



**Escuela Superior
de Ingeniería y Tecnología**
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Sistema inteligente de apoyo al conductor para la reducción de los colapsos de tráfico

*Intelligent driver support system to reduce traffic
collapses*

Edgar Figueroa González

La Laguna, 8 de septiembre de 2021

D. **Iván Castilla Rodríguez**, con N.I.F. 78.565.451-G profesor Contratado Doctor adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

D. **Rafael Arnay del Arco**, con N.I.F. 78.569.591-G profesor Contratado Doctor adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor

C E R T I F I C A (N)

Que la presente memoria titulada:

“Sistema inteligente de apoyo al conductor para la reducción de los colapsos de tráfico”

Ha sido realizada bajo su dirección por D. **Edgar Figueroa González**, con N.I.F. 43.389.520-C.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 8 de septiembre de 2021

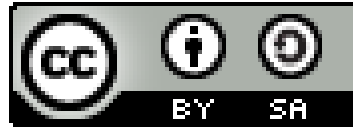
Agradecimientos

A mi familia y a mi pareja por el apoyo incondicional y por mantenerme en calma en los momentos de estrés.

A Iván Castilla y Rafael Arnay por tutorizarme y motivarme en todo momento durante el desarrollo de este proyecto.

A todas las personas que, de algún modo u otro, me han llevado al lugar donde estoy hoy.

Licencia



© Esta obra está bajo una licencia de Creative Commons Reconocimiento 4.0 Internacional.

Resumen

Las retenciones kilométricas que sufren los miles de personas que, cada mañana, circulan por la autopista norte se han convertido en uno de los principales problemas de movilidad por carretera de Tenerife. Dada la repercusión que ha presentado dicha problemática, el objetivo principal de este trabajo será ofrecer un sistema inteligente de apoyo al conductor que sirva de ayuda para minimizar el tiempo de los desplazamientos hacia los grandes núcleos poblacionales. Para ello, se ha desarrollado una interfaz, mediante el uso de simuladores y el estudio de diversos datos, que ofrece a los conductores un horario aproximado de salida, en función de sus puntos de origen y la hora a la que desean llegar al destino. A su vez, los resultados obtenidos sirven de base para posibles líneas futuras de investigación con las que se desee seguir aportando pequeños granos de arena a una cuestión que concierne a toda una población.

Palabras clave: TF-5, SUMO, simulación, sistema inteligente de apoyo, tráfico.

Abstract

The kilometre-long traffic jams experienced by the thousands of people who travel along the northern motorway every morning have become one of the main problems of road mobility in Tenerife. Given the repercussions of this problem, the main objective of this work is to offer an intelligent driver support system to help minimise the time taken to travel to large population centres. To this end, an interface has been developed, through the use of simulators and the study of various data, which offers drivers an approximate departure time, depending on their points of origin and the time at which they wish to arrive at their destination. At the same time, the results obtained serve as a basis for possible future lines of research with which to continue contributing small grains of sand to an issue that concerns an entire population.

Keywords: TF-5, SUMO, simulation, Intelligent Support System, traffic.

Índice general

1. Introducción	1
1.1. Motivación	2
1.2. Antecedentes	2
1.3. Objetivos	4
2. Desarrollo	5
2.1. Descripción del proyecto	5
2.2. Tecnologías	6
2.3. Simulación	9
2.3.1. Primeros pasos con SUMO	9
2.3.2. Importación de matrices O/D	9
2.3.3. Puntos de origen y destino en la TF-5	12
2.3.4. Ejecución de la simulación	14
2.3.5. Salidas de la simulación	15
2.4. Procesamiento de datos	17
2.5. Algoritmo de cálculo de intervalo de salida	20
2.6. Pruebas unitarias	22
3. Funcionamiento	23
3.1. Solicitar intervalo de salida	23
3.2. Visualización del intervalo obtenido	25
3.3. Despliegue con Docker	25
3.3.1. Dockerizando el Back-End	26
3.3.2. Dockerizando el Front-End	27
3.3.3. Unión de backend y frontend con Docker-Compose	27
4. Conclusiones y líneas futuras	30
5. Summary and future work	32
6. Presupuesto	34
7. Apéndice A	35
Repositorio	35

Índice de figuras

Figura 1. Esquema general de funcionamiento	5
Figura 2. Definición de un fichero TAZ	10
Figura 3. Ejemplo de un fichero OD	10
Figura 4. Fichero de configuración OD2TRIPS	10
Figura 5. Archivo de configuración duarouter	11
Figura 6. Archivo .sumocfg	11
Figura 7. Mapa de la autopista TF-5	12
Figura 8. Paso 1 en la ejecución de la simulación	14
Figura 9. Paso 2 en la ejecución de la simulación	14
Figura 10. Simulación	15
Figura 11. Fichero de salida de una simulación de 6 a 7	16
Figura 12. Información franja horaria de 6 a 7	16
Figura 13. Datos recogidos de cada vehículo con el tiempo total del trayecto	17
Figura 14. Valores obtenidos en el intervalo 6:00 - 6:15 para el destino 96	18
Figura 15. Datos del intervalo para el destino elegido por el usuario	19
Figura 16. Algoritmo que calcula el intervalo de salida	20
Figura 17. Diagrama de flujo del algoritmo de cálculo de intervalo de salida	21
Figura 18. Código fuente de las pruebas unitarias	22
Figura 19. Ejecución de las pruebas unitarias	22
Figura 20. Aspecto visual de la interfaz	23
Figura 21. Solicitud de intervalo de salida	23
Figura 22. Lugar de salida	24
Figura 23. Lugar de destino	24
Figura 24. Franja horaria prevista para llegar al destino	24
Figura 25. Alerta recibida con el intervalo de salida	25
Figura 26. Información de los trayectos solicitados	25
Figura 27. Dockerfile del contenedor del servidor	26
Figura 28. Dockerfile del contenedor del frontend	27
Figura 29. Fichero docker-compose.yml	28
Figura 30. Backend a la espera	29
Figura 31. Web desplegada con Docker-Compose	29

Índice de tablas

Tabla 1. Cronograma de las actividades a realizar en el proyecto	4
Tabla 2. Puntos de origen	13
Tabla 3. Puntos de destino	13
Tabla 4. Presupuesto	34

Capítulo 1

Introducción

Las retenciones de tráfico se han convertido en un suplicio cotidiano para los conductores que cada mañana se dirigen a la zona metropolitana de Tenerife por la Autopista Norte (TF-5). Los horarios comprendidos entre las 06:00 y las 08:30 de la mañana son los encargados de dejar atrapados a los miles de personas que se disponen a acudir a su lugar de trabajo o estudio. Las cifras corroboran el caos circulatorio existente y Santa Cruz de Tenerife, según el *TomTom Traffic Index* [1], es la tercera ciudad de España con más atascos. Asimismo, la duración de los recorridos se cuadruplica cuando se recorre en las horas punta. Como botón de muestra, el tramo entre Tacoronte y Santa Cruz de Tenerife se suele completar en 15 minutos, estando en condiciones normales, mientras que si sufres una de estas retenciones puedes llegar a tardar una hora de trayecto [2].

Nos encontramos ante una situación que cada año parece ir en aumento y no solo en Tenerife, sino en todo el archipiélago. Un estudio confirma que Canarias tiene una media altísima de 821 vehículos por cada millar de habitantes. Por su parte, Tenerife es la isla con más problemas ya que supera en 3,5 puntos la media nacional en cuanto a densidad de tráfico. Estos números tan elevados han llevado al archipiélago a ser comparado con otros territorios nacionales e incluso internacionales. Como prueba de ello se encuentra la declaración del coordinador de Izquierda Unida en Canarias, Ramón Trujillo, en la cual ha asegurado que “*si Canarias fuera un país, sería el sexto del mundo con más coches en relación a su población*”[3].

Las autoridades han planteado sobre la mesa varias soluciones que ayuden a reducir estos colapsos, pero la mayoría no han podido ser implementadas hasta el momento. El presidente de FEPECO (Federación Provincial de Entidades de la Construcción de Santa Cruz de Tenerife), Óscar Izquierdo, opina que “*el futuro no existe, cuando el presente no avanza*”, por lo que aboga por implementar carreteras eficientes, seguras y sostenibles. A su juicio, la vía variante del municipio de San Cristóbal de La Laguna es “*un primer paso para empezar a acabar con las colas de la TF-5, sin olvidarnos de que la autopista norte aún requiere un tercer carril para desahogar el tráfico*” [4].

Se trata de un problema de congestión de tráfico que llevan sufriendo los tinerfeños más de 25 años y para el que, hasta ahora, no existe una solución a la vista. Por su parte, el tráfico es una cuestión que nos concierne como sociedad y más allá de las posibles estrategias que puedan llevar a cabo las administraciones públicas,

nosotros, como ciudadanos, también podemos tomar la iniciativa y buscar recursos que sirvan para el beneficio de la ciudadanía.

Dentro de las posibles soluciones a los atascos, considero esencial apoyarnos en la tecnología. Hoy en día, el avance de la misma y nuestra adaptación a ella es evidente, por lo que utilizarla como recurso para solventar un problema de esta índole se vuelve una iniciativa interesante y pertinente. Por estos motivos, en esta línea de investigación se quiere desarrollar un sistema de recomendación de franjas horarias de salida para los conductores de la autopista TF-5 con el objetivo de reducir los picos más altos de tráfico que ocasionan colas y retenciones.

El sistema consultará datos de sus propios usuarios para predecir el estado del tráfico y recomendar horas de salida de forma que se intente optimizar el tiempo de llegada al destino. Se hará uso de la simulación para llevar a cabo las recomendaciones y simular diferentes escenarios posibles.

1.1 Motivación

Esta propuesta de Trabajo de Fin de Grado busca aportar una vía de escape a las retenciones matinales en la TF-5. Es un tema del que se ha hecho eco en reiteradas ocasiones a través de los medios de comunicación de masas y a pesar de las repercusiones y los inconvenientes que causan a toda la población de la isla, a día de hoy se continúa sin tener una solución clara al problema. Resulta paradójico que Tenerife se promocione hacia el exterior como un destino en el que poder descansar mientras que un problema de congestión de tráfico repercute a 200.000 personas cada día, según el Cabildo Insular de Tenerife.

La trascendencia del asunto a nivel regional ya hace la temática de este proyecto atractiva, sin embargo, a ello le podemos añadir que durante mi etapa como estudiante universitario he vivido en primera persona dicha problemática, convirtiendo así la asistencia a clase en toda una odisea. Por ello, este TFG se me ha presentado como una oportunidad para contribuir a resolver un inconveniente que afecta a todas aquellas personas que, como yo, residen en el norte de Tenerife. El poder ofrecer una solución que sirva de mejora para minimizar las colas kilométricas a las que nos enfrentamos cada vez que debemos desplazarnos a las zonas metropolitanas, ya sea para trabajar, estudiar o realizar algún trámite, me parece un reto bastante interesante a afrontar.

1.2 Antecedentes

Como antecedentes a este Proyecto de Trabajo de Fin de Grado se han llevado a cabo varios estudios que abordan el mismo tema y comparten un mismo objetivo: hacer uso de simuladores para gestionar el tráfico en situaciones de congestión.

Este es uno de los métodos más utilizados para analizar y buscar soluciones a los problemas del tráfico. Por lo tanto, lo más común en este tipo de proyectos es realizar simulaciones en distintas franjas horarias, comprobando el comportamiento de la autopista y los datos resultantes para analizar dónde se encuentra el problema o si el estudio llevado a cabo está siendo el correcto.

Por un lado, nos encontramos ante un TFG realizado por un compañero de la Escuela Superior de Ingeniería y Tecnología, que trata de aliviar la congestión del tráfico en la rotonda del Padre Anchieta, en La Laguna. El proyecto consiste en estudiar la instalación de semáforos en la rotonda, optimizando la duración de las fases de estos gracias a un algoritmo evolutivo [5]. En mi opinión, la idea es muy buena e interesante, no obstante considero que dicho estudio no supone una mejora sustancial para el tránsito de vehículos en la rotonda. Las congestiones de tráfico no tienen su punto de origen en la propia rotonda, ya que la mayoría de las retenciones que se producen dentro de ésta derivan de los colapsos en todo el tramo de la autopista norte e inmediaciones.

Además del mencionado, existe otro trabajo relacionado con el tema, realizado también por un compañero de la escuela, que, a través de simulaciones de la autopista TF-5 y su posterior análisis del comportamiento de la misma, plantea la solución de incluir semáforos inteligentes en las entradas de la autopista, así como en los carriles de baja velocidad de la misma [6]. Con ello, busca minimizar el impacto social, económico y natural. Este proyecto se asemeja más a lo que estamos buscando en el trabajo actual porque su objetivo concuerda con el nuestro: desarrollar una propuesta tecnológica que, de alguna manera, ayude a gestionar el tráfico y trate de reducir los colapsos en situaciones de congestión.

Dados los estudios previamente realizados sobre esta misma temática, la presente propuesta de TFG se centrará en abordar la problemática existente en la TF-5, en vez de focalizar nuestra investigación en un solo punto, como ha sido el caso del trabajo centrado en el Padre Anchieta. De la misma manera, estos antecedentes nos han servido como base, pues con ellos hemos podido identificar los resultados obtenidos con cada propuesta y analizar si realmente su implementación supondría una mejora al problema existente. A su vez, conocer las dificultades o ventajas que han tenido mis compañeros con los diferentes softwares de uso ha agilizado el proceso de selección de herramientas para el proyecto.

A partir de aquí, se plantea la posibilidad de suavizar los picos de tráfico en las horas punta del día. Se tratará de buscar soluciones para reducir el colapso de vehículos y la intensidad de las colas. Todo ello con el objetivo de que el tráfico tome un aspecto más lineal en las franjas horarias donde suelen haber más retenciones en la autopista TF-5.

1.3 Objetivos

Como ya se ha mencionado en apartados anteriores, este proyecto busca diseñar e implementar un sistema inteligente de asistencia a los usuarios de la autopista TF-5 que elija el mejor intervalo horario para su desplazamiento, consiguiendo así una mejora en la reducción de los colapsos ocasionados por el tráfico.

Para el cumplimiento del objetivo general del Trabajo de Fin de Grado, se deben llevar a cabo una serie de actividades, entre las cuales se encuentran:

- **Estudio de herramientas:** Familiarizarse con las herramientas a utilizar y buscar herramientas similares.
- **Análisis y depuración de datos:** Estudiar, entender y clasificar los datos obtenidos.
- **Simulación y calibración:** Realizar una simulación a partir de los datos obtenidos.
- **Aplicación:** Desarrollar una aplicación que sirva de apoyo al conductor, implementando un sistema de franjas horarias de salida.

Asimismo, se ha elaborado un cronograma compuesto por las actividades a realizar durante el proyecto y en el cual se establece la duración del mismo, con la fecha de inicio y final de cada tarea. A través de él, buscamos simplificar la organización del trabajo y nos sirve como hoja de ruta para alcanzar el objetivo propuesto (véase Tabla 1).

Actividades/Semanas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Estudio de las Herramientas	✓	✓	✓	✓													
Análisis y depuración de datos					✓	✓	✓	✓	✓	✓							
Simulación y calibración											✓	✓	✓	✓	✓		
Aplicación																✓	✓

Tabla 1. Cronograma de las actividades a realizar en el proyecto

Capítulo 2

Desarrollo

2.1 Descripción del proyecto

El proyecto se basa en crear una interfaz web programada en Vue.js donde el usuario podrá seleccionar los puntos de salida y llegada y la franja horaria a la que pretende llegar al destino elegido.

Una vez seleccionados los campos, el usuario pulsará sobre el botón para que el sistema calcule una aproximación acerca del intervalo de salida. Cuando se pulsa el botón, el sistema automáticamente se conecta a un servidor Websocket. En el servidor es donde se realiza la parte principal del procesamiento de datos. Inicialmente, se recogen los parámetros seleccionados por el usuario y se procede a leer y procesar la información con las salidas de las simulaciones. Seguidamente, dependiendo de los valores introducidos, el programa calcula una franja horaria de salida y automáticamente se la envía como resultado a la interfaz para su posterior visualización. El proceso de selección de ese resultado se explicará en profundidad en los siguientes apartados.

A continuación, en la Figura 1 se aprecia un esquema con el funcionamiento del sistema. Podemos diferenciar varias capas: por un lado el front-end, la capa de visualización, conectada con el servidor, que a su vez está conectado y tiene acceso a la información contenida en la capa de simulación.

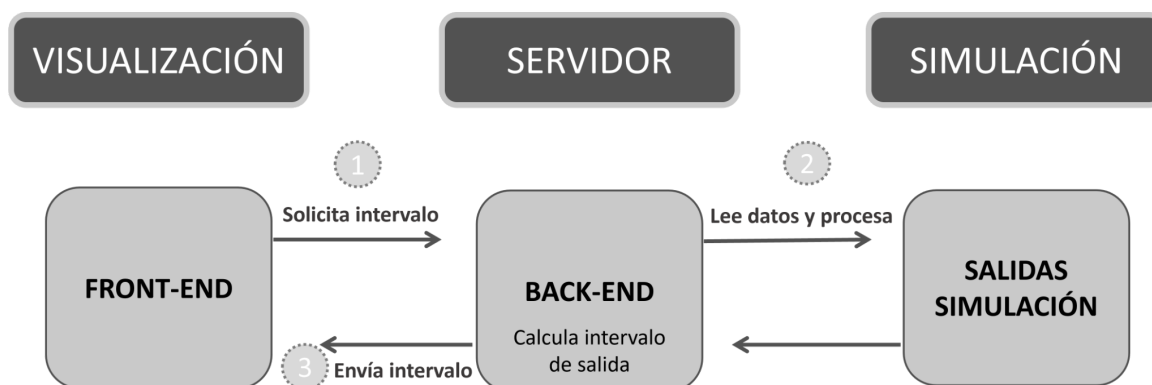


Figura 1. Esquema general de funcionamiento

2.2 Tecnologías

En este apartado se verán las herramientas utilizadas en el proyecto y otras que en un principio se llegaron a utilizar pero finalmente fueron descartadas en el proceso de selección.

SUMO

Simulation of Urban MObility (SUMO) es un paquete de simulación de tráfico multimodal, microscópico, portable y de código abierto diseñado para funcionar con grandes redes. Cuenta con una considerable fuente de documentación y con herramientas muy potentes que permiten realizar simulaciones de alto nivel, rendimiento y eficacia [7]. Por estas razones, se hizo uso de él antes que otras herramientas de simulación. Se utilizó en el proyecto para simular el estado de la autopista TF-5 en las franjas horarias donde más retenciones hay por el tráfico.

AnyLogic

AnyLogic es una herramienta que incluye todos los métodos de simulación más comunes hoy en día [8]. No se eligió por varias razones, una de ellas porque no es open source, aunque tiene una versión gratuita que permite simular hasta una hora. Otra de las razones por la cual no fue elegida fue porque en un proyecto anterior, un compañero la utilizó y tuvo muchos problemas con ella y con algunas de sus librerías. Por lo tanto, se optó por utilizar SUMO.

OpenStreetMap

OpenStreetMap (OSM) es un proyecto colaborativo para crear mapas editables y libres. En lugar del mapa en sí, los datos generados por el proyecto se consideran su salida principal. Los mapas se crean utilizando información geográfica capturada con dispositivos GPS móviles, ortofotografías y otras fuentes libres [9]. En el proyecto de este Trabajo de Fin de Grado se utilizó para exportar el mapa de la autopista TF-5.

JOSM

Java Open Street Map es un software libre de escritorio programado en Java para la edición de datos en el proyecto OpenStreetMap [10]. Finalmente no fue utilizada porque con OpenStreetMap fue suficiente para la exportación del tramo de autopista que queríamos obtener.

NETEDIT

NETEDIT [11] es una aplicación proporcionada por SUMO utilizada para editar redes de carreteras y el flujo de vehículos. Se puede utilizar para crear redes desde cero y modificar todos los aspectos de las redes existentes. Con una potente interfaz de selección y resaltado, también se puede utilizar para depurar atributos de red. Se utilizó en el proyecto principalmente para atribuir identificadores a cada tramo de la autopista.

WebSockets

WebSockets [12] es una tecnología avanzada que hace posible abrir una sesión de comunicación interactiva entre el navegador del usuario y un servidor. Con esta API, se pueden enviar mensajes a un servidor y recibir respuestas controladas por eventos sin tener que consultar al servidor para una respuesta, por ello el motivo de su selección para este proyecto. Esta tecnología ha sido utilizada para el intercambio de información entre la interfaz y el servidor. Desde la interfaz inicialmente se envía la información de los campos seleccionados por el usuario y posteriormente el servidor envía un mensaje con la franja horaria de salida.

JavaScript

JavaScript es un lenguaje de programación ligero, interpretado con funciones de primera clase [13]. Se ha utilizado para implementar el servidor. Además de elegirlo por los amplios conocimientos adquiridos durante la carrera acerca de este lenguaje, también lo he elegido porque posee módulos que dan la posibilidad de leer datos de un fichero de excel o csv de una manera sencilla y óptima.

Vue.js

Vue.js es un framework de JavaScript de código abierto para la construcción de interfaces de usuario y aplicaciones de una sola página [14]. Ha sido utilizado para desarrollar la interfaz.

Hojas de cálculo de Google

Google Sheets es un programa de hoja de cálculo incluido como parte del paquete gratuito de editores de documentos de google basado en la web que ofrece Google. Se ha utilizado para recoger la información de las salidas del simulador. Se han almacenado datos como por ejemplo el número de vehículos que transcurren desde un punto a otro, el tiempo promedio que tardan los vehículos en llegar a su destino, etc.

Python

Lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional [15]. Es cierto que muchos scripts de SUMO están programados en Python, pero finalmente no se hizo uso del lenguaje en el proyecto, porque al ver que con JavaScript ya era posible implementar lo que se tenía en mente para este trabajo, no vi necesario utilizarlo, además de no tener un nivel de conocimiento medio-alto acerca de él.

Git

Git es una herramienta de control de versiones distribuida capaz de trabajar con grandes proyectos a gran velocidad [16].

Mocha - Unit Testing

Mocha es un framework de pruebas para Node JS que se puede ejecutar desde la consola o desde un navegador. Como su propia web indica, permite realizar pruebas asíncronas de manera sencilla [17]. Al ejecutar los test, permite la presentación de informes flexibles y precisos. Se utiliza en el proyecto para comprobar a través de los test que el intervalo de salida que el servidor va a enviar a la interfaz es el correcto. He elegido este framework de pruebas porque es con el que más he trabajado a lo largo de la carrera.

Docker

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos [18]. Fue elegido para este proyecto porque al disponer de dos partes bien diferenciadas, como son back-end y front-end, vi la oportunidad de introducir Docker que es una tecnología muy de mi agrado y la he utilizado durante la carrera.

2.3 Simulación

2.3.1. Primeros pasos con SUMO

Durante las primeras semanas de desarrollo del proyecto, he estado buscando información y documentación acerca del simulador para ver cómo era su funcionamiento y hasta dónde podía llegar con las características que posee. Inicialmente, buscaba generar simulaciones de forma aleatoria, con archivos de ejecución que SUMO proporciona [19], como por ejemplo *“randomTrips.py”* [20], que genera un conjunto de viajes aleatorios para una red dada, eligiendo el borde de origen y destino de manera uniforme al azar o con una distribución modificada en forma de parámetros que se le pasan al programa por línea de comandos.

Después de entender cómo funcionaba exactamente el simulador y según lo que teníamos en mente para el proyecto, llegamos a la conclusión de que había que establecer unos puntos de entrada en la simulación. Es necesario tener en cuenta que la simulación va a depender de factores como, por ejemplo, la velocidad media de los vehículos, la cantidad de vehículos que circulan por la autopista, la duración media de los trayectos de los vehículos, el momento en el que sale cada vehículo, la cantidad de vehículos que han completado su recorrido en un tiempo determinado, etc.

Por lo tanto, podemos decir que en ese momento ya contaba con una pequeña simulación, basada en una franja horaria concreta, donde los puntos de tráfico quizás eran algo irregulares dependiendo de la hora. El siguiente paso fue investigar cómo modificar los parámetros de entrada en la simulación, para que la simulación fuera modificada respecto a los parámetros recibidos de forma externa. Para ello se utilizó las matrices Origen-Destino.

2.3.2. Importación de matrices O/D

SUMO contiene la herramienta OD2TRIPS que puede leer archivos de matrices Origen-Destino[21] en un formato concreto además de convertirlos al formato de archivo de tráfico que más nos convenga para nuestra simulación. OD2TRIPS asume que la matriz O/D se codificará como cantidades de vehículos que conducen de un distrito o zona de asignación de tráfico (TAZ) a otro dentro de un período de tiempo determinado. Para generar la simulación con esta herramienta es necesario disponer de los siguientes archivos de configuración:

- **TAZ (Traffic Analysis Zone) file [22]:** Es un fichero que se describe por su id (un nombre arbitrario) y listas de bordes de origen y destino. En este fichero se encuentran los identificadores asignados a cada una de las salidas del tramo de autopista escogido para este proyecto (véase la Figura 2).

```

<tazs>
  <taz id="<TAZ_ID>" edges="<EDGE_ID> <EDGE_ID> ..."/>
  ... further traffic assignment zones (districts) ...
</tazs>

```

Figura 2. Definición de un fichero TAZ

- **OD Matrix file [23]:** Es un fichero de configuración que enumera cada origen y cada destino junto a la cantidad de vehículos que circulan desde el punto de origen hasta el punto de destino (véase la Figura 3).

```

$OR;D2
* From-Time  To-Time
7.00 8.00
* Factor
1.00
* some
* additional
* comments
      1      1      1.00
      1      2      2.00
      1      3      3.00
      2      1      4.00
      2      2      5.00
      2      3      6.00
      3      1      7.00
      3      2      8.00
      3      3      9.00

```

Figura 3. Ejemplo de un fichero OD

- **OD2TRIPS Config file:** Fichero de configuración .xml al cual se le pasa como entrada el fichero TAZ con la información de los identificadores y el fichero OD con la cantidad de vehículos para cada par de identificadores origen - destino (véase la Figura 4).

```

<configuration>
  <input>
    <taz-files value="TAZ_file.taz.xml"/>
    <od-matrix-files value="OD_file.od"/>
  </input>
</configuration>

```

Figura 4. Fichero de configuración OD2TRIPS

- **Duarouter config file:** El archivo de configuración duarouter toma como entrada el archivo de red y el archivo OD Trips y genera como salida el archivo con las rutas (véase la Figura 5).

```
<configuration>
<!-- The duarouter configuration file takes as input your network and the OD Trips

<input>
  <net-file value="mapatf-5.net.xml"/> <!-- Your SUMO Network File -->
  <route-files value="od_file.odtrips.xml"/> <!-- Your SUMO OD Trips File -->
</input>

<output>
  <output-file value="od_route_file.odtrips.rou.xml"/>
</output>

<report>
  <xml-validation value="never"/>
  <no-step-log value="true"/>
</report>

</configuration>
```

Figura 5. Archivo de configuración duarouter

Estos ficheros son necesarios para utilizar la herramienta OD2TRIPS, pero a parte de estos archivos, existen otros que son imprescindibles para lanzar simulaciones:

- **Archivo de red .net.xml:** Es el fichero de red que contiene todos los tramos que hay en el mapa importado, con sus correspondientes identificadores, las conexiones que hay entre ellos y demás. Es un archivo de extensión larga con las características de cada tramo.
- **Fichero .sumocfg [24]:** Es el archivo que se ejecuta para lanzar la simulación y contiene como entrada el fichero de red y el de rutas. Además, en el mismo archivo es donde se le pasa el tiempo de simulación (véase la Figura 6).

```
<configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
sumoConfiguration.xsd">

  <input>
    <net-file value="mapatf-5.net.xml"/>
    <route-files value="od_route_file.odtrips.rou.alt.xml"/>
  </input>
  <time>
    <begin value="0"/>
    <end value="10800"/>
  </time>

</configuration>
```

Figura 6. Archivo .sumocfg

2.3.3. Puntos de origen y destino en la TF-5

Como mencionamos anteriormente, cada tramo corresponde a un identificador. En este apartado se describen los identificadores asignados a cada una de las salidas de la autopista. El tramo tomado se aprecia en la Figura 7 y empieza en Los Realejos y termina en dirección Santa Cruz de Tenerife.

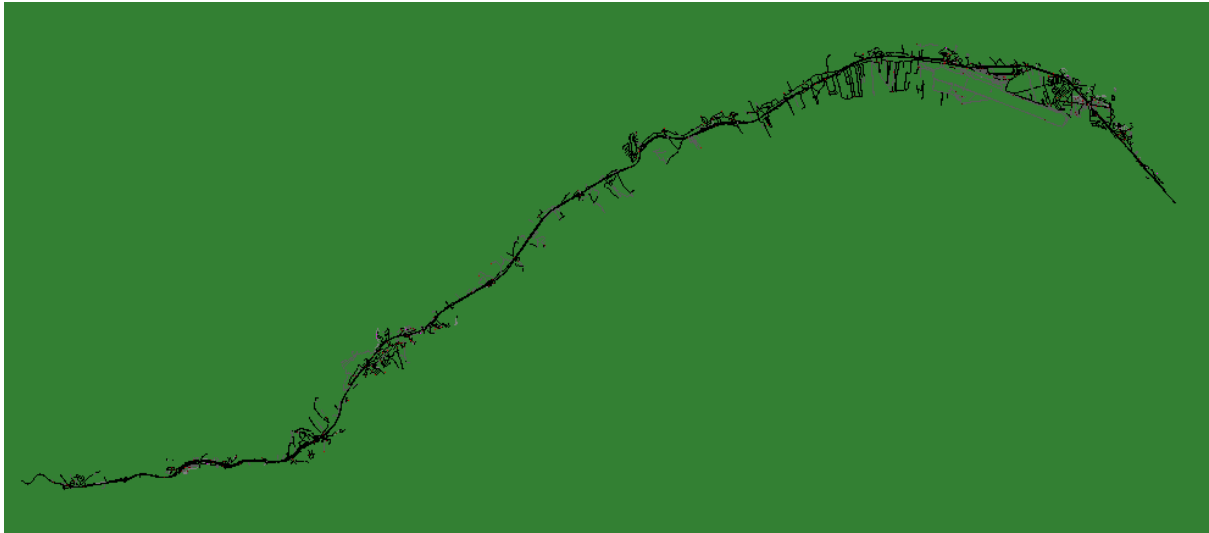


Figura 7. Mapa de la autopista TF-5

Como se ve en la Tabla 2, se han elegido prácticamente todas las salidas como posibles puntos de origen, desde Los Realejos hasta Guajara que sería el último tramo antes de llegar al final de la simulación.

Por otro lado, en la Tabla 3, se pueden apreciar los cuatro puntos de destino elegidos, comprendidos entre el Aeropuerto Tenerife Norte y dirección Santa Cruz. Han sido escogidos estos destinos porque en las horas punta donde más retenciones hay, la mayoría de coches suelen ir dirección La Laguna / Santa Cruz. A partir de ahí, no se añadieron más destinos porque las retenciones en la autopista son ocasionadas en los tramos anteriores, principalmente entre El Sauzal y Padre Anchieta. Al pasar el tramo del Padre Anchieta, la autopista se divide en 3 carriles y ahí la densidad del tráfico es más moderada. Por lo tanto, por el objetivo que tiene este Trabajo de Fin de Grado, no veía lógico añadir muchos más destinos.

Puntos de origen:

IDENTIFICADOR	ORIGEN
1	Los Realejos
3	Higuerita - Los Realejos
12	Alcampo La Villa
17	Polígono San Nicolás
26	El Ramal - La Orotava
36	La Cuesta de la Villa
43	La Quinta - Santa Úrsula
47	Alteza - Santa Úrsula
52	La Victoria de Acentejo
59	La Matanza de Acentejo
70	El Sauzal
76	Tacoronte
82	Los Naranjeros - Tacoronte
92	Guamasa - La Laguna
96	Aeropuerto Tenerife Norte
108	Padre Anchieta - La Laguna
113	Guajara

Tabla 2. Puntos de origen

Puntos de destino:

IDENTIFICADOR	DESTINO
96	Aeropuerto Tenerife Norte
108	Padre Anchieta - La Laguna
113	Guajara
120	Dirección Santa Cruz

Tabla 3. Puntos de destino

2.3.4. Ejecución de la simulación

Tal y como describimos en uno de los apartados anteriores, para generar las simulaciones son necesarios los ficheros de configuración correspondientes a matrices Origen-Destino. Con estos archivos, se procede a realizar la ejecución de las distintas simulaciones.

En primer lugar, hay que obtener el fichero de los viajes con la información de cada uno. Ese fichero tomará como entrada los datos contenidos en TAZ y el fichero OD. Para su creación, se hace uso del comando **od2trips** [25], a través de línea de comandos (véase la Figura 8), pasando los siguientes argumentos como opciones:

- -c: Carga la configuración (od2trips.config.xml)
- -n: Carga el fichero TAZ (TAZ_file.taz.xml)
- -d: Carga la matriz O/D (OD_file.od)
- -o: Escribe en un fichero de salida la información de los viajes (od_file.odtrips.xml)

```
edgar@edgar-X540LA:~/Escritorio/TFG/simulation$ od2trips -c od2trips.config.xml
-n TAZ_file.taz.xml -d OD_file.od -o od_file.odtrips.xml
Loading configuration ... done.
Success.time 3598.95
```

Figura 8. Paso 1 en la ejecución de la simulación

Seguidamente, después de realizar la ejecución, se vuelve a ejecutar otra sentencia (véase la Figura 9) esta vez con el comando **duarouter** [26], para calcular las rutas de los vehículos. Como argumento se le pasa la opción -c, seguido del fichero de configuración duarcfg, que como mencionamos antes, contiene el archivo de red y el de los viajes. Ejecutando esto se creará el de rutas, que se va a pasar como entrada en el fichero que lanza la simulación.

```
edgar@edgar-X540LA:~/Escritorio/TFG/simulation$ duarouter -c duarcfg_file.trips2routes.duarcfg
Loading configuration ... done.
Success.
```

Figura 9. Paso 2 en la ejecución de la simulación

Una vez lanzada la simulación (Figura 10), se procede a recoger las salidas de ésta. Hay que destacar un detalle importante y es que inicialmente en la simulación se desarrolla un calentamiento de coches en la autopista [27], es decir, no se encuentra el tramo de autopista totalmente vacío, hay vehículos que circulan por ella. Esto se hace porque no tendría sentido introducir los vehículos de golpe en la autopista, la simulación no sería del todo real.

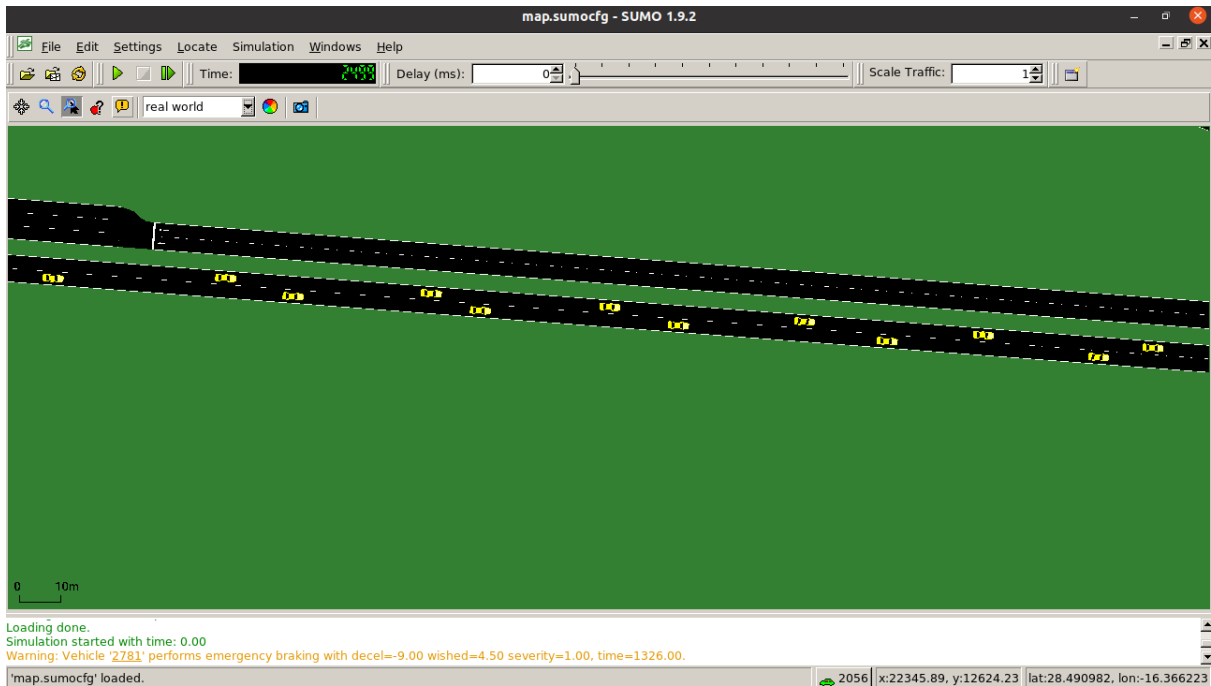


Figura 10. Simulación

Como el objetivo principal de este proyecto es reducir y optimizar los picos más altos de tráfico, se ha decidido establecer como horas punta el intervalo de tiempo comprendido entre las seis y las nueve de la mañana. Por lo tanto, las simulaciones han sido divididas en franjas horarias. Concretamente, tres franjas horarias: de seis a siete, de siete a ocho y de ocho a nueve. Esto quiere decir que se han lanzado simulaciones para cada uno de los intervalos.

Por añadir un ejemplo, en el primer intervalo se lanzan tres, cuatro simulaciones cada quince minutos y así para el resto de las franjas. En total, son muchas las simulaciones lanzadas, pero los resultados que queremos obtener no varían demasiado. Inmediatamente, las salidas de las simulaciones se van recogiendo en un fichero de salida indicado por línea de comandos en la ejecución.

2.3.5. Salidas de la simulación

En el lanzamiento de la simulación, se ejecuta el comando **sumo-gui** [28] seguido de la opción **-c** y el nombre del fichero de configuración **.sumocfg**. Para obtener salidas de la simulación, se utiliza lo que se llama en SUMO **VehRoutes** [29]. Con esta opción, se puede tener la información necesaria para el procesamiento de datos, porque este fichero contiene la salida con las rutas de los vehículos. Para ello se utiliza la opción **--vehroute-output <FILE>**. Por lo tanto, un ejemplo de comando para lanzar una simulación sería:

```
$ sumo-gui -c map.sumocfg --vehroute-output 6-7.xml
```


donde 6-7.xml es el fichero que contiene las salidas de la simulación. En la Figura 11 se muestra el aspecto de la salida:

```
<routes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/r
<vehicle id="1143" depart="24.00" departLane="0" departSpeed="21.06" fromTaz="113" toTaz="120" arrival="89.00">
  <route edges="e113 e114 e115 e116 e117 e118 e119 e120"/>
</vehicle>

<vehicle id="1050" depart="60.00" departLane="0" departSpeed="26.24" fromTaz="113" toTaz="120" arrival="113.00">
  <route edges="e113 e114 e115 e116 e117 e118 e119 e120"/>
</vehicle>
```

Figura 11. Fichero de salida de una simulación de 6 a 7

Con las instancias creadas dentro del fichero, vamos a tener la información necesaria para poder calcular correctamente el tiempo de duración del trayecto, así como el intervalo de salida al que deberá salir el vehículo para llegar a su destino. Los siguientes parámetros son los que se van a utilizar para ello:

- **id:** identificador del vehículo
- **depart:** tiempo en segundos en que se emitió el vehículo a la red
- **arrival:** tiempo en segundos en que el vehículo se retiró de la simulación, es decir, llegó al final de su ruta
- **fromTaz:** identificador del punto origen del cual sale el vehículo
- **toTaz:** identificador del punto destino al cual llega el vehículo

La salida de la simulación está en formato XML, pero para el procesamiento de datos, será más cómodo convertirlo a formato excel o csv. Por lo tanto, vamos a tener un fichero csv para cada franja horaria. Cada fichero contendrá los datos de las rutas en esa franja para cada vehículo, con su identificador, su tiempo de salida y llegada y los puntos origen y destino (véase la Figura 12).

A	B	C	D	E
Vehicle ID	Depart Time	From ID	To ID	Arrival Time
0	2489.00	1	108	5722.00
1	478.00	1	108	2760.00
2	1848.00	1	108	6094.00
3	755.00	1	108	2518.00
4	2951.00	1	108	7256.00
5	56.00	1	108	1262.00
6	1818.00	1	108	4925.00

Figura 12. Información franja horaria de 6 a 7

2.4 Procesamiento de datos

Una vez obtenidos los valores con la información de cada vehículo, se calcula el tiempo del trayecto desde el punto de origen hasta el punto de destino. Para realizar el cálculo, simplemente se toman los valores de salida y llegada en segundos, se restan y se divide entre 60. Con esto obtenemos el total en minutos del viaje de cada coche.

$$\text{Total (minutes)} = \text{Arrival Time} - \text{Depart Time} / 60$$

	A	B	C	D	E	F
1	Vehicle ID	Depart Time	From ID	To ID	Arrival Time	Total (minutes)
2	0	2489	1	108	5722	53,88333333
3	1	478	1	108	2760	38,03333333
4	2	1848	1	108	6094	70,76666667
5	3	755	1	108	2518	29,38333333
6	4	2951	1	108	7256	71,75
7	5	56	1	108	1262	20,1
8	6	1818	1	108	4925	51,78333333
9	7	382	1	108	3143	46,01666667

Figura 13. Datos recogidos de cada vehículo con el tiempo total del trayecto

Como podemos observar en la Figura 13, en una hora puede haber vehículos que hagan el mismo trayecto, pero algunos tardan más y otros tardan menos. Por lo tanto, para obtener más eficacia acerca de los datos, vamos a crear matrices para recoger los datos cada 15 minutos. Las matrices contienen las siguientes variables:

- **Inicio:** Principio del intervalo (segundos)
- **Fin:** Final del intervalo (segundos)
- **Origen:** Identificador del punto origen
- **Destino:** Identificador del punto destino
- **N:** Número de vehículos
- **Promedio:** Media de la duración del trayecto de los vehículos desde el origen X hasta el destino Y (minutos)
- **Intervalos de confianza:** Predicción del 95%

Como son matrices de 15 en 15 minutos, habrán 4 matrices en el intervalo de una hora, es decir, 4 matrices para el intervalo de 6 a 7, otras 4 para el intervalo de 7 a 8 y 4 más de 8 a 9. En cada intervalo de 15 minutos, se calcula el número de vehículos, el intervalo de confianza y el tiempo promedio para cada destino.

Por ejemplo, en el primer cuarto de hora, se obtienen los resultados para los destinos 96 (Aeropuerto Tenerife Norte), 108 (Padre Anchieta), 113 (Guajara) y 120

(Dirección Santa Cruz) y así sucesivamente en los siguientes intervalos de 15 minutos. Como el estudio está enfocado para varias franjas horarias, debemos tener un fichero excel para cada una y deben contener las 4 matrices de 15 minutos. Concretamente, en la Figura 14, se aprecian los valores obtenidos de 6:00 a 6:15, para el destino con identificador 96 (Aeropuerto Tenerife Norte).

Inicio		0			
Fin		900			
Destino		96			
Origen	N	Promedio	IC95% Inferior	IC95% Superior	
1	47	29,58049645	17,38583333	55,47583333	
12	55	26,04909091	16,355	57,68916667	
17	37	27,91621622	13,49833333	60,70166667	
26	34	22,58186275	13,03875	44,94791667	
3	68	29,75931373	17,93	66,12041667	
36	29	13,92298851	10,92833333	21,51833333	
43	25	14,492	10,02333333	21,80333333	
47	23	13,17101449	9,61833333	20,75166667	
52	23	13,05869565	9,02083333	20,00416667	
59	21	9,431746032	7,075	15,15	
70	39	7,011111111	5,52333333	10,62583333	
76	7	5,55	4,914166667	6,931666667	
82	50	3,818	3,1725	4,33875	
92	33	3,887373737	3,47	4,473333333	

Figura 14. Valores obtenidos en el intervalo 6:00 - 6:15 para el destino 96

Al tener los datos de esta manera, es más sencillo para el servidor poder procesar estos datos y utilizarlos para calcular el intervalo de salida según lo que introduzca el usuario desde la interfaz.

Cuando el usuario introduce el origen, destino y la hora de llegada y pulsa el botón, el servidor recibe los valores. Inmediatamente lo que hace es obtener el identificador del punto origen, el identificador del punto destino y convertir la hora para tener el intervalo de inicio y fin. Teniendo esos datos, los busca en el excel. Para ello, se hace uso del paquete de JavaScript llamado “*read-excel-file*” [30].

Con este paquete, el servidor puede leer el contenido del fichero y buscar los valores en él. Utiliza una promesa que retorna un array con toda la información. Lo que hace el servidor es filtrar los valores que recibe del usuario (id destino, inicio y fin del intervalo) para que la función retorne los datos correspondientes a ese intervalo y con el destino concreto (véase la Figura 15).

```

> server@1.0.0 start /home/edgar/Escritorio/Trabajo-Fin-de-Grado/backend
> node server.js

# New client [3cbc4dc5-8d17-462a-89fe-db840411503c] connected
id salida: ID 1: Los Realejos
id destino: ID 113: Guajara
hora: 6:45 - 7:00
Información intervalo: [ [ 'Destino', 113 ],
  [ 'Origen', 'N', 'Promedio', 'IC95% Inferior', 'IC95% Superior' ],
  [ 1, 58, 57.59655172, 51.20166667, 76.215 ],
  [ 108, 7, 1.14047619, 0.7608333333, 1.649166667 ],
  [ 12, 17, 68.23039216, 46.51, 87.23333333 ],
  [ 17, 43, 48.81007752, 38.1125, 58.61416667 ],
  [ 26, 36, 40.35601852, 31.49583333, 51.79583333 ],
  [ 3, 71, 59.15610329, 51.22083333, 76.1 ],
  [ 36, 35, 29.4147619, 20.29666667, 42.09333333 ],
  [ 43, 8, 22.89791667, 19.16333333, 35.735 ],
  [ 47, 14, 30.175, 18.7775, 41.97833333 ],
  [ 52, 31, 34.61182796, 19.01666667, 40.64583333 ],
  [ 59, 21, 26.91984127, 23.04166667, 31.70833333 ],
  [ 70, 39, 17.20555556, 12.88916667, 21.93916667 ],
  [ 76, 37, 13.3490991, 8.716666667, 14.74 ],
  [ 82, 31, 6.152150538, 5.5875, 6.6875 ],
  [ 92, 17, 4.756862745, 4.453333333, 5.463333333 ],
  [ 96, 11, 3.225757576, 2.954166667, 3.575 ],
  [ ] ]

```

Figura 15. Datos del intervalo para el destino elegido por el usuario

Una vez tiene la información, toma el valor promedio del trayecto para el origen que seleccionó el usuario:

$$promedio = resultados.map(x => x).filter(x => x[0] == origen)$$

Por lo tanto, ya teniendo el tiempo medio que tarda un vehículo desde el punto de origen X al punto de destino Y (X e Y introducidos por el usuario), lo siguiente es llamar a la función que calcula el intervalo de salida y seguidamente enviarlo como solución al usuario.

2.5 Algoritmo de cálculo de intervalo de salida

```
1  Funcion resultado <- calcularIntervaloSalida (promedio, hora, inicio_intervalo, final_intervalo)
2  valor_promedio <- promedio * 60
3  valor_promedio <- final_intervalo - valor_promedio
4  intervalo <- [] // Array que contiene los posibles rangos de salida
5  contador <- 0
6  intervalo_salida <- []
7
8  Si valor_promedio < inicio_intervalo Entonces
9    Mientras valor_promedio < inicio_intervalo Hacer
10     inicio_intervalo <- inicio_intervalo - 900
11     contador <- contador + 1
12     intervalo_salida <- (Longitud(intervalo) - contador)
13   Fin Mientras
14   resultado <- intervalo_salida
15 SiNo
16   resultado <- hora
17 Fin Si
18 Fin Funcion
```

Figura 16. Algoritmo que calcula el intervalo de salida

Como muestra la Figura 16, la función de cálculo del intervalo de salida recibe 4 parámetros: el valor promedio, la hora con el intervalo de llegada elegido por el usuario y los valores iniciales y finales del intervalo (puede ser 0, 900, 1800, 2700 o 3600, debido a que el estudio está realizado por franjas de 15 en 15 minutos). La variable **resultado**, va a guardar el intervalo de salida calculado por el algoritmo.

En primer lugar (línea 2), se convierte el tiempo promedio en segundos. Luego, se resta el valor final del intervalo menos el promedio (línea 3). Este número se utiliza para realizar la comparación en el if. Seguidamente, se crea un array (**intervalo**, línea 4) que contiene las posibles horas a las que se puede salir según la opción seleccionada por el usuario. Después se declaran las variables **contador** y **intervalo_salida** que se van a utilizar dentro del bucle while.

El algoritmo consiste en lo siguiente: si el valor obtenido anteriormente es menor que el inicial, quiere decir que está situado entre el inicio y el fin, por lo tanto, entra dentro del if, donde hay un bucle while. El bucle while no terminará hasta que el valor sea mayor o igual al inicial. Mientras sea menor, en cada pasada al inicio se le restará 900 (línea 10), para que pase a ser el final del intervalo anterior. El contador irá incrementando (línea 11).

En **intervalo_salida** (línea 12) se encuentra el posible resultado porque en cada pasada del bucle su valor es **intervalo[intervalo.length - contador]**, es decir, el último rango horario dentro del array de los intervalos. Dependiendo de cuantas veces se recorre el bucle, será el último rango o el anterior y así sucesivamente. Cuando termine el while, el intervalo de salida se guarda en la variable **resultado**.

En el caso de que el valor obtenido en la línea 2 sea mayor o igual al valor inicial, no entrará en el if, sino que directamente el resultado pasa a ser el mismo intervalo que eligió el usuario. Eso quiere decir que ya se encuentra dentro del rango inicial y final, por lo que no es necesario comprobar nada más. El funcionamiento del algoritmo se puede ver reflejado en el siguiente diagrama de flujo:

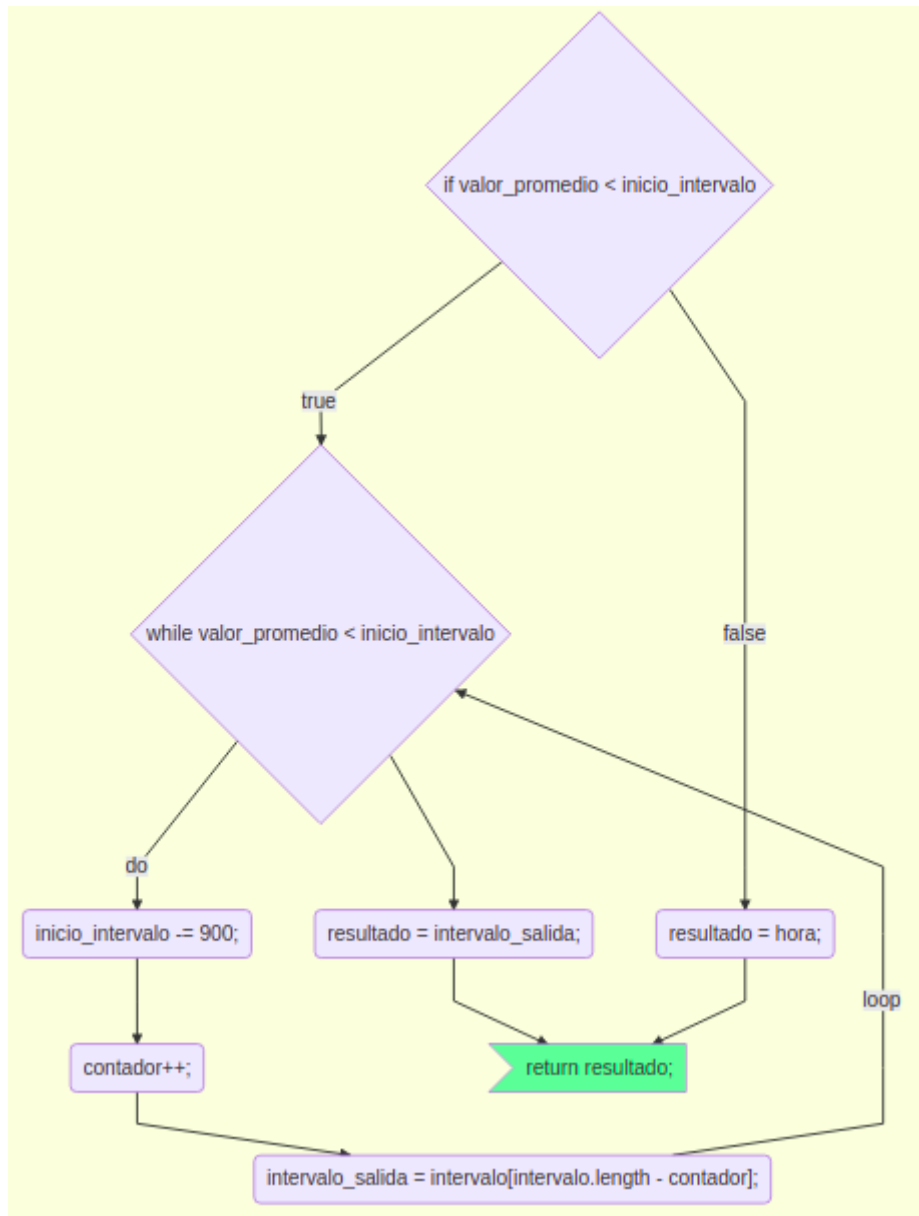


Figura 17. Diagrama de flujo del algoritmo de cálculo de intervalo de salida

Una vez guardado el resultado, ese rango se enviará a través de la conexión websockets al front-end para su posterior visualización. El usuario lo podrá ver de forma inmediata. En el Capítulo 3, se describe cómo es el funcionamiento y cómo se aprecia visualmente desde la interfaz.

2.6 Pruebas unitarias

Uno de los factores importantes dentro del proyecto, después de tener el servidor funcionando y capaz de calcular un intervalo de salida a modo de solución, es comprobar que ese resultado obtenido tiene un sentido y es el correcto. Para ello, se utilizan las pruebas unitarias (Mocha). Se han diseñado los test en la parte del back-end (véase Figura 18).

```
// PRUEBAS UNITARIAS

module.exports.departureInterval = function () {
  it('La Victoria -> Padre Anchieta - La Laguna, Hora prevista de llegada: 7:30 - 7:45 ', done => {
    expect(calcularIntervaloPrueba(32.80888889, '7:30 - 7:45', 1800, 2700)).toEqual('7:00 - 7:15')
    done();
  })
  it('Los Realejos -> Dirección Santa Cruz, Hora prevista de llegada: 7:30 - 7:45', done => {
    expect(calcularIntervaloPrueba(65.20555556, '7:30 - 7:45', 1800, 2700)).toEqual('6:30 - 6:45')
    done();
  })
  it('La Cuesta de la Villa -> Guajara, Hora prevista de llegada: 8:00 - 8:15', done => {
    expect(calcularIntervaloPrueba(14.01715686, '8:00 - 8:15', 0, 900)).toEqual('8:00 - 8:15')
    done();
  })
  it('Los Naranjeros - Tacoronte -> Aeropuerto Tenerife Norte, Hora prevista de llegada: 8:30 - 8:45', done => {
    expect(calcularIntervaloPrueba(4.194444444, '8:30 - 8:45', 1800, 2700)).toEqual('8:30 - 8:45')
    done();
  })
  it('La Matanza -> Guajara, Hora prevista de llegada: 8:15 - 8:30', done => {
    expect(calcularIntervaloPrueba(21.46746032, '8:15 - 8:30', 900, 1800)).toEqual('8:00 - 8:15')
    done();
  })
  it('Polígono San Nicolás -> Padre Anchieta - La Laguna, Hora prevista de llegada: 8:45 - 9:00', done => {
    expect(calcularIntervaloPrueba(33.60263158, '8:45 - 9:00', 2700, 3600)).toEqual('8:15 - 8:30')
    done();
  })
}
```

Figura 18. Código fuente de las pruebas unitarias

En el desarrollo de las pruebas, se pretende comprobar que el resultado que retorna el método *calcularIntervaloPrueba* es el esperado. Para lanzar las pruebas se utiliza el comando *npm run test*, obviamente, situado dentro del directorio del servidor. En la Figura 19 se muestra la ejecución exitosa de las pruebas unitarias.

```
Calcular intervalo de salida
✓ La Victoria -> Padre Anchieta - La Laguna, Hora prevista de llegada: 7:30 - 7:45
✓ Los Realejos -> Dirección Santa Cruz, Hora prevista de llegada: 7:30 - 7:45
✓ La Cuesta de la Villa -> Guajara, Hora prevista de llegada: 8:00 - 8:15
✓ Los Naranjeros - Tacoronte -> Aeropuerto Tenerife Norte, Hora prevista de llegada: 8:30 - 8:45
✓ La Matanza -> Guajara, Hora prevista de llegada: 8:15 - 8:30
✓ Polígono San Nicolás -> Padre Anchieta - La Laguna, Hora prevista de llegada: 8:45 - 9:00

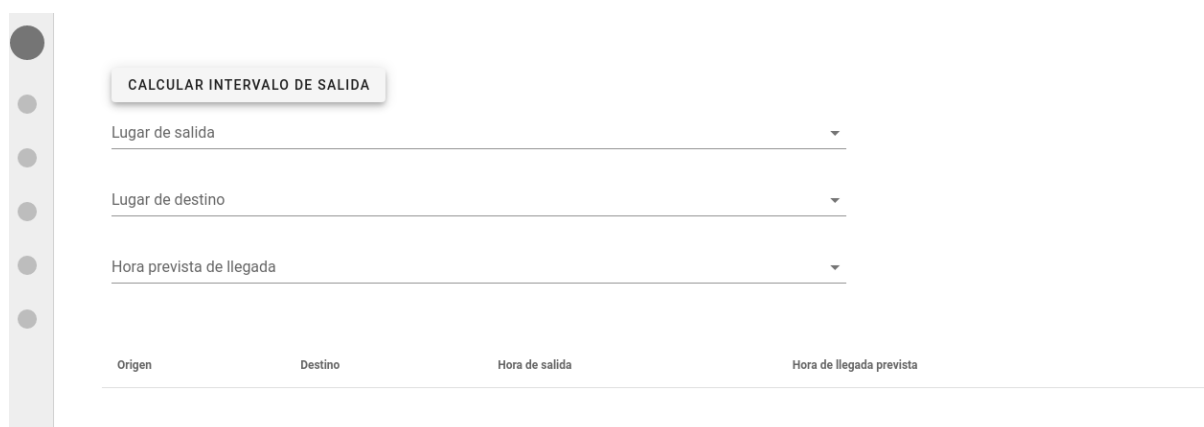
6 passing (12ms)
```

Figura 19. Ejecución de las pruebas unitarias

Capítulo 3

Funcionamiento

Hasta ahora hemos visto que el servidor es capaz de procesar los datos de las salidas de la simulación, leyéndolos y obteniéndolos, según la franja horaria elegida por el usuario. Una vez el servidor haya calculado el intervalo de salida, como mencionamos anteriormente, ese resultado lo envía al front-end para su visualización. Aquí es donde queda plasmado el intervalo de salida, en una sencilla interfaz, donde el usuario puede interactuar, inicialmente solicitando el intervalo y seguidamente visualizando la solución propuesta por el sistema. En la Figura 20 se muestra el aspecto visual que tiene la interfaz:



The screenshot shows a web interface with a vertical sidebar on the left containing five grey circles. The main content area features a button labeled 'CALCULAR INTERVALO DE SALIDA'. Below the button are three dropdown menus: 'Lugar de salida', 'Lugar de destino', and 'Hora prevista de llegada'. At the bottom, there is a table with four columns: 'Origen', 'Destino', 'Hora de salida', and 'Hora de llegada prevista'.

Figura 20. Aspecto visual de la interfaz

3.1 Solicitar intervalo de salida



This screenshot shows the same interface as Figure 20, but with the dropdown menus expanded. The 'Lugar de salida' dropdown is open, showing a list of options. The 'Lugar de destino' and 'Hora prevista de llegada' dropdowns are also open, showing their respective options. The table at the bottom is not visible in this view.

Figura 21. Solicitud de intervalo de salida

El usuario que quiera que el sistema le responda calculando un intervalo de salida (Figura 21), deberá seleccionar el lugar de donde pretende salir (Figura 22) y el lugar hasta donde quiere llegar (Figura 23), además de la franja horaria a la que pretende llegar al destino (Figura 24).

Lugar de salida

- ID 1: Los Realejos
- ID 3: Higuera - Los Realejos
- ID 12: Alcampo La Villa
- ID 17: Polígono San Nicolás
- ID 26: El Ramal - La Orotava
- ID 36: La Cuesta de la Villa

Figura 22. Lugar de salida

Lugar de destino

- ID 96: Aeropuerto Tenerife Norte
- ID 108: Padre Anchieta - La Laguna
- ID 113: Guajara
- ID: 120: Dirección Santa Cruz

Figura 23. Lugar de destino

Hora prevista de llegada

- 6:00 - 6:15
- 6:15 - 6:30
- 6:30 - 6:45
- 6:45 - 7:00
- 7:00 - 7:15
- 7:15 - 7:30

Figura 24. Franja horaria prevista para llegar al destino

Una vez el usuario elija las opciones, pulsando el botón “Calcular intervalo de salida”, enviará la información de los campos al servidor. Ahí es cuando el servidor se pone en marcha siguiendo los pasos explicados en el Capítulo 2. Seguidamente, envía al front-end el mensaje con el intervalo de salida.

3.2 Visualización del intervalo obtenido

Cuando se pulsa el botón, inmediatamente se recibe el mensaje a través de una alerta que aparece por pantalla (véase la Figura 25), con el intervalo de salida propuesto por el servidor.

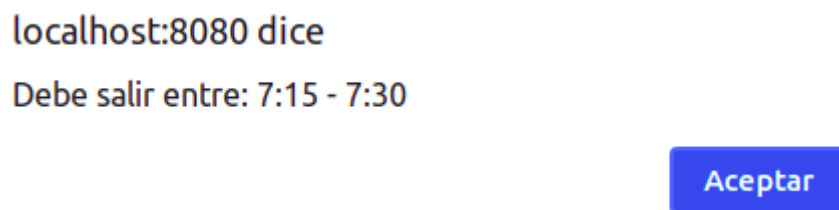


Figura 25. Alerta recibida con el intervalo de salida

El resultado se almacena en una tabla de Vue.js que contiene los siguientes campos: “Origen”, “Destino”, “Hora de salida” y “Hora de llegada prevista” (véase la Figura 26). En esta tabla se van almacenando todos los posibles casos que el usuario vaya introduciendo y solicitando el servidor. Si realiza varias consultas, todas se van añadiendo a la tabla con la información seleccionada en el formulario.

Origen	Destino	Hora de salida	Hora de llegada prevista
ID 1: Los Realejos	ID: 120: Dirección Santa Cruz	Debe salir entre: 6:45 - 7:00	7:30 - 7:45
ID 52: La Victoria	ID: 120: Dirección Santa Cruz	Debe salir entre: 7:00 - 7:15	7:30 - 7:45
ID 36: La Cuesta de la Villa	ID 113: Guajara	Debe salir entre: 8:00 - 8:15	8:15 - 8:30
ID 82: Los Naranjeros - Tacoronte	ID 96: Aeropuerto Tenerife Norte	Debe salir entre: 8:30 - 8:45	8:30 - 8:45

Figura 26. Información de los trayectos solicitados

3.3 Despliegue con Docker

En cuanto al despliegue de la aplicación, se puede desplegar por un lado el servidor (backend) y por otro lado la interfaz (front-end). Esto se hace ejecutando en el directorio del backend el comando `npm run start` y simultáneamente en el directorio frontend el comando `npm run serve`. La interfaz se debe cargar en la dirección web `localhost:8080`.

Para que el despliegue no se haga de forma independiente, se utiliza la tecnología Docker. Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software.

3.3.1. Dockerizando el Back-End

En primer lugar, hay que crear el contenedor con el servidor. Con esto podemos tener funcionando la parte del backend en dicho contenedor y así no se necesitaría tenerlo en nuestro sistema anfitrión para hacer funcionar la aplicación. Para ello, vamos a crear un fichero de configuración llamado **Dockerfile** [31] donde se describe la información para generar el contenedor Docker para el servidor (véase la Figura 27). En nuestro Dockerfile vamos a tener lo siguiente:

- La imagen de una de las últimas versiones de Node. Por ejemplo la 12.2.0-alpine
- La carpeta sobre la que vamos a trabajar en el contenedor (/app). Es ahí donde se copian los ficheros necesarios de nuestro backend, el fichero del servidor y los ficheros con los datos de las simulaciones.
- Instalación de las dependencias necesarias para el funcionamiento y el puerto donde va a estar ejecutándose el contenedor del servidor, en este caso, el puerto 8181.
- Por último, se ejecuta el comando *npm run start* para mantener el servidor escuchando.

```
1 # base image
2 FROM node:12.2.0-alpine
3
4 # set working directory
5 WORKDIR /app
6
7 # add `/app/node_modules/.bin` to $PATH
8 ENV PATH /app/node_modules/.bin:$PATH
9
10 # install and cache app dependencies
11 COPY package.json /app/package.json
12 COPY server.js /app/server.js
13 COPY data/* /app/data/
14 RUN npm config set unsafe-perm true
15 RUN npm install
16 EXPOSE 8181
17
18 # start app
19 CMD ["npm", "run", "start"]
```

Figura 27. Dockerfile del contenedor del servidor

3.3.2. Dockerizando el Front-End

Por otro lado, hay que crear el contenedor para almacenar la interfaz creada con Vue.js (véase la Figura 28). En este caso, debemos crear una imagen multi-stage. Para ello, en el **Dockerfile** se introduce lo siguiente:

- En el primer stage, se copian los archivos del proyecto y también se instalan las dependencias que sean necesarias, tal cual hicimos anteriormente para el contenedor del servidor.
- En el segundo stage, se monta un servidor web, en este caso Nginx, que va a obtener el build del stage anterior y copiará el contenido en la carpeta del servidor web.
- Por último, se arranca el nginx para servir la web frontend.

```
1 FROM node:alpine as develop-stage
2
3 WORKDIR /usr/src/app
4
5 COPY package.json ./
6
7 RUN yarn install -all
8
9 COPY . ./
10
11 FROM develop-stage as build-stage
12 RUN yarn build
13
14 FROM nginx:1.12-alpine as production-stage
15 COPY --from=build-stage /usr/src/app/dist /usr/share/nginx/html
16 EXPOSE 80
17 CMD ["nginx", "-g", "daemon off;"]
```

Figura 28. Dockerfile del contenedor del frontend

3.3.3. Unión de backend y frontend con Docker-Compose

En este último paso, se unen los contenedores que tenemos creados, backend y frontend, mediante una red interna de Docker. Esto se hace fuera de las carpetas de ambos contenedores, en el directorio raíz del proyecto, con la creación de los ficheros: *docker-compose.yml* (véase la Figura 29) y *run.sh*. Primero, se empieza con la configuración docker-compose [32]. En este fichero se hace lo siguiente:

- Crear los dos servicios: frontend y backend. Ambos se enlazan mediante la misma network.

- Abrir el puerto del servidor (puerto 8181) y el puerto del frontend (8081).
- Indicar en *build* el contexto de cada servicio al construir la imagen de Docker. Se le pasa el nombre de la imagen y también la ruta de las carpetas backend y frontend. Con esto se le comunica a docker-compose que cada servicio se va a construir a partir del Dockerfile que hay dentro de los directorios.

```
1  version: '3.7'
2  services:
3    frontend:
4      build:
5        context: "./frontend"
6        dockerfile: Dockerfile
7        image: tfg/frontend
8        container_name: "frontend"
9      ports:
10       - "8081:80"
11     networks:
12       - app
13   backend:
14     build:
15       context: "./backend"
16       dockerfile: Dockerfile
17       image: tfg-backend
18       container_name: "backend"
19     ports:
20       - "8181:8181"
21     networks:
22       - app
23   networks:
24     app:
25       name: app_docker
```

Figura 29. Fichero docker-compose.yml

El script *run.sh* construye y levanta los servicios definidos en el fichero *docker-compose.yml*:

```
docker-compose build && docker-compose up
```

Después de lanzar el script, los contenedores se conectan y se crea la interfaz. Cuando se ejecuta, se construyen ambos contenedores y el servidor se mantiene a la espera de recibir solicitudes por parte del frontend (véase la Figura 30):

```
Successfully built 0d9965a155eb
Successfully tagged tfg-backend:latest
Starting frontend ... done
Starting backend ... done
Attaching to frontend, backend
backend |
backend | > server@1.0.0 start /app
backend | > node server.js
backend |
```

Figura 30. Backend a la espera

Mientras se esté ejecutando el servidor, haciendo una petición a la dirección *localhost:8081*, podemos tener acceso a la web desplegada (véase la Figura 31):

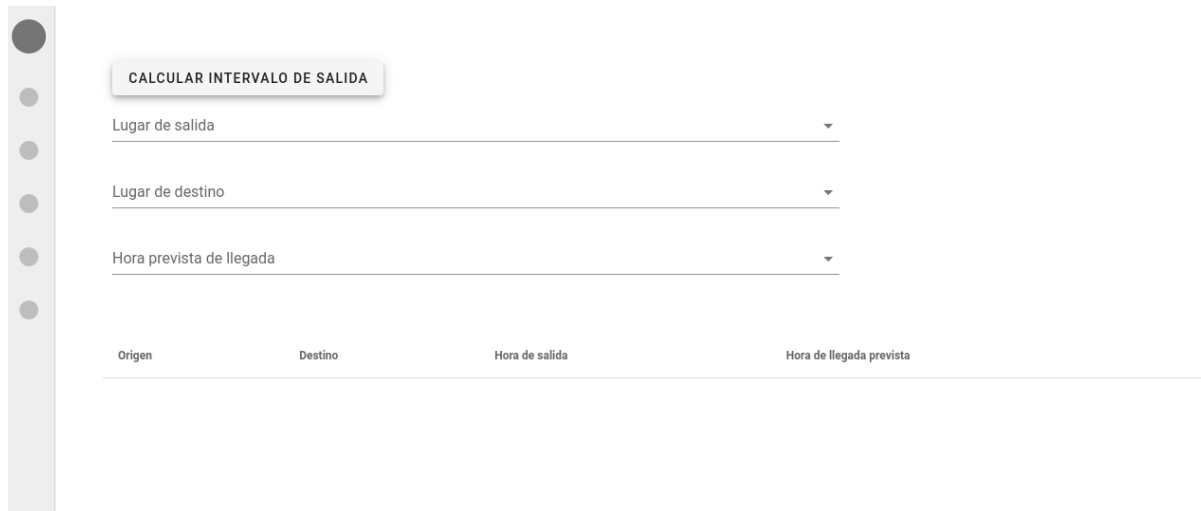


Figura 31. Web desplegada con Docker-Compose

Capítulo 4

Conclusiones y líneas futuras

Con el estudio realizado y el desarrollo de la interfaz, podemos decir que se ha llegado a cumplir con el objetivo general de nuestra investigación, es decir, proporcionar al conductor de la autopista TF-5 un sistema inteligente de apoyo para ayudar a minimizar los colapsos y retenciones ocasionados por el tráfico. Para llevar a cabo el proyecto, se empleó SUMO como simulador de tráfico y se hizo un análisis y procesamiento de la salida de las distintas simulaciones, para estudiar el comportamiento que tomaba la autopista en situaciones de congestión.

Seguidamente, teniendo el estudio, con los datos obtenidos, procesados y clasificados por franjas horarias, el servidor se pone en marcha para obtener el intervalo horario de salida según los puntos de origen y destino y la hora de llegada prevista por el conductor. Una vez calculado el intervalo, lo envía como mensaje al front-end para que lo visualice en la interfaz, donde el conductor podrá ver la recomendación de salida hecha por el sistema.

Con esto podemos afirmar que el planteamiento inicial del proyecto ha sido cumplido, ya que la idea principal era definir 3 capas bien diferenciadas y cada una realizando una función concreta. Por un lado la capa de simulación, para estudiar el comportamiento de la autopista en momentos de colapsos, por otro lado la capa del servidor, para procesar la información obtenida de las salidas de la simulación y por último la capa de visualización, conectada con el servidor, y en la que se muestra el intervalo horario de salida calculado.

Además de lo nombrado, con el trabajo realizado también se cumplen una serie de resultados esperados con la ejecución de la aplicación. Por ejemplo, con la implementación del sistema inteligente, el conductor ahorra un tiempo considerable, porque el sistema le predice un intervalo de salida y ese resultado es 95% fiable, calculado con los intervalos de confianza en las matrices cada 15 minutos. El conductor debe saber que mientras salga dentro de ese rango, va a llegar a la hora que desea. Es un ahorro de tiempo, porque muchos de los usuarios de la autopista, para llegar a su destino, suelen salir dos horas o incluso antes de lo normal para intentar evitar las colas y retenciones. Si esto lo hacen todos los usuarios de la autopista, van a generar más colapsos en ella. Por lo tanto, mientras cada conductor comience su trayecto dentro del intervalo establecido por el sistema, se agilizará el tránsito en horas punta y por consecuencia, se puede llegar a minimizar el impacto ambiental que originan los atascos, al igual que evitar posibles infracciones causadas por la frustración generada en situaciones de congestión.

Finalmente, este estudio ha conseguido llevar a cabo todas las metas planteadas, aunque también se han de mencionar algunas líneas futuras de trabajo que permitan la posibilidad de mejorar el proyecto realizado y encontrar una solución más eficaz para esta problemática que llevamos sufriendo durante tanto tiempo:

- Modificar la interfaz desarrollada y añadirle más funcionalidades como por ejemplo el control de inicio de sesión para que pueda ser utilizada por muchos más usuarios, donde cada uno pueda visualizar sus datos personales, sus horas frecuentes de salida, sus desplazamientos, etc.
- Realizar el estudio con simulaciones distintas para cada día de la semana, porque en la interfaz indica la hora prevista de llegada, pero el día no lo especifica. El sistema entiende que el día del trayecto va a ser el día siguiente o la hora más próxima al momento cuando el usuario solicita el intervalo de salida. Una vez conseguido el estudio, estaría bien establecer un calendario para que el usuario pueda elegir además del intervalo horario, el día de la semana.
- Lanzar simulaciones y recoger los datos para todas las franjas horarias donde más retenciones hay en la autopista, no solo las matinales.
- Podría ampliarse la zona de simulación, puesto que podrían producirse atascos en otras vías ajenas a la autopista que no se han incluido en el archivo de red.
- Permitir al conductor que pueda elegir sobre más puntos de origen y de destino, incluso que pueda introducir un lugar concreto y que la aplicación tenga ese punto analizado y si no es así, que tome el lugar más cercano al elegido por el conductor. Por lo tanto, eso significa que habría que añadir al estudio más puntos tanto de salida como de llegada.
- Realizar más simulaciones en tiempos más cortos para que el estudio sea más eficaz aún, y que a la hora de calcular el intervalo horario de salida, en vez de ser un intervalo de 15 minutos, sea un rango más corto.
- Quizás se podría realizar la simulación en sentido contrario, desde Santa Cruz hasta el Puerto de La Cruz o Los Realejos. En este tramo no son tan comunes las retenciones pero sí que hay algunas horas en el día donde suelen formarse colapsos en el tráfico, sobre todo a la hora del mediodía o por la tarde-noche cuando las personas terminan su jornada laboral o de estudio.

Capítulo 5

Summary and future work

With the study carried out and the development of the interface, we can say that the general objective of our research has been achieved, i.e. to provide the driver of the TF-5 motorway with an intelligent support system to help minimise traffic congestion and delays. To carry out the project, SUMO was used as a traffic simulator and the output of the different simulations was analysed and processed to study the behaviour of the motorway in situations of congestion.

After the study, with the data obtained, processed and classified by time slots, the server starts to obtain the departure time interval according to the points of origin and destination and the driver's expected arrival time. Once the interval has been calculated, it sends it as a message to the front-end for display on the interface, where the driver can see the departure recommendation made by the system.

With this we can affirm that the initial approach of the project has been fulfilled, since the main idea was to define 3 well-differentiated layers, each one performing a specific function. On the one hand, the simulation layer, to study the behaviour of the motorway in times of collapse, on the other hand, the server layer, to process the information obtained from the simulation outputs and, finally, the visualisation layer, connected to the server, which shows the calculated hourly exit interval.

In addition to the above, the work carried out also fulfils a number of expected results with the implementation of the application. For example, with the implementation of the intelligent system, the driver saves considerable time, because the system predicts a departure interval and this result is 95% reliable, calculated with confidence intervals in the matrices every 15 minutes. The driver should know that as long as he leaves within that range, he will arrive at the time he wants. This is a time saving, because many motorway users, in order to reach their destination, often leave two hours or even earlier than normal to try to avoid queues and traffic jams. If this is done by all motorway users, it will create more congestion on the motorway. Therefore, as long as each driver starts their journey within the interval set by the system, it will speed up traffic at peak times and consequently minimise the environmental impact of traffic jams, as well as avoid possible violations caused by frustration in congested situations.

Finally, this study has managed to achieve all the goals set out, although some future lines of work should also be mentioned that allow for the possibility of improving the project and finding a more effective solution to this problem that we have been suffering from for so long:

- Modify the developed interface and add more functionalities such as login control so that it can be used by many more users, where each user can view their personal data, their frequent departure times, their journeys, etc.
- Carry out the study with different simulations for each day of the week, because the interface indicates the estimated time of arrival, but does not specify the day. The system assumes that the day of the journey will be the next day or the hour closest to the time when the user requests the departure interval. Once the study has been achieved, it would be good to set up a calendar so that the user can choose not only the time slot, but also the day of the week.
- Launch simulations and collect data for all the time slots where there are the most traffic jams on the motorway, not just the morning ones.
- The simulation area could be extended, as traffic jams could occur on other roads outside the motorway that are not included in the network file.
- Allow the driver to choose over more origin and destination points, even to enter a specific location and the application would have that point analysed and if not, take the location closest to the one chosen by the driver. Therefore, this means that more departure and arrival points should be added to the study.
- Carry out more simulations at shorter times to make the study even more effective, and that when calculating the departure time interval, instead of a 15-minute interval, it should be a shorter range.
- Perhaps the simulation could be carried out in the opposite direction, from Santa Cruz to Puerto de La Cruz or Los Realejos. On this stretch, traffic jams are not so common, but there are some hours of the day when traffic jams tend to occur, especially at midday or in the afternoon and evening when people finish their work or study day.

Capítulo 6

Presupuesto

El presupuesto de este proyecto se basa en el tiempo dedicado para la elaboración del mismo, por lo tanto, el presupuesto agrupa los costes humanos, es decir, las horas empleadas en el desarrollo del proyecto.

Tarea	Horas	Precio (30€/h)
<i>Simulación y calibración</i>	150	4.500
<i>Análisis y procesamiento de datos</i>	200	6.000
<i>Desarrollo de la interfaz web</i>	100	3.000
Total	500	13.500

Tabla 4. Presupuesto

Capítulo 7

Apéndice A

7.1 Repositorio

Repositorio Github con el código fuente del proyecto:

<https://github.com/edfigo98/Trabajo-Fin-de-Grado>

Bibliografía

[1] TomTom. (2020). *Traffic congestion ranking*. TomTom Traffic Index. Recuperado el 18 de agosto de 2021, de

https://www.tomtom.com/en_gb/traffic-index/ranking/?country=ES

[2] González, L. (2015, 14 noviembre). *Tiempo de Canarias - El digital de las islas*. Tiempo de Canarias. Recuperado el 18 de agosto de 2021, de

<https://tiempodecanarias.com/mas-de-20-anos-atascados-con-la-tf-5>

[3] elDiario.es. (2018, 3 octubre). Si Canarias fuera un país, sería el sexto del mundo con más coches por cada mil habitantes. elDiario.es. Recuperado el 19 de agosto de 2021 de,

https://www.eldiario.es/canariasahora/tenerifeahora/sociedad/canarias-sexto-mundo-coches-habitantes_1_1911679.html

[4] eldia.es. (2020, 15 marzo). Acabar con las colas de la TF-5. *eldia.es*.

<https://www.eldia.es/tenerife/2020/03/15/acabar-colas-tf-5-22442403.html>

[5] González, S. E. M. (2020, 28 septiembre). *Planificación optimizada de un sistema de semáforos mediante algoritmos evolutivos: una aplicación a la rotonda del Padre Anchieta, en Santa Cruz de Tenerife*. RIULL. Recuperado el 25 de agosto de 2021, de <https://riull.ull.es/xmlui/handle/915/21321>

[6] del Arco, A. R. (2019, 15 octubre). *Simulación y gestión del tráfico en una autopista en situaciones de congestión*. RIULL. Recuperado el 25 de agosto de 2021, de <https://riull.ull.es/xmlui/handle/915/16505>

[7] Álvarez López *et al.*, "Microscopic Traffic Simulation using SUMO," 2018 21st *International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575-2582, doi: 10.1109/ITSC.2018.8569938

[8] Anylogic.com. 2021. AnyLogic: Simulation Modeling Software Tools & Solutions for Business. [online] Recuperado el 24 de agosto de 2021, de <https://www.anylogic.com/>

[9] Haklay, M., & Weber, P. (2008). Openstreetmap: User-generated street maps. *IEEE Pervasive computing*, 7(4), 12-18

[10] colaboradores de Wikipedia. (2020, 6 septiembre). *JOSM*. Wikipedia, la enciclopedia libre. Recuperado el 24 de agosto de 2021, de <https://es.wikipedia.org/wiki/JOSM>

[11] *netedit - SUMO Documentation*. (s. f.). SUMO. Recuperado el 15 de marzo de 2021, de <https://sumo.dlr.de/docs/Netedit/index.html>

[12] Mozilla and individual contributors. (2005). *WebSockets - Referencia de la API Web* | MDN. MDN Web Docs. Recuperado el 25 de mayo de 2021, de https://developer.mozilla.org/es/docs/Web/API/WebSockets_API

[13] Mozilla and individual contributors. (2005). *JavaScript* | MDN. MDN Web Docs. Recuperado el 25 de mayo de 2021, de <https://developer.mozilla.org/es/docs/Web/JavaScript>

[14] You, E. (2020). *Vue.js* (3.0) [Framework progresivo para construir interfaces de usuario.] Recuperado el 31 de mayo de 2021, de <https://vuejs.org/>

[15] colaboradores de Wikipedia. (2021, 28 agosto). Python. Wikipedia, la enciclopedia libre. Recuperado el 24 de agosto de 2021, de <https://es.wikipedia.org/wiki/Python>

[16] Torvalds, L. (2007). *Git* (2.32.0) [Software de control de versiones]. Recuperado el 20 de abril de 2021, de <https://git-scm.com/>

[17] Netscape Communications Corporation. (2011). *Mocha - the fun, simple, flexible JavaScript test framework*. Recuperado el 26 de julio de 2021, de <https://mochajs.org/>

[18] Hykes, S. (2013). *Docker* (20.10.8) [Software libre y de código abierto]. Recuperado el 2 de agosto de 2021, de <https://www.docker.com/>

[19] GitHub, Inc. (2008). *GitHub-Eclipse-SUMO*. Recuperado el 24 de febrero de 2021, de <https://github.com/eclipse/sumo>

[20] German Aerospace Center (DLR) and others. (2021, 4 agosto). *Trip - SUMO Documentation*. SUMO. Recuperado el 15 de marzo de 2021, de <https://sumo.dlr.de/docs/Tools/Trip.html>

[21] German Aerospace Center (DLR) and others. (2021). *Importing O/D Matrices - SUMO Documentation*. SUMO. Recuperado el 23 de mayo de 2021, de https://sumo.dlr.de/docs/Demand/Importing_O/D_Matrices.html

[22] German Aerospace Center (DLR) and others. (2021). *Describing the TAZ - SUMO Documentation*. SUMO. Recuperado el 23 de mayo de 2021, de https://sumo.dlr.de/docs/Demand/Importing_O/D_Matrices.html#describing_the_taz

- [23] German Aerospace Center (DLR) and others. (2021). *The O-format (VISUM/VISSIM) - SUMO Documentation*. SUMO. Recuperado el 25 de mayo de 2021, de https://sumo.dlr.de/docs/Demand/Importing_O/D_Matrices.html#the_o-format_visum_vissim
- [24] German Aerospace Center (DLR) and others. (2021). *Configuration SUMOCFG - SUMO Documentation*. SUMO. Recuperado el 15 de marzo de 2021, de https://sumo.dlr.de/docs/Tutorials/Hello_SUMO.html#configuration
- [25] German Aerospace Center (DLR) and others. (2021). *od2trips - SUMO Documentation*. SUMO. Recuperado el 18 de mayo de 2021, de <https://sumo.dlr.de/docs/od2trips.html>
- [26] German Aerospace Center (DLR) and others. (2021). *DUAROUTER - SUMO Documentation*. SUMO. Recuperado el 18 de mayo de 2021, de <https://sumo.dlr.de/docs/duarouter.html>
- [27] German Aerospace Center (DLR) and others. (2021). *Definition of Vehicles, Vehicle Types, and Routes - SUMO Documentation*. SUMO. Recuperado el 1 de junio de 2021, de https://sumo.dlr.de/docs/Definition_of_Vehicles%2C_Vehicle_Types%2C_and_Routes.html
- [28] German Aerospace Center (DLR) and others. (2021). *sumo-gui - SUMO Documentation*. SUMO. Recuperado el 10 de marzo de 2021, de <https://sumo.dlr.de/docs/sumo-gui.html>
- [29] German Aerospace Center (DLR) and others. (2021). *VehRoutes - SUMO Documentation*. SUMO. Recuperado el 1 de junio de 2021, de <https://sumo.dlr.de/docs/Simulation/Output/VehRoutes.html>
- [30] Schlueter, I. (2021). *NPM (7.7.5) [Sistema de gestión de paquetes]*. Recuperado el 10 de junio de 2021, de <https://www.npmjs.com/package/read-excel-file>
- [31] Docker Inc. (2013). *Dockerfile reference*. Docker Documentation. Recuperado el 2 de agosto de 2021, de <https://docs.docker.com/engine/reference/builder/>
- [32] Docker Inc. (2013). *Overview of Docker Compose*. Docker Documentation. Recuperado el 2 de agosto de 2021, de <https://docs.docker.com/compose/>