https://repository.uaeh.edu.mx/revistas/index.php/investigium/issue/archive

Boletín Científico INVESTIGIUM de la Escuela Superior de Tizayuca

InvESTigium

Publicación semestral, Vol. 8, No. Especial (2022) 102-110

ISSN: 2448-4830

Primer Congreso de Investigación e Innovación en Tendencias Globales
26 – 28 de octubre de 2022

Planificador de viajes de transporte público utilizando el estándar GTFS

Public transport trip planner using the GTFS standard

Alicia Martínez-Rebollar[a], Carlos A. Ruíz-Gutierrez[b], Hugo Estrada-Esquivel[c], Yasmin Hernández-Pérez[d]

Abstract:

Nowadays, the design and planning of road layouts in modern cities represents a great challenge. In addition, the massive use of public transportation and the current complexity of the different routes that operate in cities has caused citizens to require help when planning a trip. This problem is addressed by various software systems that allow citizens to reach a destination using a private car, such as Waze or Google Maps. Although these systems are of great help to end users, they have the disadvantage of being closed systems owned by a company. This means that it is not possible for developers who wish to create mobility applications to reuse these resources. Currently, many transportation agencies use GTFS (General Transit Feed Specification) data as the standard format for publishing their data. In this paper we present a trip planner, which uses the standard GTFS format. We present this planner using the case study of Mexico City.

Keywords:

Trip planner, GTFS, Public transport, Urban mobility

Resumen:

Hoy en día, el diseño y planeación de los trazados viales en las urbes modernas representa un gran desafío. Además, el uso masivo del transporte público y la actual complejidad de las distintas rutas que operan en las ciudades ha ocasionado que los ciudadanos requieran de ayuda al momento de planificar una ruta. Esta problemática es atendida por diversos sistemas de software que permiten que los ciudadanos puedan alcanzar un destino utilizando auto particular, como es el caso de Waze o Google Maps. Si bien estos sistemas son de gran ayuda para los usuarios finales, tienen el inconveniente de ser sistemas cerrados propiedad de una empresa. Esto ocasiona que no sea posible reutilizar estos recursos por desarrolladores que deseen crear aplicaciones para movilidad. Actualmente, muchas agencias de transporte utilizan los datos GTFS (General Transit Feed Specification) como formato estándar para publicar sus datos. En este artículo se presenta un planificador de viajes, el cual utiliza el formato estándar GTFS. Nosotros presentamos este planificador utilizando el caso de estudio de la Ciudad de México.

Palabras Clave:

Planificador de viajes, GTFS, Transporte público, Movilidad urbana

1. Introducción

Actualmente, las grandes ciudades son consideradas como las principales fuentes económicas del país. Esta consideración se debe a que en las ciudades se pueden encontrar un mayor número de comercios, aglomeraciones industriales y servicios, tales como, transporte, tecnología, telecomunicaciones, etc. [1]. La movilidad en las ciudades es uno de los factores que más ha favorecido el crecimiento económico [2].

a TECNM/CENIDET, https://orcid.org/0000-0002-1071-8599, Email: alicia.mr@cenidet.tecnm.mx

b TECNM/CENIDET, https://orcid.org/0000-0002-6912-8016, email: m21ce066@cenidet.tecnm.mx

c TECNM/CENIDET, https://orcid.org/0000-0002-1466-7581, Email: hugo.ee@cenidet.tecnm.mx

d TECNM/CENIDET, https://orcid.org/0000-0002-8842-0899, Email: yasmin.hp@cenidet.tecnm.mx

Fecha de recepción: 07/10/2022, Fecha de aceptación: 10/10/2022, Fecha de publicación: 05/12/2022

DOI: https://doi.org/10.29057/est.v8iEspecial.9994

Hoy en día, el diseño y planeación de los trazados viales en las urbes modernas representan un gran desafío, el cual puede llegar a constituirse en un problema para el usuario que pretende determinar las rutas de transporte óptimas [3].

El uso masivo del transporte público y la actual complejidad de las distintas rutas que operan en las ciudades ha ocasionado que los ciudadanos requieran de ayudas al momento de planificar una ruta que le ayude a ahorrar tiempo [4]. El sistema de transporte público en la Ciudad de México es una red compleja que integra diferentes tipos de transporte en diferentes rutas que recorren toda la ciudad [5]. En esta red multimodal es muy complicado saber qué combinación de transportes públicos se debe tomar para llegar más rápido a un destino [5]. El uso masivo del transporte público y la actual complejidad de las distintas rutas que operan en las ciudades ha ocasionado que los ciudadanos requieran de ayudas para planificar una ruta que le ayude a ahorrar tiempo [4].

En este artículo se propone un método para la generación de rutas para alcanzar un destino en una ciudad utilizando la red de transporte público. Esta red de transporte tiene que estar especificada en el estándar de especificación general de alimentación en tránsito (General Transit Feed Specification (GTFS), por sus siglas en inglés).

2. Fundamentos Teóricos y estado del arte

2.1 Especificación general de alimentación en tránsito

GTFS (General Transit Feed Specification) es un estándar que define un formato común para los horarios de transporte público y la información geográfica asociada a ellos. El uso de GTFS permite que las empresas de transporte publiquen sus datos de transporte y que los programadores escriban aplicaciones que los consuman [6].

Un feed GTFS se compone de una serie de archivos de texto recopilados en un archivo ZIP. Cada archivo modela un aspecto específico de la información de transporte público: paradas, rutas, viajes y otros datos relacionados con los horarios. La especificación GTFS se puede utilizar para impulsar a los planificadores de viajes, editores de horarios y programadores de diversas aplicaciones [7].

En 2005, con el fin de crear un planificador de viajes de tránsito utilizando la aplicación web Google Maps, Google y TriMet en Portland formularon la Especificación General de Alimentación de Tránsito (GTFS) [8]. En 2007, Google publicó las especificaciones de alimentación de tránsito y animó a las agencias de tránsito a utilizar el formato GTFS para crear datos de tránsito y publicarlos en la web como fuentes abiertas para uso público. El formato GTFS es fácil de crear para las agencias, cómodo para los programadores y lo suficientemente completo para que el público entienda un sistema de tránsito [9].

El formato GTFS ha sido adoptado por el sector del transporte público como estándar para el intercambio de datos de horarios, y cada vez son más las agencias de transporte público que suben datos GTFS a Google. En la actualidad, más de 170 agencias de transporte generan y publican sus horarios en formato GTFS [10].

2.2 Planificadores de viajes

Un sistema de planificación de viajes ayuda a reducir considerablemente el tiempo que comúnmente lleva planificar un viaje ante lo complejo que puede llegar ser una red de transporte público [11]. El sistema de planificación de viajes usa analíticas y algoritmos informáticos para identificar la ruta de entrega más eficiente posible entre todas las opciones existentes y en función de variables como tráfico, distancias entre los puntos de entrega, giros e intersecciones, entre otros. Es importante mencionar que, mientras más paradas tienes, más compleja es la operación y mayor beneficio dará el sistema [11].

Realizar una planeación con un nivel de detalle generalmente le lleva al sistema desde algunos segundos, incluso cuando se trata de una operación que es grande o compleja. Toda esta planificación, sin duda tomaría mucho más tiempo hacerla en forma manual [12].

2.3 Movilidad Urbana

La movilidad se refiere a la sostenibilidad, la seguridad y la eficiencia de las infraestructuras y sistemas de transporte, así como a la accesibilidad local, nacional e internacional en las ciudades [13].

El crecimiento de la población en las ciudades, así como los nuevos hábitos de vida están presionando los sistemas de transporte para que aumenten su capacidad y ofrezcan un servicio más enfocado a los ciudadanos. Por ello, es razonable que otro conjunto de soluciones especialmente relevante sean aquellas que ayudan a gestionar las redes de autobuses y en general los medios de transporte urbanos, mejorando su eficiencia, permitiendo predecir mejor la demanda para optimizar el uso, reduciendo los costes operacionales, aumentando la seguridad y en general, mejorando la experiencia de usuario [13].

2.4 Estado del arte

La movilidad urbana es un tema de gran interés y ha sido abordada desde diferentes perspectivas. A continuación, se presentan algunos de los trabajos en el área de sistemas de información para movilidad en ciudades.

Una propuesta de planificación de viajes es la de [14]. Los autores proponen un algoritmo para encontrar todas las

rutas posibles entre dos puntos cualesquiera haciendo uso de los datos GTFS (General Transit Feed Specification). En la investigación hacen uso de una técnica para reducir la sobrecarga del servidor definiendo una estructura de datos en Redis para almacenar todos los posibles resultados de la búsqueda.

También se han llevado a cabo análisis sobre los datos GTFS [15] donde se observa la conformidad sobre las rutas GTFS y los datos de posicionamiento de los autobuses en varias ciudades. Los datos GTFS se enfrentan a dos problemáticas principalmente, por un lado, versiones desactualizadas de los datos. Algunas agencias de transporte no proporcionan GTFS con la misma frecuencia que cambia el tránsito. Por otro lado, ocurre una discrepancia con los datos de posicionamiento enviados por los autobuses.

Se han realizado propuestas de planificadores con otro enfoque [16]. Este enfoque consiste de dos etapas la predicción de los requisitos de viaje y la planificación de rutas dinámicas. En primer lugar, analizar los comportamientos de viaje de los usuarios para obtener características predictivas. En segundo lugar, los autores diseñaron un algoritmo que genera rutas dinámicas y óptimas con destinos fijos en múltiples autobuses. Está pensado para que funcione como una especie de taxi colectivo donde prioriza la comodidad del usuario al optimizar una ruta a través de predicciones de rutas.

Se han realizado aplicaciones de planificación de rutas de transporte público para dispositivos móviles [17], donde menciona la importancia de tener acceso a la información cuando realizas un viaje o durante el mismo. Tiene como objetivo ser consultado de una manera práctica a través de una aplicación móvil la cual consume datos GTFS que le permite funcionar sin una conexión a internet.

Otro enfoque fue realizado por [18] donde propuso un algoritmo de consulta de la ruta de tránsito del autobús basado en los tiempos de menor transferencia. Se basó en los algoritmos de Dijkstra y el de la hormiga para la consulta de las paradas del autobús. Su objetivo fue la optimización para seleccionar la ruta de viaje del autobús que le permita encontrar los menores tiempos de transferencia y paradas del autobús.

3. Planificador de viajes propuesto

Un sistema de planificación de viajes ayuda a reducir considerablemente el tiempo que comúnmente lleva planificar un viaje ante lo complejo que puede llegar ser una red de transporte público.

Realizar una planeación con un nivel de detalle generalmente le lleva al sistema desde algunos segundos, incluso cuando se trata de una operación que es grande o compleja. Toda esta planificación, sin duda tomaría mucho más tiempo hacerla en forma manual.

En este artículo se presenta nuestra propuesta de generación de rutas de la red de transporte público. Nuestra propuesta está formada de tres principales procesos: (i) La construcción del modelo conceptual GTFS; (ii) Planificador de viajes; (iii) Representación de paradas del transporte público en mapas utilizando la librería Leaflet.

3.1 Construcción del modelo conceptual GTFS

La construcción del modelo conceptual o base de datos GTFS, implica la descarga de los archivos GTFS y la construcción del modelo. A continuación, se detallan cada uno de estos procesos llevados a cabo.

Descarga de Archivos GTFS y sus características

Los archivos GTFS son un conjunto de archivos de texto que contienen información acerca de viajes de transporte local, rutas, paradas, etc. en un formato txt. La adquisición de estos archivos puede realizarse a través de las agencias de tránsito de la ciudad correspondiente. Estas agencias se encargan de subir la información a sus plataformas para su consulta o uso de estos datos abiertos. También pueden descargarse consultando Google Transit [7], la cual incluye una lista de las empresas de transporte público que proporcionan feeds públicos. Los datos GTFS incluyen tablas que son obligatorias, opcionalmente obligatorias y opcionales [19]. Los feeds pueden cambiar en el número de archivos de texto que contengan.

En este artículo se llevó a cabo la adquisición de los datos GTFS de la Ciudad de México a través de la Secretaría de Movilidad (SEMOVI) [20]. Los archivos GTFS obtenidos se muestran en la Tabla 1, la cual muestra 8 archivos de texto plano con la siguiente información: Nombre del archivo, los campos que contiene y la descripción del archivo. En algunos casos, podrían ser más archivos, esto se debe a la flexibilidad que tienen los datos GTFS de incluir más o menos según las necesidades de las agencias o la aplicación.

Modelo conceptual de los datos GTFS

La construcción del modelo conceptual o base de datos a partir de los datos GTFS, se llevó a cabo de la siguiente manera:

Se analizó la información contenida en cada uno de los archivos. Por ejemplo, cada archivo tiene una referencia a los otros archivos relacionados. Por ejemplo, *trips.txt* hace referencia al archivo *routes.txt* utilizando el campo ID de la ruta como clave foránea (atributo: *route_id*) [21]. Esto facilita el procesamiento de estos archivos como tablas en una base de datos.

Las relaciones entre estos archivos se construyen usando el ID de referencia. Esto facilita el procesamiento de estos archivos como tablas en una base de datos.

Tabla 1. Archivos GTFS básicos [19]

Nombre	Campos	Descripción
Agency.txt	agency_id agency_name agency_url	Define las empresas de transporte público que tienen un servicio representado en este conjunto de datos.
Stops.txt	stop_id stop_name stop_lat stop_lon wheelchair_boarding	Define las paradas en las que los vehículos recogen o dejan pasajeros. También indica las estaciones y las entradas de las estaciones.
Trips.txt	trip_id route_id service_id direction_id shape_id	Viajes para cada ruta. Un viaje es una secuencia de dos o más paradas que ocurre durante un período específico.
Routes.txt	route_id route_short_name route_long_name route_type	Rutas de transporte público. Una ruta es un grupo de viajes que se muestra a los pasajeros como un solo servicio.
Calendar.txt	service_id dias_semana star_date	Indica las fechas de servicio especificadas mediante un horario semanal, con fechas de inicio y finalización.
Stop_time.txt	trip_id arrival_time departure_time stop_id stop_sequence	Proporciona las horas en las que un vehículo llega a una parada y sale de ella en cada viaje.
Shapes.txt	shape_id shape_dist shape_lat shape_lon shape_sequence	Define las reglas para asignar las rutas de viaje de los vehículos, también conocido como alineamientos de rutas.
Frequencies.txt	trip_id end_time exact_times headway_sec start_time	Indica el tiempo entre viajes para los servicios basados en intervalos o una representación comprimida de un servicio con horarios fijos.

La base de datos nos permite evitar la duplicidad de información, determinar qué campos son requeridos y que opciones utilizan según la especificación.

La importación de los datos *txt* a un manejador de base de datos puede llevarse a cabo a través de varios gestores de base de datos. En este trabajo de investigación se utilizó MySQL Workbench por ser un software open source. Dentro del gestor se debe crear una nueva base de datos e importar cada uno de los archivos GTFS, con la función “import”.

Para visualizar cada una de las tablas, las cuales corresponden a cada archivo GTFS, se puede realizar de dos formas diferentes. Una forma es realizar la consulta directamente en línea de comandos con la sentencia ´SELECT * FROM´ seguido del nombre de la tabla. Un ejemplo de la consulta es ´SELECT * FROM stops´ donde nos devolverá todos los registros que existe en la tabla paradas. Otra forma es abrir la base de datos desde el gestor como se observa en la Figura 1. En donde se puede analizar los atributos de la tabla gráficamente.

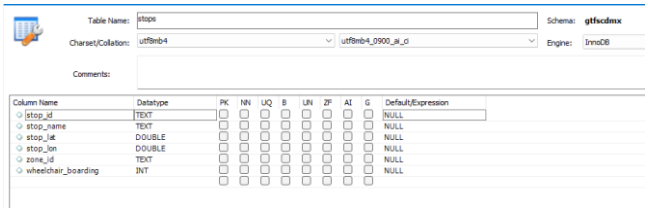

Figura 1 generación de la tabla Stops y sus atributos

Con las tablas creadas a partir de los archivos GTFS, y sus relaciones entre las tablas el modelo conceptual final puede verse en la Figura 2.

Esta base de datos se encuentra lista para ser consultada en la planificación de un viaje.

3.2 Planificador de viajes

Nuestra propuesta de un planificador de viajes se llevó a cabo utilizando el algoritmo Dijkstra el cual encuentra el camino más corto entre dos puntos [21]. Este prototipo consta de dos módulos principales. El módulo de origen-destino y el módulo de planificación de viajes.

El módulo Origen destino

Este módulo es el encargado de obtener la información de la posición de origen en la cual el usuario desea iniciar el viaje, así como el punto destino en el cual se desea llegar. Este módulo permite consumir datos de la base de datos GTFS. En primer lugar, se lleva a cabo una consulta a la tabla stops para mostrarle al usuario la lista de todas las paradas disponibles para que seleccione la parada de inicio, la cual es almacenada en una variable llamada origen.

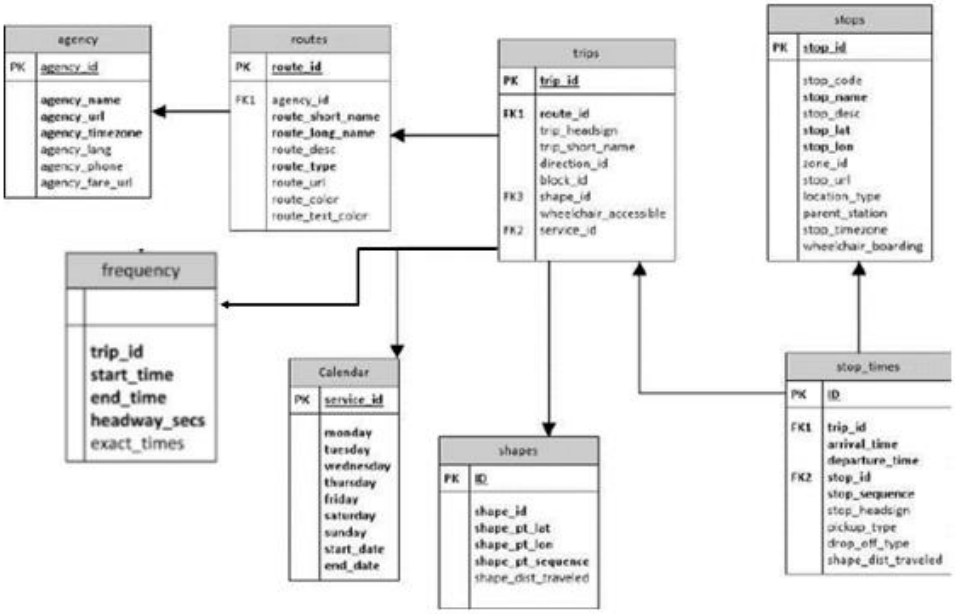

Figura 2 Modelo conceptual de los archivos GTFS

La Figura 3 muestra la instrucción realizada para realizar esta consulta. Realizamos una función llamada ´list´ la cual recibe como parámetro el nombre de la tabla en este caso hacemos referencia a la tabla paradas. Dentro de nuestra función list encontramos la consulta query hacia la base de datos la cual nos realiza el listado de los datos de esa tabla.

```
function list(table){
    return new Promise( (resolve, reject) =>{
        connection.query(`SELECT * FROM ${table}`, (err, data)
```

Figura 3. Consulta a la base de datos para listar las paradas

*Se realiza e*sta misma *consulta* para obtener la información de la parada destino, la información obtenida es almacenada en una variable denominada destino.

Módulo de planificación de viajes

El módulo planificador de viajes se llevó a cabo a través de dos procesos principales: la construcción del grafo, y la implementación del algoritmo para la planificación de viajes. El primer proceso de planificación consiste en construir un grafo que contenga todas las paradas del servicio de transporte público como vértices del grafo. Las distancias entre las paradas actúan como el peso de las aristas. La distancia puede ser en metros o kilómetros, pero siempre números positivos. Para construir el grafo se deben preparar los datos. En una nueva tabla almacenamos todas nuestras paradas en forma ordenada con sus distancias entre cada una de ellas.

Para este ejemplo tomamos un pequeño conjunto que corresponde a las estaciones del metro de la Ciudad de México. La nueva tabla creada nos queda con tres columnas que son 'Parada', 'SigParada' y 'Longitud' como se muestra en la Figura 4.

	Parada	SigParada	Longitud de interestación
0	Pantitlán	Zaragoza	1320.0
1	Zaragoza	Gómez Farías	762.0
2	Gómez Farías	Boulevard Puerto Aéreo	611.0
3	Boulevard Puerto Aéreo	Balbuena	595.0
4	Balbuena	Moctezuma	703.0

Figura 4. Tabla estaciones con los datos GTFS

Para crear el grafo nos ayudamos de Networkx que es una librería hecha en el lenguaje de programación Python [22]. Para formar el grafo hacemos uso de la función '*from_pandas_edgelist*' que se encuentra dentro de Networkx la cual recibe como parámetros un conjunto de datos, un atributo '*source'* y un atributo '*target'* y el atributo 'edge_attr'. Estos atributos vienen especificados en la documentación [23].

Para la construcción del grafo se utilizó el conjunto de datos de todas las estaciones del transporte colectivo metro y a su vez el atributo de distancia entre cada una de las paradas como se muestra en la Figura 5. La implementación para nuestro ejemplo está compuesta por los siguientes elementos:

Asignamos la función a un elemento que para este ejemplo lo nombramos GrafoMetro donde guardaremos

nuestro grafo. Siguiendo con la función le indicamos los parámetros que nos solicita el cual le indicamos como primer elemento nuestro conjunto de datos que llamamos stops. Para el parámetro source que solicita le indicamos el nombre de la columna que en nuestro caso nombramos como 'Parada'. Para el parámetro target que solicita le indicamos el nombre de la columna que en nuestro caso nombramos como 'SigParada' y como último parámetro el cual le asigna el peso a nuestras aristas y que corresponde a nuestra tercera columna de la tabla el cual hace referencia a *Edge_attr*.

Únicamente deberíamos ejecutar la función para la construcción del grafo.

```
GrafoMetro = nx.from_pandas_edgelist(stops,
    source='Parada',
    target='SigParada',
    edge_attr='Longitud de interestación')
```

Figura 5. Función para crear un grafo con Networkx

El segundo proceso consiste en la implementación del algoritmo para la planificación de viajes. El algoritmo de Dijkstra, calcula los caminos más cortos entre un vértice y los demás. Para que el algoritmo pueda realizar un cálculo se deben cumplir ciertos criterios. El primer criterio es que las longitudes de las aristas deben ser números enteros y positivos. Otro criterio es que el grafo debe ser dirigido, es decir, las aristas solamente pueden recorrer en una dirección.

El algoritmo Dijkstra consiste en ir explorando todos los caminos más cortos que parten del vértice origen y que llevan a todos los demás vértices. Cuando se obtiene el camino más corto desde el vértice origen, al resto de vértices que componen el grafo, el algoritmo se detiene. El pseudocódigo de este algoritmo se muestra en la Figura 6.

```
G = (V,E) donde V es el conjunto de vértices y E son aristas
S = es le conjunto de vértices cuyos caminos más cortos
    al origen han sido ya determinados.
V-S Es elresto de vértices
d:  Array de estimaciones de caminos más cortos a los vértices
pr: Array de predecesors para cada vértice

    <inicializar d y pr>
    <Poner S = Ø> // Aún no se a estudiado ningún vértice

        While  (V-S) ≠ Ø //mientras queden vértices sin
                         determinar su camino mínimo al origen
               <Ordenar lo vértices en V-S analizar a la menor
               distancia al origen>
               <Añadir u, el vértice más cercano en V-S, a S>//S = S (u)
                   <Recalcular distancia a todos los vértices
                   adyacentes a u>
```

Figura 6. Pseudocódigo del algoritmo de Dijkstra [24].

Se recorren todos los vértices adyacentes al punto de origen, excepto los vértices marcados. Tomamos como próximo nodo actual el de menor valor al nodo adyacente (puede hacerse almacenando los valores en una cola de prioridad) y volvemos al paso 3 mientras existan nodos no marcados como visitados. Una vez terminado el algoritmo, el arreglo estará completamente lleno.

Como resultado de la ejecución el algoritmo nos arrojará como resultado un arreglo de elementos con todas las paradas que nos llevan desde el origen hasta el destino.

3.3 Representación de paradas del transporte público en mapas utilizando la librería Leaflet

Para representar las paradas del transporte público en un mapa se hace uso de algunas herramientas como la cartografía del mapa o librerías para agregar capas al mapa. En esta propuesta de investigación, para representar las rutas se utiliza Openstreetmap y la librería Leaflet. Leaflet es la principal biblioteca JavaScript de código abierto para mapas interactivos [25].

Estas librerías consultan la base de datos GTFS para poder representar las paradas, por lo que se hace uso de las tablas 'stops' y 'shapes'. Estas dos tablas contienen la información de todas las paradas y puntos geográficos de la ciudad a la que pertenece el GTFS. Los datos de las tablas mencionadas se deberán convertir a un formato GeoJson que es un formato estándar abierto diseñado para representar elementos geográficos sencillos. Con ayuda de una librería como es gtfs2geojson realizamos esta conversión. Para implementarlo solo debemos indicar el objeto a tratar como se muestra en la Figura 7.

```
gtfs2geojson.stops(fs.readFileSync('stops.txt', 'utf8'), function(result)
    res.json(result);
```

Figura 7. Función para convertir datos a GeoJson

Una vez que se realizó la conversión al nuevo formato se puede utilizar el plugin *leaflet-ajax*. Este plugin permite hacer una llamada AJAX a un servicio que retorne un documento tipo JSON y cargar la respuesta en un mapa. Para utilizar la Leaflet solo la debemos importar a nuestro proyecto como un script más.

Para representar una ruta dentro de un mapa se debe hacer uso de la librería Leaflet. Le debemos proporcionar las coordenadas donde deseamos colocar el marcador. Las coordenadas están comprendidas por latitud y longitud. Una vez proporcionadas las coordenadas se especifica el tipo de marcador que se quiere colocar. La función 'marker' recibe los parámetros de longitud y latitud seguido de addTo para visualizarlo en el mapa.

```
L.marker([latitud, longitud], {opción1, opción2...})
```

Figura 8. Función para agregar marcadores en un mapa

Los marcadores pueden ser predeterminados o personalizados. Además de especificar el marcador es posible agregar algún tipo de información adicional con un

cuadro de texto que al dar clic sobre el marcador se muestre la información.

Por último, solo haría falta realizar la llamada en el documento html para visualizar nuestra ruta en un navegador. Abrimos nuestro navegador y en la barra de direcciones le indicamos la ruta del archivo que deseamos visualizar típicamente siempre suele ser predeterminada como http://localhost:3000, pero esto se puede modificar si lo desea.

4. Casos de Estudio

El caso de estudio utilizado en este trabajo de investigación fue los datos GTFS de la Ciudad de México. Esta información es constantemente actualizada, la última actualización se llevó a cabo el 28 de septiembre del 2022 [20].

La visualización de todas las paradas del servicio público de la Ciudad de México se muestra en la Figura 9. Nosotros utilizamos las librerías Leaflet y Openstreetmap para representar estas paradas en el mapa de la Ciudad.

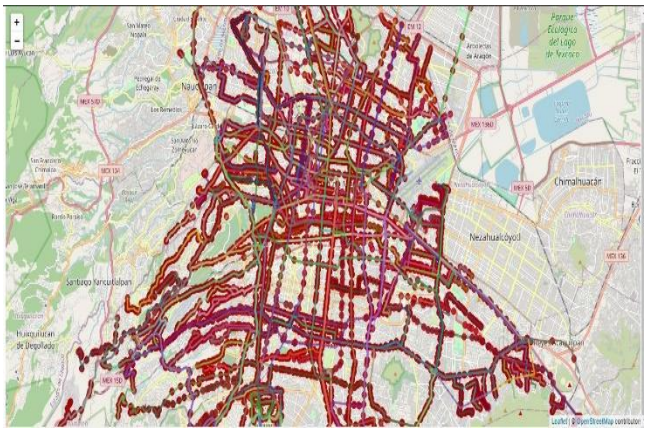

Figura 9. Representación de las paradas de la ciudad en un mapa

Una de las consultas que se hicieron para la evaluación de nuestra propuesta fue consultar las paradas del metro de la Ciudad de México:

Punto de Origen: Cuatro caminos (línea 2)

Punto destino: Deportivo Oceanía (línea B)

La Figura 10 muestra las líneas del metro en donde se encuentran señaladas estos puntos origen 1 – destino 1.

El resultado de esta consulta al algoritmo implementado para la planificación del viaje dio como resultado 20 estaciones del metro (Figura 11):

Linea 2: Cuatro caminos, Panteones, Tacuba, Cuitlahuac, Popotla, Colegio Militar, Normal, San Cosme, Revolución, Hidalgo, Bellas Artes

Linea 8: Garibaldi,

Línea B: Lagunilla, Tepita, Morelos, San Lazaro, Ricardo Flores Magón, Romero Rubio, Ocenania, Deportivo Oceanía

Figura 10. Líneas del trasporte colectivo metro

```
['Cuatro Caminos',
 'Panteones',
 'Tacuba',
 'Cuitláhuac',
 'Popotla',
 'Colegio Militar',
 'Normal',
 'San Cosme',
 'Revolución',
 'Hidalgo',
 'Bellas Artes',
 'Garibaldi',
 'Lagunilla',
 'Tepito',
 'Morelos',
 'San Lázaro',
 'Ricardo Flores Magón',
 'Romero Rubio',
 'Oceanía',
 'Deportivo Oceanía']
```

Figura 11. Ruta 1 obtenida con el algoritmo Dijkstra

La representación de este viaje en el mapa de la ciudad se muestra en la Figura 12. Los marcadores se personalizaron con la imagen de las estaciones.

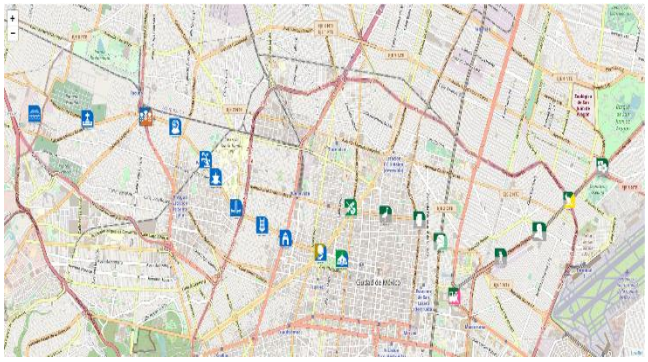

Figura 12. Representación de la ruta 1 en un mapa usando Leaflet

Otra de las consultas que se hicieron para la evaluación de nuestra propuesta fue consultar las paradas del metro de la Ciudad de México:
Punto de Origen: Bellas Artes (línea 2)
Punto destino: Lazaro Cardenas (línea B)
La Figura 10 muestra las líneas del metro en donde se encuentran señaladas estos puntos origen 2 – destino 2.
El resultado de esta consulta al algoritmo implementado para la planificación del viaje dio como resultado 8 estaciones del metro (Figura 13):
Linea 2: Bellas Artes, (Linea 2, línea 8)
San juan de Letrán, Salto del Agua, (Línea 1)
Isabel la Católica, Pino Suárez, San Antonio Abad (Línea 2)
Chabacano, Lazaro Cárdenas (Línea 9)

```
['Bellas Artes',
 'San Juan de Letrán',
 'Salto del Agua',
 'Isabel la Católica',
 'Pino Suárez',
 'San Antonio Abad',
 'Chabacano',
 'Lázaro Cardenas']
```

Figura 13. Ruta 2 obtenida con el algoritmo Dijkstra

La representación de este viaje en el mapa de la ciudad se muestra en la Figura 14.

Figura 14. Representación de la ruta 2 en un mapa usando leaflet

5. Conclusiones y trabajos futuros

Las ciudades se han convertido en una enorme fuente de datos. Uno de los temas que más relevancia ha tenido en las últimas fechas son los datos asociados a movilidad urbana, que permite que los ciudadanos puedan acceder a bienes y servicios pero que también genera múltiples problemas a los ciudadanos. Uno de los temas principales de la ciudad es la movilidad de los ciudadanos utilizando el sistema de transporte público, el cual es una intrincada red de sistemas interconectados. Las ciudades exponen sus datos de rutas de transporte en estándares como es GTFS. Sin embargo, estos datos no siempre son utilizados por desarrolladores. En este artículo presentamos un planificador de rutas que permite que un ciudadano pueda encontrar la ruta más corta para alcanzar un destino en la ciudad utilizando la especificación GTFS como base para obtener los diferentes sistemas de transporte, paradas y destinos. El artículo presenta la forma en la cual un desarrollador puede replicar la obtención de datos GTFS de una ciudad y replicar el procesamiento de datos para generar un planificador de rutas. Actualmente nos encontramos generando una versión de código abierto del planificador de rutas, de tal forma que éste planificador pueda ser utilizado en forma abierta por cualquier desarrollador que quiera integrarlo en aplicaciones de movilidad inteligente. Este planificador se desarrolla utilizando tecnologías de código abierto de tal forma que no existan restricciones para su uso y modificación.

Referencias

[1] redeuslac.org, «Economía Urbana - REDEUSLAC,» 2021. [En línea]. Available: https://redeuslac.org/lineas-de-accion/economia-urbana/#:%7E:text=Las%20ciudades%2C%20que%20concentran%20capi-tal,productividad%20y%20la%20capacidad%20innovadora. [Último acceso: 2 Octubre 2022].

[2] R. Montezuma, «Ciudad y transporte: la movilidad urbana,» *Cepal*, 2003.

[3] C. Velásquez, «Espacio público y movilidad urbana: Sistemas Integrados de Transporte Masivo (SITM),» pp. 21-90, 2015.

[4] A. Bull, Congestión de tránsito el problema y cómo enfrentarlo, 206 ed., Cepal, 2003.

[5] D. López, A. Lozano, H. González, A. Guzmán y F. Maldonado, «Hiperpuma: Sistema Multimodal de Información al Viajero,» de *La movilidad en la Ciudad de México Impactos, conflictos y oportunidades*, IG, 2018, pp. 119-151.

[6] A. Colque, R. Valdivia, M. Navarrete y S. Aracena, «Un sistema de información geográfico para el transporte público basado en el estandar GTFS realtime,» *Scielo*, vol. 29, pp. 51-62, 2019.

[7] Google Transit, «Descripción general de GTFS estáticas | Transporte público estático,» 2021. [En línea]. Available: https://developers.google.com/transit/gtfs?hl=es. [Último acceso: 2 octubre 2022].

[8] M. Catala, «Expanding the Google Transit Feed Specification to Support Operation and Planning,» *National Center for Transit Research*, 2011.

[9] Google transit, «Guías GTFS,» 2021. [En línea]. Available: https://developers.google.com/transit/gtfs/guides. [Último acceso: 28 Septiembre 2022].

[10] J. Wong, R. Landon, W. Kari y H. Regan, «Open Transit Data: State of the Practice and Experiences from Participating Agencies in the United States,» *Transportation Research Broad*, 2013.

[11] Quadminds, «¿Qué es un Sistema de Planificación de Rutas?,» 1 Junio 2022. [En línea]. Available: https://www.quadminds.com/blog/sistema-de-planificacion-de-rutas/. [Último acceso: 4 Octubre 2022].

[12] B. Sánchez, «6 beneficios de un sistema de planificación de rutas,» 22 Junio 2022. [En línea]. Available: https://www.netlogistik.com/es/blog/6-beneficios-que-brinda-un-sistema-de-planificacion-de-rutas. [Último acceso: 4 Octubre 2022].

[13] Fundación Telefónica, «Smart Cities: un primer paso hacia la internet de las cosas,» *Ariel*, 2011.

[14] A. Vagner, «Trip planning algorithm for GTFS data with NoSQL structure to improve the performance,» *Jatit*, vol. 99, nº 10, p. 10, 2021.

[15] A. Queiroz, V. Santos, D. Nascimento y C. Pires, «Conformity Analysis of GTFS Routes and Bus Trajectories,» *UFCG*, pp. 1-6, 2019.

[26] gtfs.org, «GTFS Schedule Reference,» 2021. [En línea]. Available: https://gtfs.org/schedule/reference/. [Último acceso: 4 Octubre 2022].

[16] X. Kong, M. Li, T. Tang, K. Tian, M. Moreira y F. Xia, «Shared Subway Shuttle Bus Route Planning Based on Transport Data Analytics,» *IEEE Transactions on Automation Science and Engineering*, vol. 15, pp. 1507-1520, 2018.

[17] T. Szincsák y A. Vágner, «Public transit schedule and route planner application for mobile devices,» *Eger*, pp. 153-161, 2015.

[18] W. Li, X. Chen y B. Yang, «Bus Travel Transit Path Query Algorithm Based on Ant Algorithm,» *Third International Conference on Genetic and Evolutionary Computing*, nº 11106229, pp. 665-669, 2009.

[19] Hmong, «Especificación general de feeds de tránsito HistoriayAplicaciones,» 2021. [En línea]. Available: https://hmong.es/wiki/General_Transit_Feed_Specification. [Último acceso: 2 Octubre 2022].

[20] SEMOVI, «Secretaría de Movilidad (SEMOVI),» 28 Septiembre 2022. [En línea]. Available: https://datos.cdmx.gob.mx/dataset/gtfs#:~:text=El%20GTFS%20est%C3%A1tico%20es%20un,y%20planificaci%C3%B3n%20de%20viajes%20multimodales.. [Último acceso: 4 Octubre 2022].

[21] Codingame, «El algoritmo de Dijkstra - Los caminos más cortos con el algoritmo de Dijkstra,» 2020. [En línea]. Available: https://www.codingame.com/playgrounds/7656/los-caminos-mas-cortos-con-el-algoritmo-de-dijkstra/el-algoritmo-de-dijkstra. [Último acceso: 2 Octubre 2022].

[22] A. Lima, «Introducción a las redes sociales usando NetworkX en Python – Acervo Lima,» 2021. [En línea]. Available: https://es.acervolima.com/introduccion-a-las-redes-sociales-usando-networkx-en-python/. [Último acceso: 2 Octubre 2022].

[23] Networkx, «networkx.org,» 2022. [En línea]. Available: https://networkx.org/documentation/stable/reference/generated/networkx.convert_matrix.from_pandas_edgelist.html#networkx.convert_matrix.from_pandas_edgelist. [Último acceso: 25 Septiembre 2022].

[24] R. Beltran y M. Pérez-Ramírez, «Simulador ROV multiplayer para escenarios petroleros submarinos,» *Research in Computing Science*, p. 74, 2017.

[25] M. Lewin, «Working with Base Layers - Leaflet.js Succinctly Ebook,» 22 Marzo 2016. [En línea]. Available: https://www.syncfusion.com/succinctly-free-ebooks/leafletjs/working-with-base-layers. [Último acceso: 2 Octubre 2022].