenaventura de Ch.	6427694	https://www.openstreetmap.org/relation/6427694
Estacion Rio Coca - 6 de Julio	6425178	https://www.openstreetmap.org/relation/6425178
6 de Julio - Estacion Rio Coca	6425211	https://www.openstreetmap.org/relation/6425211
Estacion Rio Coca - Agua Clara	6425215	https://www.openstreetmap.org/relation/6425215
Agua Clara - Estacion Rio Coca	6425238	https://www.openstreetmap.org/relation/6425238
Estacion Rio Coca - Comite del Pueblo	6425265	https://www.openstreetmap.org/relation/6425265
Comite del Pueblo - Estacion Rio Coca	6425321	https://www.openstreetmap.org/relation/6425321
Estacion Rio Coca - Cumbaya	6425344	https://www.openstreetmap.org/relation/6425344
Cumbaya - Estacion Rio Coca	6425597	https://www.openstreetmap.org/relation/6425597
Estacion Rio Coca - La Luz	6425647	https://www.openstreetmap.org/relation/6425647
La Luz - Estacion Rio Coca	6425666	https://www.openstreetmap.org/relation/6425666
Estacion Rio Coca - Monteserrin	6427241	https://www.openstreetmap.org/relation/6427241
Monteserrin - Estacion Rio Coca	6427271	https://www.openstreetmap.org/relation/6427271
Estacion Rio Coca - Nayon	6427277	https://www.openstreetmap.org/relation/6427277
Nayon - Estacion Rio Coca	6427307	https://www.openstreetmap.org/relation/6427307
Estacion Rio Coca - Zambiza	6427309	https://www.openstreetmap.org/relation/6427309
Zambiza - Estacion Rio Coca	6427362	https://www.openstreetmap.org/relation/6427362
Estacion La Y - Cotocollao	6433746	https://www.openstreetmap.org/relation/6433746
Cotocollao - Estacion La Y	6433756	https://www.openstreetmap.org/relation/6433756
Ruminahui - Estacion La Y	6433757	https://www.openstreetmap.org/relation/6433757
Estacion La Y - Ruminahui	6435307	https://www.openstreetmap.org/relation/6435307
Estacion La Y - Kennedy	6435311	https://www.openstreetmap.org/relation/6435311

Kennedy - Estacion La Y	6435317	https://www.openstreetmap.org/relation/6435317
Estacion La Y - Comite del Pueblo	6435320	https://www.openstreetmap.org/relation/6435320
Comite del Pueblo - Estacion La Y	6435323	https://www.openstreetmap.org/relation/6435323
Estacion La Y - Laureles	6435329	https://www.openstreetmap.org/relation/6435329
Laureles - Estacion La Y	6423482	https://www.openstreetmap.org/relation/6423482

ORDEN DE EMPASTADO