IMPACTO DE LA VELOCIDAD Y MODELO DE MOVILIDAD EN UNA COMUNICACIÓN DE DATOS DE UNA RED VEHICULAR

Daniel Felipe Chavarro Piamba
Oscar Arley Orozco Sarasti

Director
Ing. Oscar J. Calderón C.

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telecomunicaciones
Línea de Investigación de Gestión Integrada de Redes, Servicios y Arquitecturas de Telecomunicaciones
Popayán, Mayo de 2013

IMPACTO DE LA VELOCIDAD Y MODELO DE MOVILIDAD EN UNA COMUNICACIÓN DE DATOS DE UNA RED VEHICULAR

Anexo de Trabajo de Grado presentado como requisito para obtener el título de Ingeniero en Electrónica y Telecomunicaciones

Daniel Felipe Chavarro Piamba
Oscar Arley Orozco Sarasti

Director
Ing. Oscar J. Calderón C.

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telecomunicaciones
Línea de Investigación de Gestión Integrada de Redes, Servicios y Arquitecturas de Telecomunicaciones
Popayán, Mayo de 2013

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ACRÓNIMOS

GTNetS *Georgia Tech Network Simulator,* Simulador de Red del Tecnológico de Georgia.

JiST/SWANS *Java in Simulation Time/Scalable Wireless Ad hoc Network Simulator,* Java en Tiempo de Simulación/Simulador Escalable para redes Inalámbricas Ad hoc.

MOVE *Mobility model generator for Vehicular networks,* Generador de modelo de movilidad para redes vehiculares.

SNS *Staged Network Simulator,* Simulador de Red por Etapas.

STRAW *Street Random Waypoint,* Punto Aleatorio en la Calle.

SUMO *Simulation of Urban MObility,* Simulación de Movilidad Urbana.

TraNS *Traffic and Network Simulation environment,* Entorno de Simulación de Tráfico y Red.

VeINS *Vehicles in Network Simulation,* Vehículos en Simulación de Red.

VISSIM *Visual Traffic Simulation,* Simulación Visual del Tráfico.

ANEXO A

A.1. MAPA CONCEPTUAL PARA LA GENERACIÓN DE MODELOS DE MOVILIDAD

El mapa conceptual para la generación de un modelo de movilidad se muestra en el Capítulo II de este trabajo de grado. Este anexo busca profundizar en las variables presentes en dicho mapa, en especial en el mapa adaptado al entorno colombiano; con el fin de aclarar posibles dudas acerca de cómo pueden llegar a afectar a los modelos de movilidad las situaciones presentes en el entorno colombiano.

Los obstáculos que puedan afectar la generación de un modelo de movilidad y en especial, los obstáculos particulares al entorno nacional se muestran en la Figura A.1.

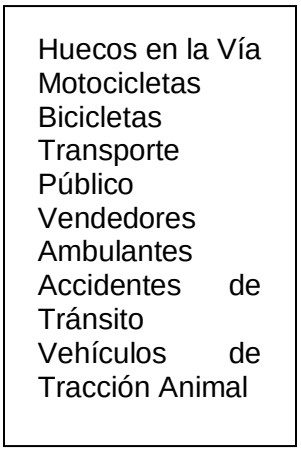

Figura A.1. Obstáculos para la generación de modelos de movilidad. Por los Autores.

La descripción acerca de cómo estos obstáculos particulares a entornos colombianos afectan la movilidad vehicular se presenta a continuación:

- **Huecos en la Vía:** los huecos en la vía afectan la movilidad de los nodos y por consiguiente, el modelo de movilidad debido a que hacen que los conductores cambien de carriles repentinamente o disminuyan su velocidad drásticamente. Esto hace que el modelo de movilidad que puedan estar siguiendo un conjunto de vehículos cambie repentina e inesperadamente, afectando con esto procesos de comunicación vehicular.

- **Motocicletas:** las motocicletas son consideradas como obstáculos debido a que, para este trabajo de grado no hacen parte de la red VANET. Es decir, no son nodos móviles de las redes vehiculares. Asimismo, debido a la agresividad de algunos conductores de motos, pueden convertirse en obstáculos para los vehículos dadas sus maniobras agresivas de adelantamiento.

- **Bicicletas:** al igual que las motocicletas, algunas maniobras imprudentes de los ciclistas pueden afectar el movimiento de los vehículos para tratar de evitarlos. Cabe destacar que, aunque es casi habitual ver motocicletas o bicicletas que afecten el movimiento de los automóviles en carreteras colombianas, no es un problema exclusivo del país: también se da en países donde el número de motos y bicicletas es alto (India y China respectivamente).

- **Transporte Público:** el transporte público es una de las causas por las cuales pueden generarse congestiones vehiculares (debido al tamaño de los vehículos). Además, la mala costumbre de parquearse a recoger o dejar pasajeros en cualquier sitio por parte de los conductores de vehículos de servicio público hace que la movilidad y los modelos de movilidad sean afectados significativamente. Esto afecta particularmente a la ciudad de Popayán en el centro histórico de la ciudad, debido a la estrechez de las vías por ése sector de la ciudad.

- **Vendedores Ambulantes:** debido a la situación económica del país, los vendedores ambulantes en semáforos, cruces e intersecciones de las vías colombianas no son novedad. En consecuencia, los conductores que compran los productos que ofrecen éstas personas pueden llegar a provocar un pequeño embotellamiento mientras realizan la transacción comercial. Aunque el tiempo que se detienen a comprar es poco, esto de alguna manera puede provocar cambios de carriles por parte de los nodos detrás del auto detenido; afectando así el modelo de movilidad.

- **Accidentes de Tránsito:** los accidentes de tránsito afectan la movilidad de las vías de todo el planeta. Al presentarse masivamente (como en Estados Unidos) puede parar totalmente el flujo de vehículos sobre la vía en cuestión. Como se dijo en el Capítulo I de este trabajo de grado, en Colombia son causantes de la muerte de alrededor de 6000 personas cada año y, para el caso de los modelos de movilidad, los afecta significativamente al modificar el estado normal de la vía.

- **Vehículos de tracción animal:** junto con las motocicletas y las bicicletas, los vehículos de tracción animal son una particularidad de las vías colombianas y, pueden afectar la movilidad vehicular al momento de ser sobrepasados por los vehículos o si presentan comportamientos agresivos. Además, es común ver en las vías colombianas motocicletas, bicicletas y vehículos de tracción animal desplazándose sobre el carril izquierdo (el carril rápido o de adelanto) de las vías.

Los obstáculos descritos anteriormente son los más significativos para la generación de modelos de movilidad tomando en cuenta características del entorno colombiano.

Siguiendo con la descripción acerca de los tópicos presentes en el mapa conceptual del Capítulo II de este trabajo de grado se tienen varios puntos de atracción y repulsión, en donde los vehículos están más cerca y lejos respectivamente. Estos puntos se muestran en la Figura A.2.

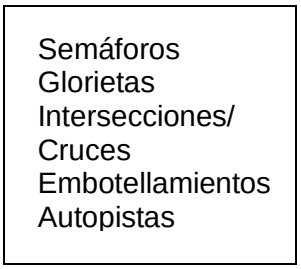

Figura A.2. Puntos de atracción/repulsión para la generación de modelos de movilidad. Por los Autores.

- **Semáforos:** los semáforos son considerados puntos de atracción debido a que los automóviles, al esperar el cambio de rojo a verde, permanecen juntos por dicho instante de tiempo. Esto puede llegar a modificar el modelo de movilidad que traían los nodos antes de realizar la parada en el semáforo.

- **Glorietas:** las glorietas también son puntos de atracción dado que los nodos se ven obligados a detenerse para ceder el paso a los nodos que estén circulando por la glorieta. De manera que los carros que vengan detrás se verán obligados a disminuir su velocidad y, por ende, puede llegar a afectar el modelo de movilidad que vengan siguiendo antes de la glorieta.

- **Intersecciones/Cruces:** al igual que las glorietas y semáforos, las intersecciones o cruces obligan a disminuir la velocidad que lleven los nodos.

- **Embotellamientos:** los embotellamientos o trancones son los puntos de atracción más críticos, debido a que la velocidad de los nodos es casi nula y no pueden circular libremente. Es la situación en donde están más tiempo los autos cerca, pudiendo llegar así a afectar el modelo de movilidad.

- **Autopistas:** al contrario de los anteriores, las autopistas pueden considerarse puntos de repulsión dadas las altas velocidades que puedan alcanzar los nodos y el movimiento casi independiente en relación a otros vehículos que circulen por dichas vías.

Pasando a las limitaciones de velocidad mostradas en el mapa del Capítulo II, se tienen en cuenta las presentadas en el entorno colombiano. Dichos elementos se muestran en la Figura A.3.

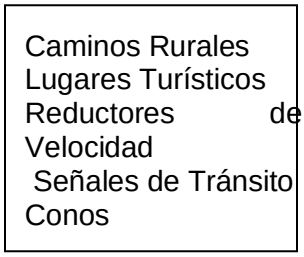

Figura A.3. Limitaciones de velocidad para la generación de modelos de movilidad. Por los Autores.

- **Caminos Rurales:** los caminos rurales en Colombia no presentan las mismas condiciones que las vías principales. Generalmente son caminos sin pavimentación y con arena o piedras, lo que hace que los vehículos que transitan por éstos caminos lo hagan a velocidades reducidas. Es por esto que la circulación de nodos por estos caminos afecta la velocidad y por ende, el modelo de movilidad.

- **Lugares Turísticos:** los puntos turísticos de las ciudades de Colombia están restringidos para la movilidad vehicular, hasta tal punto de prohibirla por completo en algunos sitios. Por los que se puede transitar con vehículos, se debe hacerlo a bajas velocidades debido al alto número de peatones presentes en la vía y sus alrededores. Es por esto que estos lugares presentan limitaciones de velocidad para los autos.

- **Reductores de Velocidad:** hacen referencia a los populares “policías acostados” y a otro tipo de elementos en la vía que cumplen la función de disminuir la velocidad de los nodos. Se ubican generalmente en vías donde la accidentalidad por circular a altas velocidades es considerable y afectan principalmente la movilidad de los vehículos al obligarlos a reducir su velocidad y/o a cambiar de carril por adelantar autos que circulan por dichos reductores a velocidades más bajas.

- **Señales de Tránsito:** aunque en Colombia, lo común es que poco se respeten las señales de tránsito, para una conducción segura deben acatarse y respetarse. Ya sean las señales que indican la velocidad máxima permitida o señales de aviso de curvas, puentes, construcciones, etc. En especial las señales que limitan la velocidad máxima son las que recaen principalmente en el contexto de limitaciones de velocidad.

- **Conos:** elementos de plástico que las autoridades viales ubican sobre las carreteras para dar señales informativas a los conductores o para delimitar carriles cuando otras opciones no sean posibles. Particularmente en el entorno colombiano, las autoridades viales ubican estos elementos cuando realizan controles o retenes a lo largo de las vías, obligando así a los conductores a disminuir su velocidad.

Por último, pero no menos importante están los hábitos sociales propios de la cultura colombiana al momento de conducir que pueden afectar un modelo de movilidad. Se muestran en la Figura A.4.

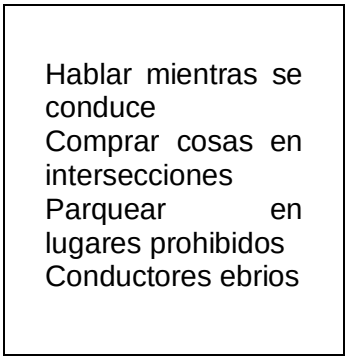

Figura A.4. Hábitos sociales que afectan la generación de un modelo de movilidad. Por los Autores.

- **Hablar mientras se conduce:** aunque los conductores hablen con las personas al interior del vehículo, no es tan relevante como cuando un conductor contesta su teléfono celular, ya que al hablar por celular se distrae de lo que acontece alrededor suyo y esto puede llegar a tener consecuencias nocivas tanto para él como para las demás personas en la vía. Lo ideal es que un conductor detenga su vehículo por completo y conteste el teléfono celular, aunque en Colombia esto poco se ve, dada la cultura de hablar manejando.

- **Comprar cosas en intersecciones:** similar a los vendedores ambulantes en semáforos e intersecciones, la acción de realizar transacciones comerciales en puntos donde dicho conductor puede generar embotellamientos y trancones, afectando con esto el modelo de movilidad que llevaban los vehículos antes de la acción.

- **Parquear en lugares prohibidos:** particularmente común en Colombia, la gente parquea sus vehículos en lugares prohibidos y esto puede llegar a generar embotellamientos o trancones.

- **Conductores ebrios:** aunque en Colombia esta sea una célebre pero triste estadística, dichos casos se presentan hacia altas horas de la noche, cuando el flujo de tráfico vehicular ha disminuido considerablemente. Es por esto que éste tópico no es de tanta relevancia como los anteriores pero, en situaciones muy específicas puede llegar a afectar la movilidad de los demás automóviles.

ANEXO B

B.1. SIMULADORES PARA REDES VANET

Los simuladores para redes VANET se clasifican en cuatro categorías, como se especifica en la Sección 3.2 del trabajo de grado. En este documento se especificaron las herramientas más significativas, por lo que en este anexo se describen otros programas disponibles para la simulación de redes VANET.

B.1.1. Simuladores de tráfico vehicular

En la Sección 3.3.1 del trabajo de grado se describen los simuladores de tráfico vehicular SUMO, MOVE y Citymob. Otros simuladores que pertenecen a esta categoría son:

- **VanetMobiSim:** es una extensión del entorno de simulación denominado CanuMobiSim que se enfoca en la movilidad vehicular tanto a nivel macroscópico como a nivel microscópico. Permite la importación de mapas provenientes de la base de datos de la Oficina de Censo de Estados Unidos (denominados TIGER) o la generación aleatoria de los mismos. Está enfocado principalmente al entorno norteamericano tomando en cuenta carreteras, mapas, edificios y demás e implementa el modelo de movilidad IDM-IM en los nodos a simular [1].

- **VISSIM (*Visual Traffic Simulation*):** el proyecto VISSIM fue desarrollado en Londres y tiene como objetivo proporcionar una implementación para aproximar el movimiento urbano de los vehículos bajo una simulación microscópica. El proyecto está desarrollado en Java y provee métodos para modelar un escenario de red y los nodos que lo componen. Aunque es un proyecto académico, el autor pone a disposición el código fuente para desarrollar nuevos elementos que originalmente no se tuvieron en cuenta (como comportamientos realistas de los nodos o la comprobación de hipótesis acerca del tráfico vehicular) [2].

- **FreeSim:** simulador desarrollado en Java que permite la representación y carga de escenarios como estructuras gráficas de múltiples sistemas de carreteras. Lleva a cabo simulaciones microscópicas y macroscópicas y permite la creación de datos de entrada (programados por el usuario) o la importación de éstos desde fuentes que proporcionen datos en tiempo real de alguna organización de transporte [3].

- **STRAW (*Street Random Waypoint*):** simulador desarrollado por la Universidad de Illinois. Está basado en el simulador de red JiST/SWANS y proporciona simulaciones realistas utilizando modelos de movilidad en ciudades estadounidenses. Sus trazas no pueden ser utilizadas en otros simuladores de red diferentes a JiST/SWANS, por lo que está bastante limitado en uso [4].

- **Mobitools:** es una suite de movilidad desarrollada en Estados Unidos. Desarrollada en Java, incorpora el uso de mapas reales para la creación de trazas de movilidad, además de un visualizador para mapas y un módulo para calcular los efectos de la propagación. Compatible con NS-2 y Qualnet [5].

- **TraNSlite:** basada en SUMO, esta herramienta genera trazas de movilidad para ser insertadas en NS-2. Utiliza mapas tipo TIGER, OpenStreetMap y Google Earth. Se considera una versión simplificada de la herramienta Trans [6].

Los simuladores descritos anteriormente, junto con los especificados en la Sección 3.2 del trabajo de grado, presentan tanto ventajas como desventajas y pueden ser útiles o no dependiendo de necesidades particulares. Es por esto que en la Tabla B.1 se presenta una comparación cualitativa de dichas herramientas teniendo en cuenta características propias del software hasta las trazas que generan.

B.1.2. Simuladores de red.

La Sección 3.3.2 del trabajo de grado presenta los simuladores de red más relevantes para redes VANET. Además de éstos, a continuación se describen las restantes herramientas software para la simulación de red.

- **SNS (*Staged Network Simulator*):** este simulador utiliza la técnica de simulación por etapas, la cual consiste en eliminar cálculos computacionales redundantes a través del uso de funciones de reutilización y caché. El simulador almacena, cuando sea posible, resultados de operaciones previamente simuladas y las vuelve a utilizar si es necesario. Está basado en NS-2 y ejecuta tareas hasta 50 veces más rápido que dicho simulador; aunque la última versión estable no está diseñada para simular escenarios de redes VANET [7].

- **GloMoSim:** es un entorno de simulación escalable para redes cableadas e inalámbricas. Fue construido en secciones o capas (similar al modelo de referencia OSI) en donde APIs estándar se utilizan entre las capas. Esto permite una rápida integración de modelos desarrollados en distintas capas por otros autores. Cabe destacar que este simulador es la versión gratuita de QualNet [8].

- **JiST/SWANS (*Java in Simulation Time/Scalable Wireless Ad hoc Network Simulator*):** es un simulador escalable de eventos discretos que se ejecuta sobre una máquina virtual de Java. Está basado en la plataforma JiST, la cual hace referencia a una máquina de altas prestaciones de simulación de eventos discretos que corren en la máquina virtual de Java. Permite una mayor cantidad de nodos en la simulación en relación a NS-2 y GloMoSim [9].

- **GTNetS (Georgia Tech Network Simulator):** desarrollado en el Instituto Tecnológico de Georgia, constituye un entorno de simulación que permite estudiar el comportamiento de redes de moderada a gran escala. Utilizado principalmente a la hora de desarrollar protocolos de enrutamiento para redes *ad hoc* [10].

La Tabla B.2 presenta una comparación cualitativa de los simuladores de red descritos anteriormente junto a los descritos en la Sección 3.3.2 del trabajo de grado.

	SUMO	VanetMobiSim	VISSIM	FreeSim	STRAW	MOVE	CityMob	Mobitools	TraNSlite
Software									
Portabilidad	✓	✓	✓	✓	✓	✓	✓	✓	✓
Freeware	✓	✓	✓	✓	✓	✓	✓	✓	✓
Opensource	✓	✓	✓	✓	✓	✓	✓	×	×
Consola de manejo	✓	×	×	×	×	✓	✓	×	×
GUI	✓	✓	✓	×	✓	✓	✓	✓	✓
Ejemplos disponibles	✓	✓	×	✓	×	✓	×	✓	×
Desarrollo continuo	✓	×	×	×	×	✓	×	×	×
Facilidad al configurar	✓	✓	✓	×	×	✓	✓	✓	✓
Facilidad de uso	×	×	✓	✓	×	×	✓	×	✓
Mapas									
Reales	✓	✓	×	✓	✓	✓	×	×	✓
Definidos por el usuario	✓	✓	✓	×	✓	✓	×	✓	×
Aleatorios	✓	✓	×	×	×	✓	✓	✓	✓
Posibilidad de importación	✓	✓	×	×	×	✓	×	✓	✓
Modelos de movilidad									
Mesoscópicos	×	✓	×	✓	×	×	×	×	×
Microscópicos	✓	✓	✓	✓	✓	✓	✓	✓	✓
Macroscópicos	×	✓	×	✓	×	×	×	×	×
Características de la vía									
Vías con múltiples carriles	✓	✓	✓	×	✓	✓	✓	✓	✓
Limitadores de velocidad	✓	✓	×	✓	✓	✓	✓	✓	✓
Semáforos	✓	✓	✓	×	✓	✓	✓	×	✓
Señales de tránsito	✓	✓	×	×	✓	✓	×	✓	✓
Gestión de intersecciones	✓	✓	×	×	×	✓	✓	×	✓
Trazas									
Basadas en XML	✓	✓	×	×	×	✓	×	×	×
Con soporte para NS-2	✓	×	×	✓	✓	✓	✓	×	✓
Con soporte para Qualnet	×	✓	×	×	×	✓	×	×	×
Con soporte para SWANS	×	×	×	×		×	×	×	×
Otras Características									
Diferentes tipos de autos	✓	×	×	×	×	✓	✓	×	×
Criterios de adelantamiento	✓	×	×	×	×	✓	×	×	✓
Cambio de carriles	✓	✓	×	×	×	✓	✓	✓	✓
	SUMO	**VanetMobiSim**	**VISSIM**	**FreeSim**	**STRAW**	**MOVE**	**CityMob**	**Mobitools**	**TraNSlite**

Tabla B.1. Comparación cualitativa de simuladores de tráfico vehicular. Por los Autores

	NS-2	GloMoSim	JiST/ SWANS	QualNet	GTNetS	OMNeT++	SNS	NS-3
Software								
Portabilidad	✓	✓	✓	✓	✓	✓	✓	✓
Freeware	✓	✓	✓	×	✓	✓	✓	✓
Open Source	✓	×	✓	×	✓	✓	✓	✓
Ejemplos Disponibles	✓	✓	✓	N.D	✓	✓	✓	×
Desarrollo contínuo	×	×	×	✓	×	✓	×	✓
Alto número de nodos	×	✓	✓	✓	×	✓	✓	✓
Manejo por consola	✓	✓	✓	N.D	×	✓	✓	✓
GUI	✓	✓	✓	✓	✓	✓	×	✓
Escalabilidad	×	✓	✓	✓	×	✓	×	✓
Facilidad al configurar	✓	✓	×	N.D	✓	✓	✓	✓
Facilidad al utilizar	×	×	×	N.D	✓	×	×	×
VANET								
802.11p	✓	×	×	✓	×	✓	×	✓
Obstáculos	×	×	×	N.D	×	✓	✓	✓
Modelos de flujo y tráfico vehicular	×	×	×	✓	×	✓	×	✓

Tabla B.2. Comparación cualitativa de simuladores de red. Por los Autores.

B.1.3. Simuladores híbridos y simuladores VANET

Los simuladores híbridos y los simuladores VANET se presentan en un menor número en relación a las dos primeras secciones. Por lo que todos se encuentran descritos en la Sección 3.3.3 y 3.3.4 del documento respectivamente.

B.1.4. Tendencias en la simulación de redes VANET

Las tendencias en simulación de redes VANET ha evolucionado desde que este tópico empezó a tomar relevancia entre la comunidad científica y académica. En un principio se modeló únicamente el movimiento aleatorio de los nodos con ayuda de la teoría de tráfico, para pasar a tener en cuenta trazas reales, definir escenarios macroscópicos y microscópicos y llegar a la tendencia actual que hace referencia a simulaciones bidireccionalmente acopladas.
Las simulaciones bidireccionalmente acopladas hacen referencia a la comunicación en tiempo real entre el generador de movilidad y el simulador de red. Esto hace que la eficiencia de simulación aumente considerablemente.
La Figura B.1 muestra dicha tendencia.

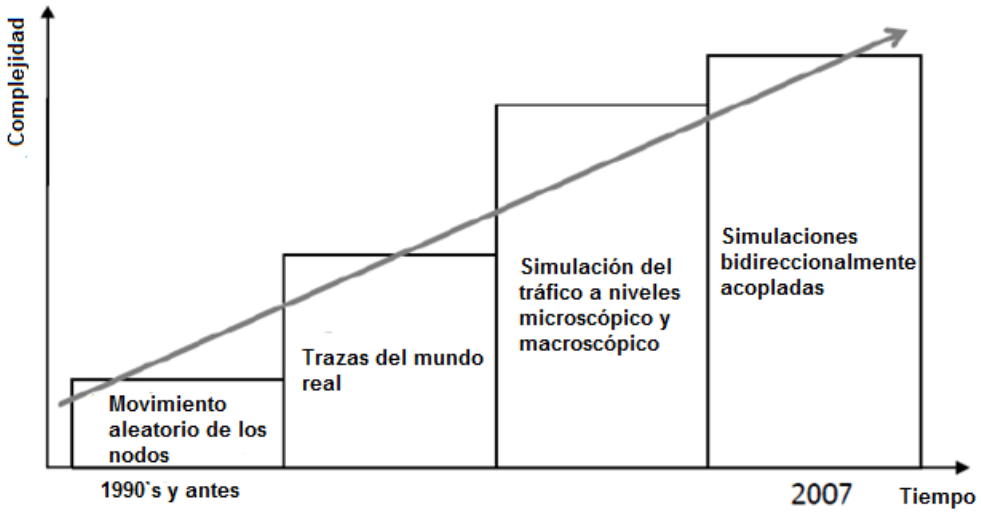

Figura B.1. Tendencias en la simulación de redes VANET. Adaptada de [11].

ANEXO C

C.1. INSTALACIÓN Y CONFIGURACIÓN DE SIMULADORES

A continuación se explica paso a paso el proceso de instalación y configuración de cada una de las herramientas de simulación elegidas previamente en el Capítulo III del trabajo de grado. La instalación se realizó en un equipo portátil *Toshiba Satellite® E205* con Windows® *7* de 64 bits y, además las siguientes características:

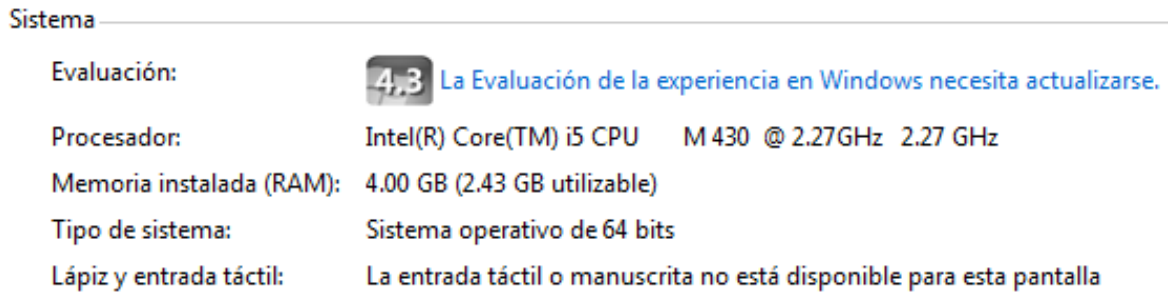

Figura C.1. Características del equipo portátil utilizado.

C.1.1. Instalación de SUMO

1. Se procede a descargar el simulador de tráfico SUMO versión 0.15.0 para Windows (64 bits). Para esto, se selecciona el paquete "sumo-winbin-0.15.0.zip" de la página:
   http://sourceforge.net/apps/mediawiki/sumo/index.php?title=Downloads.

2. Una vez que el paquete se descargue, se copia el archivo en el directorio que se desee (en este caso es en el disco C:\). Posteriormente se procede a descomprimirlo, con lo cual genera una carpeta con el nombre "sumo-0.15.0" con todos los archivos necesarios para su instalación.

3. Para realizar la instalación del software abrir la ventana de comandos (cmd.exe), ir a la carpeta donde se generó la carpeta de Sumo (C:/sumo-0.15.0) y ejecutar los siguientes comandos como se observa en la Figura C.2.

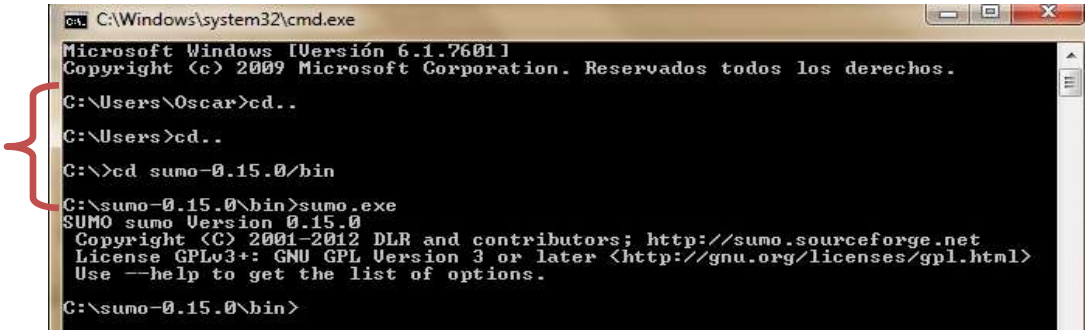

Figura C.2. Comandos de instalación de SUMO.

4. Para confirmar que la instalación no haya tenido problemas, se ejecuta el comando "sumo-gui.exe*"*. En la Figura C.3 se observa la Interfaz Gráfica de Usuario (GUI: *Graphic User Interface*) de SUMO, confirmando que la instalación ha sido exitosa.

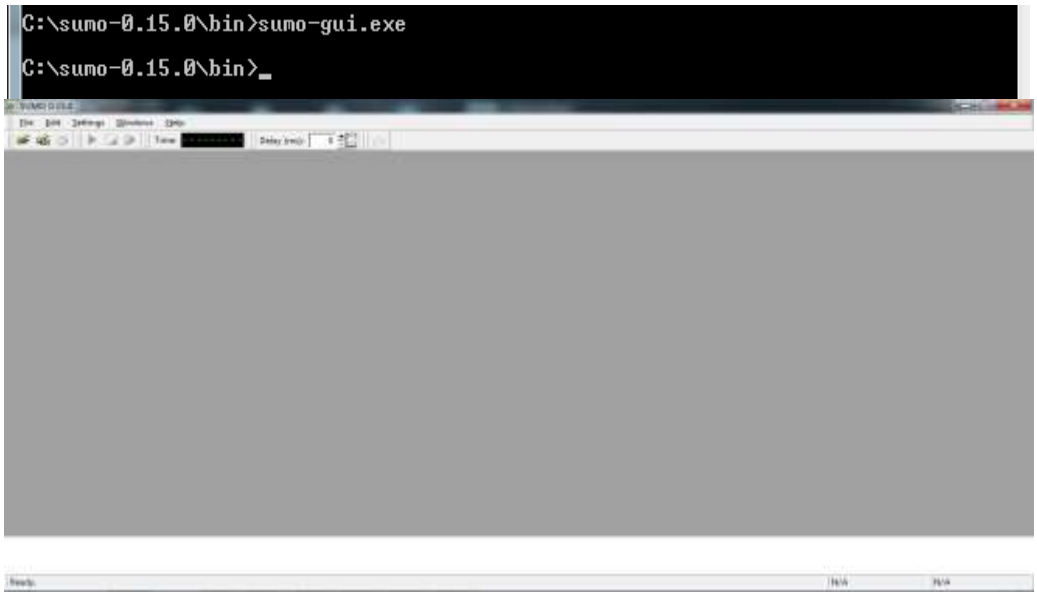

Figura C.3. Interfaz gráfica de SUMO 0.15.0.

La GUI de SUMO también se puede abrir directamente donde se creó originalmente la carpeta, dentro de la carpeta "bin" (C:\sumo-0.15.0\bin → sumo-gui.exe (Aplicación)).

C.1.2. Instalación e OMNeT++

1. Se procede a descargar el simulador de red OMNeT++ versión 4.2.2 para Windows (64 bits). Para esto, se selecciona el paquete "omnetpp-4.2.2-src-windows.zip" de la página: http://www.omnetpp.org/.

2. Una vez que el paquete se descargue, se copia el archivo en el directorio que se desee (para este caso es en el disco C:\). Posteriormente se dan permisos de administrador (si es necesario) y se procede a descomprimirlo, con lo cual genera una carpeta con el nombre "omnetpp-4.2.2" con todos los archivos necesarios para su instalación.

3. Para realizar la instalación del software abrir la ventana de comandos (cmd.exe), ir a la carpeta donde se genero la carpeta de OMNET++ (C:/omnetpp-4.2.2) y ejecutar el comando "*mingwenv.cmd"* para abrir la consola de configuración, como se observa en la Figura C.4.

Figura C.4. Ejecutando consola de configuración de OMNeT++.

4. Dentro de la consola de configuración se ejecuta el comando "*./configure"* (ver Figura C.5) para realizar la configuración necesaria de las librerías para la instalación del software. Una vez que se confirme que fue exitosa la configuración, se ejecuta el comando "*make"* (ver Figura C.6) para construir los archivos necesarios para el correcto funcionamiento de OMNET++.

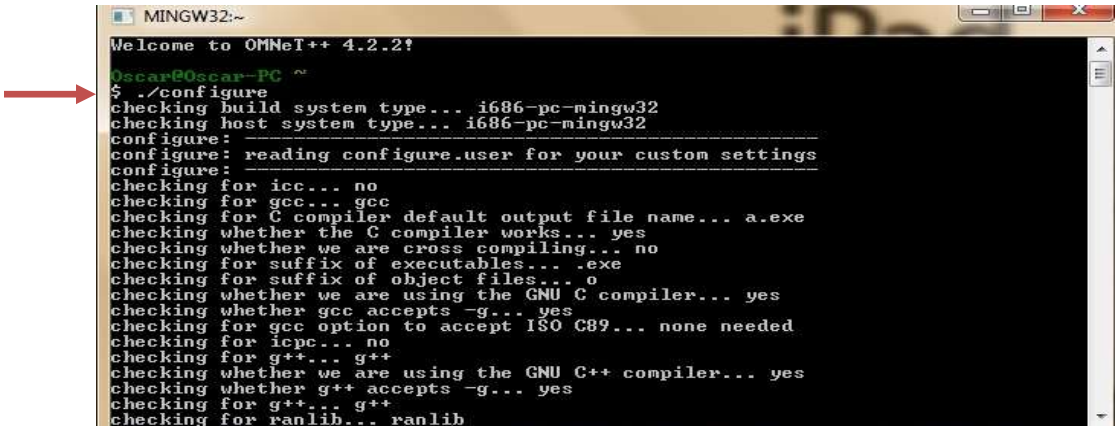

Figura C.5. Consola de configuración – comando *"./configure"*.

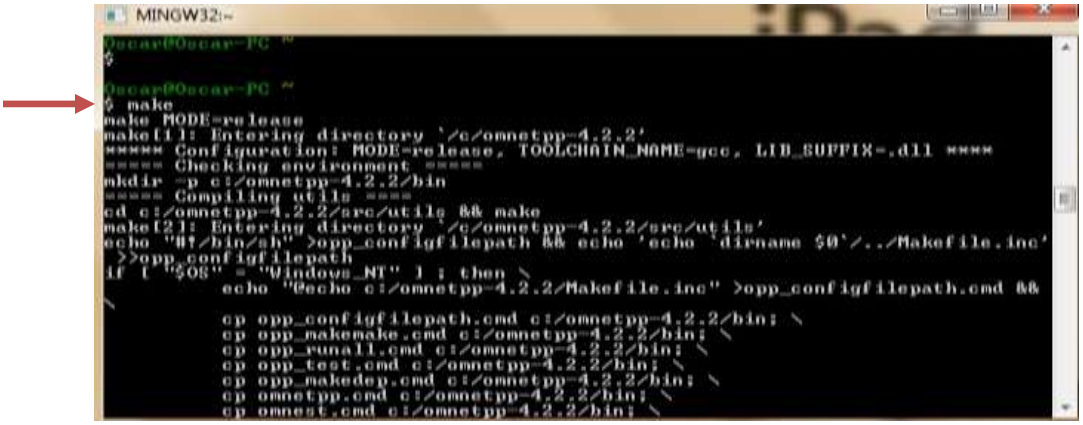

Figura C.6. Consola de configuración – comando "*make"*.

5. Para confirmar que la instalación no haya tenido problemas, se ejecuta el ejemplo "*Aloha"* mediante los comandos marcados con las flechas azules (Figura C.7) dentro de la consola de configuración.

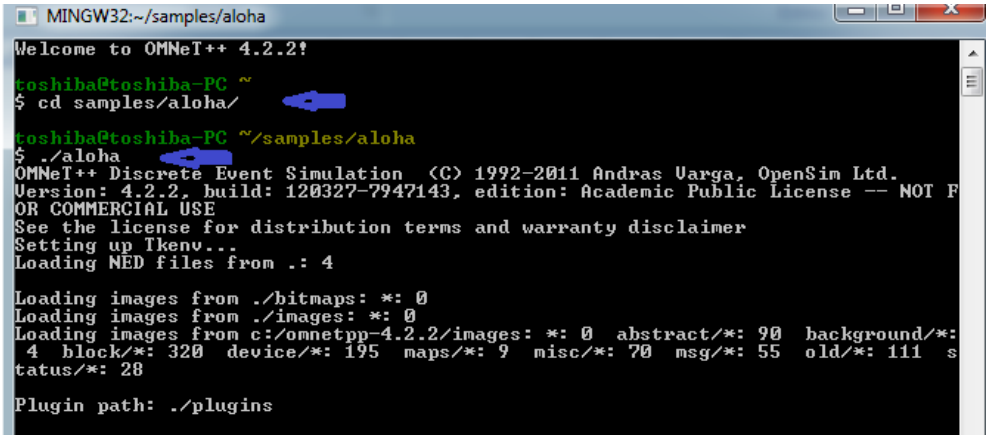

Figura C.7. Ejecutando ejemplo *Aloha*.

6. Una vez ejecutado el ejemplo *Aloha*, se debe abrir la GUI de OMNeT++ denominada "*Tkenv"* junto a una ventana de configuración de la cual se va a escoger la opción por defecto *"pure Aloha, overloaded"* (ver Figura C.8). Una vez que se haga clic en *OK*, se observa de manera grafica el ejemplo escogido como se ilustra en la Figura C.9.

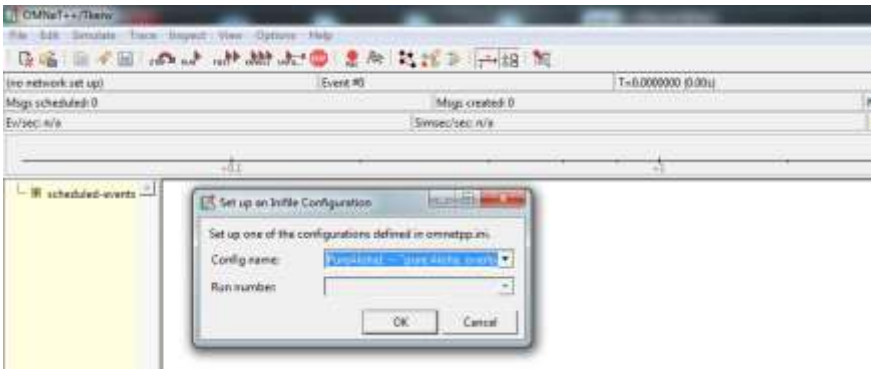

Figura C.8. GUI de OMNET++.

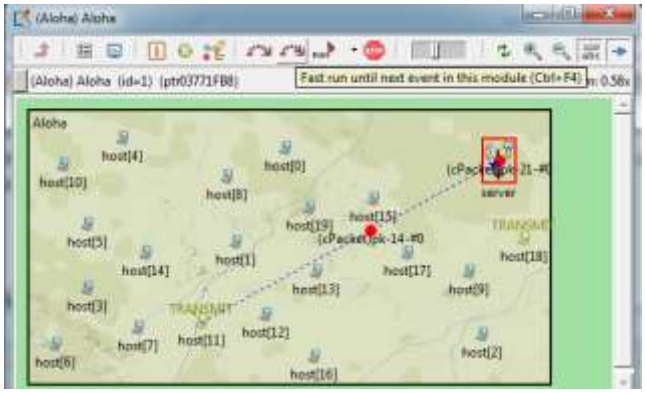

Figura C.9. Interfaz grafica del ejemplo *Aloha*.

C.1.3. instalación de Veins y comunicación entre simuladores

1. Se procede a descargar el simulador híbrido Veins versión 2.0-rc2 para Windows (64 bits). Para esto, se selecciona el paquete “veins-2.0-rc2.zip” de la página: http://veins.car2x.org/download/

2. Al igual que en las herramientas anteriores, se copia el archivo en el directorio que se desee (para este caso es en el disco C:\). Posteriormente se dan permisos de administrador (de ser necesario) y se procede a descomprimirlo, el cual va a generar una carpeta con el nombre “veins-2.0-rc2” con todos los archivos necesarios para establecer la comunicación entre los simuladores.

3. Abrir la interfaz de OMNeT++ ubicada en “C:\omnetpp-4.2.2\bin\omnetpp”. Posteriormente presionar *OK* en la ventana emergente. Después, dar clic en *File* → *Import* → *General* y dar doble clic en *Existing Projects into Woorkspace* (Como se muestra en la Figura C.10).

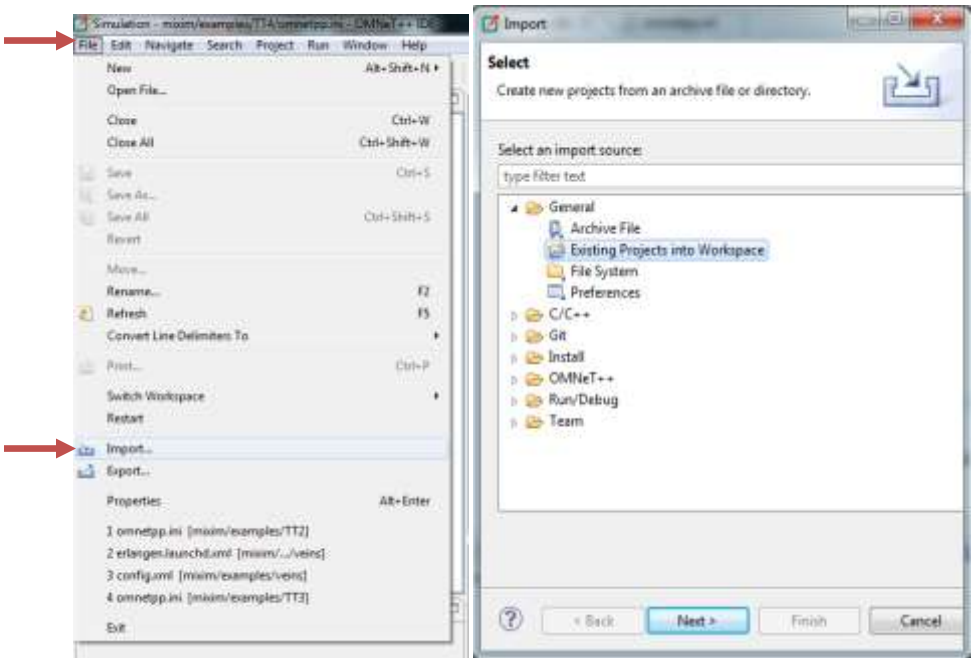

Figura C.10. Interfaz de trabajo de *OMNET++*.

Luego, en la opción *Select root directory* escoger el directorio de Veins (C:\veins-2.0-rc2), presionar *Finish*. Una vez realizado lo anterior, se crea dentro del explorador de proyectos de la interfaz de OMNET++ un archivo denominado “*mixim”* como se muestra en la Figura C.11.

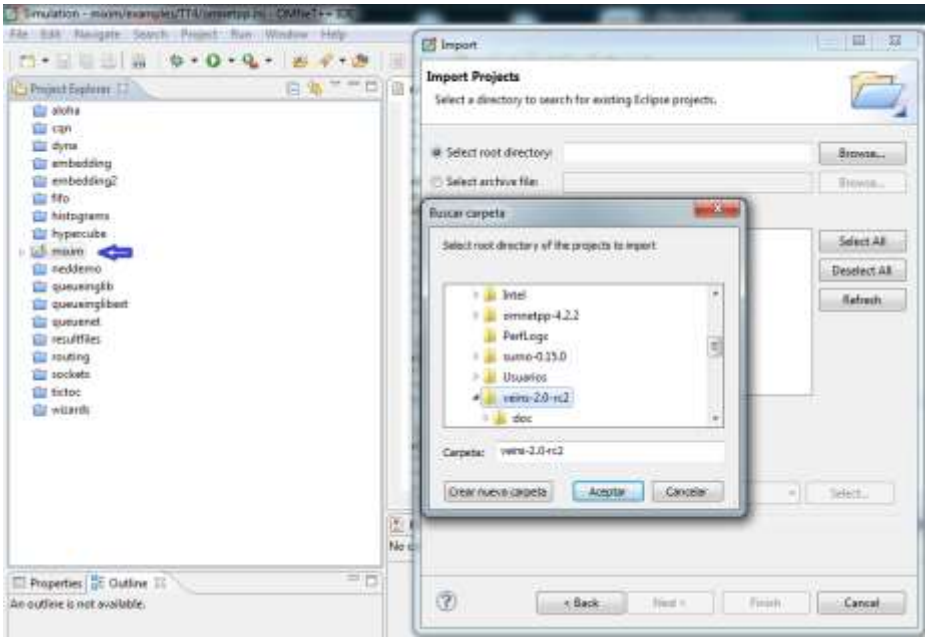

Figura C.11. Proyecto "*mixim"* de VEINS.

4. Para finalizar el proceso de importación de Veins en el explorador de proyectos de OMNeT++, se debe ir a la pestaña *Project → Build All*.

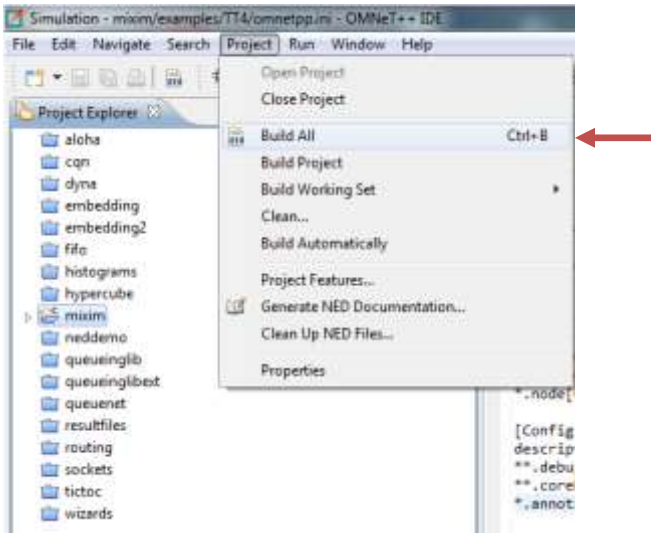

Figura C.12. Importando VEINS.

5. Para terminar, se comprueba el correcto funcionamiento del simulador híbrido interactuando con OMNeT++ Y SUMO, para lo cual existe un script en *python,* el cual se ejecuta dentro de la ventana de comandos con la línea "*/c/veins-2.0-rc2/sumo-launchd.py -vv -c /c/sumo-0.15.0/bin/sumo.exe"*, para que se habilite el puerto 9999, como se muestra en la Figura C.13.

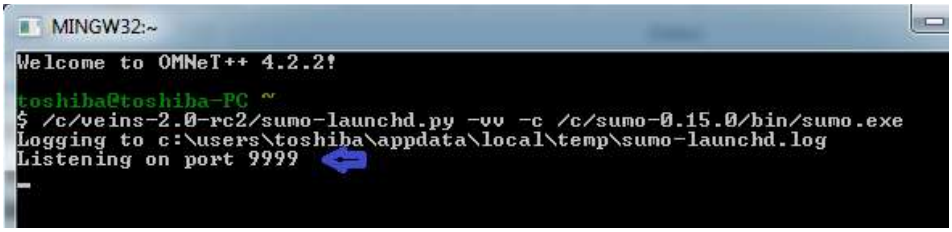

Figura C.13. Habilitando el puerto de comunicación.

Una vez que el puerto por defecto esté habilitado para comunicar a los simuladores, se ejecuta dentro de la interfaz de OMNET++ el ejemplo *mixim → examples → veins* y dentro de ésta se presiona clic derecho sobre "*omnetpp.ini"* y se selecciona *Run As → OMNeT++ Simulation*.

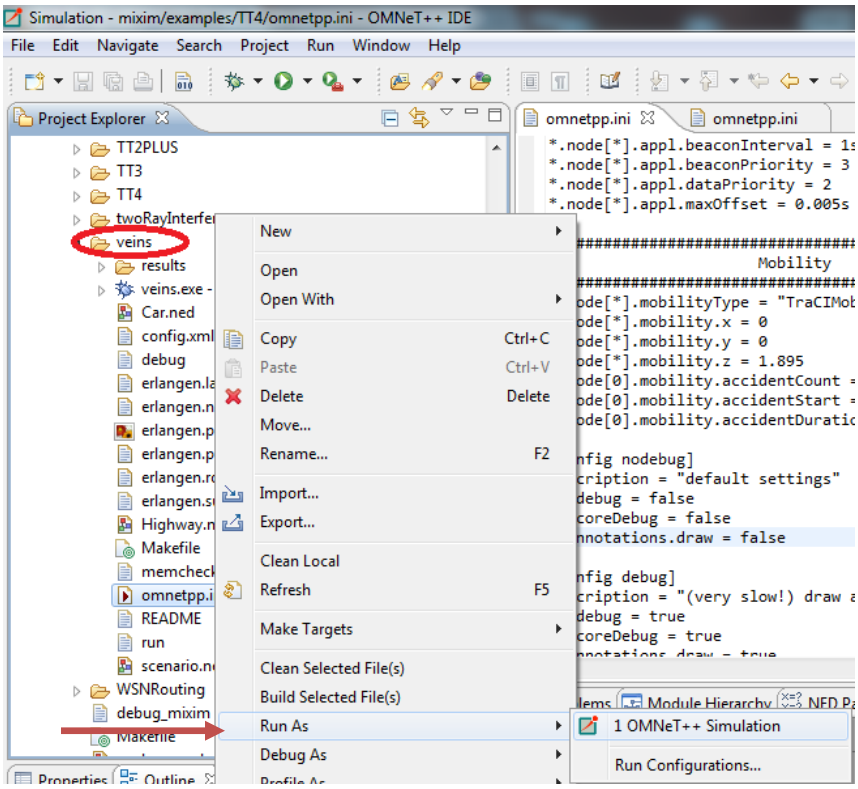

Figura C.14. Ejecutando la simulación de Veins.

Para una primera ejecución del proyecto, es necesario ingresar la ruta del archivo ejecutable de VEINS, como lo muestra la Figura C.15.

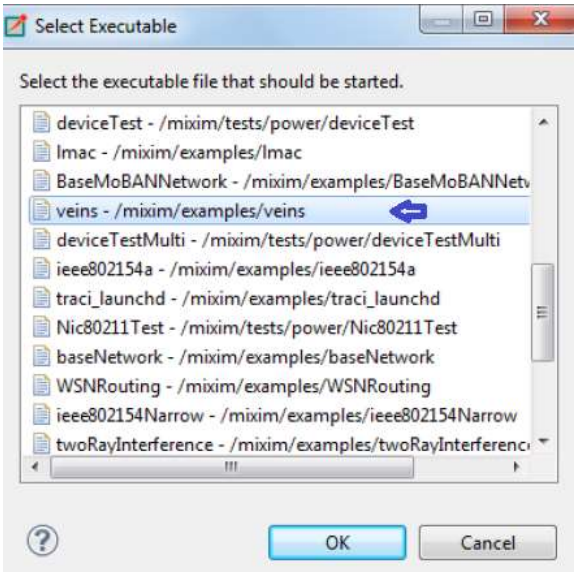

Figura C.15. Ejecutando la simulación de Veins.

Posteriormente se presiona en la flecha RUN en la ventana que emerge (Figura C.16) para que inicie la simulación y, de esta manera quede preparado el simulador *híbrido* para efectuarse la comunicación entre los otros dos simuladores.

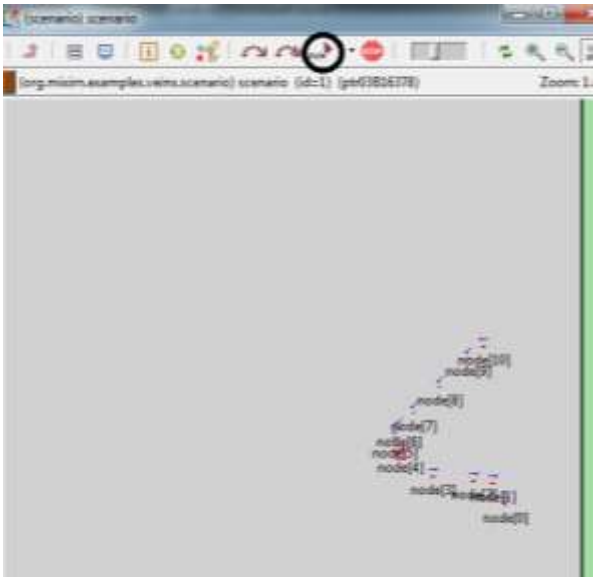

Figura C.16. Ejecutando la simulación.

C.2. MANUAL DE SUMO VERSIÓN 0.15.0

SUMO permite generar una red de movilidad compuesta por una red vial y por el tráfico vehicular que la atraviesa.

C.2.1. Generación de la red vial

Consiste en el trayecto por el cual se movilizan los vehículos en una región determinada y existen diferentes maneras de generar dicho recorrido. En este caso, se enfocó en la importación de mapas mediante fuentes topográficas para trabajar con escenarios reales.

- **Importación de mapas**

Se pueden realizar la importación de mapas de fuentes topográficas tales como *VISUM*, *openDRIVE*, *MATsim*, *OpenStreetMap*, etc. Para este proyecto, como se describe en el Capítulo III sección 3.5.1, se importa el mapa correspondiente a la región abarcada por la glorieta de Santa Clara, en la ciudad de Popayán (Cauca-Colombia) mediante *OpenStreetMap*. Para integrar el mapa de la región escogida con *SUMO*, se deben seguir los siguientes pasos:

1. Como se muestra en la Figura C.17 se procede a ubicar la glorieta de Santa Clara en la página http://www.openstreetmap.org/

Figura C.17. Glorieta de Santa Clara.

2. Luego, hacer clic en la pestaña Exportar y escoger la opción Datos formato *OpenStreetMap XML*, para que sea compatible con *SUMO*. Por último presionar Exportar, el cual el mapa se guardará en la carpeta de descargas con el nombre de “map.osm”.

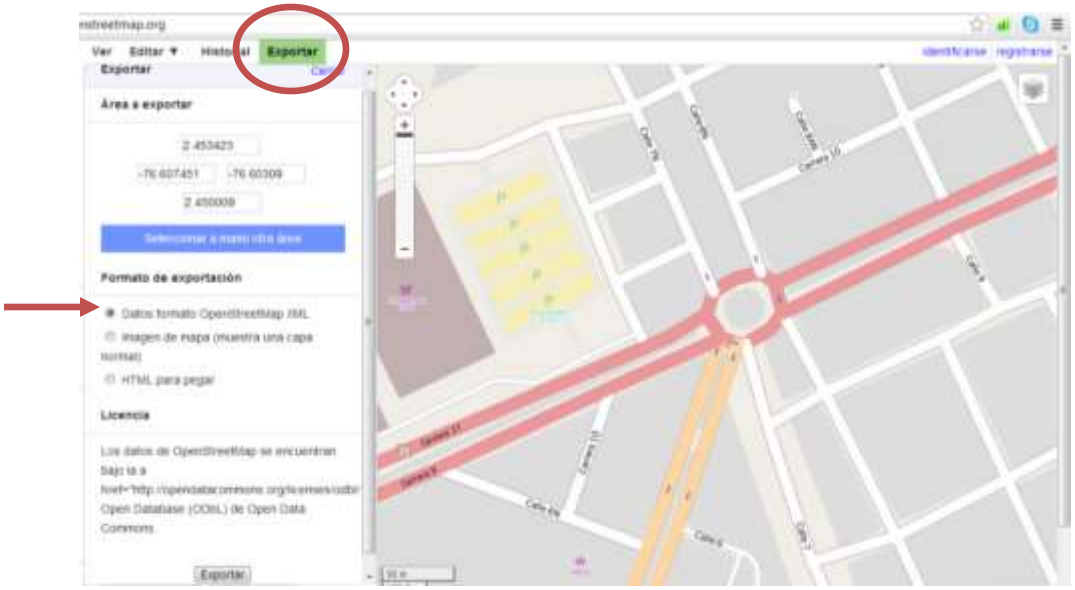

Figura C.18. Exportación del mapa.

3. Copiar el archivo “map.osm” desde la carpeta de descargas hacia la carpeta “bin” dentro de “sumo-0.15.0”.

4. Finalmente, convertir “map.osm” a un archivo de red con extensión <nombre>.net.xml mediante la herramienta NETCONVERT (perteneciente a SUMO) dentro de la ventana de comandos como se ve en la Figura C.19.

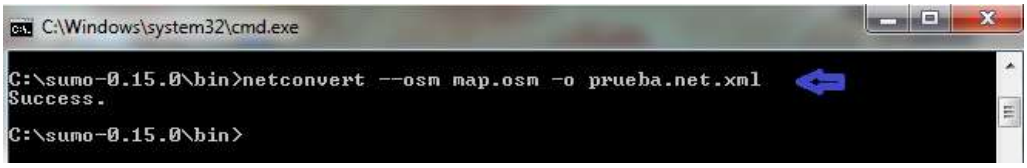

Figura C.19. Generando archivo de red en SUMO.

5. Para comprobar que el proceso fue exitoso, se abre la GUI de SUMO y en la pestaña *File* → *Open Network*, abrir “prueba.net.xml” el cual tiene que mostrar una imagen parecida a la de la Figura C.20.

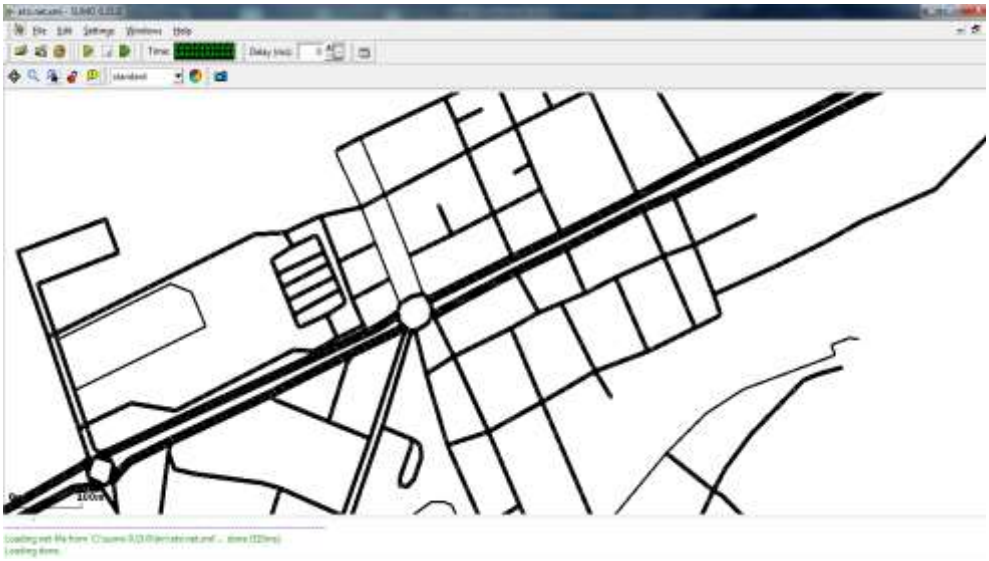

Figura C.20. Mapa “prueba.net.xml”.

- **Editor de mapas de OpenStreetMap de Java (JOSM)**

En caso de que se necesite realizar ajustes o correcciones al mapa, el editor de mapas *JOSM* es una herramienta muy útil para este propósito. A continuación se explica cómo manejar este programa, haciendo exclusivamente énfasis en corregir las imperfecciones que tuvo el escenario de trabajo descrito en este trabajo de grado.

1. Se procede a descargar y posteriormente instalar el editor de mapas *JOSM* para Windows de la página: http://josm.openstreetmap.de/

2. Después, se abre el archivo al que se le quieren realizar los cambios, para este caso “prueba.osm”.

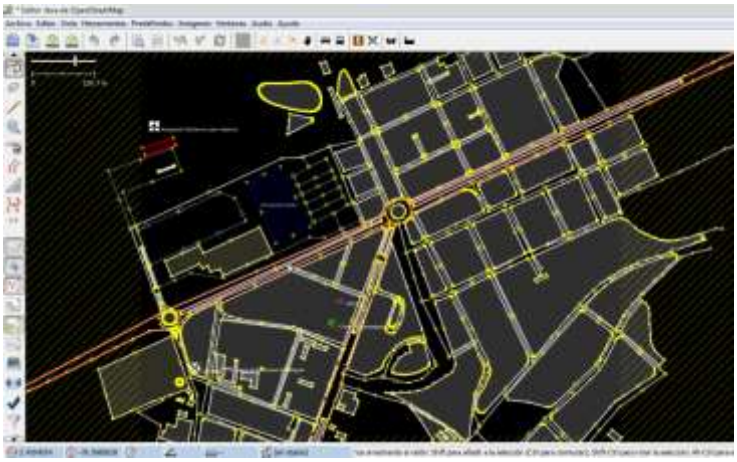

Figura C.21. GUI de JOSM.

3. Para eliminar bloques del mapa que no son importantes en el enfoque del trabajo, simplemente se selecciona la región y se presiona la tela suprimir del teclado.

4. Para que la avenida principal tenga un solo sentido con dos carriles de norte a sur y de sur a norte se debe seleccionar cada uno de las secciones que hacen parte de ella, presionar “Añadir” (en la pestaña de Propiedades) y ubicar *Lanes* con un valor de 2 para que tenga 2 carriles y *oneway* con un valor de yes.

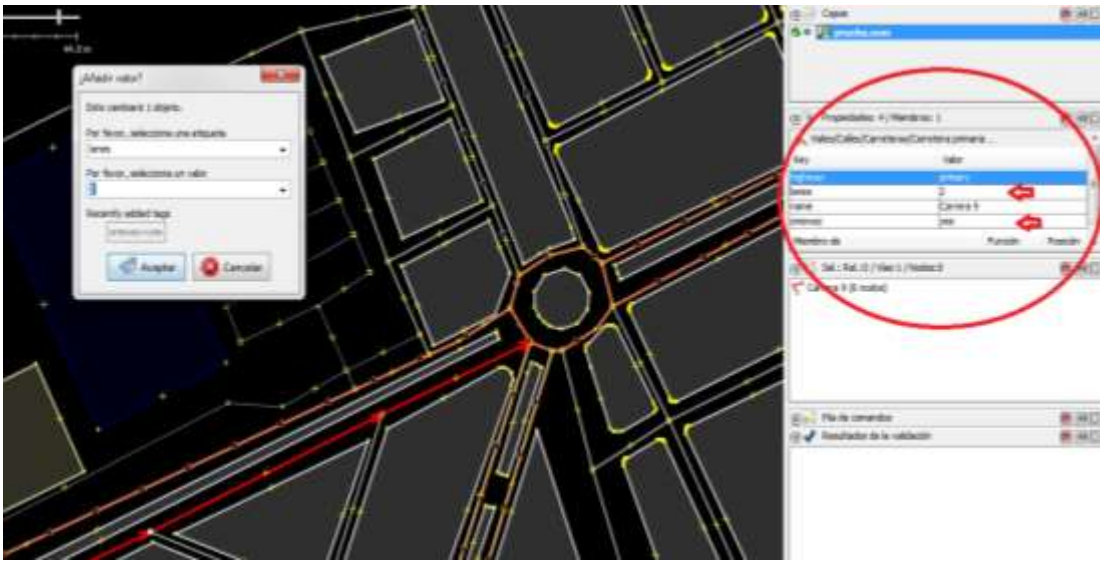

Figura C.22. Edición de carriles.

5. Para modificar el sentido de una vida que no vaya acorde a la realidad, se presiona la tecla R y se selecciona la opción No aplicar cambios.

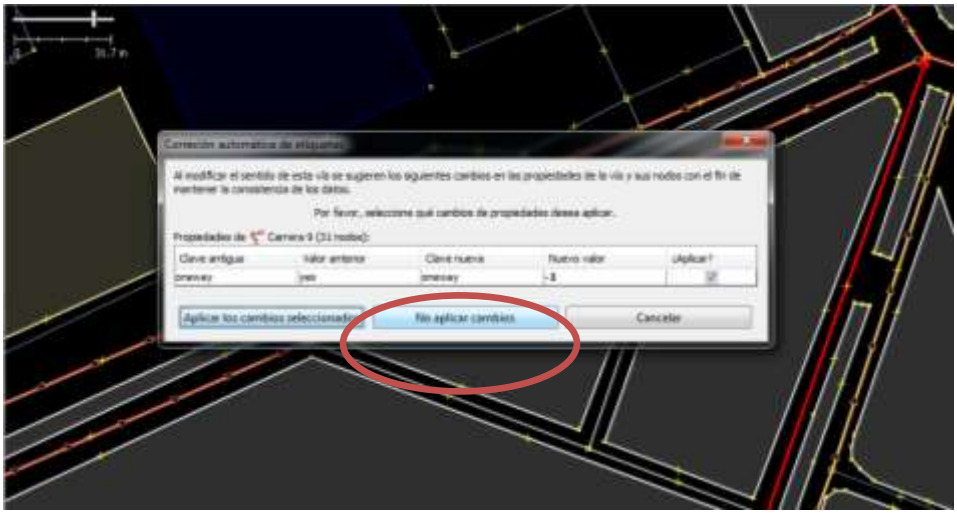

Figura C.23. Edición del sentido de un carril.

6. Para agregar semáforos a una intersección o vía, se debe seleccionar el nodo de interés y seguir la siguiente ruta: Predefinidos → Viales → Nodos de Vías → Semáforo

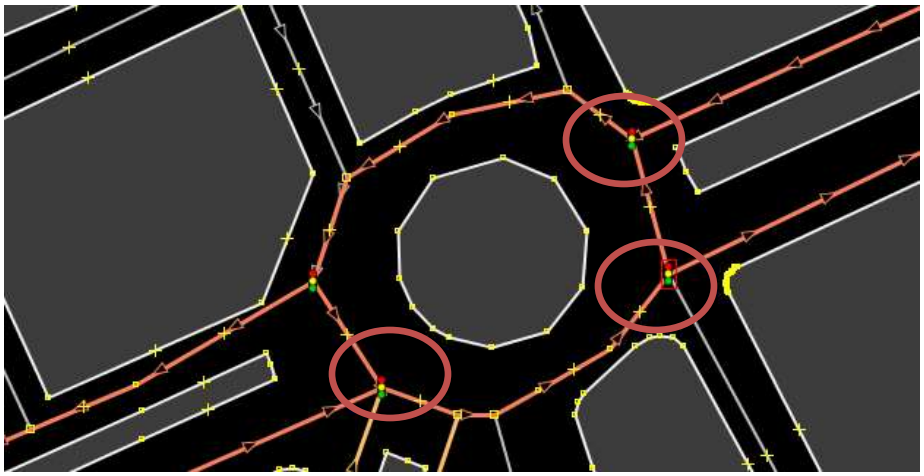

Figura C.24. Adición de semáforos.

Por último se guardan todos los cambios realizados y se convierte el archivo con *NETCONVERT* (como se explicó en el paso 4 de la sección B.2.1 de este anexo). En este caso el mapa se guarda con el nombre de "glorieta.net.xml" con el fin de hacerlo compatible con SUMO.
El tiempo de intervalo entre verde a rojo y viceversa en los semáforos se modifica dentro del archivo "glorieta.net.xml" mediante el código mostrado en la Figura C.25 (este código configura únicamente los 3 semáforos seleccionados con el círculo rojo de la Figura C.24).

```
<tlLogic id="1220068949" type="static" programID="0" offset="0">
    <phase duration="32.04" state="GGrr"/>
    <phase duration="1.82" state="yyrr"/>
    <phase duration="41.54" state="rrGG"/>
    <phase duration="2.4" state="rryy"/>
</tlLogic>
<tlLogic id="799914979" type="static" programID="0" offset="10">
    <phase duration="61.85" state="GGGrrr"/>
    <phase duration="1.85" state="yyyrrr"/>
    <phase duration="71.34" state="rrrGGG"/>
    <phase duration="2.34" state="rrryyy"/>
</tlLogic>
<tlLogic id="980267387" type="static" programID="0" offset="0">
    <phase duration="59.72" state="rrrGGG"/>
    <phase duration="2.72" state="rrryyy"/>
    <phase duration="50" state="GGGrrr"/>
    <phase duration="1.8" state="yyyrrr"/>
</tlLogic>
```

Figura C.25. Configuración de semáforos dentro de "glorieta.net.xml".

- **id:** es la ubicación del semáforo dentro del mapa de trabajo (código único dado por SUMO).

- **type:** hay 3 tipos de semáforo, para este caso es “static”.

- **programID:** el nombre que se quiere que tenga cada uno de los semáforos.

- **offset:** es el tiempo inicial (en segundos) en el que el semáforo empieza a funcionar dentro de la simulación.

- **duration:** es el tiempo de duración (en segundos) de cada una de las fases.

- **state:** Cada una de las fases que conforman el semáforo:

  - ✓ **G:** representa el color verde. Los carros tienen prioridad para pasar la intersección.

  - ✓ **r:** representa el color rojo. Los carros deben estar detenidos.

  - ✓ **y:** representa el color amarillo o anaranjado. Los carros empieza a desacelerar al llegar a una intersección.

C.2.2. Generación del tráfico vial

Una vez que se haya generado el mapa con todas las características necesarias para el desarrollo del proyecto, se procede a generar el patrón de movimiento que se va ajustar a ambientes realísticos de la vida cotidiana. Para cumplir con esto, es necesario crear diferentes archivos con la ayuda de la herramienta MOVE de acuerdo al siguiente procedimiento:

- Generar el flujo de tráfico
- Ejecutar la herramienta MOVE
- Generar el patrón de movilidad
- Simular el patrón de movilidad

a. Generar el flujo de tráfico

Mediante un editor de texto (en este caso se trabajó con *Notepad*++), se genera el patrón de tráfico, el cual define la ruta trazada para un determinado número de carros siguiendo los siguientes pasos:

1. Abrir *Notepad*++ y seguir la sintaxis de la Figura C.26. La configuración que se muestra en dicha figura hace referencia al modelo de movilidad utilizado y varía dependiendo del modelo.

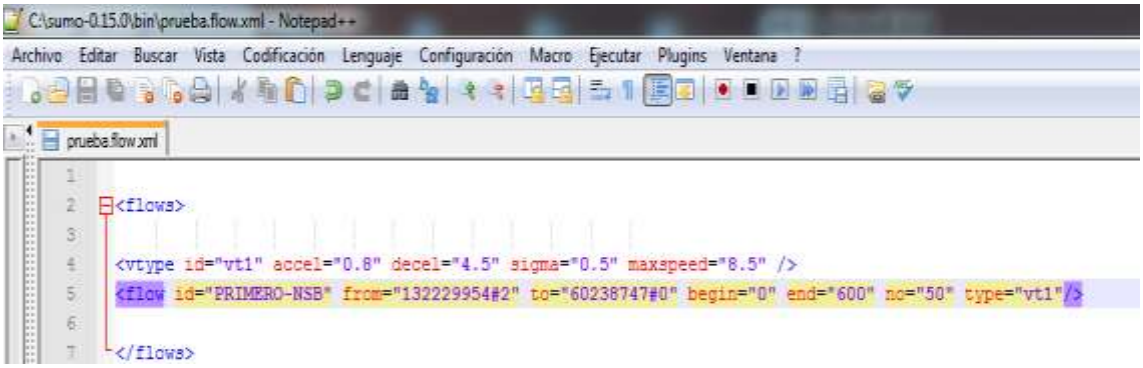

Figura C.26. Sintaxis del flujo de tráfico.

Tanto la información que va en "*from"* (origen) como en "*to"* (destino) de la figura C.26 se extrae de las calles correspondientes al mapa de la sección C.2.1. Para esto, es necesario abrir el mapa de la GUI de SUMO, dar clic derecho sobre la calle y presionar la opción *Copy name to clipboard*.

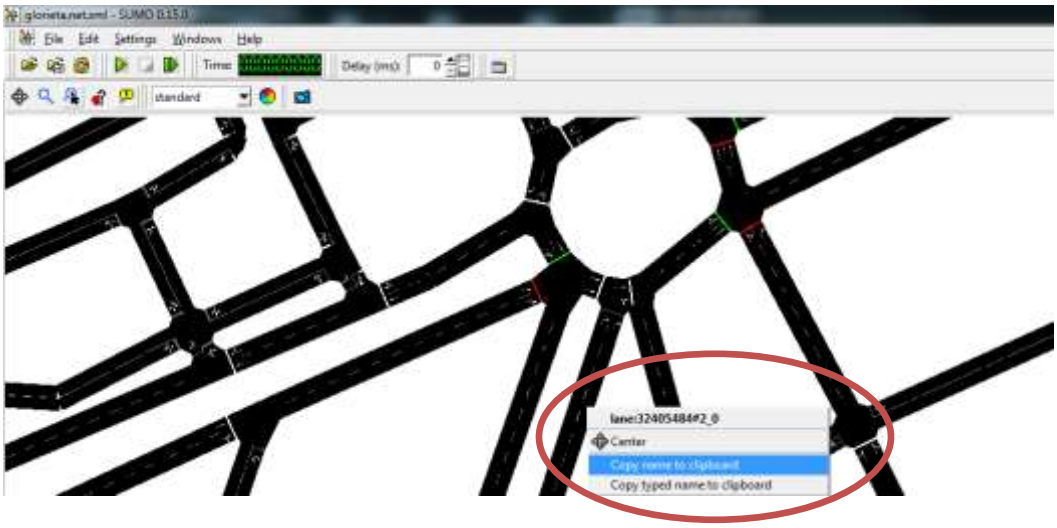

Figura C.27. Calle de origen/destino.

2. Guardar el archivo con la extensión ".flow.xml" en Archivo → Guardar como dentro de la carpeta "sumo-0.15.0\bin". Para este caso se llama "prueba.flow.xml".

b. Ejecutar la herramienta MOVE

MOVE (Mobility model generator for VANET) permite generar el patrón de movilidad de manera rápida, eficiente y fundamentalmente con todas las características necesarias para que el movimiento de los vehículos se ajusten a la realidad y se integre fácilmente con la red vial establecida anteriormente. Para ejecutar esta herramienta es necesario:

1. Descargar el programa MOVE versión 2.9 de la página: http://lens.csie.ncku.edu.tw/Joomla_version/index.php/research-projects/past/18-rapid-vanet.

2. Una vez que el paquete se descargue, se copia el archivo en la carpeta donde esté instalado SUMO (carpeta “sumo-0.15.0”).

3. Para que MOVE pueda utilizar todas las herramienta que tiene SUMO, se ejecuta el siguiente comando en la ventana de comandos:

Figura C.28. Ejecución de MOVE.

4. Posteriormente se abre una ventana en la que se debe escoger “*Mobility Model*” (Figura C.29) para que aparezca la GUI de MOVE (Figura C.30).

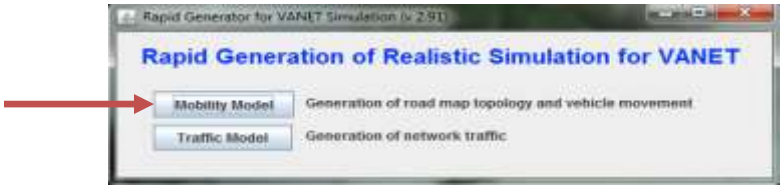

Figura C.29. Generación del modelo de movilidad.

Figura C.30. GUI de MOVE.

5. Por último se da clic en *File* → *Set* SUMO *Binaries Path* e indicar el directorio donde se encuentra instalado SUMO.

Figura C.31. Directorio de ubicación de SUMO.

c. Generar el patrón de movilidad

Una vez que se tenga el archivo ".flow.xml", es necesario generar las rutas definitivas por donde el tráfico vehicular se va a desplazar a lo largo del mapa.
En la interfaz de MOVE, se da clic en el botón "*Create Vehicle"* para abrir el generador de rutas y luego, seleccionar la opción "*Dynamic Router"* para que la ruta sea dinámica y los vehículos eviten tomar una ruta inesperada (esto sucede al seleccionar la opción "*Junction Turning Ratios"*, la cual no es de interés para este trabajo de grado) y sean descartados al llegar al destino. El procedimiento a seguir es el siguiente:

1. Seleccionar la opción "*Using flow definitions file"* y ubicar el archivo de flujo creado anteriormente. En este caso "prueba.flow.xml".

2. En el campo "*Map File"* ubicar el mapa definitivo creado anteriormente. En este caso "glorieta.net.xml".

3. Por último, en el campo "*Set Output File"* colocar el nombre de cómo se llamará al archivo de rutas, con extensión ".rou.xml" (en este caso "prueba.rou.xml"). Después se especifica el tiempo que va a durar la simulación y se presiona el botón *OK*. Una vez que el programa termine de compilar deberá aparecer "*Success"* y quedará guardado dentro de la carpeta "bin" de SUMO, como se muestra en la Figura C.32.

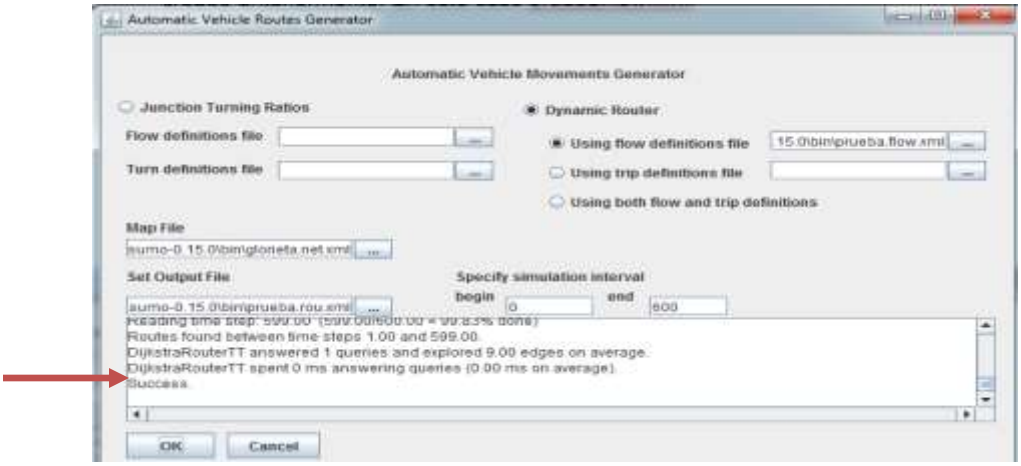

Figura C.32. Generando archivo de rutas.

d. Simular el patrón de movilidad

Una vez que se tenga el archivo rou.xml, se deben efectuar los siguientes pasos:

1. Presionar el botón "*Configuration"* en la sección "*Simulation"* de MOVE, para generar el archivo de configuración, el cual es el encargado de poseer todos los datos necesarios para realizar la simulación. Al igual que en el caso anterior, se debe fijar el tiempo de simulación y se debe ubicar el archivo de mapas en el campo "*Map File"* y el archivo de rutas en el campo "*Routes File"*.

Figura C.33. Generando archivo de configuración.

2. Luego, ir a *File* → *Save as*, darle un nombre con la extensión ".sumo.cfg" y guardarlo en la carpeta "bin" de SUMO. Para este caso se llama "prueba.sumo.cfg".

3. Para la implementación de las paradas de buses se creó un archivo adicional llamado "additional.add.xml" que gestiona los buses que paran, el lugar de detención y el tiempo. Este archivo se muestra en la Figura C.34

```
<additional xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://sumo.sf.net/xsd/additional_file.xsd">
    <busStop id="busstop1" lane="132229954#1_0" startPos="35" endPos="45" lines="100"/>
    <vType id="BUS" accel="0.8" decel="4.5" sigma="0.5" length="8" maxSpeed="8.5" color="1,1,0" guiShape="bus"/>

    <vehicle id="bus0" type="BUS" depart="0" color="1,1,0">
        <route edges="132229954#1 132229954#2 65583516#1 65583516#2 65583516#3 132093236#0 132093236#1 132093236#2"/>
        <stop busStop="busstop1" duration="10"/>
    </vehicle>

</additional>
```

Figura C.34. Generación de paradas de buses.

A continuación se presenta una descripción de los parámetros de la Figura C.34:

- **id:** nombre de la variable
- **lane:** ubicación de la parada en un carril específico del mapa.
- **startPos/endPos:** tamaño inicial/final de las paradas de bus.
- **lines:** nombre de la parada de bus.
- **type:** tipo de vehículo que realiza las paradas.
- **depart:** tiempo de inicio del vehículo en la simulación.
- **color:** color del vehículo en particular.
- **edges:** ruta que sigue el vehículo dentro del mapa.
- **busStop:** instrucción que invoca a la sentencia "id" inicial.
- **duration:** tiempo en segundos que dura la parada de los buses.

De la Figura C.34 es preciso insistir en que ésta muestra únicamente la configuración para la parada de un bus. Si se requiere puede aumentarse el número de paradas dependiendo de los requerimientos particulares.

4. Abrir el archivo creado “prueba.sumo.cfg” con el editor de textos y en la línea de código “<additional-files value” agregar la ruta del archivo “additional.add.xml” descrita en el numeral anterior, como lo muestra la Figura C.35.

```
<configuration>

    <input>
      <net-file value="C:\sumo-0.15.0\bin\glorieta.net.xml"/>
      <route-files value="C:\sumo-0.15.0\bin\glorieta.rou.xml"/>
      <additional-files value="C:\sumo-0.15.0\bin\additional.add.xml"/>
      <junction-files value=""/>
    </input>

    <output>
      <netstate-dump value=""/>
      <tripinfo-output value="output-tripinfos.xml"/>
      <emissions-output value="output-emissions.xml"/>
      <vehroute-output value="output-vehroutes.xml"/>
    </output>
```

Figura C.35. Adicionando ruta del archivo “additional.add.xml”.

5. Desde la interfaz de MOVE en la sección “*Simulation”*, presionar el botón de “*Visualization”* para abrir la GUI de SUMO e ir a *File* → *Open Simulation* y seleccionar el archivo de configuración creado (en este caso “prueba.sumo.cfg”), el cual abrirá el mapa de la región de estudio. La simulación empezará al presionar el triángulo verde ubicado en la barra de herramientas, mostrando unos nodos amarillos circulando a través del mapa como se observa en la Figura C.36.

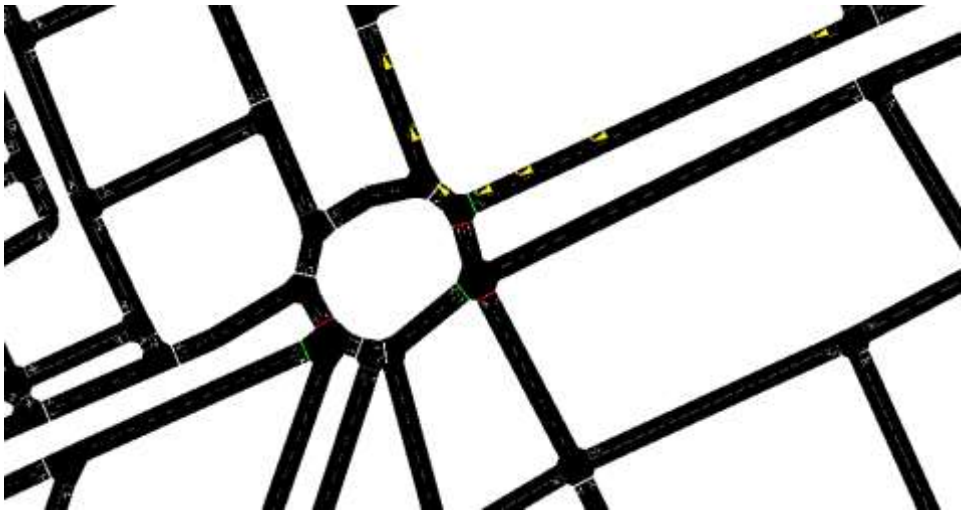

Figura C.36. Simulando una red con tráfico vehicular.

6. Para que SUMO muestre la forma de los vehículos y evite las formas triangulares que se observan en la Figura B.36 se deben seguir los siguientes pasos:

   i. En la GUI de SUMO ubicar la paleta de colores en la barra de herramientas, dar clic en dicha paleta e ir a la pestaña “vehicles” como se muestra en la Figura C.37.
   ii. En la pestaña “vehicles” seleccionada anteriormente establecer la opción “Show As” a “simple shapes” y en la opción “Color by” seleccionar “given/assigned vehicle color”. En la parte superior izquierda de ésta ventana dar clic en el ícono del disquete rojo y establecer un nombre personalizado para ésta configuración. Para este caso se llamó “escenario”. Dar clic en OK y en Use para establecer la nueva configuración. Este proceso se muestra en la Figura C.38.
   iii. Ya con la configuración guardada se puede iniciar la simulación del tráfico vehicular y se muestran los distintos tipos de vehículos recorriendo el mapa, caracterizados por los diversos colores previamente configurados.

Figura C.37. Variación en la forma y color de los vehículos en SUMO.

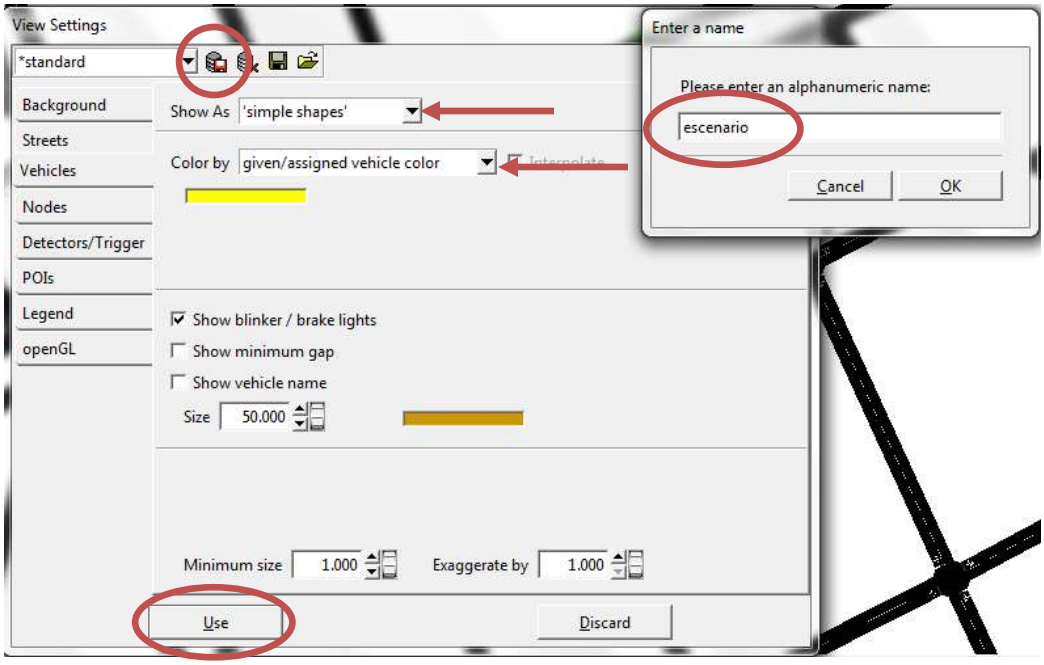

Figura C.38. Finalizando configuración de forma y color de nodos.

La Tabla C.1 especifica los colores soportados por SUMO y su respectivo código descriptivo.

CÓDIGO	COLOR
0,0,0	INVISIBLE
0,0,1	AZUL
0,1,0	VERDE CLARO
0,1,1	AZUL CLARO
1,0,0	PÚRPURA
1,1,0	AMARILLO
1,1,1	BLANCO

Tabla C.1. Códigos de colores para nodos en SUMO. Por los Autores.

C.3. MANUAL DE OMNeT++ Y Veins

Debido a que Veins es un modulo integrado con OMNeT++, el cual permite caracterizar redes VANET y además permite interactuar con SUMO mediante mensajes de comunicación, es indispensable editar el modelo de Veins para generar una red de datos en torno a nuestra área específica de trabajo. Es importante dejar intacto el modelo original en caso de ser necesario.

C.3.1. EDICIÓN DEL MODELO

Para editar en OMNeT++ un modelo sin afectar al original y posteriormente poder reutilizarlo, se debe realizar el siguiente procedimiento:

1. Copiar el modulo de Veins ubicado en "C:\veins-2.0-rc2\examples" y pegarlo dentro de la misma dirección, el cual toca renombrarlo; para este caso se va a llamar "TT" haciendo alusión al trabajo de tesis el cual es la finalidad de este proyecto.

2. Se deben editar los scripts de los archivos "scenario.ned", "Highway.ned" (características generales de la red) y "Car.ned" (comunicación entre nodos) únicamente del módulo que se acaba de crear para evitar errores con el modelo original. De los archivos mencionados anteriormente se debe cambiar la ruta del nuevo paquete como se muestra en la Figura C.39 presionando clic derecho sobre cada uno de ellos y escoger la opción *Open With → Text Editor*.

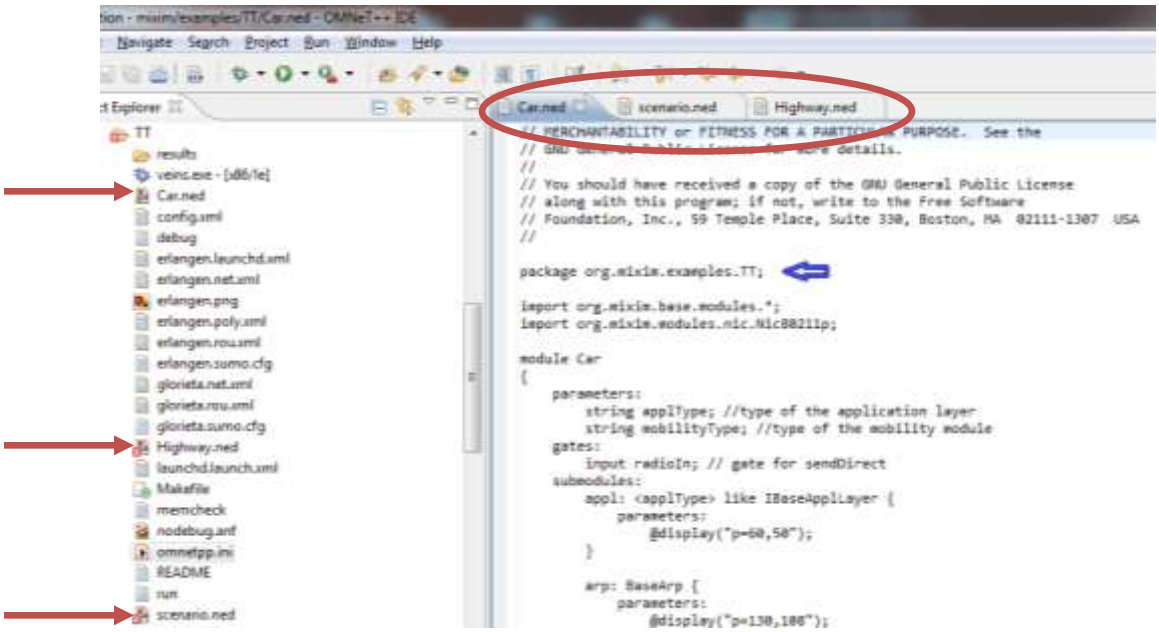

Figura C.39. Edición del script "Car.ned".

3. En "omnetpp.ini" (información de la simulación) se deben realizar los cambios únicamente en el modelo TT, para que el escenario llame al nuevo módulo *Car* y para que el gestor de VEINS llame a los archivos de SUMO respectivamente como se muestra en la Figura C.40.

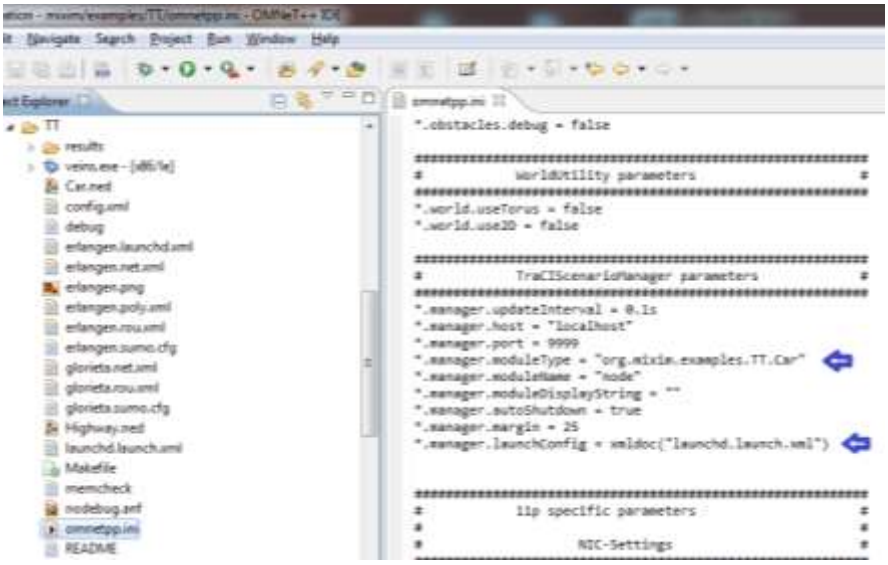

Figura C.40. Editando el archivo “omnetpp.ini”.

4. En el archivo “omnetpp.ini”, dentro de la sección *Mobility*, se va a generar un accidente el cual será detectado por el nodo 9 y empezará 20 segundos después de haberse creado el nodo, con una duración de 20 segundos.

```
##########################################################
#                       Mobility                         #
##########################################################
*.node[*].mobilityType = "TraCIMobility"
*.node[*].mobility.x = 0
*.node[*].mobility.y = 0
*.node[*].mobility.z = 1.895
*.node[9].mobility.accidentCount = 1
*.node[9].mobility.accidentStart = 20s
*.node[9].mobility.accidentDuration = 20s
```

Figura C.41. Creación de accidentes.

5. Para que el tiempo de simulación coincida con el que se configuró en SUMO, se debe dar clic derecho en “omnetpp.ini” y seleccionar la opción *Open With → OMNet++ Ini File Editor*.

Figura C.42. Editando tiempo de simulación.

6. Copiar los archivos de ruta (“prueba.rou.xml”), mapa (“glorieta.net.xml”), adicional (“additional.add.xml”) y el de configuración (“prueba.sumo.cfg”) dentro de la carpeta “C:\veins-2.0-rc2\examples\TT”. Luego, modificar el archivo “prueba.sumo.cfg” eliminando la ruta local de acceso de *.net.xml y *.rou.xml, además borrar la instrucción <srand value = “0.1”> y agregar < step-length value= “0.1”> para evitar que se inhabilite la comunicación entre SUMO y OMNeT++. Este procedimiento se muestra en la Figura C.43.

```
1  <configuration>
2
3      <input>
4        <net-file value="glorieta.net.xml"/>
5        <route-files value="glorieta.rou.xml"/>
6        <additional-files value="C:\sumo-0.15.0\bin\additional.add.xml"/>
7        <junction-files value=""/>
8      </input>
9
10     <output>
11       <netstate-dump value=""/>
12       <tripinfo-output value="output-tripinfos.xml"/>
13       <emissions-output value="output-emissions.xml"/>
14       <vehroute-output value="output-vehroutes.xml"/>
15     </output>
16
17     <time>
18       <begin value="0"/>
19       <end value="600"/>
20       <time-to-teleport value="-1"/>
21       <step-length value="0.1"/>
22       <route-steps value="-1"/>
23     </time>
24
25     <reports>
26       <print-options value="false"/>
27     </reports>
28
29 </configuration>
```

Figura C.43. Archivo “prueba.sumo.cfg” en Veins.

7. Abrir el script “erlangen.launchd.xml” dentro del proyecto TT y modificar las rutas de los archivos como aparecen en la Figura C.44. Después de hacer los cambios respectivos, se guarda el script con el nombre “launchd.launch.xml” para que el gestor los ubique correctamente.

launchd.launch.xml

```
<?xml version="1.0"?>
<!-- debug config -->
<launch>
    <copy file="glorieta.net.xml" />
    <copy file="glorieta.rou.xml" />
    <copy file="glorieta.sumo.cfg" type="config" />
</launch>
```

Figura C.44. Archivo “launchd.launch.xml”.

8. Seguir el paso 5 de la sección C.1.3 de este anexo, pero únicamente hasta la Figura C.15 y, con la diferencia de que ahora el script de python va a ser “*/c/veins-2.0-rc2/sumo-launchd.py -vv -c /c/sumo-0.15.0/bin/sumo-gui.exe”*.

9. Se deben desplegar las GUI de SUMO y OMNeT++ para poder visualizar la simulación establecida. Para ello, primero se ejecuta SUMO para que se encuentre listo para recibir los mensajes y posteriormente el de OMNeT++.

10. Verificar que los vehículos se mueven únicamente siguiendo la ruta de análisis dentro de la GUI de OMNeT++.

11. Una vez finalizada la simulación, para desplegar los resultados en el explorador de proyectos de OMNeT++ ubicar la carpeta con el nombre del proyecto (para este caso es TT), expandir dicha carpeta y abrir la carpeta “Results” para visualizar el archivo “nodebug-0.vec”. Dar doble clic a este último archivo, el programa despliega una ventana emergente donde se pedirá que se cree un archivo con extensión “.anf”. Dar clic en *Finish* en dicha ventana y se despliegan los resultados. Dichos resultados pueden ser filtrados por nodo, variable, tipo de gráfico, entre otros. La Figura C.45 muestra dicho procedimiento.

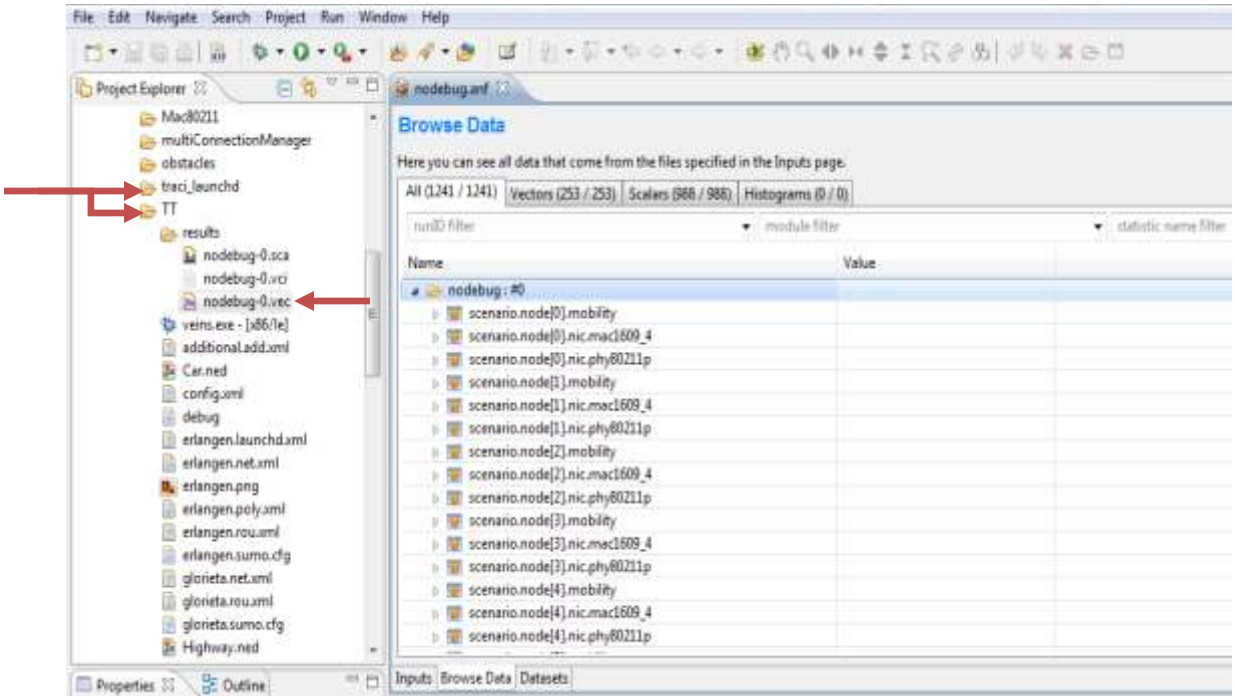

Figura C.45. Desplegando resultados en OMNeT++.

Una vez realizado todos los ajustes explicados anteriormente, el simulador VANET queda listo para empezar a realizar los escenarios para obtener los resultados definitivos del proyecto objeto de estudio.

REFERENCIAS

[1] J. Härri et al., "VanetMobiSim: generating realistic mobility patterns for VANETs," en *Proc. of the 3rd ACM International Workshop on Vehicular Ad Hoc Networks (VANET'06),* Septiembre 29, 2006, Los Angeles, USA.

[2] Visual Traffic Simulation, Tim Fotherby, [Online]. Disponible: http://www.tomfotherby.com/Websites/VISSIM/. [Consultado: 4 Mar 2013]

[3] FreeSim – A Free Real-Time Freeway Traffic Simulator, Jeffrey Miller, Ph.D, [Online]. Disponible: http://www.freewaysimulator.com/index.html [Consultado: 4 Mar 2013].

[4] STRAW – Street Random Waypoint – vehicular mobility model for network simulations. [Online]. Disponible: http://www.aqualab.cs.northwestern.edu/resources/9-projects/144-straw-street-random-waypoint-vehicular-mobility-model-for-network-simulations-e-g-car-networks [Consultado: 4 Mar 2013].

[5] G. Marfia, P. Lutterotti y G. Pau, "MobiTools: An Integrated Toolchain for Mobile Ad Hoc Networks," Reporte Técnico N.070019, Universidad de California Los Angeles, Departamento de Ciencias Computacionales, Octubre 2007.

[6] Traffic and Network Simulator Environment. [Online]. Disponible: http://lca.epfl.ch/projects/trans/. [Consultado: 4 Mar 2013].

[7] SNS: Staged Simulation in NS2. [Online]. Disponible: http://www.cs.cornell.edu/people/egs/sns/ [Consultado: 5 Mar 2013].

[8] About GloMoSim. [Online]. Disponible: http://pcl.cs.ucla.edu/projects/glomosim/. [Consultado: 5 Mar 2013].

[9] JiST – Java in Simulation Time / SWANS - Scalable Wireless Ad hoc Network Simulator. R. Barr. [Online]. Disponible: http://jist.ece.cornell.edu/. [Consultado: 5 Mar 2013].

[10] GTNetS – Home. B. Murray. [Online]. Disponible: http://www.ece.gatech.edu/research/labs/MANIACS/GTNetS/. [Consultado: 5 Mar 2013].

[11] R. Hernández, "Evaluación de herramientas de simulación de Redes Vehiculares," Tesis de Maestría, Escuela Técnica Superior de Ingeniería de Telecomunicación, Universidad Politécnica de Cartagena, Cartagena, España, Octubre 2010.