Universidad del Bío-Bío
Facultad de Ciencias Empresariales
Departamento de Sistemas de Información

Análisis y Simulación gráfica de Algoritmos de encubrimiento para Redes Inalámbricas en Áreas Urbanas

Alumno : Gonzalo Ignacio Saavedra Ramos.
Rut : 17.343.309-3.
Profesor guía : Patricio Galdames Sepúlveda.

Resumen

Este proyecto se presenta para dar conformidad a los requisitos exigidos por la Universidad del Bío-Bío en el proceso de titulación para la carrera de Ingeniería Civil en Informática.

El proyecto titulado "Análisis y Simulación gráfica de algoritmos de encubrimiento para redes inalámbricas en áreas urbanas" fue desarrollado para la Universidad del Bío-Bío, con la finalidad de brindar un estado del arte referente a las técnicas de encubrimiento para Servicios Basados en la Localización (SBL), abordando el problema relacionado con la privacidad de la información de localización de los usuarios de estos, de tal forma de garantizar a cada usuario que solicita un servicio un mayor grado de privacidad de su información utilizando los métodos propuestos en los trabajos estudiados. Finalmente, se desarrolló un simulador gráfico en Java que permite la visualización de diferentes algoritmos de encubrimiento desarrollados por diversos autores. Este simulador permite la visualización de estas técnicas simulando un área urbana basada en la zona céntrica de la ciudad de Concepción, generando usuarios (tanto vehículos como personas) que se mueven constantemente en el área definida. En base a esto, se puede simular la ejecución de cada técnica con respecto de estos usuarios y la zona donde se encuentran.

Este proyecto otorga una base en lo que a Servicios Basados en la Localización respecta y, especialmente, lo que se refiere a las técnicas de encubrimiento de localización para estos. Eliminando en parte el grado de desinformación que se tiene sobre estos y los potenciales peligros que presentan a la privacidad y seguridad de cada usuario.

Abstract

This Project was developed to satisfy the requirements defined and demanded by "Universidad del Bío-Bío" to be able to obtain the title of "Civil Engineer in Computer Science".

The project called "Analysis and Graphic Simulation of cloaking algorithms for wireless networks in urban areas" was developed for "Universidad del Bío-Bío" with the purpose of giving a state of the art in the subject of Location Cloaking Techniques for Location Based Services (LBS), considering the problem associated to the privacy of the location information of an LBS user, to guarantee a higher level of protection to the location information disclosed by an LBS user with the use of the techniques studied and analyzed in this work. Finally, a graphic simulator was developed in the programming language Java to simulate the execution of various algorithms proposed by different authors in their works. This simulator allows to graphically see the execution of a cloaking algorithm by simulating an urban area based on the topology of the centric area of the city of Concepcion, Chile. This simulator generates users (vehicles and regular persons) that constantly move around the defined area. Based on this, it is possible to simulate the execution of each algorithm taking into consideration a specific user and his location on the "map".

This project serves as a base for the subject of Location Based Services, and specifically, covers and brings information on Location Cloaking Techniques developed for these services. With this work, we eliminate some of the misinformation on this subject and identify the problems they bring to the privacy and security of a LBS user.

Diccionario de abreviaciones:

Se definen las siglas que serán utilizadas en este documento.

SBL: Servicios Basados en la Localización, definidos a partir de su denominación en inglés *Location Based Services* (LBS).

Wi-fi: Conexión inalámbrica entre dispositivos. Se usará esta nomenclatura para referirse a este término.

Ad-hoc network: Red de área construida a medida que se conectan nodos. En este trabajo, se hará referencia principalmente a redes móviles de este tipo.

Anonymizer: Servidor de confianza encargado de realizar la comunicación entre el usuario y un Servicio Basado en la Localización.

TC: Trajectory Cloaking (en español: "Encubrimiento de trajectoria"), se refiere al encubrimiento asociado a Servicios Basados en la Localización continuos.

CR: Cloaking Region (en español: "Región de encubrimiento"), se refiere a la región que generan los algoritmos de encubrimiento para ocultar al usuario del servidor.

RW: Random Walk (en español: "Caminata Aleatoria"), uno de los modelos de movimiento estudiados en este proyecto.

RWP: Random Waypoint (en español: "Punto de recorrido Aleatorio"), uno de los modelos de movimiento estudiados en este proyecto.

GPS: Global Positioning System (en español: "Sistema de Posicionamiento Global").

ÍNDICE GENERAL

ÍNDICE TABLAS

ÍNDICE FIGURAS

1 INTRODUCCIÓN

Con los avances en la tecnología y reducción de precios en lo que se conoce hoy en día como tecnologías móviles, cada vez son más personas las que poseen y utilizan diariamente dispositivos móviles. Es por este motivo que se han desarrollado y continúan desarrollando hasta la fecha diversas aplicaciones para estos dispositivos con distintas finalidades, como lo son las aplicaciones que utilizan la información de localización de sus usuarios, más conocidas como Servicios Basados en la Localización (SBL), que tienen la finalidad de otorgar soluciones a problemas del día a día de cada persona, como, por ejemplo: ¿Dónde se encuentra la gasolinera más cercana a mi posición actual? ¿Cuál es el restaurant chino más cercano de donde me encuentro? ¿Dónde se encuentra el hotel más cercano respecto de donde me encuentro?. Esta información es de gran ayuda y utilidad para las personas. En efecto, ayudan a simplificar y mejorar la vida de cada individuo de manera significativa. Sin embargo, la información de localización necesaria para ejecutar estos servicios puede quedar expuesta a las manos de incontables adversarios (tales como personas externas, usuarios maliciosos, o inclusive y el que se considerará en este estudio, el mismo proveedor del SBL), lo cual genera un peligro inmediato a la privacidad y seguridad de cada usuario.

En el aspecto de su privacidad, la información de localización podría delatar fácilmente a una persona, ya que, si se sabe que su ubicación es la oficina de una determinada institución, un adversario puede rápidamente deducir que el usuario que solicitó el servicio es muy probablemente un empleado de dicha institución.

Así también, si un usuario realiza varias consultas en un periodo de tiempo desde una determinada ubicación (por ejemplo, un mall o centro comercial), un adversario puede deducir a partir de esto el estilo de vida, gustos, entre otras cosas de aquella persona. En el aspecto de la seguridad, si un adversario sabe la posición exacta de un dispositivo determinado puede localizar y destruir físicamente dicho dispositivo o cliente, lo cual es de gran importancia especialmente en campos de batalla digitales.

En este trabajo abordamos estos problemas específicamente en el contexto de las redes inalámbricas, en donde un usuario puede obtener su información de localización utilizando por ejemplo el sistema GPS de su dispositivo móvil, y realizar entonces la comunicación directa con un Servicio Basado en la Localización y mediante el mensaje que se genera en esta comunicación

entregarle su información de localización exacta a este con el fin de obtener una respuesta deseada. Es en este momento en que su identidad se ve comprometida, ya que, aunque utilice seudónimos u otras formas similares de ocultar sus datos del servidor esto no es suficiente para proteger su privacidad, debido a que la información de localización contenida en este mensaje puede contener información sensible del usuario que el SBL puede utilizar y finalmente re identificarlo, para posteriormente usar estos datos con fines de beneficio propio o inclusive entregar esta información a terceros.

A partir de estas problemáticas, surge este proyecto, que tiene la finalidad de dar a conocer algunos problemas que existen en la actualidad con los servicios basados en la localización en relación a los riesgos asociados al *grado de anonimato, privacidad de localización* y *seguridad física de localización* que pueden traer a los usuarios de redes inalámbricas, como así también estudiar y analizar diferentes técnicas propuestas por distintos autores para poder enfrentar y resolver satisfactoriamente estos problemas. Finalmente, se desarrollará una aplicación Java implementando dos técnicas en particular, las cuales se seleccionarán luego de realizar el estudio de cada una, con el fin de seleccionar las que otorguen una solución más completa y eficiente a la problemática presentada.

Este trabajo se divide entonces de la siguiente manera:
En el capítulo 1, se presentan los objetivos del proyecto y una breve introducción al área y temática de los Servicios Basados en la Localización (SBL), desde sus orígenes hasta su actual popularización y utilización masiva alrededor del mundo. En el capítulo 2 se presenta en detalle el problema de despersonalización de la ubicación presente en los Servicios Basados en la Localización. También se mencionan y describen trabajos relacionados a esto, dando una breve descripción de lo que desarrollan y los resultados obtenidos en estos. Posteriormente se presenta un análisis y descripción detallada de cada técnica estudiada en este proyecto. Luego, se plantean otros problemas que hemos mencionado con anterioridad, se describe, define y analiza lo que corresponde a los problemas relacionados al grado de anonimato de los usuarios, de privacidad de localización y por último los problemas de la seguridad física de localización. Este capítulo se da por finalizado con la realización de una comparación de cada técnica analizada en base a criterios que hemos definido, mediante la generación de una tabla comparativa que muestre claramente las ventajas, desventajas y alcances de cada técnica, para así seleccionar dos de estas que consideremos que entreguen una solución óptima para su posterior simulación. En el capítulo 3 realizamos la presentación y desarrollo de las simulaciones gráficas (esto en forma de una aplicación Java de

escritorio) de las técnicas seleccionadas con anterioridad, teniendo en cuenta una que tome en cuenta las restricciones de movimiento de un usuario y otra que permita el movimiento libre de este en el mapa, y así obtener conclusiones claras sobre cómo se comportan los diferentes algoritmos implementados en la simulación. Para poder realizar estas simulaciones, definimos de forma breve, consideramos y utilizamos los algoritmos de movimiento de usuarios "*Random Walk*" y "*Random Waypoint*" los cuales generan y simulan el movimiento de una cantidad determinada de usuarios en un mapa previamente generado para la implementación de las simulaciones. En el capítulo 4, se realiza la especificación de requerimientos de software. En el capítulo 5, se presenta el análisis de software, en el cual se presentan los diagramas asociados a este, como el Diagrama de Clases y de Casos de Uso, así también como la descripción y especificación de Casos de Uso. Finalmente, se concluye este trabajo con trabajos futuros que puedan surgir a partir de los temas que se abordaron en esta investigación, y que complementen y ahonden aún más en esta área, además de otorgar las conclusiones pertinentes al análisis que se ha realizado a lo largo de nuestra investigación.

Para finalizar, esperamos que este trabajo sea de ayuda y elimine un poco el desconocimiento que se tiene referente al tema de la seguridad de la información de localización, específicamente de los propios usuarios que utilizan SBL, debido a que la mayoría de las personas que los utilizan no están al tanto de los riesgos a los que se ven expuestos de forma directa o indirectamente. Y así, remarcar la importancia que tiene el mantener seguros los datos de cada usuario.

2 DEFINICION DE LA EMPRESA O INSTITUCIÓN

2.1 Descripción de la empresa

Este trabajo se desarrolla para la Universidad del Bío-Bío, esta tiene sus orígenes en el año 1947, cuando fue creada bajo el nombre de "Universidad Técnica del Estado" (UTE). Posteriormente esta fue adaptándose y creciendo, hasta convertirse en lo que se conoce hoy en día como "Universidad del Bío-Bío", siendo la única institución de educación superior pública y estatal de la región.

Antecedentes generales de la Empresa

- Nombre: Universidad del Bío-Bío.
- Dirección: Avenida Collao 1202, Concepción.
- Rubro: Educación.
- Productos – Servicios que ofrece: La Universidad del Bío-Bío ofrece servicios de educación superior en el estado de Chile

Visión:

Ser reconocida a nivel nacional e internacional como una Universidad pública, responsable socialmente y regional que, comprometida con su rol estatal, desde la Región del Biobío, forma personas integrales de excelencia y aporta a través de su quehacer al desarrollo sustentable de la región y el país.

Misión:

La Universidad del Bío-Bío, a partir de su naturaleza pública, responsable socialmente y estatal, tiene por misión, desde la Región del Biobío, aportar a la sociedad con la formación de personas integrales, a través de una Educación Superior de excelencia. Comprometida con los desafíos de la región y del país, contribuye a la movilidad e integración social por medio de; la generación y transferencia de conocimiento avanzado, mediante la docencia de pregrado y postgrado de calidad, la investigación fundamental, aplicada y de desarrollo, la vinculación bidireccional con el medio, la formación continua y la extensión. Asimismo, impulsa el emprendimiento y la innovación, el fortalecimiento de la internacionalización y el desarrollo sustentable de sus actividades, basada en una cultura participativa centrada en el respeto a las personas.

3 DEFINICIÓN PROYECTO

3.1 Objetivos del proyecto

Objetivo general:

- Simular gráficamente el uso de técnicas de encubrimiento de ubicación para usuarios de redes inalámbricas en un entorno urbano, como la zona céntrica de la ciudad de Concepción, Region del Bío-Bío, Chile.

Objetivos específicos:

- Comparar las técnicas de encubrimiento de ubicación desde el punto de vista de la forma de la región de encubrimiento que generan (puntos, área, segmentos), tamaño de la región de ocultamiento, tipo de datos requeridos (número "k" específico, área pública, entre otros), tipo de arquitectura de red utilizada (centralizada, distribuida, entre otras), tipo de protección que brindan (anonimato, protección de privacidad, seguridad de localización), resistencia frente diferentes ataques a SBL, entre otros puntos.
- Seleccionar el uso de dos técnicas de encubrimiento. Una que no asume restricciones en las direcciones de movimiento de los usuarios y otra que si las considere.
- Estudiar dos modelos de movimiento de usuarios: Random Walk y Random Waypoint.
- Implementar un simulador gráfico que visualice la ejecución de las técnicas seleccionadas con los modelos de movimiento propuestos en un área basada en la ciudad de Concepción.

3.2 Historia de los Servicios Basados en la Localización

Breve definición de Servicios Basados en la Localización:

Los servicios basados en la localización son programas computacionales que utilizan la *información de localización* para otorgar así una variedad de servicios a sus usuarios finales. Pertenecen por tanto a la categoría de *servicios de información* y poseen hoy en día un sinfín de usos en las redes sociales como un servicio netamente de entretención, el cual es accesible a través de dispositivos móviles por medio de *las redes móviles* (Wi-Fi, 3G, 4G) en los cuales utiliza la información correspondiente a la *posición geográfica* del dispositivo móvil. Estas tecnologías se han hecho cada vez más importantes y populares con el avance tecnológico y, particularmente, la disminución en los precios de los dispositivos móviles como Smarthphones y tablets en el mercado actual.

Los SBL son usados en una variedad de áreas y contextos, como lo son: salud, búsqueda de objetos, entretenimiento, trabajo e inclusive estilos de vida y la vida personal de un individuo.

Particularmente, los SBL incluyen servicios para identificar la localización de una persona u objeto, y así, otorgar un servicio de utilidad a cada persona. Como, por ejemplo: ¿Dónde se encuentra el cajero automático más cercano a mi posición? ¿Cuál es el restaurant más cercano a donde me encuentro actualmente? ¿Qué tan lejos se encuentra de mi posición actual uno de los empleados de mi empresa? Entre otros.

Historia:

Antiguamente, los SBL se encontraban disponibles e implementados en una pequeña fracción de las aplicaciones y funcionalidades computacionales existentes. Sin embargo, hoy en día se encuentran prácticamente en todas las aplicaciones de control a nivel computacional, y específicamente tecnologías móviles utilizadas alrededor del mundo. Han evolucionado de ser simples modelos de servicios basados en la sincronización a ser sistemas complejos de autentificación para implementar casi cualquier modelo de servicios basados en la localización.

Son activamente utilizados hoy en día en todo el mundo, y corresponden a una de las aplicaciones de decisión más usadas en computación actualmente.

Algunos de los precursores de los SBL de hoy en día incluyen al *Infrared Active Badge System (1989-1993)*, el *Ericsson-Europolitan GSM SBL Trial* por *Jörgen Johansson (1995)*, y la tésis de Magíster escrita por el empleado de Nokia *Timo Rantaleinen (1995)*.

El primer servicio basado en la localización como nosotros lo conocemos fue lanzado en el año 2001 por TeliaSonera en Suecia (Encuentra a un amigo, páginas amarillas, posición del hogar, localización de llamada de emergencia, etc), y por EMT en Estonia (localización de llamada de emergencia, encuentra a un amigo, juego de TV). TeliaSonera y EMT basaron su servicio en el Sistema de Posicionamiento Móvil Ericsson (MPS).

En esta misma fecha diferentes empresas en diversos países comenzaron a lanzar SBL similares a los mencionados anteriormente, y fue creciendo su popularidad y aceptación por parte de los usuarios en todo el mundo. La mayor ventaja que poseían estas aplicaciones y que las hacían atractivas para todas las personas fue que no era necesario introducir ningún código ZIP ni nada parecido, inclusive al cambiar de localización. El sistema GPS se encargaba de otorgar estos datos usando la web móvil.

3.2.1 ¿Qué es un Servicio Basado en la Localización?

Un Servicio Basado en la Localización puede ser definido de forma simple como servicios que integran la información de localización de un usuario (móvil, generalmente) con otra información de tal manera de otorgar una respuesta con un valor agregado a un usuario en particular.

Los Servicios Basados en la Localización utilizan y aprovechan esta información de localización para brindar a un usuario un servicio específico, como, por ejemplo, una aplicación capaz de encontrar a una persona en particular (estas aplicaciones son conocidas como *Friend Finder*), ubicar la clínica de salud u hospital más cercano al usuario, la estación de gasolina que se encuentra más cerca, el restaurant de comida China que se encuentra más cerca, etc. Todas estas aplicaciones poseen la característica en común de que entregan información de utilidad para el día a día de las personas, y solo necesitan la información de localización de un usuario, lo cual es fácilmente accesible utilizando las tecnologías móviles de hoy en día, mediante los sistemas web y de GPS que estas poseen.

Existen hoy en día una variedad de estos servicios y a su vez, proveedores de estos servicios (SBL Service Provider). Desde aplicaciones desarrolladas por las mismas empresas de telecomunicaciones para teléfonos móviles, hasta aplicaciones desarrolladas por empresas externas e inclusive freelancers (en el caso del sistema Android). Algunos SBL utilizan y requieren del registro de cada usuario, solicitando información adicional tal como el nombre completo, edad, estado civil y cedula de identidad del usuario, sin embargo, este no es un requerimiento común para los SBL de hoy en día especialmente, y más aún, no es necesario para que este tenga un correcto funcionamiento.

Esta clase de SBL es la que presenta una mayor cantidad de riesgo para cada persona, debido a la cantidad de información personal que maneja, la cual puede ser fácilmente expuesta y extraída por los ataques de un adversario. Es por esta razón, que la gran mayoría de los proveedores de servicios optaron por mantener esta información oculta, otorgando a cada usuario un "alias" de forma de esconder su información real del ataque de un adversario. Sin embargo, a pesar de estas medidas de seguridad, los SBL presentan grandes problemas de privacidad y seguridad que hasta hoy en día investigadores alrededor del mundo buscan el método óptimo y más efectivo para poder resolverlos.

3.2.2 Desarrollo de tecnologías móviles e incremento de la accesibilidad a estas

Inicialmente, las tecnologías móviles solamente existían como aparatos analógicos capaces de otorgar la posibilidad de comunicación entre usuarios, mediante la utilización de ondas de radio que permitía realizar llamadas de voz. Esta tecnología es conocida hasta el día de hoy como **1G** (First Generation). Luego del éxito del lanzamiento de estos innovadores dispositivos, que originalmente fueron lanzados con el fin de mantener a las tropas comunicadas en la segunda guerra mundial, comenzó a surgir la necesidad de crear masivamente estos dispositivos para mantener en contacto a importantes empresarios, quienes necesitaban estar comunicados las 24 horas del día y en cualquier lugar.

En el año 1990, en conjunto con la consolidación de las computadoras personales y las redes informáticas, nace en Europa lo que se conoce como **2G** (Second Generation), la segunda generación de teléfonos celulares capaces de mantener comunicada a las personas todo el tiempo. Con el perfeccionamiento y aumento en el conocimiento tecnológico, 2G presentó considerables mejoras respecto de su predecesor. Principalmente, la diferencia radicaba en que esta es una tecnología completamente digital, y los protocolos de comunicación son mucho más sofisticados.

Posteriormente a la expansión a nivel mundial y éxito rotundo de las tecnologías 2G, comenzaron a surgir mejoras en esta misma, lo cual es considerado como **2.5G**. Las principales optimizaciones y actualizaciones que se realizaron fueron netamente en el ámbito de la mensajería, mejorando el actual sservicio SMS con EMS, el cual permite enviar melodías e iconos dentro de los mensajes, a diferencia de SMS que solo soportaba texto. A su vez, se incorporó MMS, el cual fue una mejora al servicio EMS, permitiendo la inclusión de música, imágenes y videos en los mensajes.

Respondiendo al continuo crecimiento tecnológico y avances en el área, en el año 2001 nace en Japón la tecnología **3G** de celulares. Superando y mejorando inclusive a la increíble 2G, estos dispositivos están basados en los SGTM (Servicios General de Telecomunicaciones Móviles), e incluyen la implementación de aplicaciones personalizadas, mejor acceso a internet, velocidad de procesamiento más rápida, entre otros.

Hoy en día, se ha llegado inclusive a superar la tecnología 3G, desarrollando y aplicando la tecnología de celulares y dispositivos móviles **4G**. Orientada completamente al uso e interacción con internet, combinando el uso también de Wi-Fi y WiMax. Realmente, no existe una definición exacta de lo que corresponde a 4G, pero se entiende que su objetivo principal es implementar y mejorar tecnologías ya conocidas y diferentes protocolos con la finalidad de garantizar la mayor velocidad de procesamiento con el uso de las redes inalámbricas más económicas y accesibles.

Aunque en un principio los primeros dispositivos móviles y teléfonos celulares eran extremadamente costosos, debido al desconocimiento general que había en el área (pocos desarrolladores y posibles usuarios), estos fueron pensados para ser utilizados solamente por personas específicas (empresarios, ejercito y gente de gran importancia). Sin embargo, con el paso del tiempo y el aumento tecnológico estos dispositivos se han vuelto accesibles para la gran mayoría de la población, e inclusive se podría decir que necesarios.

Esto ha influido directamente en el crecimiento exponencial del uso de SBL, debido a que la mayor parte de estos corresponden a aplicaciones móviles de interés para cada usuario.

3.2.3 Popularización de SBL para tecnologías móviles

Los SBL comenzaron siendo simples modelos de Servicios Basados en Sincronización (Synchronization Based Services), y han evolucionado hasta lo que hoy en día conocemos, siendo complejos sistemas y aplicaciones computacionales.
Desde el surgimiento y exponencial crecimiento de las tecnologías móviles, se abrió a su vez una ventana de oportunidad en un nuevo mercado para distintos proveedores de servicios y desarrolladores de aplicaciones. Poco a poco, a medida que evolucionaban las tecnologías celulares 1G, 2G, 3G, 4G, los usuarios comenzaron a requerir servicios que pudieran utilizar en sus dispositivos móviles que les brindarán soluciones y cubrieran sus demandas en el momento.
Es así como comienza el aumento y variedad de SBL que otorgaban exactamente lo que los usuarios necesitaban

Con la capacidad de tener acceso a internet desde cualquier lugar en el que se encuentre una persona, mediante el uso de un teléfono celular o dispositivo móvil (Tablet, iPad, etc), aumentó la cantidad de información a la que un usuario puede tener acceso en cualquier momento, y también las acciones que puede realizar con su dispositivo. Utilizando el servicio de internet, un SBL puede solicitar la información de localización del usuario y ocuparla para otorgar una solución eficiente y oportuna a una necesidad especifica del usuario. Ya no es necesario que el usuario identifique manualmente (mediante una búsqueda en google, por ejemplo) un hotel que se encuentre en su ciudad y cercanías; el SBL realiza este trabajo rápido y automáticamente.

Es por este motivo, y gracias a la gran variedad de áreas de interés que pueden cubrir los SBL, que estos corresponden actualmente a una de las aplicaciones móviles más utilizadas a nivel mundial, sino las más usadas. Estas se encuentran en la gran mayoría de dispositivos móviles accesibles para hombres y mujeres promedio, y son usadas diariamente por millones de personas.

4 ESTADO DEL ARTE

4.1 Problema de despersonalización de la ubicación en redes inalámbricas

Aunque la invención, utilización y posterior popularización de los Servicios Basados en la Localización han traído un gran número de mejoras y beneficios para los usuarios de Smarthphones y diferentes tipos de dispositivos móviles, estos también traen consigo peligros y problemas de gran importancia para las personas. Específicamente, tan solo al momento de solicitar un servicio a un servidor de SBL, la privacidad de la información de localización de un usuario se ve comprometida. Esto ya que, al momento de revelar su localización exacta, un usuario está exponiendo su identidad, información asociada a él (gustos, forma de vida) entre otros datos al SBL.

Esto es debido principalmente a que han surgido diversos trabajos dedicados a proteger la identidad e información de localización de un usuario ([2], [3], [5], [16], [18], [19]), mediante la *despersonalización* de esta. Considerando que el problema consiste en que cada usuario revela su información de localización con demasiada precisión, es necesario entonces hacerla menos precisa, es decir, despersonalizarla al punto de que no se sepa con certeza a que usuario pertenece realmente.

De aquí entonces se propone y da nacimiento el término "k-anonimato" (*K-Anonimity*, como se menciona en [2]), en el cual se propone despersonalizar la información de localización de un usuario, por medio de la creación de una "región de encubrimiento", la cual corresponde a un área de mayor resolución donde se encuentra el usuario que solicita el servicio SBL y a su vez k-1 otros usuarios, de esta forma el usuario que realizó la solicitud se mantiene encubierto en una región con k posibles usuarios que realizaron la solicitud, otorgando un grado de protección a su información de localización de posibles adversarios que deseen usarlos con fines maliciosos o simplemente no consentidos por el usuario al que pertenecen.

4.2 Trabajos relacionados

A continuación, se brinda una breve descripción referente a trabajos relacionados al tema de este proyecto, realizados por investigadores de diversas Universidades y centros de estudio, los cuales han sido considerados dentro de este trabajo como una base para el desarrollo de este mismo.

- Marko Gruteser, Dirk Grunwald "Anonymous usage of Location-Based services through and temporal cloaking":

Este trabajo propone e introduce el concepto de "K-anonimidad" para usuarios de SBL en redes inalambricas. Este concepto hace referencia a la creación de un área o región de encubrimiento que contenga al menos k-1 otros usuarios que se encuentren en esa zona (K usuarios en total en conjunto con el usuario que solicitó el servicio) de tal forma que así quede protegida la identidad del usuario que realizó la consulta por k-1 usuarios presentes en dicha región. [2]

- Toby Xu, Ying Cai "Feeling-based Location Privacy Protection for Location-based Services":

Este trabajo se basa en la utilización del concepto conocido y estudiado anteriormente de "K-anonimidad", y utiliza este mismo para la creación de una región de encubrimiento de tal forma de proteger la privacidad de localización de los usuarios. A diferencia de métodos existentes para la creación de esta, se introduce el nuevo concepto de "sensibilidad" al momento de elegir qué área un usuario siente como segura, por lo que se propone la utilización de una región pública que el usuario seleccione para así determinar el nivel "K" de privacidad que este requiere, y así crear su región de encubrimiento correspondiente. [3]

- Toby Xu, Ying Cai "Location Cloaking for Safety Protection of Ad Hoc Networks":

Este trabajo introduce un nuevo concepto denominado "Seguridad de Localización" (Location Safety) que hace referencia a la necesidad de proteger la seguridad de localización de un determinado nodo en una red Ad Hoc, debido a que puede ser atractivo para un adversario encontrar dicho nodo utilizando su información de localización y posteriormente, destruirlo. Se introduce también el concepto de "Nivel de Seguridad" (Safety Level) y se define como la densidad total, de los nodos que se encuentran en un área y el radio de esta. Mientras mayor sea el Nivel de Seguridad de esta zona, esta será menos atractiva para un adversario de investigar debido a los elevados costos de búsqueda que enfrentaría. Esta técnica utiliza también el concepto de "K-anonimato" para crear la región de seguridad de cada nodo. [5]

- Ben Niu, Qinghua Li, Xiaoyan Zhu, Guohon Cao, Hui Li "Achieving k-anonimity in Privacy-Aware Location-Based Services":

Este trabajo utiliza el concepto explorado y desarrollado de K-anonimidad de tal forma de crear una región de encubrimiento que proteja la identidad de un usuario que solicita un SBL. Sin embargo, este propone la utilización de una arquitectura de red diferente a la utilizada por la mayoría de los otros trabajos que utilizan un anonymizer (servidor de confianza) para generar la región de encubrimiento y posteriormente enviarla al SBL como la ubicación de la persona en cuestión. Aquí, se realiza la creación de esta región de una forma distribuida, por medio de puntos de acceso ubicados en diferentes lugares geográficos. Así también, se propone la utilización de las probabilidades de query de una determinada región para seleccionar K diferentes "sujetos" (Dummys) y construir la región de encubrimiento con estos en vez de usuarios reales que se encuentren en el área. [18]

- Chow C-Y, Mokbel MF, Bao J, Liu X. "Query-aware location anonymization for road networks":

Este trabajo desarrolla algoritmos novedosos y eficientes con el fin de proteger la privacidad de localización para usuarios que se encuentran en el ambiente de redes de caminos (autopistas, calles inter urbanas, etc) y requiere los servicios de un SBL. Se utiliza también el concepto de k-anonimato y se extiende a lo que se denomina "Encubrimiento de Trayectoria" (Trajectory Cloaking), que hace referencia al encubrimiento de un usuario que se encuentra en movimiento y realiza varias actualizaciones de su posicionamiento en un periodo de tiempo. [16]

- Toby Xu, Ying Cai. "Exploring Historical Location Data for Anonymity in Location-based Services":

Este trabajo propone un nuevo acercamiento a lo que se conoce como "Protección de k-Anonimato" (k-Anonymity Protection) para SBL. En este, se genera la creación de una región de encubrimiento para cada usuario utilizando k-diferentes "huellas" (footprints), esto es, información histórica de usuarios que han pasado por el área desde donde el usuario solicita un servicio. Debido a esto, un adversario no podrá saber en qué momento en el tiempo estuvieron ahi los usuarios, así como también la identidad y localización de quien realizó la consulta al SBL.

Además, se generan algoritmos que son los primeros en garantizar "Protección de Trayectoria" (Trajectory Cloaking) en caso de que un usuario genere solicitudes en el tiempo mientras se va moviendo continuamente. [19]

4.3 Análisis y descripción de técnicas

A continuación, se realiza una descripción detallada de cada una de las técnicas de interés a analizar en este trabajo, a modo de ahondar en cada técnica para así tener una clara idea a lo que estas apuntan y por qué fueron estas incluidas en este proyecto.

Anonymous Usage of Location Based-Services Through Spatial and Temporal Cloaking:

El autor Gruteser [2] propone un algoritmo que otorgará "k-anonimidad" a los usuarios (subjetcs) que entregan información de localización referente a su ubicación en un determinado tiempo, en donde esta información pasa primero por un mix-router que reordena cada mensaje que han enviado todos los usuarios a este y luego permite su salida hacia el servidor SBL, de tal forma que no pueda ser reindentificado fácilmente por diferentes adversarios. La k-anonimidad se logra mediante la creación de un espacio de privacidad relacionado al lugar donde se encuentra el sujeto en cuestión, pero a su vez podemos encontrar k-1 otros sujetos en un periodo de tiempo determinado. Se sugiere que mientras mayor sea la cantidad de sujetos k que están presentes, mayor es el nivel de anonimidad que se logra crear para el individuo. El sistema utiliza tres pares de tuplas para generar la zona de anonimidad k ([x1,x2], [y1,y2], [t1,t2]), en donde [x1,x2] e [y1,y2] corresponden a un área de dos dimensiones donde el sujeto se encuentra o encontraba, y [t1,t2] el periodo de tiempo cuando estaba en dicha zona.

Un problema con este algoritmo de k-anonimidad [2] surge y es de gran importancia, en el caso en que varias solicitudes converjan en una misma área a la vez (tiempo y espacio). Esto es, suponiendo los siguientes datos de prueba arbitrarios, habiendo cuatro solicitudes con los siguientes datos:

1- ([0,1], [0,1], [t1,t2])
2- ([1,2], [0,1], [t1,t2])
3- ([0,1], [1,2], [t1,t2])
4- ([0,2], [0,2], [t1,t2])

Estas tuplas convergen en el mismo espacio y tiempo. Las primeras tres tuplas hacen referencia a cuadrantes adyacentes, y la última por lo tanto refiere a un cuadrante más largo que cubre a los otros 3.

Un adversario podría entonces concluir de esta manera que la solicitud número 4 se originó del cuadrante ([1,2],[1,2]), debido a que de lo contrario el algoritmo hubiera elegido un área más pequeña.

Esta situación viola el candado de anonimidad propuesto, y demuestra que un adversario puede obtener información de tuplas que convergen en tiempo y espacio. Esta situación corresponde al caso en que vehículos entreguen su información de localización, específicamente, estas tuplas pertenecen a 4 vehículos diferentes. Los intervalos de tiempo en este caso son demasiado pequeños para que un vehículo se mueva una distancia considerable. Esto deja en evidencia por tanto que el algoritmo puede otorgar protección efectiva en el caso de una solicitud individual, pero no en el caso de múltiples solicitudes en un corto intervalo de tiempo.

Aún más, es posible que un adversario inteligente pueda montar un ataque de identificación si logra asociar múltiples solicitudes al mismo sujeto en cuestión y puede así obtener la información de localización del sujeto de otras fuentes. Esto se hace posible fácilmente si dichas solicitudes vienen de SBL raramente accesados o de baja popularidad. Cabe destacar también, que es posible que un adversario obtenga información de localización y más de un sujeto especifico mediante información detallada en el mensaje enviado por dicho individuo. Sin embargo, Gruteser [2] aclara que el algoritmo planteado solo optimiza información referente a la localización del sujeto ingresada a través de un SBL determinado, por lo que la información en los mensajes es una preocupación adicional paralela, no investigada ni solucionada en este trabajo en específico, pero de gran importancia para lo que vendrían a ser trabajos relacionados a futuro.

El algoritmo presentado por Gruteser opera en términos sencillos con los datos que se presentan a continuación:

Datos de localización	Tamaño área ocultamiento	Datos área de ocultamiento	Datos de tiempo de solicitud	Constante de anonimidad k-min
(x1,y1)	(Tamaño máximo establecido)	([x1,x2], [y1],[y2], [t1 - delta,t2])	t2, t1	K

Tabla 1: Variables necesarias algoritmo Gruteser

Finalmente, es importante destacar la arquitectura con la que trabaja esta técnica, la cual consiste en la utilización de un servidor de confianza el cual denominan "anonymizer" [2]. Este entonces obtiene los datos de localización específicos entregados por un usuario, y utilizando esta información genera la región de encubrimiento asociada a este usuario y se la envía posteriormente al SBL, el cual provee una respuesta considerando esa región como la posición del usuario que solicitó el servicio. Es decir, el usuario no tiene contacto directo con el SBL al momento de enviar su información, sino que es este servidor de confianza el que se encarga de realizar dicha comunicación.

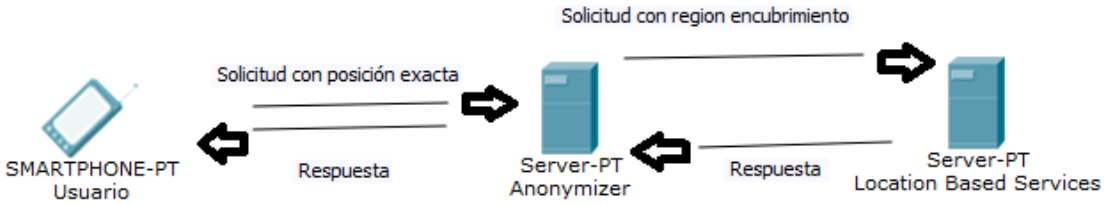


Figura 1: Arquitectura cliente-anonymizer-servidor SBL

Achieving k-anonimity in Privacy-Aware Location-Based Services:

El autor Ben Niu [18] propone la utilización de dos novedosas técnicas de encubrimiento presentadas en su investigación. Mediante DLS (Dummy-Location Selection), se intenta conseguir k-anonimidad para usuarios de dispositivos móviles, pero de una forma diferente al algoritmo presentado por Gruteser, el cual utiliza un anonymizer (servidor en el que se confía planamente) para generar una zona de ocultamiento (CR). El autor hace énfasis en que es este mismo el que ocasiona el mayor problema referente a dicha técnica, ya que si se tiene conocimiento y/o acceso a este anonymizer los datos de los usuarios pueden quedar expuestos debido a que se tiene acceso a información parcial que corresponde principalmente a la probabilidad de query por región en cada área presente en el anonymizer, o peor aún, se tiene acceso a la información de localización exacta de cada usuario.

Es por esto que el algoritmo DLS propone generar la región de encubrimiento sin recurrir a la utilización de un anonymizer. Para poder medir el grado de anonimidad se aplican métricas de entropía. Esta se puede medir como la no-certeza de determinar la localización actual de un individuo dentro de varios otros sujetos (denominados "sujetos"), es decir, el desconocimiento de la ubicación real actual del usuario en cuestión.

Por lo tanto, el mayor nivel de anonimidad se alcanza mientras mayor sea el grado de la entropía.
Es importante mencionar que estos sujetos no necesariamente corresponden a usuarios/personas reales en cada región, sino que pueden efectivamente ser "usuarios ficticios" mientras cumplan las condiciones necesarias para ser considerados en el algoritmo. Una región es considerada como una opción válida cuando esta presenta una probabilidad de query similar a la del usuario que solicita k-anonimidad en cuestión, es así como se alcanza el mayor grado posible de entropía y por consecuencia, de k-anonimidad esperado.

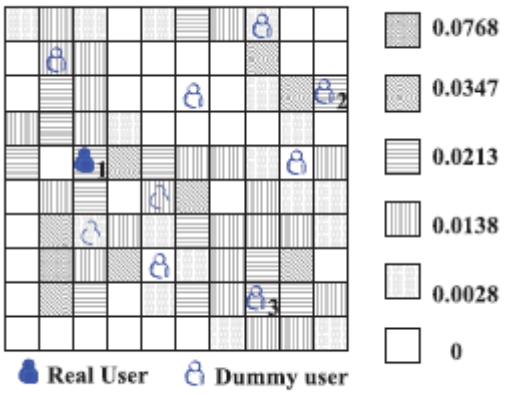


Figura 2: Usuario "sujetos" a considerar según su probabilidad de query. [18]

El algoritmo **enhanced-DLS** se diferencia principalmente del algoritmo principal (DLS) debido a que en este se intenta lograr que los dummy seleccionados se encuentren lo más separados posibles entre si y del usuario real, creando una región de encubrimiento aún más segura.

El usuario realiza una query incluyendo la siguiente información:

- Identificador de consulta [ID].
- Localización exacta [x,y].
- Interés de la consulta [tipo de consulta, por ejemplo: restaurants cercanos, gasolineras, etc].
- Rango de la región consultada [región consultada enviada donde está el usuario].
- Otros antecedentes.

El grado de k-anonimidad óptimo esperado para cada usuario es equivalente a 1/k. Los algoritmos DLS y enchanced-DLS consiguen otorgar este debido a que las probabilidades de query de cada k-1 sujeto y la posición real del usuario son netamente iguales (diferencia mínima) lo que conlleva a que localizar al verdadero usuario dentro de las k posiciones totales es 1/k para cada región.

Feeling-based Location Privacy Protection for Location-based Services:

El autor Tobi Xu [3] propone una técnica de encubrimiento que otorga k-anonimato al usuario de una forma diferente a la conocida hasta el momento. Hasta ahora, los algoritmos se basaban en el uso e ingreso (la mayor parte) de un número específico "k" para así poder definir el nivel de privacidad que un usuario en cuestión requiera. Sin embargo, este trabajo señala que esto mismo puede ser un problema a nivel de los usuarios, debido a que la privacidad más que un número es un "sentimiento", por lo que es difícil para una persona cuantificar su grado de privacidad con el cual se sentirán seguros con un número específico. Digamos, por ejemplo, que una persona ingresará como su nivel de privacidad "k" el número 19, ¿Cómo sabe él/ella que este número le otorgará la privacidad que requiere? Y si este número llegara a satisfacer sus niveles de privacidad ¿Por qué no 20? ¿O 18? Es de esta premisa que el autor propone un nuevo enfoque para poder calcular el nivel de privacidad de una persona en particular, para así poder crear la región de encubrimiento correcta para lo que esta persona siente como una zona segura.

En este trabajo, se presenta y utiliza la idea de que cada usuario señale una *región que ellos consideran como segura, como su región de encubrimiento*. Es decir, cada usuario deberá ingresar en el sistema un área específica (como podría ser, por ejemplo, el Mall Plaza del Trébol, ubicado acá en la región del Bío Bío.) la cual el sistema reconocerá como su región de encubrimiento deseada, es de esta entonces, de la cual se obtendrán datos (utilizando métricas de entropía) referentes a su popularidad para finalmente cuantificar el grado de seguridad requerido por el usuario para así crear su región de encubrimiento final que será utilizada y reportada al SBL correspondiente como su localización real.

Es importante mencionar y destacar, que esta técnica es utilizable en el área de *Trajectory Cloaking* (TC), por medio de la implementación de algoritmos desarrollados específicamente por el autor para este ámbito, en casos en que un usuario necesite hacer constantes actualizaciones de su posición en un periodo de tiempo sin saber el recorrido que este realiza realmente. La popularidad de la región entregada es calculada usando las "muestras" recolectadas (footprints o huellas) de la siguiente manera:

Tomemos que **R** sea una región espacial específica y $\mathbf{S(R) = \{u_1, u_2, u_3, \ldots, u_m\}}$ corresponden a los usuarios o personas que tienen footprints en la región **R**. Se define entonces $\mathbf{n_i}$ ($1 <= i <= m$) como la cantidad de footprints que un usuario $\mathbf{u_i}$ posee un **R**, y $\mathbf{N} = \sum_{i=1}^{m} n_i$ es la sumatoria de todas las

footprints de cada usuario en la región **R**. Entonces, se define la entropía de **R** como **E(R)= -** $\sum_{i=1}^{m} n_i$**/N log** n_i**/N** y la popularidad de **R** como **P(R)= 2**$^{E(R)}$.

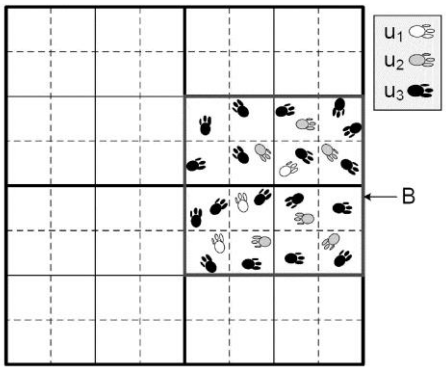


Figura 3: Huellas en estructura propuesta por Toby Xu. [3]

Historical location data for anonimity preservation in Location-Based Services:

El autor Tobi Xu [19] propone un nuevo acercamiento a lo que se refiere a protección por medio de k-anonimity en SBL. Esta nueva técnica despersonaliza la información de localización mediante la utilización de una región de encubrimiento que se envía finalmente al SBL en cuestión que contiene k diferentes "footprints". Esto es, localizaciones históricas o "huellas" de diferentes nodos móviles que en algún momento determinado se encontraban en ese lugar en específico. Esto marca una clara diferencia con técnicas existentes hasta el momento, debido a que estas se basan en la utilización de la localización actual de k host "vecinos" (usuarios que se encuentren a una determinada distancia) del usuario que está solicitando el servicio. Es debido a esto que esta técnica presenta una mejora al reducir el área de encubrimiento, lo que por ende significa que se tiene una reducción del procesamiento de cada query y "overhead" de comunicación para devolver los resultados de la query al host que las solicita. Cabe mencionar también, que las técnicas existentes también requieren constantes actualizaciones por parte de los vecinos del usuario, no importando si estos requieren un SBL o no, lo cual significa una "molestia" adicional para estos usuarios.

Además de todas estas ventajas ya mencionadas, lo más importante que Tobi Xu plantea en esta técnica es la capacidad de esta de brindar ***k-anonimity trajectory protection*** para asegurar anonimidad en caso de que un usuario móvil requiera solicitar SBL a medida que se encuentra en movimiento. Esto se logra mediante la despersonalización de la ***trajectoria*** de un usuario (una serie de localizaciones de un usuario en el tiempo) basándose en las trajectorias históricas de otros usuarios (footprints de trajectorias).

Location Safety:

El autor **Tobi-Xu** y **Ying-Cai** [5] proponen una técnica y a su vez introducen a un nuevo concepto denominado "***Location Safety***" (seguridad de localización). Esto hace referencia a la necesidad de proteger la seguridad de localización de un nodo en una red ad hoc (ad hoc network) debido a que un adversario podría adueñarse de esta información y finalmente destruir físicamente dicho nodo. La técnica propuesta está basada en el concepto de que mientras mayor sea el grado de ***seguridad*** de un área, es menos atractivo para un adversario el buscar por un nodo en esta zona y destruirlo, debido a que los costos de búsqueda serian demasiado elevados. El grado de seguridad mencionado con anterioridad, corresponde al radio del área en cuestión y la cantidad de nodos que se encuentran en esta.

Para garantizar que una región de encubrimiento de un nodo tiene los requerimientos de seguridad necesarios y, a su vez, es tan pequeña como sea posible de modo de no causar problemas a nivel de respuesta de la consulta, cada área o dominio donde se encuentra un nodo es dividida en subdominios los cuales poseen también un nivel de seguridad no menor al requerido. Este proceso llega a su fin cuando las particiones siguientes de un determinado dominio poseerian un nivel de seguridad menor que el estandar definido. Es así entonces, que cada nodo toma el subdominio correspondiente a su posición y que no pueda seguir siendo particionado más y lo utiliza como su ***caja de encubrimiento***.

De esta forma se asegura que la ubicación de cada nodo es segura ante la presencia de un adversario debido a que este no podrá buscar por su ubicación específica debido a elevados costos de búsqueda.

Query-Aware location anonymization for road networks:

El autor Chin-Yin Chow [16] propone una técnica específicamente diseñada para proteger la **privacidad de localización** de un usuario que se encuentra en el ambiente de las redes de caminos (autopistas, calles inter urbanas, etc) y requiere los servicios de un SBL. Esto es, debido a que la utilización de las técnicas existentes hasta ahora (en su gran mayoría basadas en k-anonimity) resultaría en problemas de filtros de privacidad y tiempos de procesamiento de query ineficientes.

El algoritmo propuesto tiene la característica de ser **query aware** (consiente a nivel de query) ya que toma en consideración el costo de ejecución de la query en un servidor de bases de datos y la calidad de esta, esto es, el número de objetos que se retornan al usuario por el servidor durante el proceso de anonimización de localización. Específicamente, se desarrolla una nueva función de costo que balancea entre los costos de ejecución de la query y la calidad de esta misma.

Es así entonces que se introducen dos versiones de este algoritmo; uno llamado ***pure greedy*** y el otro ***randomized greedy*** que tienen por objetivo minimizar la función de costo creada a su vez satisfacer los requerimientos de privacidad especificados por el usuario. De tal forma de acomodar los intervalos con una gran carga de trabajo, se introduce también un paradigma que incrementa la escalabilidad del algoritmo de anonimización de localización presentado anteriormente y además del servidor de bases de datos utilizado para aceptar una gran cantidad de querys recibidas en un corto periodo de tiempo.

4.4 Clasificación técnicas de encubrimiento

En la literatura actual se investigan y desarrollan técnicas capaces de otorgar al usuario un cierto nivel de protección para su información de localización. Efectivamente, como se ha podido apreciar en las técnicas analizadas en la sección anterior, cada autor se preocupa de despersonalizar la ubicación e información relacionada a un usuario incluida en esta misma mediante la implementación de diferentes algoritmos, en su gran mayoría basadas en el principio de K-anonimidad. Sin embargo, es difícil poder comprender realmente qué **tipo** de seguridad es la que nos ofrecen estas técnicas. ¿Está la totalidad de mi información de localización segura utilizando este encubrimiento? ¿Qué tan difícil será para un adversario obtener información sensible de cada usuario? Son algunas de las preguntas que surgen al momento de pensar en utilizar una determinada técnica.

Para poder entender y describir el concepto relacionado al tipo de seguridad que se otorga con cada algoritmo entonces, hemos utilizado definiciones creadas en primera instancia por el autor **Toby Xu** ([3], [5]) para poder entonces diferenciar claramente qué clase de protección nos ofrece cada algoritmo.

Se considera entonces que cada técnica puede brindar **protección de anonimato**, **protección de privacidad de localización**, y **protección de seguridad de localización**. La clasificación de cada una en una de estas tres clasificaciones dependerá principalmente de como funcione el algoritmo de creación de regiones de encubrimiento, y en el principio en el que se basa cada técnica (K-anonimato, etc).

A continuación, se da una definición clara sobre cada una de estas categorías.

4.4.1 Encubrimiento de anonimato

Si bien se puede interpretar el término "anonimato" como mantener cierta información (identidad, etc) oculta de determinados sujetos, entidades u otros, en el caso de la utilización de SBL no basta tan solo con utilizar por ejemplo, un seudónimo para "camuflar" la identidad real de un usuario, esto es debido a que el tan solo hecho de revelar su información de localización (ubicación precisa) significa poner en riesgo su identidad, ya que un adversario puede utilizar esta información para deducir u obtener de otras formas información sensible del usuario, similar al caso de cómo se obtienen datos de usuarios "anónimos" en las redes de internet. Un usuario podría entonces no querer que su identidad sea expuesta directamente como el suscriptor de un SBL. Esto es importante ya que, un usuario en particular queda conectado de forma directa con un servicio específico, lo cual puede significar que dicha información caiga fácilmente en manos de terceros y utilizada para propósitos que una persona no desee (marketing, o inclusive acoso).

Para solucionar entonces estos problemas de seguridad, se propuso el uso de técnicas basadas en el principio de K-anonimato de tal forma de ocultar a un usuario dentro de una región que contiene al menos k-1 otros usuarios **reales** adicionales. De esta forma, un adversario tendrá dificultades en identificar cuál de todos los usuarios que se encuentran en dicha área fue el que realizó la solicitud del servicio.

Las técnicas que pertenecen a esta categoría, son capaces de garantizar de esta forma entonces que la identidad del usuario que solicitó el servicio se mantiene oculta. Pero, no son capaces de garantizar que su información de localización este completamente segura. Esto ya que, aunque para un adversario sea difícil saber cuál de todos los usuarios es que el realiza la consulta, él sabe que todos estos se encuentran ahí en el momento de la solicitud del servicio.

Debido a esto, no solo se compromete la información de localización del usuario que hace la solicitud, sino que también de todos los otros usuarios que fueron utilizados para crear su región de encubrimiento [3].

De esta forma entonces se define el encubrimiento de anonimato como todas aquellas técnicas que utilizan el principio de K-anonimato y requieren de k-1 usuarios **reales** adicionales al usuario que solicita un SBL para generar cada región de encubrimiento.

4.4.2 Encubrimiento de privacidad de localización

Aunque otorgar encubrimiento de anonimato es una manera efectiva de proteger la identidad de cada usuario, no puede garantizar que la localización del usuario que solicitó el servicio e inclusive del resto de los usuarios que se encuentran en la región de encubrimiento se encuentre a salvo. Esto podría significar entonces que esta información sea utilizada para re identificar a los usuarios presentes en el área de encubrimiento, generando filtros de información que no debe revelarse.

Este problema ha sido investigado por un gran número de autores ([3]) y se han presentado diversas soluciones en la literatura. Utilizando la misma base de las técnicas propuestas con anterioridad que brindan K-anonimato [2] a los usuarios, se han desarrollado y derivado técnicas que garantizan a un usuario obtener su nivel deseado de seguridad mediante la utilización de k-1 diferentes "**footprints**" (huellas) de usuarios que alguna vez estuvieron por el área en la que un usuario se encuentra solicitando un SBL en este momento. Se genera entonces la región de encubrimiento pertinente para el usuario en cuestión, la cual, al igual que en lo que corresponde al encubrimiento de anonimato, es enviada al servidor de SBL como lo que corresponde a la ubicación de la persona que solicitó el servicio. Cada huella corresponde entonces a información histórica de usuarios que estuvieron en algún momento en el área.

Debido a la utilización de estas "huellas" y no de la ubicación de otros sujetos en el mismo momento en que un determinado usuario solicita un SBL, es posible garantizar a cada persona la privacidad de su información de localización. Esto es ya que, no importa si un adversario es capaz de identificar a cada usuario que se encuentra en el área entregada como región de encubrimiento, aun así, él no será capaz de deducir *quien estuvo ahí y en qué periodo de tiempo*, por medio de la correlación con *espacios restringidos* (**Restricted Spaces Re-identification**). Lo que no solo brinda privacidad de localización para el usuario que está ejecutando el servicio, sino que también para los usuarios vecinos que otorgaban su información de localización para generar la región de encubrimiento en las técnicas correspondientes a encubrimiento de anonimato, debido a la naturaleza de las "huellas".

Se define el concepto de encubrimiento de privacidad de localización como todas aquellas técnicas que protegen la información de localización de un usuario mediante la creación de una región de encubrimiento que contiene al usuario que solicitó el servicio y k-1 otros usuarios que se encontraban en algún momento en el área en que se solicitó el servicio.

4.4.3 Encubrimiento de seguridad de localización

Un problema de gran impacto e importancia estudiado en lo que corresponde a la implementación y utilización de redes ad-hoc, por el autor **Toby Xu** [5], en donde existen nodos que envían mensajes de comunicación entre ellos en un área específica, es mencionado por este mismo bajo el nombre de "**Location Safety**" (seguridad de localización).

En este trabajo se describe una red conectada con una cantidad conocida de nodos, entonces es posible y de interés para un adversario en particular el poder **destruir** físicamente uno o más de estos, lo cual es posible por medio de la adquisición y conocimiento de su información de localización especifica. Esta información puede ser interceptada por medio de los mensajes que estos mismos nodos envían u de otras formas, y utilizada entonces por terceros para hallar y dañar físicamente dichos nodos.

Se introduce entonces el termino de *nivel de seguridad* [5] y se define como el resultado de la relación entre el área y la cantidad de nodos que se encuentran en esta. Mientras mayor sea este nivel de seguridad, menos atractivo será para un adversario buscar por un nodo en particular en el área y destruirlo. Utilizando un principio similar a K-anonimato, sin embargo, con una diferencia importante de notar. Pese a que se busca ocultar un nodo dentro de un área de encubrimiento que contiene otros nodos dentro de ella también, al igual como se busca hacer en K-anonimato, esta área debe tener el mayor tamaño posible y al mismo tiempo, tener la menor cantidad de nodos en esta. Es decir, se tiene un principio **inverso** respecto de K-anonimato tradicional.

Se define entonces el concepto de encubrimiento de seguridad de localización para todas aquellas técnicas que implementen una región de encubrimiento utilizando la menor cantidad posible de nodos (reales, por lo menos en las técnicas presentes hasta el momento) en un área de gran tamaño.

4.5 Ataques que afectan a las técnicas

Existen diversas formas por las cuales un adversario puede obtener la información de localización de un usuario e identificarlo a partir de esta, esto es especialmente sencillo cuando la información dentro del SBL corresponde a la localización exacta de un usuario y esta no se encuentra encriptada por medio del uso de técnicas criptográficas [17], o no corresponde a una región de encubrimiento, como proponen diversas técnicas con el fin de garantizar un mayor grado de seguridad a su información de localización. En estos casos, basta con que el adversario tome control del servidor SBL o tenga acceso a este de una u otra forma (por ejemplo, hackeandolo y burlando la seguridad del SBL o, inclusive, siendo el mismo SBL quien desea utilizar esta información con otras finalidades). Sin embargo, inclusive al utilizar estos métodos mencionados para incrementar la seguridad de la información de localización de un usuario, está aún puede encontrarse en peligro. Esto ya que, por medio de distintas clases de **ataques** y análisis a la información que tiene acceso un adversario (una región de encubrimiento con K usuarios en el caso de las técnicas estudiadas en este proyecto), este puede llegar a identificar al usuario especifico que realizó la solicitud del servicio y/o aprender más información sobre este mismo.
En este trabajo, analizamos algunos de estos ataques que constituyen un problema para la seguridad de la información de localización de las personas.

Los problemas ([2], [3], [16], [18]) que describiremos y analizaremos son los siguientes:

- *Identificación de Espacios Restringidos* (**Restricted Space Identification)**.
- *Ataque de Observación* (**Observation Attack**).
- *Ataque de Centro del Área de Encubrimiento* (**Center of Cloaked Area**).
- *Ataque de Repetición* (**Replay Attack**).
- *Ataque de Colusión* (**Colluding Attack**).
- *Ataque de Inferencia* (**Inference Attack**).

Identificación de Espacios Restringidos: Aunque la información de localización de un usuario enviada a un SBL sea anónima (por medio del uso de un seudónimo u otros métodos para ocultar la identidad), esta persona aún corre el riesgo de ser re identificado por un adversario. Esto es posible por medio de la correlación con espacios restringidos, como lo son por ejemplo una oficina de trabajo o un hogar. Si una solicitud recibida por un SBL corresponde a la localización de una oficina, por ejemplo, entonces lo más probable es que el usuario que envió esa consulta sea un

trabajador de dicha oficina, o si esta corresponde a la localización de una propiedad privada (un hogar, departamento, terreno, etc) entonces el usuario que solicitó ese servicio corresponde muy probablemente al dueño de esa propiedad o un miembro de su familia. Este ataque se basa en el uso de este principio para así, posteriormente, re identificar al sujeto que solicitó el servicio en cuestión.

Ataque de Observación: Este ataque se basa en el hecho de que, un adversario interesado en descubrir la identidad de un sujeto que solicitó un SBL, especialmente cuando este se encuentra oculto dentro de una región de encubrimiento que posee K usuarios distintos pero no se tiene completa certeza en qué periodo de tiempo estos se encontraban efectivamente en esa área, puede acotar el número de posibles personas que solicitaron el servicio mediante la observación directa (en persona) del área en cuestión recibida por el SBL. Mediante este método, un adversario es capaz de saber efectivamente la cantidad de personas y quien/quienes se encuentran realmente en la zona al momento en que se realizó el servicio.

Ataque de Centro del Área de Encubrimiento: Este ataque se basa en que, en varias técnicas de encubrimiento que emplean la creación y utilización de un área de encubrimiento para proteger la privacidad y/o identidad de un usuario que solicita un servicio a un SBL, la posición exacta de este en la región de encubrimiento se encuentra extremadamente próxima o precisamente en el centro de esta. Por lo que, mediante un estudio del tamaño de la región de encubrimiento entregada al SBL, entre otros análisis, es posible para un adversario identificar al usuario exacto que solicitó el servicio en un momento determinado.

Ataque de Repetición: Este ataque se basa en que, teniendo en cuenta que un adversario sepa la posición exacta de los usuarios en una región de encubrimiento (sin embargo, desconoce su identidad), este quiera saber quién de estos usuarios fue el que realizó la consulta. Por lo que, este realiza consultas utilizando estas posiciones de tal forma de generar nuevas regiones de encubrimiento y así descubrir asociaciones entre estas, la región que se generó al comienzo y la nueva región que se ha generado. Cabe destacar que mientras mayor sea el grado de información adicional que el adversario posea (por ejemplo, conocimiento sobre el algoritmo de anonimización parcial o completo, estadísticas utilizadas por las funciones de costo asociadas a la query, etc), mayor probabilidad de éxito tendrá.

Ataque de Colusión: Este ataque se basa en el concepto de que un adversario (pasivo en la mayoría de los casos) puede aliarse o coludir con usuarios de tal forma de poder obtener información adicional sobre otros usuarios en las cercanías o inclusive coludir con el mismo servidor SBL para predecir información sensible de usuarios existentes. Cabe destacar que mientras mayor sea el grupo con el que se colude, mayor será la probabilidad del usuario de obtener la información de interés de otros usuarios.

Ataque de Inferencia: Este ataque se basa en la capacidad de un adversario (activo en la mayoría de los casos) de inferir información de interés para él sobre un usuario. En este ataque, se considera que el SBL es el adversario, por lo que este posee acceso completo a la información de cada query, desde las probabilidades de query de cada área geográfica, queries históricas y queries actuales las cuales contienen el identificador del usuario, la mezcla de localizaciones de usuarios reales y usuarios "dummy" (en caso de existir), los intereses, rangos de query, entre otra información. Es gracias a toda esta información que el adversario puede realizar este tipo de ataques y obtener así información sensible de un usuario específico.

Existen otros ataques que pueden ser realizados por un adversario para obtener información sensible sobre los usuarios de SBL ([3], [5], [16]), sin embargo, consideramos que estos son los de mayor relevancia con respecto a las técnicas que hemos analizado en este trabajo.

4.6 Comparación de técnicas

De modo de obtener una idea más clara y concreta sobre cómo actúa cada técnica estudiada en este trabajo, se realizará una comparación lineal acerca de cada una según la región de encubrimiento que estas generan y, a su vez, como es generada, utilizando las clasificaciones que se hicieron con anterioridad. Utilizamos también la definición de dos tipos de arquitecturas de red, como detallamos a continuación:

Las dos arquitecturas de red consideras en este trabajo son del tipo **centralizado** y **distribuido**.

Arquitectura centralizada: la mayoría de las técnicas de encubrimiento de localización existentes se basan en la utilización de servidores de anonimato de confianza (Anonymizer) el cual protege la información de localización e identidad específica de cada usuario de un servidor no confiable (SBL). La idea principal de este enfoque es hacer uso de un anonymizer de modo que sea un intermediario entre el usuario y el SBL de tal forma de prevenir que este último conozca la localización exacta e identidad de los usuarios. Es entonces el anonymizer el encargado de computar la región de encubrimiento que corresponde a cada usuario y enviar esta al SBL como la información de localización de dicha persona.

Arquitectura distribuida: debido a que la propia utilización de un anonymizer para crear la región de encubrimiento de un usuario trae consigo potenciales peligros y problemas a este mismo, varios estudios presentan opciones para construir una región de encubrimiento sin la utilización de un anonymizer.

Diversas técnicas proponen diferentes métodos para cumplir esta tarea, sin embargo, la gran mayoría se basan en utilización del propio dispositivo móvil del usuario. Mediante la comunicación de un usuario con otro cercano a su posición actual se realiza un intercambio de información, utilizando esta información y cuando se obtienen suficientes usuarios para satisfacer las condiciones de seguridad requeridas por el usuario que solicita el servicio se crea la región de encubrimiento correspondiente, esta técnica es propuesta y descrita detalladamente por el autor Ben-Niu.

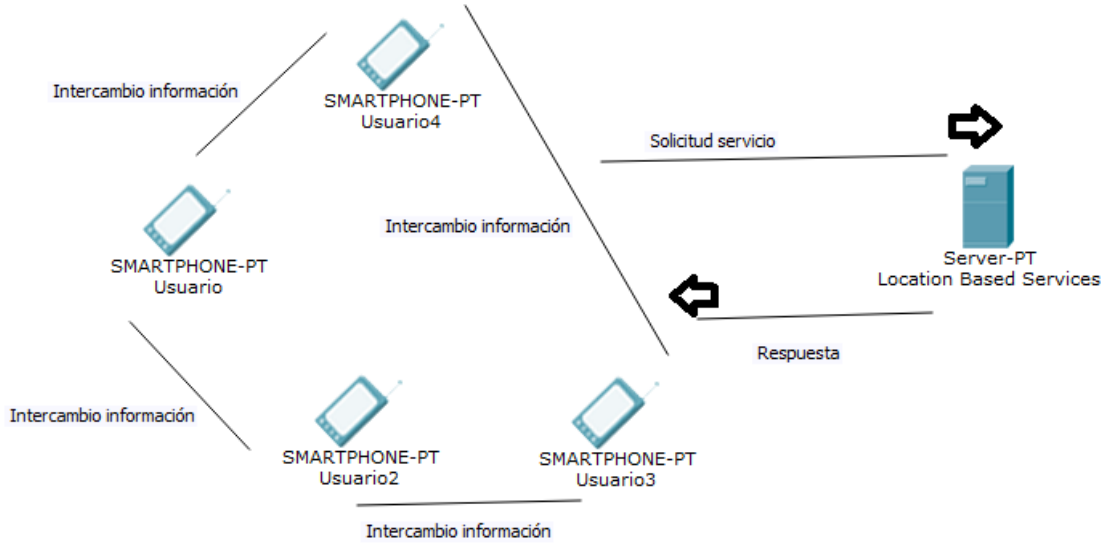


Figura 4: Arquitectura distribuida

Teniendo en cuenta estas dos arquitecturas, utilizamos adicionalmente las clasificaciones de técnicas definidas previamente en este trabajo, pudiendo estas utilizar cualquiera de los dos tipos de arquitectura de red.

Encubrimiento de anonimato: la idea principal de las técnicas de encubrimiento que utilizan este enfoque es ocultar la localización exacta de un usuario agregándola a una *región de encubrimiento* más grande y hacer de este usuario indistinguible respecto de las localizaciones de otros usuarios (reales) presentes en esta. Para este tipo de técnica son aplicables los dos tipos de arquitectura de red.

Encubrimiento de privacidad de localización: la idea principal de estas técnicas de encubrimiento es proteger y otorgar privacidad a la localización de un usuario. Pese a que las técnicas de encubrimiento de anonimato ofrecen protección para la identidad en cuestión de una persona, un adversario aún puede saber dónde se encuentra dicho individuo, aunque no pueda diferenciar inmediatamente cual de todas las localizaciones presentes en la región de encubrimiento corresponde al usuario en cuestión, él sabe que está ahí junto con k-1 usuarios, y podría eventualmente descifrar su posición exacta mediante diversos métodos.

Encubrimiento de seguridad de localización: la idea principal de estas técnicas es eliminar completamente el riesgo de identificación de la localización de un usuario específico de modo de evitar que esto resulte en daños reales a este, como podría ser la destrucción completa de dicha ubicación. La técnica perteneciente a esta categoría solo posee una arquitectura del tipo distribuido, por lo menos en lo que corresponde a las que fueron analizadas en este trabajo.

De acuerdo a los estudios y técnicas analizadas en este trabajo, se obtienen los siguientes resultados: Teniendo en cuenta las definiciones descritas previamente, es posible realizar una categorización de cada técnica incluida en este trabajo, con la finalidad de poder ver claramente las similitudes y diferencias que estas poseen para posteriores estudios e implementaciones de cada una, de tal forma de conocer plenamente los alcances y detalles pertinentes a cada una.

4.6.1 Tabla comparativa

Nombre técnica	Clasificación	Resistencia frente ataques	Query Aware	Soporte para trajectory cloaking	Datos requeridos	Forma región encubrimiento
Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking	Encubrimiento de Anonimato	Ataque de Observación	No	No	Número K variable.	Cuadrado o rectangular.
Achieving K-Anonimity in Privacy Aware Location Based Services	Encubrimiento de Anonimato	Ataques de Colusión, Ataques de Inferencia	No	Si. Aunque no especifica mucho sobre esto	Número K variable.	Puntos (posiciones de cada usuario).
Feeling-based Location Privacy Protection for Location-based Services	Encubrimiento de Privacidad de Localización	Identificación de Espacios Restringidos	No	Si. Incluye un algoritmo diseñado para esto.	Región pública que será utilizada para determinar k.	Cuadrado y Segmentos dependiendo del algoritmo.
Historical Location Data for Anonimity Preservation in Location Based Services	Encubrimiento de Privacidad de Localización	Identificación de Espacios Restringidos	No	Si. Incluye algoritmos diseñados con este fin.	Número K diferente para cada usuario.	Cuadrado y Segmentos dependiendo del algoritmo.
Location Cloaking for Safety Protection of Ad Hoc Networks	Encubrimiento de seguridad de Localización	Ataque de Correlación	No	No	Número K diferente para cada nodo.	Cuadrado.
Query Aware Location Anonymization for Road Networks	Encubrimiento de anonimato	Ataques de Repetición, Ataques de Centro del Área de Encubrimiento	Si	Si	Número K diferente para cada usuario en movimiento	Segmentos.

Tabla 2: Comparación técnicas estudiadas

4.6.2 Explicación tabla comparativa

En este trabajo se ha decidido entregar los resultados finales del análisis de cada técnica mediante una tabla comparativa que muestre los alcances e información importante a destacar de cada trabajo estudiado. Para esto utilizamos diferentes criterios que consideramos son de vital importancia destacar y tener presente. Específicamente, hemos decidido que para cada técnica es importante clarificar la siguiente información:

- Clasificación.
- Resistencia frente a ataques.
- Query Aware.
- Soporte para trajectory cloaking.
- Datos requeridos.
- Forma región de encubrimiento.
- Arquitectura de red.

Consideramos entonces estos atributos para comparar cada técnica y tener una visión clara de las ventajas y desventajas que poseen estas, con el fin de aclarar cada término utilizado en la tabla comparativa explicamos a continuación el significado de cada uno de ellos.

Clasificación: Este aspecto hace referencia a la clasificación de cada técnica con respecto al tipo de protección que estas realmente brindan. Es importante notar que en la literatura actual se tiene un cierto grado de desconocimiento y se confunde con bastante frecuencia cuando una técnica brinda **protección de anonimato**, **protección de privacidad de localización** y **protección de seguridad física**. Como hemos mencionado anteriormente, una técnica que brinda **protección de anonimato** corresponde a aquella que es capaz de garantizar que la *identidad* del usuario que solicitó el servicio permanezca oculta de un adversario. Esto se logra, en el contexto de los algoritmos que utilizan el concepto de k-anonimato y crean una región de encubrimiento que contenga al usuario y otros k-1 usuarios adicionales, mediante la utilización de k-1 usuarios *reales*, es decir, personas que se encuentren dentro del área donde se realiza la consulta en el momento en que esta fue ejecutada. Por lo que, pese a que un adversario pueda identificar cada punto de localización en la región de encubrimiento, este no sabrá la identidad de cada uno de ellos.

Por otro lado, las técnicas que proporcionan **protección de privacidad de localización**, son aquellas que son capaces de garantizar al usuario que su localización en cuestión permanecerá

oculta de un adversario. Esto lo logran, dentro del contexto de los algoritmos que utilizan el concepto de k-anonimato, por medio de la utilización de la información **histórica** de los usuarios que han visitado una zona determinada, también llamadas "footprints" (denominadas así por el autor Toby Xu) o "probabilidades de query" (término utilizado por el autor Ben Niu) y corresponden a la cantidad de veces que un usuario a visitado una región, es con estos datos que finalmente se crea la región de encubrimiento de cada usuario, la cual contiene entonces a la persona que solicitó el servicio como a k-1 otros "usuarios" que son realmente footprints de estos. Debido a esto, aunque un adversario averigüe la identidad de cada usuario en la zona, el no sabrá quien estuvo ahí en que momento en el tiempo, garantizando que sus localizaciones se mantengan a salvo.

Finalmente, las técnicas que proporcionan protección de seguridad física son aquellas que son capaces de garantizar al usuario que un adversario no será capaz de destruirlo físicamente utilizando su información de localización. Estas técnicas se basan en el concepto de que para un adversario es de interés poder descubrir la localización de cada usuario y así poder destruir físicamente uno o más de estos. Debido a esto, se introduce un nuevo concepto denominado **nivel de seguridad** que se define como el radio del área donde se encuentran los usuarios y la densidad de estos en dicha zona. Mientras mayor sea el nivel de seguridad de un área, menos interesante es para un adversario buscar y destruir un usuario en específico, debido a que los costos asociados a esto serian demasiado grandes. Esta es la principal diferencia con respecto de las otras clasificaciones, debido a que mientras mayor sea el número de usuarios en el área, menor será su nivel de seguridad, por lo que estás técnicas concluyen que, un nivel de usuarios lo menos elevado posible conlleva a un nivel de seguridad más alto y, por ende, más seguro este se encuentra físicamente ante un adversario.

Resistencia frente a ataques: Este aspecto hace referencia a la capacidad de una técnica de ocultamiento de proporcionar protección de la información de localización de un usuario inclusive al enfrentarse a un adversario que realiza un determinado ataque a la información que posee el SBL (en este trabajo, mayoritariamente una región de encubrimiento) con el fin de obtener información más precisa (posición exacta, identidad, etc) del usuario que solicitó el servicio. Como hemos mencionado y descrito en el capítulo 3, en el punto 3.4 específicamente, la información de localización de un usuario puede ser utilizada para obtener la identidad en cuestión de la persona, y aunque esta se encuentre protegida por medio de una región de encubrimiento existen métodos que un adversario puede usar para refinar esta área, disminuir los posibles sujetos que pudieron haber ejecutado el servicio y finalmente re identificar la identidad y ubicación exacta del usuario que solicitó este servicio. Es por esto entonces que, a medida que se van desarrollando y mejorando

técnicas de ocultamiento, estas se crean garantizando protección contra ciertos ataques, otorgando así un grado de protección aún mayor de la información de localización e identidad del usuario. Pese a que ninguna técnica hasta el momento es completamente resistente frente a todos los posibles ataques de los que se tiene conocimiento hasta la fecha, se puede apreciar un incremento en la preocupación y énfasis en este tema por parte de los diferentes autores, por lo que consideramos este punto en este trabajo como un tema de gran importancia y relevancia en el ámbito de las regiones de encubrimiento.

Query Aware: Este aspecto hace referencia a un atributo o característica especifica de algunas técnicas de ocultamiento desarrolladas por diferentes autores, las cuales se preocupan tanto del tiempo de ejecución de una query, el tiempo de respuesta de esta y los costos finales para el sistema asociados a la ejecución de dicha query. Este concepto se utiliza y menciona como "query aware" en la literatura y define la capacidad de un algoritmo de ocultamiento de brindar protección de anonimato, privacidad de localización o seguridad física al mismo tiempo de tener en consideración los tiempos de respuesta de la query que generan, tiempos de ejecución y el costo final de esta. De tal forma, que luego de optimizaciones al algoritmo (en caso de ser necesarias), garanticen que este es factible de implementar en un Servicio Basado en la Localización real brindando los menores costos posibles a nivel de query y generando tiempos de respuesta óptimos para el usuario, elevando lo menos posible estos valores en comparación con los que posee una query estándar en estos servicios.

Consideramos que este es un criterio importante al momento de buscar algoritmos de ocultamiento aplicables en servicios basados en la localización reales, debido a que, por ejemplo, un servicio que demore demasiado tiempo en entregar la respuesta que un usuario necesita en el momento es menos atractivo para estos mismos y además, muy posiblemente signifique que esta query tenga un costo demasiado elevado para el sistema y genere por ende respuestas de este estilo con gran frecuencia, lo cual genera un servicio entregado ineficiente.

Soporte para trajectory cloaking: Este aspecto hace referencia a un término utilizado mayoritariamente por el autor **Tobi Xu** ([2], [19]) para describir y definir lo que corresponde al ocultamiento de usuarios en movimiento continuo y que realizan actualizaciones constantes de su posición conforme se van moviendo. Formalmente, se conoce que en SBL continuos un usuario o cliente móvil realiza una secuencia de actualizaciones de su posición en el tiempo, y esto genera lo que se conoce como una ***trajectoria***. De aquí aparece entonces el término "**Trajectory Cloaking**", con el cual se apunta a realizar el encubrimiento correspondiente para este tipo de usuarios, lo cual

significa entonces a su vez que se debe calcular su región de encubrimiento correspondiente. Es importante destacar que, las técnicas y algoritmos comúnmente utilizadas para realizar el encubrimiento de usuarios "estáticos" o llamados más comúnmente "esporádicos" no son eficientes al ser aplicadas en usuarios que generan actualizaciones constantes de su posición en el tiempo, y generan finalmente una región de encubrimiento menos eficiente y con un menor grado de seguridad para estos usuarios.

Es debido a esto que, muchos de los autores que han publicado técnicas más actuales, consideran una parte de su investigación exclusivamente al desarrollo de técnicas especializadas para este tipo de casos, generando algoritmos capaces de entregar K-anonimidad para usuarios de SBL continuos.

Debido a la complejidad e importancia de este tema, lo hemos tratado en otro capítulo de este trabajo con el fin de dar información más a fondo sobre este.

Datos requeridos: Este aspecto hace referencia a los atributos que son requeridos por las técnicas de ocultamiento de usuarios, más específicamente, por los algoritmos desarrollados por los diferentes autores para generar la región de encubrimiento u ocultamiento requerida por el usuario que solicitó el servicio de SBL. Comúnmente, en las técnicas basadas en k-anonimato se tiene una variable estándar necesaria para poder generar dicha área, y esta es el número k con el que las personas desean que su región de encubrimiento sea calculada. Sin embargo, como hemos podido apreciar y explicado anteriormente, esto no es siempre así. Específicamente, en la técnica propuesta por el autor Tobi Xu [2] la cual llamaremos "Feeling Based Cloaking" utiliza una región pública que el usuario considera como un área segura y desea tener una región de encubrimiento como esa, es de esta entonces que el algoritmo obtiene su "popularidad" y lo utiliza como el número "k" para generar finalmente la región de encubrimiento. Estas variables corresponden a datos ingresados por el usuario directamente y seleccionados por sus propios criterios.

Así también, existen otros datos requeridos y utilizados por las técnicas y algoritmos que quizás el propio usuario no este consiente de que así es. Un claro ejemplo de esto es la ubicación en cuestión del usuario, la cual es enviada al servidor de confianza u otros sitios por medio de la función GPS de su teléfono móvil, y en caso de que soporten Trajectory Cloaking (y dependiendo del tipo de Trajectory Cloaking el cual permitan, esto es, con conocimiento previo del recorrido completo del usuario o sin conocimiento de este) podrían requerir el recorrido en forma de trayectoria que el usuario realizará, además de su posición actual. Es probable que algunas técnicas puedan necesitar

otros datos además de los mencionados previamente, pero estos son los más comunes e imprescindibles para calcular una región de encubrimiento por medio de k-anonimato.

Forma región de encubrimiento: Este aspecto hace referencia a la forma final que posee la región de encubrimiento generada por cada una de las técnicas analizadas en este trabajo. Al momento de generar una región de encubrimiento, esta es construida siguiendo generalmente un patrón geométrico que determina la forma del área generada correspondiente a la región de ocultamiento de un usuario, esta puede ser, por ejemplo, de forma cuadrada o rectangular (uno de los métodos de construir regiones de encubrimiento más comunes) de tal forma que dentro de dicha figura se encuentren los k-1 usuarios y el usuario que solicitó el servicio. Así mismo, existen otras formas geométricas que son utilizadas al momento de crear las regiones de encubrimiento (círculos, etc) e inclusive algunas formas que no se basan en figuras geométricas, como la generación de segmentos (específicamente, un **conjunto de segmentos**), utilizados especialmente en los algoritmos que soportan Trajectory Cloaking ([2], [3]) para representar su región de encubrimiento. Otra forma (menos común) en la que pueden ser generadas las áreas de ocultamiento sin recurrir a una forma geométrica es la generación de puntos. Esto es, generar puntos que corresponden a las **posiciones geográficas** de cada "usuario" que fue seleccionado para estar dentro de la región de encubrimiento en conjunto con el usuario que realizó la solicitud del servicio.

Existen otras formas en las que son representadas las regiones de encubrimiento, y es un punto de gran importancia de notar y tener en consideración, debido a que la forma final de la región de encubrimiento está ligada directamente con los tipos de ataques que son efectivos en estas, y, por ende, finalmente, se relacionan directamente con la **eficiencia** y **seguridad final** que estas proveen.

Arquitectura de red: este aspecto hace referencia al tipo de arquitectura de red para la que estas técnicas están diseñadas y en la que se basan para su funcionamiento. Por lo general, estas técnicas están desarrolladas para trabajar en redes del tipo P2P (Peer to Peer) inalámbricas, redes tipo ad hoc, entre otras. La arquitectura de red en general entonces la describimos como **centralizada** y **distribuida** para describir técnicas enfocadas a un sistema con una entidad de confianza que mantiene los datos de todos los usuarios (Anonymizer, etc) de otras que no utilizan este tipo de dispositivos.

4.6.3 Conclusiones tabla comparativa

Por medio del desarrollo de la tabla comparativa hemos podido notar y destacar características importantes de las técnicas de ocultamiento estudiadas en este proyecto.

Una de las características más importantes que hemos descrito y utilizado para realizar la comparación es la clasificación a la que cada técnica pertenece. Consideramos este punto uno de los más relevantes y recalcamos su importancia al momento de investigar y comprender el área de las técnicas de ocultamiento para usuarios de SBL en redes inalámbricas (específicamente las basadas en el principio de k-anonimato). Es importante poder responder al momento de estudiar y más aún, implementar una técnica, a la pregunta: ¿Qué tipo de protección real obtienen los usuarios con este encubrimiento?
Es claro que el objetivo final de todas estas técnicas y el resultado de sus algoritmos es generar una región de encubrimiento con una mayor resolución geográfica con respecto a la localización especifica del usuario que solicitó el servicio, de tal forma de **ocultar** a este dentro de otros k-1 usuarios y de esta forma mantener su información de localización "a salvo", sin embargo, debido a la naturaleza de cada técnica y los métodos utilizados en los algoritmos empleados en cada una de estas el grado y tipo de protección que estas brindan finalmente. Estos tipos de protección corresponden, como mencionábamos con anterioridad, a los siguientes:

- **Encubrimiento de anonimato.**
- **Encubrimiento de privacidad de localización.**
- **Encubrimiento de seguridad de localización.**

Técnicas que brindan protección del tipo "**encubrimiento de anonimato"** poseen la característica principal de que estas protegen la identidad del usuario en cuestión mediante la creación de una región de encubrimiento que posee K usuarios, teniendo en cuenta al usuario que solicita el servicio y k-1 otros usuarios **reales** que se encuentran **en el momento** en las cercanías del usuario. Efectivamente, esto brinda un determinado grado de protección para el usuario en cuestión (k-anonimato), pero, no puede garantizar que su localización en cuestión está protegida. Más aún, técnicas de este estilo pueden inclusive comprometer la seguridad e información de localización de todos los otros usuarios que fueron utilizados para crear la región de encubrimiento, muchas veces incluyéndolos en esta sin su consentimiento previo. Esto es debido a que, pese a que el SBL no puede saber con certeza de inmediato cuál de los usuarios presentes en esa región fue el que realizó

la solicitud del servicio, este sabe que estos se encuentran efectivamente ahí en el momento en que se realizó el servicio, por lo que su información de localización se encuentra expuesta en cierta medida. Además, al utilizar para la creación del área de ocultamiento usuarios reales, esto puede generar un problema en lo que a la respuesta de la query corresponde, debido a que si no se hay k-1 usuarios en el instante habrá que esperar hasta que se lleguen al área. Mayoritariamente, estas técnicas también utilizan la arquitectura de red **centralizada**, esto es, la utilización de un servidor de confianza (anonymizer) el cual contienen la información de localización de todos los usuarios que han solicitado servicios y genera las regiones de encubrimiento para cada persona, por lo que se ven además expuesta a los problemas y riesgos de seguridad que involucra utilizar estos dispositivos [3].

Como hemos mencionado anteriormente, un anonymizer sufre principalmente de dos tipos de problemas, este puede ver su rendimiento afectado al convertirse en un ***cuello de botella*** (**Bottle Neck**), cuando existen usuarios que están haciendo solicitudes simultáneamente, o inclusive usuarios que realizan actualizaciones constantes de su posición, y también, y más importante, se convierten en un punto único de ataque, si un adversario gana control del anonymizer este tiene acceso a la información de localización, queries, entre otros, de todos los usuarios que han realizado solicitudes al SBL.

Técnicas que brindan protección del tipo "**encubrimiento de privacidad de localización**" tienen la característica principal de que estas protegen la identidad del usuario en cuestión y su información de localización mediante la creación de una región de encubrimiento que posee K usuarios totales, considerando al usuario que realizó la solicitud del servicio y k-1 otros usuarios. Estos usuarios no corresponden a usuarios que se encuentran en el momento en el área de ocultamiento que se creó, sino que, corresponden a "huellas" (footprints), o información histórica de usuarios que alguna vez estuvieron en ese lugar, creando de esta forma "sujetos" (dummys) los cuales son incluidos en la región de encubrimiento de cada usuario que realiza una solicitud. Debido a esto, aunque el adversario identificara a los usuarios que se encuentran en la región, no podrá saber quien solicitó el servicio ni en qué momento en el tiempo se encontraba allí. De esta forma se evita también el problema presente en las técnicas que proveen encubrimiento de anonimato referente a exponer a su vez a los usuarios utilizados para crear el área de ocultamiento, debido a que estos no se encuentran ahí en el mismo momento en que un usuario solicita un servicio [3].

Este tipo de técnicas utiliza una base de datos de huellas, las cuales son "cargadas" inicialmente con los datos que se han recolectado de los usuarios en el área específica, y posteriormente se actualiza transcurrido un periodo de tiempo determinado a medida que más usuarios llegan a ciertas áreas utilizando sus dispositivos móviles. También, utilizan en su mayoría métricas de entropía para calcular la región de encubrimiento de forma óptima y escoger las huellas apropiadas para utilizarlas en el área de ocultamiento generada en conjunto con el usuario.
Por medio de la utilización de estos métodos garantizan al usuario que su localización se mantendrá segura frente a una gran cantidad de adversarios, y otorgan finalmente un mayor grado de seguridad a la información de localización de cada usuario.

Es por esto por estos motivos y más, que consideramos en este trabajo que este tipo de técnicas son de las más eficientes al momento de proteger la información de localización de los usuarios de SBL, y además tienden a ser más resistentes frente a una mayor cantidad de ataques (dependiendo de su arquitectura de red).

Técnicas que brindan protección del tipo "**encubrimiento de seguridad de localización**" tienen la característica principal de que proveen a los usuarios o "nodos" con protección a su información de localización de tal forma que no sea posible para un usuario identificar un nodo y destruirlo físicamente. Estas técnicas usan un principio diferente del que hemos visto en aquellas que utilizan e implementan k-anonimato, debido a que aquí se requiere un número k con un valor alto que corresponde a la cantidad de usuarios que se encontrarán en el área de encubrimiento que se generará para un usuario especifico, siendo esta región de un tamaño pequeño. En cambio, el encubrimiento de seguridad de localización genera una región de encubrimiento para cada nodo presente en una red la cual contiene una cierta cantidad de nodos adicionales, pero para la creación de esta se basa en una teoría la cual considera que un nodo tiene un nivel de seguridad "S", el cual corresponde al tamaño del área que adopta cada nodo dividido la cantidad de nodos que están en esta misma. Esto quiere decir, que mientras **más grande** sea el área y **menor** la cantidad de nodos, mayor será su nivel de seguridad, y mientras mayor sea el nivel de seguridad de u nodo, menos atractivo es para un adversario buscar por algún nodo en el área y destruirlo, debido a que esto conllevaría a costos demasiado elevados.

Por tanto, el encubrimiento de seguridad de localización se basa en un principio **inverso** al utilizado en las técnicas basadas en k-anonimato, debido a que al adversario no le interesa realmente saber las identidades de cada nodo, sino que solamente desea saber su posición y así poder destruirlo [5].

Teniendo esto en consideración, hemos concluido que las técnicas que pertenecen a la clasificación de "encubrimiento de privacidad de localización" ofrecen un grado de seguridad mayor para los usuarios de SBL. No solo protegen la identidad del usuario que solicita el servicio y su información de localización de una forma eficiente, sino que también evitan poner en riesgo a otras personas mediante la utilización de la información histórica de usuarios que han estado por los alrededores en un determinado momento. Además, estas brindan protección contra los ataques de "Identificación de Espacios Restringidos", el cual es una de las formas más fáciles y accesibles para un adversario de obtener información sensible de los usuarios de SBL.

Las técnicas que pertenecen a la clasificación de "encubrimiento de seguridad de localización" brindan un tipo de protección orientado a un escenario diferente referente a las otras dos clasificaciones. Este tipo de técnicas está orientado a lo que corresponde a campos de batalla virtuales donde un adversario solo desea destruir nodos en un área determinada. Es por esto que nos centraremos en los otros tipos de técnicas para elegir las cuales simularemos. Sin embargo, es importante tenerlas en consideración para trabajos futuros.

Con todo lo mencionado anteriormente, hemos decidido seleccionar en primera instancia la técnica del autor Ben Niu "***Achieving k-Anonimity in Privacy-Aware Location-based Services***". Esta técnica garantiza "encubrimiento de seguridad de localización" y, además, utiliza métodos de entropía para seleccionar a los usuarios que formaran parte de una región de encubrimiento.

Luego de esto, hemos decidido seleccionar como segunda técnica para simular la técnica "***Query Aware Location Anonymization for Road Networks***" [16], esto principalmente por dos motivos. El primero consiste en que es una de las técnicas que considera y soporta trajectory cloaking y garantiza un resultado eficiente al momento de generar las regiones de encubrimiento para cada usuario. Y segundo, debido a que es una de las pocas técnicas que es "Query Aware", es decir, tiene en consideración tanto los tiempos de ejecución de las query que se reciben para generar regiones de encubrimiento, como así también los costos asociadas a estas, otorgando así soluciones óptimas en este aspecto.

Contribuciones de Trabajo

Teniendo en cuenta las clasificaciones que se han descrito para las técnicas de ocultamiento en Servicios Basados en la Localización es posible identificar de forma más clara el tipo de protección que brindan estas a los usuarios de SBL y otras características asociadas a estas, de la misma forma que saber los problemas que generalmente están relacionados a cada técnica por medio de su clasificación. En base a esto entonces surgen las siguientes preguntas: ¿existe una técnica que pertenezca a más de una clasificación a la vez? ¿es cada clasificación excluyente una de otra? ¿hay una relación entre cada clasificación?

En este trabajo otorgamos una respuesta a estas preguntas, considerando las definiciones que hemos realizado de cada técnica basándonos en lo que el autor Toby Xu [3] ha definido previamente y también, la forma en la que opera cada técnica.

Retomando las definiciones de cada clasificación, tenemos entonces que cada técnica que pertenece al **encubrimiento de anonimato** corresponde a aquella que define una región de encubrimiento para un usuario "U" mediante la inclusión de K-1 otros usuarios **reales** en una región generada a partir de la posición del usuario que solicitó el servicio. Esto quiere decir, que finalmente en la región de encubrimiento se encontrarán K usuarios en total, los cuales están en esa zona al momento en que se generó la consulta. Esta es la característica principal de este tipo de técnicas, según lo considerado en este trabajo.

Cada técnica que pertenece al **encubrimiento de privacidad de localización** corresponde a aquella que define una región de encubrimiento para un usuario "U" por medio de la inclusión de K-1 **huellas** (footprints), información histórica de usuarios que han estado en las cercanías de la zona donde se encuentra el usuario que solicitó el servicio, de esta forma se genera una región que contiene al usuario y las huellas correspondientes. Esto significa entonces que, finalmente, en la región de encubrimiento generada por este tipo de técnicas se encontrará el usuario junto K-1 usuarios "ficticios" (huellas correspondientes a usuarios que alguna vez estuvieron en la zona, pero no necesariamente en este momento). Consideramos por ende esta como la característica distintiva y principal de las técnicas que pertenecen a esta clasificación

Cada técnica que pertenece al **encubrimiento de seguridad de localización** corresponde a aquella que define una región de encubrimiento para un nodo o usuario "U" por medio de la inclusión de nodos adicionales **reales**, generando así una región de encubrimiento del **mayor** tamaño posible y que contenga la **menor** cantidad posible de nodos dentro de ella. Esto debido a que, a diferencia de los problemas relacionados a la seguridad de localización y grado de anonimato, un adversario no está realmente interesado en saber la identidad de un usuario especifico, sino que simplemente desea saber su ubicación y destruirlo. Consideramos entonces esta como la característica principal que define a las técnicas pertenecientes a esta clasificación.

Observando las definiciones y características principales de cada clasificación, podemos apreciar una similitud interesante entre **encubrimiento de anonimato** y **encubrimiento de seguridad de localización**; en efecto, ambas clasificaciones basan la generación de su región de encubrimiento mediante la utilización de usuarios **reales** adicionales al usuario que solicitó el servicio. También notamos y deducimos que, dado a que las técnicas que brindan **encubrimiento de anonimato** protegen la identidad del usuario que solicitó el servicio al ocultarlo dentro de una región con otros k-1 usuarios, las técnicas que proveen **encubrimiento de privacidad de localización** son también capaces de proteger la identidad del usuario ya que, pese a que los k-1 usuarios utilizados para generar la región de encubrimiento no son usuarios que se encuentran en ese momento, el adversario no sabe quién o quienes estuvieron ahí y en qué momento, por lo que todos corresponden a usuarios que pudieron haber realizado la consulta, efectivamente ocultando la identidad del usuario que solicitó el SBL. Por ende, podemos decir que este tipo de técnicas también brindan **encubrimiento de anonimato**. Sin embargo, es importante destacar que, debido a un ataque especifico mencionado con anterioridad, el llamado "Ataque de Observación" [2] no es posible asegurar que estas técnicas garanticen siempre encubrimiento de anonimato, ya que estas no brindan protección contra este ataque en particular según se puede apreciar en la tabla comparativa, viéndose la identidad del usuario comprometida en caso de que un adversario utilice este ataque.

5 SIMULACIONES

Habiendo realizado los estudios y análisis correspondientes a las técnicas de ocultamiento que hemos incluido en este trabajo, se realizará un simulador gráfico que permita ver claramente el comportamiento de las técnicas que hemos seleccionado finalmente. El simulador ha sido desarrollado en el lenguaje **Java**, implementando específicamente una aplicación **Java de escritorio** de tal forma que pueda ser visualizada en múltiples plataformas de acuerdo a las necesidades del usuario.

La aplicación genera caminos o rutas que representan a un área urbana en pantalla, cargando cada posición mediante un archivo .txt que contiene coordenadas (x,y).

Figura 5: Pantalla de inicio aplicación.

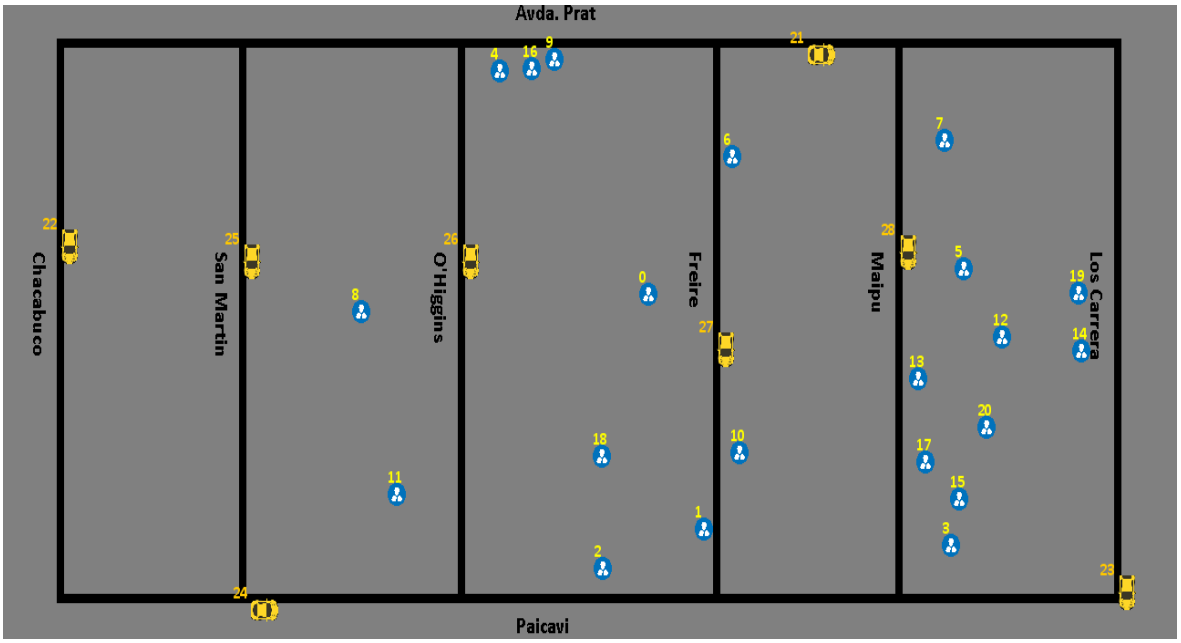


Figura 6: Rutas generadas con simulador

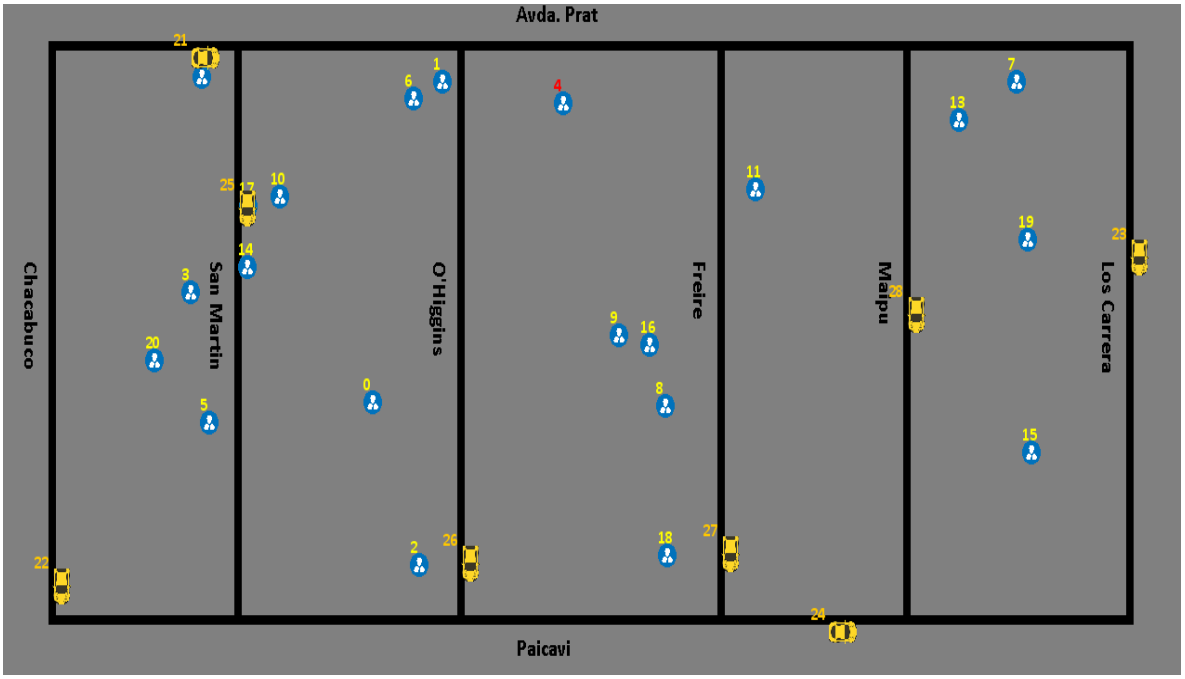


Figura 7: Usuario seleccionado dentro de caminos nuevos generados.

Los caminos generados se aprecian en las figuras 6 y 7. El usuario seleccionado actualmente se señala con el color **rojo** en su índice correspondiente. En este caso, este corresponde al usuario con índice **4**.

Figura 8: Botonera simulador gráfico.

La figura 8 muestra los botones y combobox con los que el usuario puede interactuar en la aplicación. El combobox "Seleccione usuario" permite la selección de un usuario presente en el área, el botón "Pausar" permite detener el movimiento de los usuarios de tal forma de visualizar más claramente la región que se crea, así como también reanudar el movimiento cuando esté pausado. "Generar región" genera la región de encubrimiento con los datos seleccionados. "Seleccione técnica" permite seleccionar la técnica a simular", e "Ingrese k" obtiene el número k con el cual se desea realizar la simulación. Finalmente, el botón "Grilla ON/OFF" se encarga de mostrar la grilla de probabilidades del área u ocultarla, en caso de que esté presente.

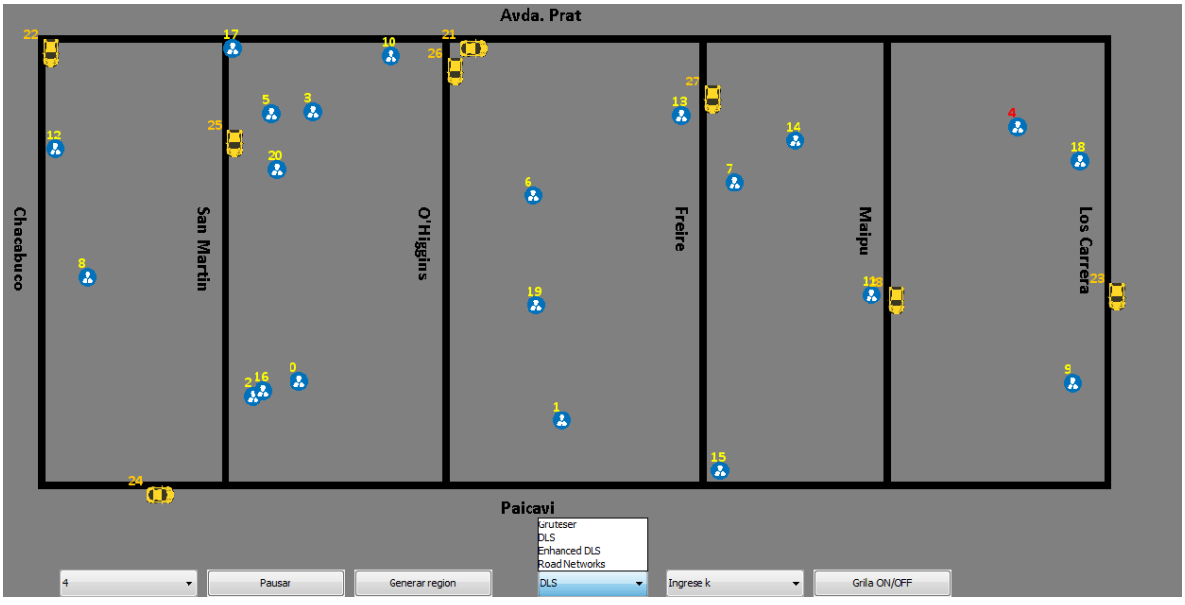


Figura 9: Listado de técnicas incluidas para simular.

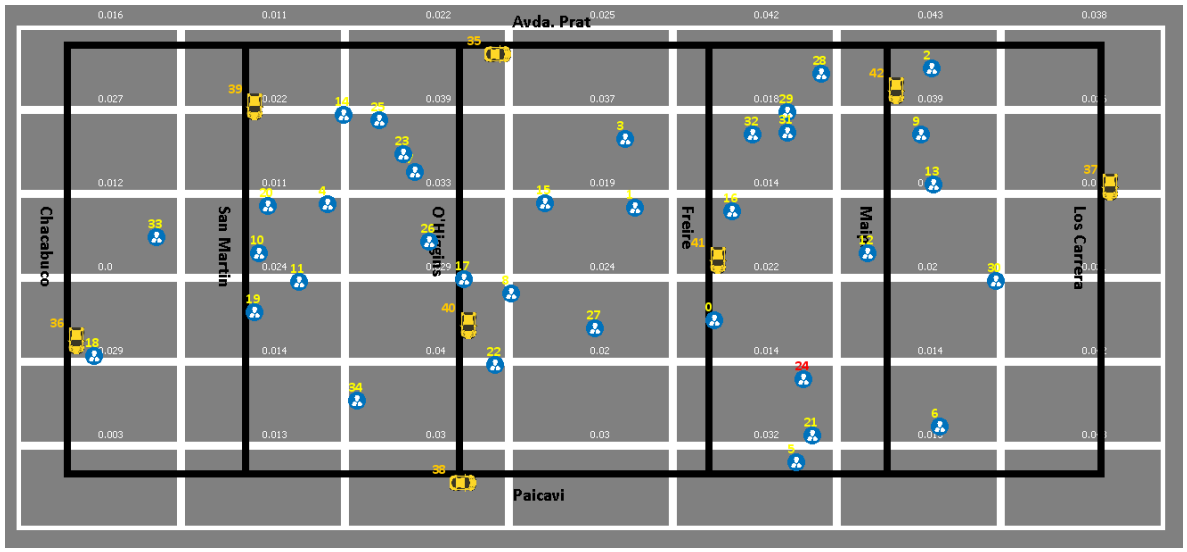


Figura 10: Grilla visualizada en el simulador.

En la figura 10 se puede apreciar la grilla que se utiliza en el simulador. Esta grilla posee las probabilidades de cada área en los caminos generados, las cuales se generan de forma aleatoria y normalizada.

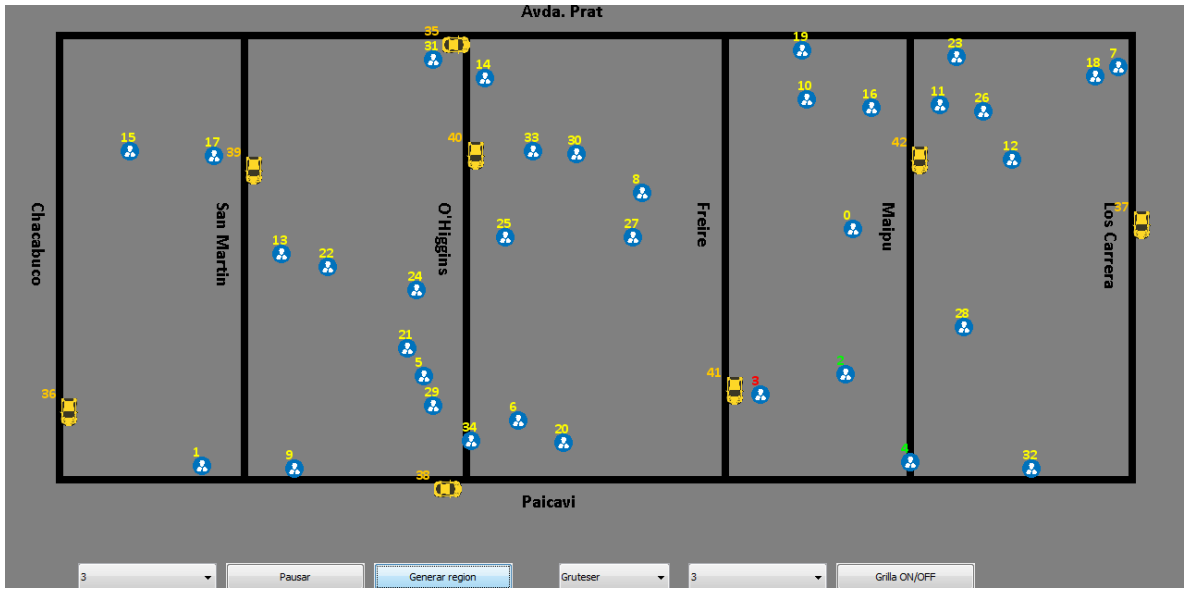


Figura 11: Simulación de la técnica del trabajo de Gruteser. [2]

En la figura 11 se aprecia una simulación de la técnica propuesta por Gruteser [2]. Se seleccionan de esta forma k-1 otros usuarios que se encuentran en el área para generar la región del usuario que solicitó el servicio (usuario seleccionado). Los k-1 usuarios seleccionados se visualizan con el color verde en sus índices, y el usuario que seleccionó el servicio con el color rojo en su índice.

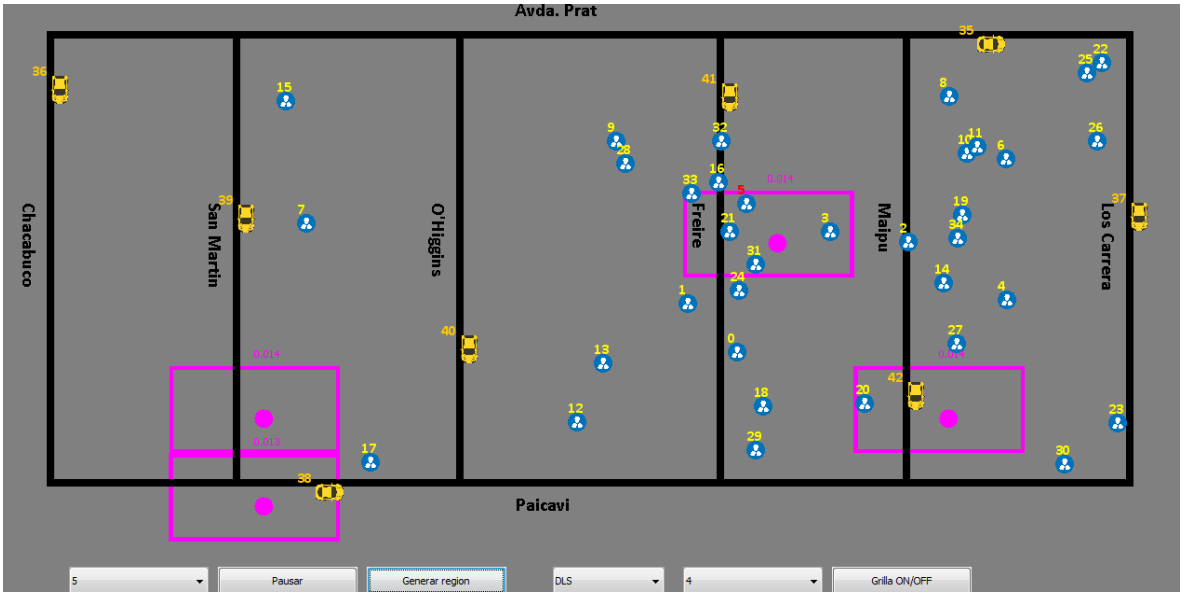


Figura 12: Simulación técnica Dummy Location Selection del autor Ben-Niu [18].

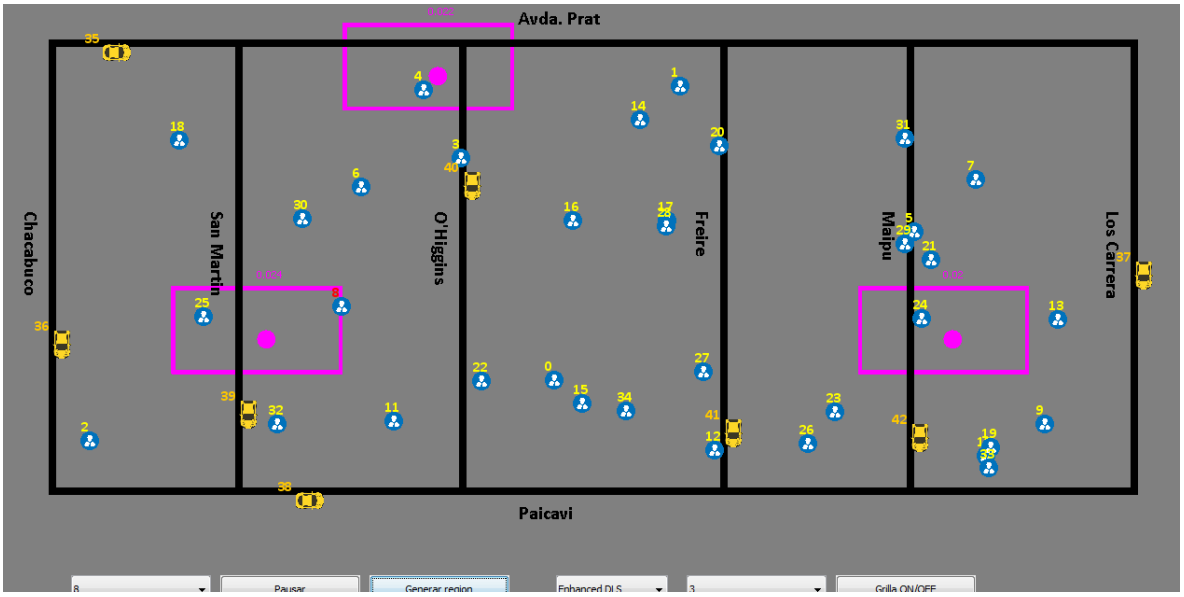


Figura 13: Simulación técnica Enhanced Dummy Location Selection por Ben-Niu [18].

En las figuras 12 y 13, se visualiza la ejecución de las técnicas de encubrimiento correspondientes a Dummy Location Selection y Enhanced Dummy Location Selection, respectivamente.

El usuario que solicitó el servicio se identifica con el color rojo en su índice, las regiones que han sido seleccionadas para su región de encubrimiento se marcan con el color magenta. En cada región seleccionada de esta forma, se puede ver un circulo de color magenta, este hace referencia a la existencia de un "sujeto" en esa zona, como describe la probabilidad asociada en la grilla.

Figura 14: Simulación técnica Road Networks del autor Chow C-Y [16].

En la figura 14 se visualiza la simulación de la técnica de encubrimiento propuesta por el autor Chow C-Y. El usuario (**solo** del tipo "vehículo" en el caso de esta técnica), se identifica con el color rojo en su índice, los caminos seleccionados para su región de encubrimiento se marcan con el color verde. Esta técnica no considera a usuarios del tipo "persona" para la generación de la región de encubrimiento.

5.1 Modelos de movimiento

Como se mencionó con anterioridad, en este proyecto se utilizarán dos algoritmos de movimiento con la finalidad de representar de una forma óptima y cercana a la realidad el movimiento de personas y vehículos en una zona urbana, como lo es por ejemplo la ciudad de Concepción, en las simulaciones gráficas que se desarrollarán.

Los dos algoritmos a implementar y utilizar para las simulaciones son los siguientes:

- *Random Walk.*
- *Random Waypoint.*

Random Walk: También conocido como "*Caminata Aleatoria*" o "*Paseo Aleatorio*" es una formalización matemática de la trayectoria que se crea al realizar sucesivos pasos aleatorios. Este concepto es aplicado en una variedad de campos para estudiar diversos fenómenos como el camino que toma una molécula a través de un líquido o gas (Física), el precio de una acción fluctuante (Economia), la posición una persona en un área definida en un determinado periodo de tiempo (Ciencias de la Computación). Es esto último lo que es de interés para este trabajo, debido a que de esta forma podemos simular el movimiento de un número específico de "personas" o "usuarios" dentro del área urbana de la Ciudad de Concepción y utilizar estos mismos como "dummys" o sujetos de prueba para los cálculos posteriores de regiones de encubrimiento en las simulaciones gráficas.

Para realizar cada movimiento, se utiliza el principio básico de los algoritmos de este estilo, el cual dice que la posición de un punto o "usuario" en un cierto instante depende **sólo** de su posición en un instante previo y una variable aleatoria que determina su nueva dirección y longitud de paso.
Para efectos del algoritmo, esto se traduce como lo describimos a continuación:

$$X(i+I) = X(i) + A(I)$$

Donde "X(i)" corresponde a una trayectoria que empieza con X(0), "A" es la variable aleatoria que determinará el siguiente paso y "I" corresponde al intervalo de tiempo que hay entre un paso determinado y el siguiente.

Es importante destacar que, un paseo aleatorio define el movimiento de un usuario de forma **libre** en el área en que este se mueve, lo cual significa que un usuario puede moverse tomando cualquier dirección y cualquier camino que desee sin importar si este sea un cruce o una vía urbana. Es debido a esto, dado que trabajaremos las simulaciones gráficas en un entorno urbano, que hemos implementado una variación de este algoritmo solo para los usuarios que corresponden a *personas* y no los vehículos, debido a que estos últimos tienen limitaciones en cuanto a los lugares por los cuales pueden moverse. A diferencia de los vehículos, consideramos que las personas son capaces de moverse netamente por cualquier lugar en el área de la ciudad, esto incluye tanto las zonas de tránsito peatonal como las rutas destinadas para el movimiento de vehículos, como lo son calles y carreteras.

En base a esta primicia implementamos el movimiento de los usuarios (personas y vehículos) de dos formas ligeramente diferentes, dependiendo finalmente de sus limitaciones de movimiento correspondientes (calles, carreteras, vías peatonales, etc), y teniendo en cuenta también las características de cada tipo de usuario, como su posición en el área y, principalmente, la **velocidad** a la que estos se están moviendo. Es esta variable la cual nos permite identificar de forma clara y precisa el tipo de usuario al que nos enfrentamos, esto ya que un usuario de tipo "vehículo" siempre tendrá una velocidad de movimiento superior a un usuario de tipo "persona", por lo que teniendo conocimiento de esta variable es fácilmente posible realizar el movimiento correcto para el tipo de usuario correspondiente en la zona donde se realizarán las simulaciones gráficas

Random Waypoint: corresponde al segundo modelo de movimiento que hemos seleccionado para realizar las simulaciones gráficas de las técnicas seleccionadas. En el manejo de movilidad (*Mobility Management*), el modelo *Random Waypoint* es un modelo aleatorio para generar el movimiento de usuarios móviles, y permite observar como su localización, velocidad y aceleración cambian con el tiempo. Este modelo es uno de los modelos de movilidad más populares utilizado mayoritariamente con propósitos de diferentes tipos de simulación, sobre todo en el área de las Redes Ad Hoc Móviles, para evaluar protocolos de rutas (*Routing Protocols*), esto es debido a su amplia difusión y simplicidad.

Una de las características más importantes de este modelo, es que se basa y permite a los nodos o usuarios moverse de forma aleatoria libremente sin restricciones. Esto quiere decir que tanto la velocidad, su localización en un determinado momento y dirección que toman para moverse son todas escogidas de forma aleatoria e independiente de otros usuarios (esto es entonces, un usuario solo depende de sí mismo para obtener sus "variables de movimiento").

El modelo finalmente se basa en lo que describimos a continuación: Cada usuario o nodo comienza su "camino" realizando una pausa de una determinada cantidad de segundos. Luego, este selecciona una nueva posición aleatoria a la que se moverá en el área de simulación y una velocidad aleatoria entre 0 y un máximo pre-definido. El usuario se mueve entonces a ese lugar y pausa de la misma manera que se describió anteriormente, espera una cantidad determinada de segundos y obtiene una nueva posición donde moverse, así como también su velocidad de movimiento. Es así entonces, como esto se repite en un ciclo mientras la simulación esté en ejecución.

Estos modelos han sido la base en la que hemos basado la construcción de las simulaciones gráficas de las dos técnicas de encubrimiento que trabajaremos de ahora en adelante en este proyecto, por lo que es importante tenerlos en cuenta al momento de observar el movimiento de los usuarios en el área de las simulaciones y posteriormente, los resultados obtenidos en estas.

5.2 Técnicas a simular

Tras haber realizado la descripción y comparación de cada técnica que se ha estudiado en este trabajo, hemos seleccionado dos técnicas de las cuales generaremos simulaciones gráficas utilizando el simulador desarrollado.

Las técnicas a simular corresponden entonces finalmente a "*Achieving k-anonimity in Privacy Aware location Based Services*" [18] y "*Query Aware Location Anonymization for Road Networks*" [16] principalmente debido a que ambas poseen una característica importante que hace que se diferencien del resto y son por ende más llamativas al momento de visualizar su funcionamiento mediante el simulador. La primera utiliza técnicas de entropía para generar cada región de encubrimiento, mientras que la última genera una región de encubrimiento diseñada específicamente para vehículos que se movilizan por caminos y solicitan un Servicio Basado en la Localización.

Asi también, se incluyó la simulación de la técnica "Anonymous Usage of Location Based Services Through Spatial and Temporal Cloaking" [2] debido a que esta técnica corresponde a una de las primeras propuestas en la literatura, y en la que muchas otras han basado su desarrollo.

6 ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE

6.1 Alcances

Este software se desarrolló con la finalidad de realizar las simulaciones gráficas de las técnicas seleccionadas en este proyecto de título. En primera instancia se encarga de la generación de caminos o rutas, simulando un área urbana como el área de Concepción centro, en la cual se encuentran usuarios y vehículos desplazándose continuamente. El desplazamiento se realiza mediante la utilización de los algoritmos de movimiento descritos con anterioridad. Una vez generados los caminos y usuarios que se mueven en la zona, permite la simulación de cada técnica según sea la elección del usuario mediante una casilla ComboBox sencilla, mostrando gráficamente cual es la región de encubrimiento de un usuario seleccionado previamente según la técnica elegida. Generada esta región, el usuario puede volver a simular cualquier otra técnica que desee con un usuario presente en el área generada. Los caminos se generan automáticamente al iniciar la aplicación, siendo controlados por un archivo .txt donde se encuentran las coordenadas de las rutas.

6.2 Objetivo del software

El software debe lograr simular gráficamente las técnicas de encubrimiento seleccionadas en este trabajo.

Esto se consigue implementando caminos o rutas similares a las presentes en el área céntrica de Concepción, generando en estos usuarios ficticios (personas y vehículos) que se mueven en esta zona. En base a estos usuarios es posible entonces seleccionar al que se desee y de esta forma entonces, la aplicación genera la región de encubrimiento correspondiente a ese usuario en base a los demás usuarios presentes en el área.

La región de ocultamiento se genera de la misma forma en la que se describe según su autor original, es decir, en el caso de la técnica propuesta por Ben Niu se mostrarán los puntos de los usuarios que están presentes en la zona, y en el caso de ser la técnica *Query Aware Location Anonymization for Road Networks* se mostrarán los segmentos que formarán entonces la región de encubrimiento.

6.3 Descripción Global del Producto

6.3.1 Interfaz de usuario

Las interfaces de usuario se diseñaron de forma que fueran sencillas y amigables al usuario.
Mediante la pantalla principal, que se tiene acceso al iniciar la aplicación, se genera una ventana de 640x520, donde se dan a conocer las opciones y la finalidad del software de forma clara. Se encuentra visible el botón "Generar caminos" con el cual se generan las rutas al momento de presionarlo, mostrando el mapa de caminos en la siguiente pantalla de la aplicación, con una resolución de 640x520 también. En esta el usuario puede seleccionar un vehiculo/persona y una técnica de encubrimiento y generar la región de encubrimiento de este la cual se mostrará en la siguiente pantalla.

6.3.2 Interfaz Software

Al ser una aplicación de escritorio desarrollada en el lenguaje de programación Java, requerirá de un Sistema Operativo con interfaz gráfica tales como Windows XP o superior, Mac OS X versión 10.7.3 o superior, Ubuntu 10.04 en adelante, etc.

Como mínimo, en el caso de la utilización de Windows XP, se requiere 64 MB de memoria RAM instalada. 128 MB de RAM son necesarios en caso de tener otro Sistema Operativo de Windows. De la misma forma, se requiere un espacio libre en disco de 124 MB para su instalación.

6.3.3 Interfaces de comunicación

Debido a que es una aplicación de escritorio, no son requeridos protocolos específicos de internet para su funcionamiento.

6.4 Requerimientos Específicos

6.4.1 Requerimientos Funcionales del sistema

Pese a que no es necesario que el software genere ninguna clase de informe, se detalla cómo se comporta este frente a cada acción del usuario. Principalmente, la aplicación responde a las elecciones del usuario de la siguiente manera:

- Generar caminos. En el momento en que el usuario presiona el botón "Generar caminos", se generan las rutas donde se encuentran los vehículos/personas moviéndose constantemente.
- Generar región de encubrimiento. En el momento en que el usuario presiona el botón "Generar región de encubrimiento", se genera la región de encubrimiento del vehiculo/persona elegido según la técnica de ocultamiento que ha sido seleccionada mediante un ComboBox.
- Volver a rutas. En el momento en que el usuario presiona el botón "Volver a rutas", se vuelve a generar el área con las rutas y usuarios presentes con anterioridad.

6.4.2 Interfaces externas de entrada

La aplicación recibirá como datos de entrada los que se describen a continuación.

Identificador	**Nombre del ítem.**	**Detalle de Datos contenidos en ítem**
DE_01	Ubicaciones de rutas	POSICION X1, POSICION Y1, POSICION X2, POSICION Y2

6.4.3 Interfaces externas de Salida

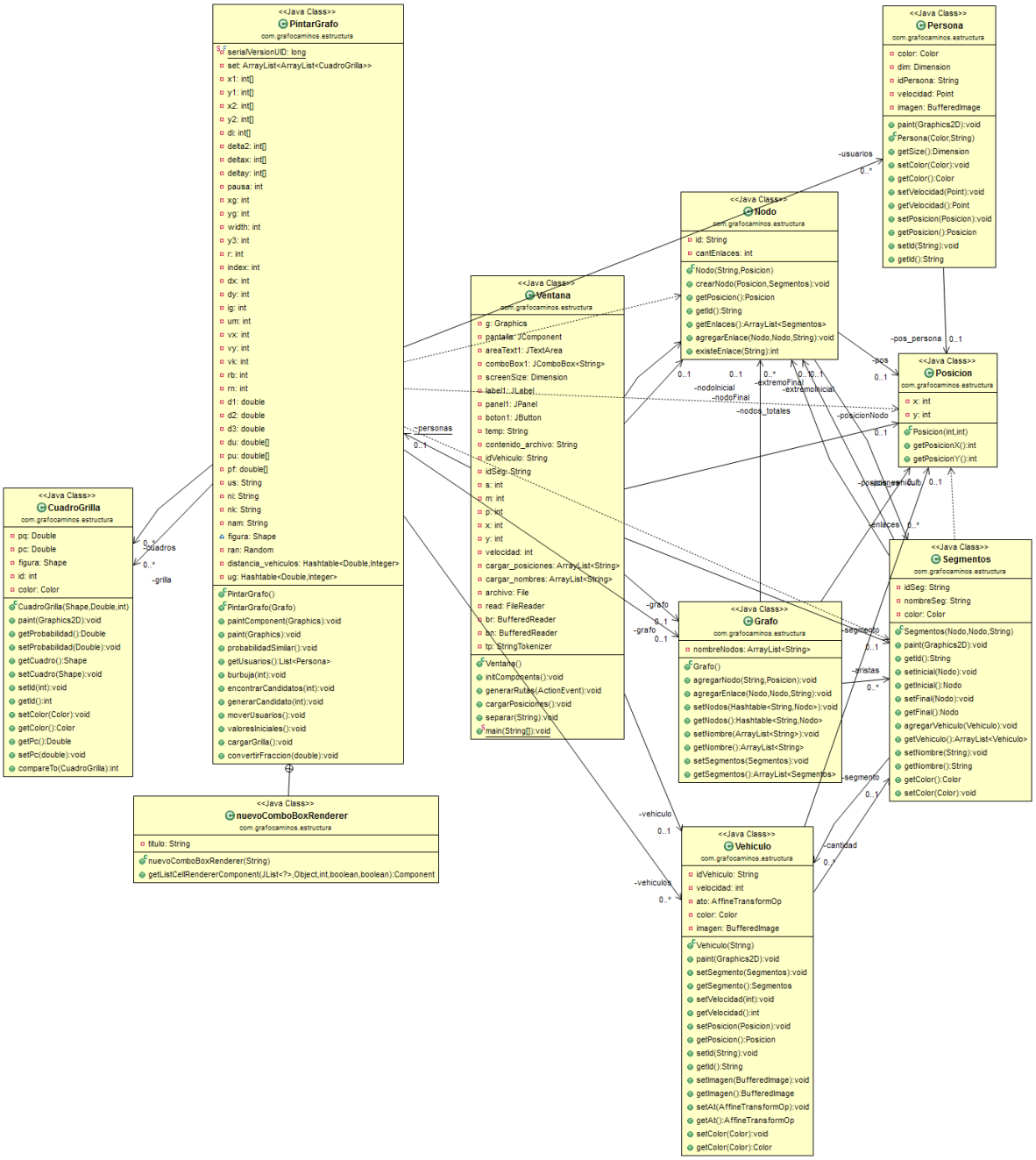
La aplicación tendrá como datos de salida los que se describen a continuación.

Identificador	**Nombre del ítem.**	**Detalle de Datos contenidos en ítem**	**Medio Salida**
IS_01	Área de caminos	RUTAS	Pantalla
IS_02	Región de encubrimiento	REGION OCULTAMIENTO	Pantalla

7 ANÁLISIS

7.1 Diagrama de Clases

El software desarrollado en Java responde al siguiente modelo en el cual se basó para su construcción.



7.2 Casos de uso

Se representan los requerimientos funcionales mediante Casos de Uso para cada uno.

7.2.1 Actores

- Usuario:

El usuario corresponde a la persona que utilizará el simulador gráfico desarrollado. En el software tiene permisos para ejecutar todas las operaciones disponibles. No requiere de grandes conocimientos técnicos, solo debe estar familiarizado con el entorno gráfico de Java y con el uso de botones y CheckBox.

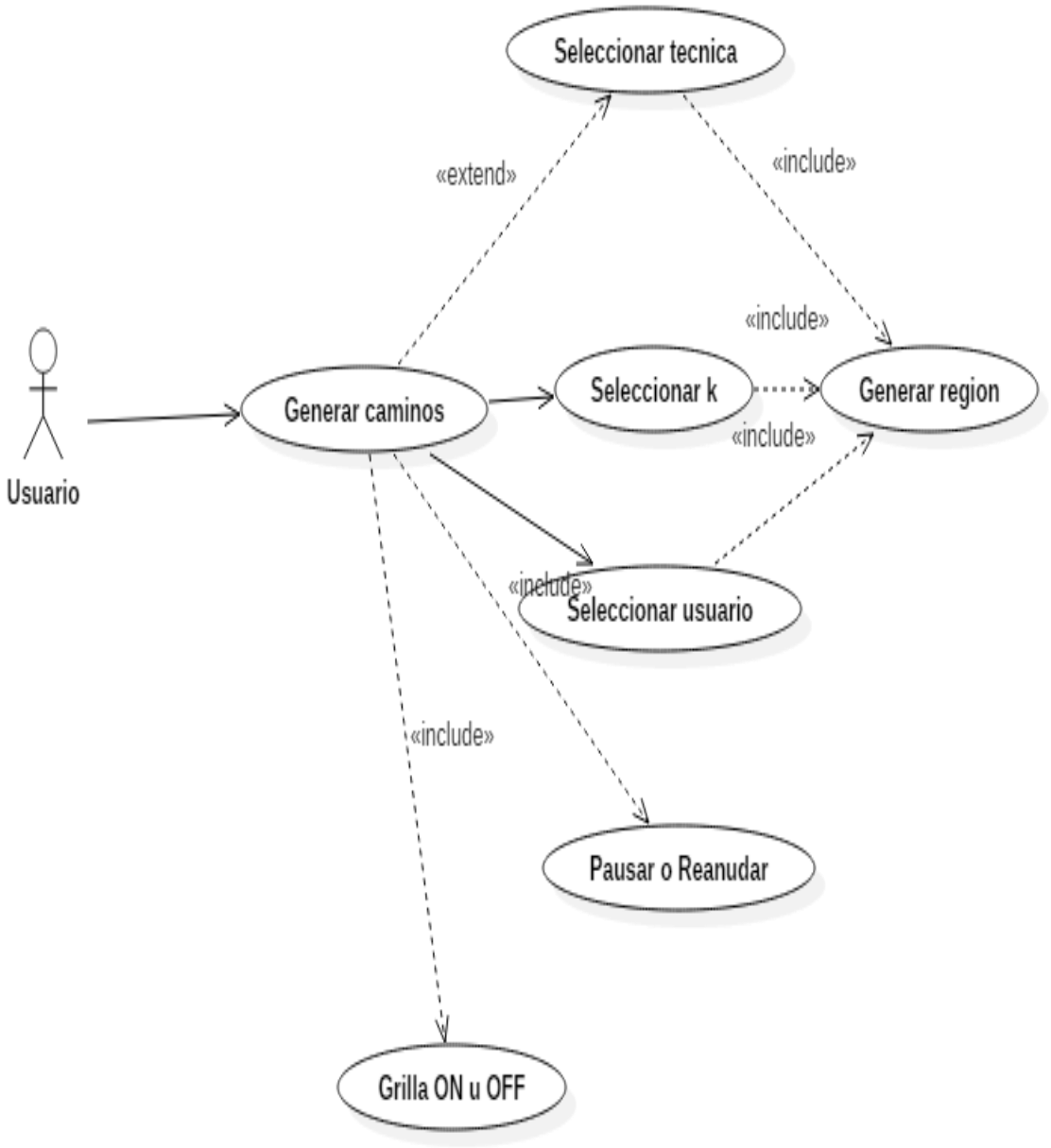
7.2.2 Casos de Uso y descripción

A continuación, se visualizan las interacciones que tienen los usuarios con la aplicación.

El actor "Usuario" es el que utiliza el software desarrollado. Este es quien ejecuta la aplicación utilizando un computador o notebook y puede ingresar nuevas posiciones para generar las rutas o caminos y visualizar así un área urbana simulada con una cantidad determinada de personas y vehículos. En base al área generada, puede finalmente generar la región de encubrimiento de un usuario que seleccione, dada una técnica que él especifique.

El usuario posee completo acceso a las funcionalidades del simulador sin la necesidad de realizar un log in.

Diagrama de Casos de Uso:



7.2.3 Especificación de los Casos de Uso

7.2.3.1 Caso de Uso: <Generar caminos >

- Descripción: Se generan las rutas o caminos.
- Pre-Condiciones: El usuario debe presionar el botón "Generar rutas".
- Flujo de Eventos Básicos:

Al actor	El sistema
	1.- Se generan las rutas o caminos definidas en el archivo .txt.
	2.- Se muestra por medio de una interfaz gráfica el área generada.

- Flujo de Eventos Alternativo:

Al actor	El sistema
	1(a).- Se informa al usuario que las rutas están vacías o no definidas.
	2(a).- Se informa que no se han podido generar el área definida por las rutas.

- Post-Condiciones: Se muestra el área generada por pantalla.

7.2.3.2 Caso de Uso: <Seleccionar usuarios >

- Descripción: Se establece una persona o vehiculo dentro de todos los presentes del que se generará su región de encubrimiento.
- Pre-Condiciones: El usuario debe seleccionar una persona o vehiculo dentro de los presentes.
- Flujo de Eventos Básicos:

Al actor	El sistema
1.- Se selecciona una de las personas o un vehículo dentro de los presentes en el área.	2.- Se establece una persona o vehiculo como "seleccionado".

- Flujo de Eventos Alternativo:

Al actor	El sistema
	2(a).- Ninguna persona o vehiculo es establecido como "seleccionado".

- Post-Condiciones: Se muestra gráficamente mediante un color determinado (rojo) la persona o vehículo que ha sido seleccionado.

7.2.3.3 Caso de Uso: <Seleccionar técnica >

- Descripción: Se define una técnica de ocultamiento la cual será utilizada para generar la región de encubrimiento.
- Pre-Condiciones: El usuario selecciona la técnica dentro de una lista previamente definida en un campo ComboBox.
- Flujo de Eventos Básicos:

Al actor	El sistema
1.- Se selecciona la técnica de ocultamiento que se desee dentro de un ComboBox.	2.- Se define la técnica de ocultamiento en la aplicación.

- Post-Condiciones: Se establece la técnica de ocultamiento en el campo ComboBox.

7.2.3.4 Caso de Uso: <Seleccionar k>

- Descripción: Se define el valor de "k" para generar la región de encubrimiento
- Pre-Condiciones: El usuario debe seleccionar un valor de "k" mediante un ComboBox luego de haber generado los caminos
- Flujo de Eventos Básicos:

Al actor	El sistema
1.- Se selecciona el valor del número "k" que se desee dentro de las opciones otorgadas por un ComboBox.	1.- Se define el valor de "k" en la aplicación

- Post-Condiciones: Se establece el valor del número "k" en el campo ComboBox.

7.2.3.5 Caso de Uso: <Pausar u Reanudar>

- Descripción: Se pausa o reanuda el movimiento de los usuarios en el "mapa" dependiendo si está activo o pausado.
- Pre-Condiciones: El usuario debe presionar el botón "Pausar/Reanudar" luego de haber generado los caminos.
- Flujo de Eventos Básicos:

Al actor	El sistema
1.- El usuario presiona el botón "Pausar/Reanudar".	1.- Se pausa el movimiento de los usuarios en caso de que esté activo.
	2.- Se reanuda el movimiento de los usuarios en caso de que esté pausado.

- Post-Condiciones: Se pausa o reanuda el movimiento de los usuarios en pantalla.

7.2.3.6 Caso de Uso: <Grilla ON u OFF >

- Descripción: Se visualiza la grilla de probabilidades del "mapa" en caso de estar invisible o se oculta en caso de estar visible.
- Pre-Condiciones: El usuario debe presionar el botón "Grilla ON/OFF" luego de haber generado los caminos.
- Flujo de Eventos Básicos:

Al actor	El sistema
1.- El usuario debe presionar el botón "Grilla ON/OFF".	1.- Se muestra en pantalla la grilla de probabilidades en caso de que se encontrara "oculta".
	2.- Se oculta la grilla de probabilidades en caso de que se encontrara visible en pantalla.

7.2.3.7 Caso de Uso: <Generar región>

- Descripción: Se genera la región de encubrimiento luego de haber elegido una persona o un vehículo y una técnica de ocultamiento.
- Pre-Condiciones: El usuario debe presionar el botón "Generar región" luego de haber elegido una persona o un vehículo y seleccionado la técnica de ocultamiento.
- Flujo de Eventos Básicos:

Al actor	El sistema
1.- El usuario presiona el botón "Generar región".	1.- Se calcula la región de encubrimiento de acuerdo a los parámetros establecidos con anterioridad.
	2.- Se muestra por pantalla la región de encubrimiento correspondiente.

- Flujo de Eventos Alternativo:

Al actor	El sistema
	1(a).- Se informa al usuario que no se ha podido calcular la región de encubrimiento debido a falta de información (elegir una persona o un vehículo, por ejemplo) o un error en los datos.

- Post-Condiciones: Se muestra la región de encubrimiento generada por pantalla.

8 CONCLUSIONES

En el proyecto desarrollado se han presentado diversos desafíos que se deben afrontar los cuales han sido abordados en los diferentes tópicos de este informe. En primera instancia, se aprecia dentro de los aportes de este proyecto el estudio y análisis bibliográfico del área de las técnicas de encubrimiento para servicios basados en la localización, el cual ha sido realizado de forma exitosa y con buenos resultados. Se ha podido brindar una descripción clara y concisa de lo que corresponde al estado del arte dentro de la literatura, de tal forma de eliminar el desconocimiento que se tiene sobre esta área en la región.

Así mismo, a partir de este estudio realizado, se consiguió realizar una comparación satisfactoria de las técnicas analizadas de forma en que se evidencie tanto las fortalezas como debilidades que poseen cada una. Esto otorga una vista más clara sobre la forma en que se comporta cada técnica, y permitió por ende cumplir el siguiente objetivo exitosamente también. Este objetivo corresponde a la selección de dos técnicas de encubrimiento para su posterior implementación dentro del simulador gráfico desarrollado, esto fue realizado de forma mucho más sencilla gracias a la tabla comparativa realizada.

Además de lo relacionado al estudio bibliográfico, en este trabajo se destacan los aportes realizados en el simulador gráfico desarrollado. Por medio de un estudio a dos modelos de movimiento (RW y RWP), se realizó la implementación de un simulador gráfico que genera un área de caminos o rutas y simula el movimiento de usuarios (personas y vehículos) dentro de esta. En base a esto se ejecutan los algoritmos de encubrimiento seleccionados para generar la región de encubrimiento correspondiente.

En este proyecto se ha complementado la literatura existente, de forma que se tengan definiciones más estandarizadas referente al tema del encubrimiento de localización para usuarios de Servicios Basados en la Localización.

9 BIBLIOGRAFÍA

[1] - Carmela Troncoso, "Design and analysis methods for privacy technologies"

[2] - Marko Gruteser, Dirk Grunwald, "Anonymous usage of Location-Based services through and temporal cloaking".

[3] - Toby Xu, Ying Cai, "Feeling-based Location Privacy Protection for Location-based Services". ACM CCS'09.

[4] - Toby Xu, Ying Cai, "Location Safety Protection in Ad Hoc Networks". Ad Hoc Networks (Elsevier) Journal, Special Issue: "Privacy and Security in Wireless Sensor and Ad Hoc Networks, Vol. 7, NO. 8, p1551~p1562, 2009.

[5] - Toby Xu, Ying Cai, "Location Cloaking for Safety Protection of Ad Hoc Networks". INFOCOM'09, April 19-25, 2009.

[6] - B. Gedik, L. Liu "A customizable k-Anonimity Model for Protecting Location Privacy". In ICDCS'05, pages 620-629.

[7] - C. Bettini, X.S. Wang, S. Jajodia "Protecting Privacy Against Location-Based Personal Indentification". In proc VLDB Workshop on Privacy Enhancing Technologies, pages 393-412, 2006.

[8] - A. Inan, Y. Saygin "Location Anonymity in Horizontally Partitioned Spatial-Temporal Data". In Master thesis, Sabanci University, Turquia, 2006.

[9] - P. Kalnis, G. Ghinita, K Mouratidis, D. Papadias "Preserving Anonymity in Location Based Services", Reporte Tecnico TRB/06, Departamento de Ciencas de la Computacion, Universidad Nacional de Singapur, 2006.

[10] - C.Y. Chow, M. Mokble, X. Xiu "A Peer-to-Peer Spatial Cloaking Algorithm for Anonymous Location-based Sevices ". In ACM GIS, November 2006.

[11] - G. Ghinita, P. Kalnis, and S. Skiadopoulos. " MobiHide: A Mobile Peer-to-Peer System for Anonymous Location-Based Queries". In SSTD'07, pages 221–238, 2007.

[12] - G. Ghinita, P. Kalnis, and S. Skiadopoulos. "PRIVE: Anonymous Location-based Queries in Distributed Mobile Systems." In Proc. of the 16th international conference on World Wide Web, pages 371–380, Banff, Alberta, Canada, 2007.

[13] - C. Chow and M. F. Mokbel. "Enabling Private Continuous Queries for Revealed User Locations". In SSTD'07, pages 258–275, 2007. http://www.ietf.org/internet-drafts/draftietf-geopriv-reqs-01.txt, 2002.

[14] - H. Hu and J. Xu. "Non-Exposure Location Anonymity". In ICDE'09, pages 1120–1131, 2009.

[15] - Rajchandar Padmanaban, "Location Privacy in Location Based Services: Unsolved Problem and Challenge ", International Journal of Advanced Remote Sensing and GIS 2013, Volume 2, Issue 1, pp. 398-404, Article ID Tech-171 ISSN 2320 - 0243.

[16] - Chow C-Y, Mokbel MF, Bao J, Liu X. "Query-aware location anonymization for road networks". Geoinformatica. 2011;15(3):571–607.

[17] - Gabriel Ghinita, Panos Kalnis, Ali Khoshgozaran, Cyrus Shahabi, Kian-Lee Tan. "Private Queries in Location Based Services: Anonymizers are not Necessary". SIGMOD'08, June 9–12, 2008, Vancouver, BC, Canada.

[18] - Ben Niu, Qinghua Li, Xiaoyan Zhu, Guohong Cao and Hui Li. "Achieving k-anonymity in Privacy-Aware Location-Based Services". IEEE INFOCOM 2014 - IEEE Conference on Computer Communications.

[19] – Tobi Xu, Ying Cai, "Exploring Historical Location Data for Anonymity Preservation in Location-based Services". IEEE INFOCOM 2008 proceedings.

10 ANEXO: CÓDIGO FUENTE SIMULADOR GRÁFICO

Clase CuadroGrilla:

```
package com.grafocaminos.estructura;

import java.awt.BasicStroke;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Point;
import java.awt.Shape;

//Clase que define y dibuja la grilla del mapa
public class CuadroGrilla implements Comparable<CuadroGrilla>{
	private Double pq;
	private Double pc;
	private Shape figura;
	private int id;
	private Color color;

	public CuadroGrilla(Shape figura, Double pq, int id){
	  this.pq = pq;
	  this.figura = figura;
	  this.id = id;
	  pc = 0.0;
	  color = Color.GRAY;
	}

	public void paint(Graphics2D gs){
	  Shape fig = getCuadro();

	  if(fig!=null){
			gs.setColor(getColor());
			gs.draw(fig);
			gs.drawString(this.getProbabilidad().toString(),	(int)fig.getBounds2D().getCenterX(),
(int)fig.getBounds2D().getY()-10);
			if(this.getColor().equals(Color.MAGENTA)){
				gs.fillOval((int)this.getCuadro().getBounds2D().getCenterX(),
(int)this.getCuadro().getBounds2D().getCenterY(), 20, 20);
			}
	  }
	}

	public Double getProbabilidad(){
	  return pq;
	}


public void setProbabilidad(Double pq){
```

```
        this.pq = pq;
    }

    public Shape getCuadro(){
        return figura;
    }

    public void setCuadro(Shape figura){
        this.figura = figura;
    }

    public void setId(int id){
        this.id = id;
    }

    public int getId(){
        return id;
    }

    public void setColor(Color color){
        this.color = color;
    }

    public Color getColor(){
        return color;
    }

    public Double getPc(){
        return pc;
    }

    public void setPc(double pc){
        this.pc = pc;
    }

    @Override
    public int compareTo(CuadroGrilla o) {
        // TODO Auto-generated method stub
        if(this.getProbabilidad()< o.getProbabilidad()) return -1;
        else if(this.getProbabilidad()> o.getProbabilidad()) return 1;
        else return 0;
    }
}
```

Clase Grafo:

```
package com.grafocaminos.estructura;

import java.util.ArrayList;
import java.util.Hashtable;

//Clase que define al grafo y las rutas que posee
public class Grafo {
  private ArrayList<String> nombreNodos;
  private ArrayList<Posicion> posiciones;
  private ArrayList<Segmentos> aristas;
  private Hashtable<String, Nodo> nodos_totales;

  public Grafo() {
        nombreNodos = new ArrayList<String>();
        nodos_totales = new Hashtable<String, Nodo>();
        aristas = new ArrayList<Segmentos>();
        posiciones = new ArrayList<Posicion>();
  }

  public void agregarNodo(String nombre, Posicion pos){
        nombreNodos.add(nombre);
        posiciones.add(pos);
        nodos_totales.put(nombre, new Nodo(nombre, pos));
  }

  //Realizar analisis EXHAUSTIVO
  public void agregarEnlace(Nodo nodoInicial, Nodo nodoFinal, String nombreSeg){
        Segmentos seg = new Segmentos(nodoInicial, nodoFinal, nombreSeg);

        aristas.add(seg);

        nodos_totales.get(nodoInicial).agregarEnlace(nodoInicial, nodoFinal, nombreSeg);
        nodos_totales.get(nodoFinal).agregarEnlace(nodoFinal, nodoInicial, nombreSeg);
  }

  public void setNodos(Hashtable<String,Nodo> nodos){
        nodos_totales = nodos;
  }

  public Hashtable<String,Nodo> getNodos(){
        return nodos_totales;
  }

  public void setNombre(ArrayList<String> nombre){
        nombreNodos = nombre;
  }


public ArrayList<String> getNombre(){
```

```
        return nombreNodos;
    }

    public void setSegmentos(Segmentos arista){
        aristas.add(arista);
    }

    public ArrayList<Segmentos> getSegmentos(){
        return aristas;
    }
}
```

Clase Nodo:

```
package com.grafocaminos.estructura;

import java.util.ArrayList;

//Clase que define un nodo o esquina y sus caracteristicas
public class Nodo {
    private String id;
    private int cantEnlaces;
    private ArrayList<Segmentos> enlaces = new ArrayList<Segmentos>();
    private Posicion pos;

    public Nodo(String newId, Posicion pos){
        id = newId;
        this.pos = pos;
        cantEnlaces = -1;
    }

    public void crearNodo(Posicion pos, Segmentos segmento){
        this.pos = pos;
        enlaces.add(segmento);
        cantEnlaces = -1;
    }

    public Posicion getPosicion(){
        return pos;
    }

    public String getId(){
        return id;
    }

    public ArrayList<Segmentos> getEnlaces() {
        return enlaces;
    }

    public void agregarEnlace(Nodo nodoInicial, Nodo nodoFinal, String nombreSeg){
```

```
            if(cantEnlaces == -1)
            {
                    enlaces.add(new Segmentos(nodoInicial, nodoFinal, nombreSeg));
                    cantEnlaces++;
            }

            else
            {
                    int posicionEnlace;
                    posicionEnlace = existeEnlace(nombreSeg);
                    if(posicionEnlace == -1)
                    {
                            enlaces.add(new Segmentos(nodoInicial, nodoFinal, nombreSeg));
                            cantEnlaces++;
                    }
            }
    }

    public int existeEnlace(String enlazar) {
            for(int i=0; i< enlaces.size(); i++)
            {
                    Segmentos esEnlace;
                    esEnlace = enlaces.get(i);
                    if(esEnlace.getId().equals(enlazar))
                    {
                            return i;
                    }
            }
            return -1;
    }

}
```

Clase Persona:

```
package com.grafocaminos.estructura;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.Graphics2D;
import java.awt.Image;
import java.awt.Point;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

import javax.imageio.ImageIO;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

public class Persona {
  private Color color;
  private Dimension dim;
  private String idPersona;
  private Point velocidad;
  private Posicion pos_persona;
  private BufferedImage imagen;

  public void paint(Graphics2D gc){
        Posicion p = getPosicion();

        if(p!= null){
              gc.setColor(getColor());
              Dimension dim = getSize();
              gc.setFont(new Font("Calibri", Font.BOLD, 16));
              gc.drawString(idPersona, p.getPosicionX(), p.getPosicionY());

              gc.drawImage(imagen, p.getPosicionX(), p.getPosicionY(), 20, 20, null);
        }
  }

  public Persona(Color color, String idPersona){
        setColor(color);

        //pos_persona = new Posicion(50,160);
        this.idPersona = idPersona;
        velocidad = new Point(10 - (int)Math.random()*20,10 - (int)Math.random()*20);
        dim = new Dimension(20,20);

        try {
              imagen = ImageIO.read(getClass().getResourceAsStream("/usuario.png"));
        } catch (IOException e) {
              // TODO Auto-generated catch block
```

```
                    e.printStackTrace();
                }
    }

    public Dimension getSize(){
                return dim;
    }

    public void setColor(Color color){
                this.color = color;
    }

    public Color getColor(){
                return color;
    }

    public void setVelocidad(Point velocidad){
                this.velocidad = velocidad;
    }

    public Point getVelocidad(){
                return velocidad;
    }

    public void setPosicion(Posicion pos_persona){
                this.pos_persona = pos_persona;
    }

    public Posicion getPosicion(){
                return pos_persona;
    }

    public void setId(String idPersona){
                this.idPersona = idPersona;
    }

    public String getId(){
                return idPersona;
    }

}
```

Clase Posicion

```
package com.grafocaminos.estructura;

//Clase que define la posicion de un punto
public class Posicion {
  private int x;
  private int y;

  public Posicion(int x, int y) {
        this.x = x;
        this.y = y;
  }

  public int getPosicionX(){
        return x;
  }

  public int getPosicionY(){
        return y;
  }

}
```

Clase Segmento

```
package com.grafocaminos.estructura;

import java.awt.Color;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Shape;
import java.util.ArrayList;

//Clase que define a un segmento y sus caracteristicas
public class Segmentos {
  private String idSeg;
  private String nombreSeg;
  private Nodo extremoInicial;
  private Nodo extremoFinal;
  private ArrayList<Vehiculo> cantidad;
  private Color color;

  public Segmentos(Nodo nodoInicial, Nodo nodoFinal, String idSeg){
        extremoInicial = nodoInicial;
        extremoFinal = nodoFinal;
        cantidad = new ArrayList<Vehiculo>();
        this.idSeg = idSeg;
        nombreSeg = "nd";
        color = Color.BLACK;
  }
```

```
  public void paint(Graphics2D gp){
            Nodo ni = getInicial();
            Nodo nf = getFinal();

            if(ni!=null && nf!=null){
                  gp.setColor(getColor());
                  gp.drawLine(ni.getPosicion().getPosicionX(),    ni.getPosicion().getPosicionY(),
nf.getPosicion().getPosicionX(), nf.getPosicion().getPosicionY());

            }
            if(nombreSeg.equals("Avda. Prat")){
                  gp.setColor(Color.BLACK);
                  gp.drawString(nombreSeg,                ni.getPosicion().getPosicionX()+500,
ni.getPosicion().getPosicionY()-20);
            }
            else if(nombreSeg.equals("Paicavi")){
                  gp.setColor(Color.BLACK);
                  gp.drawString(nombreSeg,                ni.getPosicion().getPosicionX()+500,
ni.getPosicion().getPosicionY()+30);
            }
  }

  public String getId(){
            return idSeg;
  }

  public void setInicial(Nodo extremoInicial) {
            this.extremoInicial = extremoInicial;
  }

  public Nodo getInicial() {
            return extremoInicial;
  }

  public void setFinal(Nodo extremoFinal) {
            this.extremoFinal = extremoFinal;
  }

  public Nodo getFinal() {
            return extremoFinal;
  }

  public void agregarVehiculo(Vehiculo vehiculo){
            cantidad.add(vehiculo);
  }


public ArrayList<Vehiculo> getVehiculo(){
            return cantidad;
  }
```

```
  public void setNombre(String nombreSeg){
        this.nombreSeg = nombreSeg;
  }

  public String getNombre(){
        return nombreSeg;
  }

  public Color getColor(){
        return color;
  }

  public void setColor(Color color){
        this.color = color;
  }

}
```

Clase Vehiculo:

```
package com.grafocaminos.estructura;

import java.awt.BasicStroke;
import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics2D;
import java.awt.image.AffineTransformOp;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

import javax.imageio.ImageIO;
import javax.swing.JOptionPane;

//Clase que define a un usuario de tipo vehiculo y sus caracteristicas
public class Vehiculo {
  private String idVehiculo;
  private int velocidad;
  private Posicion pos_vehiculo;
  private Segmentos segmento;
  private AffineTransformOp ato;
  private Color color;
  private BufferedImage imagen;

  public Vehiculo(String idVehiculo){
        color = Color.ORANGE;
        this.idVehiculo = idVehiculo;
        try {
              imagen                                                    =
ImageIO.read(this.getClass().getResourceAsStream("/vehiculo2.png"));
```

```
            } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
            }
            ato = null;
    }

    public void paint(Graphics2D gi){
            Posicion po = getPosicion();

            if(po!=null && ato==null){
                    gi.setColor(color);
                    gi.setFont(new Font("Calibri", Font.BOLD, 16));
                    gi.drawString(idVehiculo,                    pos_vehiculo.getPosicionX()-20,
pos_vehiculo.getPosicionY());

                    //gi.fillRect(pos_vehiculo.getPosicionX(), pos_vehiculo.getPosicionY(), 30, 20);
                    gi.drawImage(imagen, po.getPosicionX(), po.getPosicionY(), 30, 20, null);
            }
            if(ato!=null){
                    gi.setColor(color);
                    gi.drawString(idVehiculo,                    pos_vehiculo.getPosicionX()-20,
pos_vehiculo.getPosicionY());
                    gi.drawImage(ato.filter(imagen, null), po.getPosicionX(), po.getPosicionY(), 20,
30, null);
            }
    }

    public void setSegmento(Segmentos segmento){
            this.segmento = segmento;
    }

    public Segmentos getSegmento(){
            return segmento;
    }

    public void setVelocidad(int velocidad){
            this.velocidad = velocidad;
    }

    public int getVelocidad(){
            return velocidad;
    }


public void setPosicion(Posicion pos_vehiculo){
            this.pos_vehiculo = pos_vehiculo;
    }

    public Posicion getPosicion(){
            return pos_vehiculo;
```

```
    }

    public void setId(String idVehiculo){
            this.idVehiculo = idVehiculo;
    }

    public String getId(){
            return idVehiculo;
    }

    public void setImagen(BufferedImage imagen){
            this.imagen = imagen;
    }

    public BufferedImage getImagen(){
            return imagen;
    }

    public void setAt(AffineTransformOp ato){
            this.ato = ato;
    }

    public AffineTransformOp getAt(){
            return ato;
    }

    public void setColor(Color color){
            this.color = color;
    }

    public Color getColor(Color color){
            return color;
    }
}
```

Clase PintarGrafo:

```
package com.grafocaminos.estructura;

import java.awt.BasicStroke;
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Component;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.ItemSelectable;
import java.awt.RenderingHints;
import java.awt.Shape;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.awt.geom.AffineTransform;
import java.awt.geom.Ellipse2D;
import java.awt.geom.Rectangle2D;
import java.awt.image.AffineTransformOp;
import java.text.DecimalFormat;
import java.text.DecimalFormatSymbols;
import java.text.NumberFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.Hashtable;
import java.util.List;
import java.util.Locale;
import java.util.Random;

import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.ListCellRenderer;
import javax.swing.Popup;
import javax.swing.Timer;

//Clase encargada de pintar lo relacionado al grafo y realizar las funciones principales
public class PintarGrafo extends JPanel{
  /**
   *
   */
  private static final long serialVersionUID = 1L;
  private ArrayList<CuadroGrilla> grilla;
```

```
  private ArrayList<ArrayList<CuadroGrilla>> set;
  private Grafo grafo;
  private int[] x1, y1, x2, y2, di, delta2, deltax, deltay;
  private int pausa = 50, xg = 25, yg = 60, width = 180, y3 = 0, r, index, dx=0, dy=0, ig, um, vx, vy,
vk, rb = 1, rn = 1;
  private double d1 = 0, d2 =0, d3;
  private double[] du, pu, pf;
  private String us, ni, nk, nam;
  Shape figura;
  private Random ran;
  private List<Persona> usuarios;
  private List<Vehiculo> vehiculos;
  private List<CuadroGrilla> cuadros;
  private Hashtable<Double, Integer> distancia_vehiculos;
  private Hashtable<Double, Integer> ug;

  public PintarGrafo(){
        setPreferredSize(new Dimension(500, 400));

  }

  public PintarGrafo(Grafo grafo){
        setPreferredSize(new Dimension(500, 400));
        setLayout(null);
        setBackground(Color.GRAY);

        usuarios = new ArrayList<Persona>();
        vehiculos = new ArrayList<Vehiculo>();
        cuadros = new ArrayList<CuadroGrilla>();
        grilla = new ArrayList<CuadroGrilla>();
        distancia_vehiculos = new Hashtable<Double, Integer>();
        ug = new Hashtable<Double, Integer>();
        ran = new Random();
        x1 = new int[65];
        y1 = new int[65];
        x2 = new int[8];
        y2 = new int[8];
        di = new int[8];
        delta2 = new int[8];
        pu = new double[42];
        this.grafo = grafo;

        //Se inicializa y carga la GRILLA
        cargarGrilla();

        int high = cuadros.size() - 1;

        burbuja(high);
        //Arreglos que contienen datos relacionados a las PERSONAS
        um = 10 + ran.nextInt(40);
        deltax = new int[um];
```

```
        deltay = new int[um];
        pf = new double[um];

        //Se generan los valores iniciales asociados a posiciones y velocidades de PERSONAS y
VEHICULOS
        valoresIniciales();

        //Arreglo que contendrá información de los VEHICULOS
        du = new double[vehiculos.size()];

        //Se define y ejecuta el proceso de movimiento de usuarios controlado por un Timer
        Timer timer = new Timer(pausa, new ActionListener(){
                @Override
                public void actionPerformed(ActionEvent e) {
                        //Se realiza el movimiento de cada usuario
                        moverUsuarios();
                }

        });
        timer.start();

        //Se definen los Botones, ComboBox y otros componentes graficos
        JLabel label1 = new JLabel();
        JButton botonp = new JButton();
        JButton botonr = new JButton();
        JButton botonh = new JButton("Grilla ON/OFF");
        JComboBox<String> cbt = new JComboBox<String>();
        JComboBox<String> cbu = new JComboBox<String>();
        JComboBox<String> cbk = new JComboBox<String>();

        cbt.addItem("Gruteser");
        cbt.addItem("DLS");
        cbt.addItem("Enhanced DLS");
        cbt.addItem("Road Networks");

        cbt.setBounds(620, 650, 120, 30);
        cbt.setSelectedIndex(0);

        cbk.setBounds(760, 650, 150, 30);

        cbu.setBounds(100, 650, 150, 30);
        for(index=0; index <usuarios.size() + 8; index++){
                cbu.addItem(String.valueOf(index));
                if(index < 10){
                        cbk.addItem(String.valueOf(index+1));
                }
        }

        //Se define para cada ComboBox la descripción de este, visible cuando no hay una
opción seleccionada
        cbu.setRenderer(new nuevoComboBoxRenderer("Seleccione usuario"));
```

```
        cbu.setSelectedIndex(-1);
        cbk.setRenderer(new nuevoComboBoxRenderer("Ingrese k"));
        cbk.setSelectedIndex(-1);
        cbt.setRenderer(new nuevoComboBoxRenderer("Seleccione tecnica"));
        cbt.setSelectedIndex(-1);

        label1.setBounds(100, 650, 150, 30);
        botonp.setText("Pausar/Reanudar");
        botonp.setBounds(260, 650, 150, 30);
        botonr.setText("Generar region");
        botonr.setBounds(420, 650, 150, 30);
        botonr.setEnabled(false);
        botonh.setBounds(920, 650, 150, 30);

        //Se definen los métodos que escuchan las selecciones del usuario para la aplicación

        //Método que escucha la selección de usuarios, y acciones relacionadas
        cbu.addItemListener(new ItemListener(){
                public void itemStateChanged(ItemEvent itemEvent){
                        int estado1 = itemEvent.getStateChange();
                        if(estado1 == ItemEvent.SELECTED){
                                us = (String) itemEvent.getItem();
                                for(index=0; index<vehiculos.size(); index++){
                                        if(vehiculos.get(index).getId().equals(us)){
                                                vehiculos.get(index).setColor(Color.RED);
                                        }
                                        else{

  vehiculos.get(index).setColor(Color.ORANGE);
                                        }
                                }
                                System.out.println("Ha seleccionado al usuario: "+ us);
                                for(index = 0; index<usuarios.size(); index++){
                                        if(usuarios.get(index).getId().equals(us)){
                                                System.out.println("Se pintara al usuario: " +
us);
                                                usuarios.get(index).setColor(Color.RED);
                                        }
                                        else{

  usuarios.get(index).setColor(Color.YELLOW);
                                        }
                                        if(index < grafo.getSegmentos().size()){

  grafo.getSegmentos().get(index).setColor(Color.BLACK);
                                        }
                                }

                                }
```

```
                        if(cbu.getSelectedIndex()!=-1 && cbk.getSelectedIndex()!=-1 &&
cbt.getSelectedIndex()!=-1){
                                botonr.setEnabled(true);
                        }
                        }
                });

        //Método que escucha la selección del número "k" y acciones relacionadas
        cbk.addItemListener(new ItemListener(){
                public void itemStateChanged(ItemEvent itemEvent){
                        int estado3 = itemEvent.getStateChange();
                        if(estado3 == ItemEvent.SELECTED){
                                nk = (String) itemEvent.getItem();
                        }

                        if(cbu.getSelectedIndex()!=-1 && cbk.getSelectedIndex()!=-1 &&
cbt.getSelectedIndex()!=-1){
                                botonr.setEnabled(true);
                        }
                }
        });

        //Método que escucha la selección de la técnica y acciones relacionadas
        cbt.addItemListener(new ItemListener(){
                public void itemStateChanged(ItemEvent itemEvent){
                        int estado2 = itemEvent.getStateChange();
                        if(estado2 == ItemEvent.SELECTED){
                                System.out.println("Ha seleccionado " + estado2 + " Item: " +
itemEvent.getItem());
                                ni = (String) itemEvent.getItem();
                                System.out.println(ni);
                        }
                        ItemSelectable is = itemEvent.getItemSelectable();

                        if(cbu.getSelectedIndex()!=-1 && cbk.getSelectedIndex()!=-1 &&
cbt.getSelectedIndex()!=-1){
                                botonr.setEnabled(true);
                        }
                }
        });

        //Método que escucha la generación de la región con los datos seleccionados por el
usuario
        botonr.addActionListener(new ActionListener(){
                public void actionPerformed(ActionEvent e){
                        if(ni == "DLS" && Integer.valueOf(us)< usuarios.size()){
                                if(Integer.valueOf(us)<usuarios.size()){
                                        for(index=0; index<cuadros.size(); index++){

  if(cuadros.get(index).getCuadro().contains((double)usuarios.get(Integer.valueOf(us)).getPosicion
().getPosicionX(), (double)usuarios.get(Integer.valueOf(us)).getPosicion().getPosicionY())){
```

```
                                                ig = cuadros.get(index).getId();
                                                //System.out.println(ig);
                                                }
                                        }
                                }

                        System.out.println("La grilla seleccionada es: " + ig);

                        for(r=0; r<cuadros.size(); r++){
                                if(cuadros.get(r).getId() == ig){
                                        System.out.println("La grilla que contiene al usuario
es: " + cuadros.get(r).getId());
                                        ig = r;
                                        break;
                                }
                        }

                        System.out.println("La grilla se encuentra en el indice: " +ig);
                        grilla.clear();
                        rb=1;
                        rn=1;

                        encontrarCandidatos(2*(Integer.valueOf(nk)));

                        generarCandidato(2*(Integer.valueOf(nk)));

                        for(r=0; r<Integer.valueOf(nk); r++){
                                set.get(0).get(r).setColor(Color.MAGENTA);
                                System.out.println("El set seleccionado contiene la grilla: " +
set.get(0).get(r).getId());
                        }

                        System.out.println("Se ejecuto DLS");

                        repaint();

                        }
                        else if(ni == "DLS" && Integer.valueOf(us) >= usuarios.size()){
                                JOptionPane.showMessageDialog(null, "Debe seleccionar un
usuario de tipo 'persona' para ejecutar DLS.");
                        }
                        else if(ni == "Enhanced DLS" && Integer.valueOf(us)<
usuarios.size()){
                                if(Integer.valueOf(us)<usuarios.size()){
                                        for(index=0; index<cuadros.size(); index++){

  if(cuadros.get(index).getCuadro().contains((double)usuarios.get(Integer.valueOf(us)).getPosicion
().getPosicionX(), (double)usuarios.get(Integer.valueOf(us)).getPosicion().getPosicionY())){
                                                ig = cuadros.get(index).getId();
                                                //System.out.println(ig);
                                                }
```

```
                                        }
                                }

                                for(r=0; r<cuadros.size(); r++){
                                        if(cuadros.get(r).getId() == ig){
                                                //System.out.println("La grilla que contiene al
usuario es: " + cuadros.get(r).getId());
                                                ig = r;
                                                break;
                                        }
                                }

                                grilla.clear();

                                encontrarCandidatos(4*(Integer.valueOf(nk)));
                                generarCandidato(4*(Integer.valueOf(nk)));

                                //EN CONSTRUCCION
                                List<CuadroGrilla> setC = new ArrayList<CuadroGrilla>();
                                System.out.println("La grilla seleccionada corresponde a: "
+cuadros.get(ig).getId());
                                setC.add(cuadros.get(ig));

                                System.out.println("El tamaño del set de candidatos es: "
+set.get(0).size());

                                int f = 0, ea = 0, is = 0, nr;
                                double d2 = 1, den = 0, pt = 0, d3, tf =0;
                                DecimalFormat df = new DecimalFormat("#.###",
DecimalFormatSymbols.getInstance(Locale.US));
                                List<CuadroGrilla> probUniformes = new
ArrayList<CuadroGrilla>();

                                d1 = 1;

                                //Se calcula el set de candidatos que formaran la region de
encubrimiento
                                for(f = 1; f<Integer.valueOf(nk); f++){
                                        den = 0;
                                        pt = 0;
                                        d2 = 1;
                                        tf = 0;
                                        ea = 0;
                                        probUniformes.clear();
                                        for(index=0; index<set.get(0).size(); index++){
                                                d2=1;
                                                for(r=0; r<setC.size(); r++){
                                                        //System.out.println("El elemento del
set candidato es: " +set.get(0).get(index).getCuadro().getBounds2D().getCenterX()+ " y el
elemento del set seleccionado es: " +setC.get(r).getCuadro().getBounds2D().getCenterX());
```

```
                                                d2 *=
Math.sqrt((double)Math.pow((set.get(0).get(index).getCuadro().getBounds2D().getCenterX() -
setC.get(r).getCuadro().getBounds2D().getCenterX()), 2) +
Math.pow((set.get(0).get(index).getCuadro().getBounds2D().getCenterY() -
setC.get(r).getCuadro().getBounds2D().getCenterY()), 2));
                                                //System.out.println("El valor del
numerador es: " +d2);
                                        }
                                        set.get(0).get(index).setPc(d2);
                                        den += d2;
                                }

                                for(index=0; index<set.get(0).size(); index++){
                                        pt =
Double.parseDouble(df.format((set.get(0).get(index).getPc()/den)));
                                        set.get(0).get(index).setPc(pt);
                                        tf+=pt;

                                        if(index==(set.get(0).size()-1)){
                                                //System.out.println("El elemento "
+set.get(0).get(index).getId()+ " tiene su probabilidad ajustada");
                                                set.get(0).get(set.get(0).size()-
1).setPc(set.get(0).get(set.get(0).size()-1).getPc() + (1-tf));
                                        }

                                        //System.out.println("La probabilidad del set
es: " +set.get(0).get(index).getPc());

                                        for(r=ea; r<(ea+(int)(pt*1000)); r++){

  probUniformes.add(set.get(0).get(index));
                                                //System.out.println("La celda "
+probUniformes.get(r).getId()+ " ha sido agregada "+probUniformes.get(r).getPc()*1000+ "
cantidad de veces");
                                        }
                                        ea+=(int)(pt*1000);
                                        //System.out.println("La probabilidad del
elemento del set es: " + pt);
                                }

                                //System.out.println("El primer elemento del set de
probabilidades corresponde a: "+probUniformes.get(0).getId());

                                nr = ran.nextInt(1000);
                                for(r=0; r<set.get(0).size(); r++){

  if(set.get(0).get(r).equals(probUniformes.get(nr))){
                                                is = r;
                                        }
                                        /*else{
```

```
                                                System.out.println("Hubo un error, el
contenido no encontrado corresponde a: " +set.get(0).get(r).getId());
                                                }*/
                                        }



                                        //System.out.println("La probabilidad total del set es: "
+ tf);
                                        setC.add(set.get(0).get(is));
                                        set.get(0).remove(is);

                                        if(f==1){
                                                System.out.println("El set seleccionado
contiene las celdas: " +setC.get(f-1).getId());
                                                setC.get(f-1).setColor(Color.MAGENTA);
                                        }
                                        if(f>=1){
                                                System.out.println("El set seleccionado
contiene las celdas: " +setC.get(f).getId());
                                                setC.get(f).setColor(Color.MAGENTA);
                                        }
                                }

                                System.out.println("Se ejecuto Enhanced DLS");
                                repaint();
                        }
                        else if(ni == "Enhanced DLS" && Integer.valueOf(us) >=
usuarios.size()){
                                JOptionPane.showMessageDialog(null, "Debe elegir un
usuario de tipo persona para simular enhanced DLS");
                        }
                        else if(ni == "Road Networks" && Integer.valueOf(us) >=
usuarios.size()){
                                for(index = 0; index<8; index++){
                                if(vehiculos.get(index).getId().equals(us)){
                                        vx                                              =
vehiculos.get(index).getPosicion().getPosicionX();
                                        vy                                              =
vehiculos.get(index).getPosicion().getPosicionY();
                                        System.out.println("Usted seleccionó al vehiculo: " +
grafo.getSegmentos().get(index).getVehiculo().get(0).getId());
                                }
                        }
                                for(index = 0; index<8; index++){
                                        d1                                              =
Math.sqrt((double)Math.pow((vehiculos.get(index).getPosicion().getPosicionY() - vy), 2) +
Math.pow((vehiculos.get(index).getPosicion().getPosicionX() - vx), 2));
                                        du[index] = d1;
                                        distancia_vehiculos.put(d1, um+index);
                                }
```

```
                                Arrays.sort(du);

                                vk = 0;
                                for(r=0; r<8; r++){
                                        //System.out.println(distancia_vehiculos.get(du[r]));
                                        for(index=0; index<8; index++){

  System.out.println(grafo.getSegmentos().get(index).getVehiculo().get(0).getId());

  System.out.println(distancia_vehiculos.get(du[r]));
                                                if(vk < Integer.valueOf(nk) &&
grafo.getSegmentos().get(index).getVehiculo().get(0).getId().equals(String.valueOf(distancia_vehi
culos.get(du[r])))){

  grafo.getSegmentos().get(index).setColor(Color.GREEN);
                                                        vk++;
                                                        System.out.println(vk);
                                                }
                                        }

                                        System.out.println("Se ejecuto Road Networks");
                                        repaint();
                                }
                                //JOptionPane.showMessageDialog(null, "La tecnica Road
Networks esta aun en desarrollo");
                        }
                        else if(ni == "Road Networks" && Integer.valueOf(us)<
usuarios.size()){
                                JOptionPane.showMessageDialog(null, "Debe seleccionar un
usuario de tipo vehiculo para ejecutar Road Networks");
                        }
                        else if(ni == "Gruteser" && Integer.valueOf(us)< usuarios.size()){
                                for(index=0; index<usuarios.size(); index++){
                                        d3                                              =
Math.sqrt((double)Math.pow((usuarios.get(index).getPosicion().getPosicionY()           -
usuarios.get(Integer.valueOf(us)).getPosicion().getPosicionY()),            2)          +
Math.pow((usuarios.get(index).getPosicion().getPosicionX()                              -
usuarios.get(Integer.valueOf(us)).getPosicion().getPosicionX()), 2));
                                        pf[index] = d3;
                                        ug.put(pf[index],
Integer.valueOf(usuarios.get(index).getId()));
                                }

                                Arrays.sort(pf);

                                for(index=1; index<Integer.valueOf(nk); index++){

  usuarios.get(ug.get(pf[index])).setColor(Color.GREEN);
                                }

                                r=0;
```

```
                                        while(r<21){
                                        for(index=0; index<usuarios.size(); index++){

  if(cuadros.get(r).getCuadro().contains((double)usuarios.get(index).getPosicion().getPosicionX(),
(double)usuarios.get(index).getPosicion().getPosicionY())){

                                                        System.out.println("La grilla: " + r + "
contiene al usuario: " + usuarios.get(index).getId());

  //System.out.println(ug.get(Integer.valueOf(usuarios.get(index).getId())));
                                                }

                                        }
                                        r++;
                                        }
                                        r=0;

                                        System.out.println("Se ejecuto Gruteser");
                                        ug.clear();
                                        repaint();
                                }
                                else if(ni == "Gruteser" && Integer.valueOf(us)>= usuarios.size()){
                                        JOptionPane.showMessageDialog(null, "Debe seleccionar un
usuario de tipo 'persona' para ejecutar Gruteser.");
                                }

                                r=0;
                        }
                });

                //Método que escucha si el usuario desea pausar o reanudar el movimiento
                botonp.addActionListener(new ActionListener(){
                        public void actionPerformed(ActionEvent e){
                                if(timer.isRunning()){
                                        timer.stop();
                                }
                                else{
                                        timer.start();
                                }
                        }
                });

                //Método que se encarga de visualizar la grilla al usuario u ocultarla
                botonh.addActionListener(new ActionListener(){
                        public void actionPerformed(ActionEvent e){
                                int nc = 0;
                                for(r=0; r<cuadros.size(); r++){
                                        if(cuadros.get(r).getColor().equals(Color.WHITE)){
                                                cuadros.get(r).setColor(Color.GRAY);
                                                nc++;
                                        }
```

```
                                else if(cuadros.get(r).getColor().equals(Color.MAGENTA)){
                                        cuadros.get(r).setColor(Color.WHITE);

                                }
                                else if(cuadros.get(r).getColor().equals(Color.GRAY)){
                                        cuadros.get(r).setColor(Color.WHITE);
                                }
                        }
                        if(nc<cuadros.size() && nc!=0){
                                for(r=0; r<cuadros.size(); r++){
                                        cuadros.get(r).setColor(Color.GRAY);
                                }
                        }
                }
        });

        this.add(botonp);
        this.add(botonr);
        this.add(botonh);
        this.add(cbt);
        this.add(cbu);
        this.add(cbk);
}

@Override
public void paintComponent(Graphics g){
        super.paintComponent(g);
        Graphics2D ga = (Graphics2D) g;

        //Se define tamaño de pincel y color para dibujar la GRILLA
        ga.setStroke(new BasicStroke(5.0f));
        ga.setColor(Color.WHITE);

        if(dx!=0 && dy!=0){
                ga.drawOval(dx, dy, 30, 30);
        }
        //Se dibuja la GRILLA
        if(y3 == 0){
                for(r=0; r<cuadros.size(); r++){
                        cuadros.get(r).paint(ga);
                }
        }
        //Se define el tamaño del pincel para dibujar el GRAFO
        ga.setStroke(new BasicStroke(8.0f));
        AffineTransform original = ga.getTransform();
        ga.setFont(new Font("Calibri", Font.BOLD, 20));
        //Se dibuja el GRAFO
        for(int i=0; i<grafo.getSegmentos().size(); i++){

//System.out.println(grafo.getSegmentos().get(i).getFinal().getPosicion().getPosicionY());
                ga.setColor(Color.BLACK);
```

```
                if(i==0){
                        nam = "Avda. Prat";
                        grafo.getSegmentos().get(i).setNombre(nam);
                }
                else if(i==1){
                        nam = "Chacabuco";
                        ga.rotate(Math.toRadians(90));
                        ga.drawString(nam, 260, -50);
                }
                else if(i==2){
                        nam = "San Martin";
                        ga.rotate(Math.toRadians(90));
                        ga.drawString(nam, 260, -250);
                }
                else if(i==3){
                        nam = "Paicavi";
                        grafo.getSegmentos().get(i).setNombre(nam);
                }
                else if(i==4){
                        nam = "O'Higgins";
                        ga.rotate(Math.toRadians(90));
                        ga.drawString(nam, 260, -490);
                }
                else if(i==5){
                        nam = "Freire";
                        ga.rotate(Math.toRadians(90));
                        ga.drawString(nam, 260, -770);
                }
                else if(i==6){
                        nam = "Maipu";
                        ga.rotate(Math.toRadians(90));
                        ga.drawString(nam, 260, -970);
                }
                else if(i==7){
                        nam = "Los Carrera";
                        ga.rotate(Math.toRadians(90));
                        ga.drawString(nam, 260, -1210);
                }
                ga.setTransform(original);
                grafo.getSegmentos().get(i).paint(ga);
        }

        ga.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);
        ga.setStroke(new BasicStroke(5.0f));

        //ga.drawString(grafo.getSegmentos().get(0).getInicial().getId(), x3, y3);
        ga.setColor(Color.YELLOW);
        for(int j=0; j<usuarios.size(); j++){
                //ga.fillOval(x1[j], y1[j]-10, 20, 20);
```

```
                usuarios.get(j).paint(ga);
        }

        for(index=0; index<8; index++){
                ga.setColor(Color.ORANGE);
                if(di[index] == 0){
                        vehiculos.get(index).paint(ga);
                }
                else{
                        AffineTransform at =
AffineTransform.getRotateInstance(Math.toRadians(90),
vehiculos.get(index).getImagen().getWidth()/3, vehiculos.get(index).getImagen().getHeight()/3);
                        AffineTransformOp nf = new AffineTransformOp(at,
AffineTransformOp.TYPE_BILINEAR);
                        vehiculos.get(index).setAt(nf);
                        vehiculos.get(index).paint(ga);
                }
                ga.setTransform(original);
        }
        ga.setColor(Color.ORANGE);
  }

  @Override
  public void paint(Graphics g){
        super.paint(g);

  }

  public void probabilidadSimilar(){
        double d;
        double h[];
        h = new double[42];
        for(r=0; r<cuadros.size(); r++){
                d = (double)(Math.round((0.71 - cuadros.get(r).getProbabilidad())*100))/100;
                h[r] = d;

                //System.out.println(h[r]);
        }
  }

  public List<Persona> getUsuarios(){
        return usuarios;
  }

  //QUICKSORT grilla, continuar AQUI
  public void burbuja(int alto){
        if(cuadros == null || cuadros.size() == 0){
                return;
        }

        if(0 >= alto){
```

```
                return;
        }

        CuadroGrilla temp;

                for(r=0; r<cuadros.size()-1; r++){
                        for(index=0; index<cuadros.size()-r-1; index++){
                                if(cuadros.get(index).compareTo(cuadros.get(index+1)) == 1){
                                        temp = cuadros.get(index);
                                        cuadros.set(index, cuadros.get(index+1));
                                        cuadros.set(index+1, temp);
                                }
                        }
                }
  }

  public void encontrarCandidatos(int c){
        rb=1;
        rn=1;
        for(r=0; r< c; r++){
                //System.out.println(cuadros.get(r).getId());
                if(r%2 == 0 && (ig-rb)>= 0 || (ig+rn)>=42){
                        grilla.add(cuadros.get(ig-rb));
                        System.out.println("El elemento justo atras del usuario seleccionado es:
" +cuadros.get(ig-rb).getId());
                        rb++;
                }
                else if(r%2 != 0 && (ig+rn)<42 || (ig-rb)<0){
                        grilla.add(cuadros.get(ig+rn));
                        System.out.println("El elemento justo delante del usuario seleccionado
es: " +cuadros.get(ig+rn).getId());
                        rn++;
                }

                System.out.println("Los candidatos son: " + grilla.get(r).getId());
        }
  }

  public void generarCandidato(int f){
        ArrayList<CuadroGrilla> set1;
        set = new ArrayList<ArrayList<CuadroGrilla>>();
        List<Integer> nr = new ArrayList<Integer>();

        set.add(grilla);

        for(r=0; r<f; r++){
                nr.add(r);
        }

        /*for(index=0; index<2*(Integer.valueOf(nk)); index++){
                Collections.shuffle(nr);
```

```
                System.out.println("Set 1: ");
                for(r=0; r<2*(Integer.valueOf(nk)); r++){
                        System.out.println("Candidato: " + nr.get(r));
                }
        }*/


        int kr = 0;
        double total = 0, subtotal = 0;
        double total2 = -1;
        for(r=0; r<f; r++){
                Collections.shuffle(nr);
                set1 = new ArrayList<CuadroGrilla>();
                for(index=0; index<(f/2); index++){
                        if(index == 0){
                                set1.add(cuadros.get(ig));
                                total += set1.get(index).getProbabilidad();
                        }
                        else{
                                set1.add(grilla.get(nr.get(index)));
                                total += set1.get(index).getProbabilidad();
                        }
                }

                for(kr=0; kr<(f/2); kr++){
                        subtotal += ((set1.get(kr).getProbabilidad()/total) *
(Math.log(set1.get(kr).getProbabilidad()/total)/Math.log(2)))*-1;
                        //System.out.println("El set " +r+ " corresponde a: "
+set1.get(kr).getId());
                }

                System.out.println("El Pij del set " + r + " corresponde a: " + subtotal);

                if(subtotal > total2){
                        set.set(0, set1);
                        total2 = subtotal;
                        //System.out.println("La mayor entropia es: " + subtotal);
                }

                total = 0;
                subtotal = 0;
        }

  }

  //Método que realiza y controla los movimientos de cada usuario
  public void moverUsuarios(){
        for(int s=0; s<usuarios.size(); s++){
                x1[s]+=deltax[s];
                y1[s]+=deltay[s];
```

```
                    if(s<8 &&
((grafo.getSegmentos().get(s).getFinal().getPosicion().getPosicionX()==vehiculos.get(0).getPosici
on().getPosicionX() &&
grafo.getSegmentos().get(s).getFinal().getPosicion().getPosicionY()==vehiculos.get(0).getPosicion
().getPosicionY()) ||
(grafo.getSegmentos().get(s).getInicial().getPosicion().getPosicionX()==vehiculos.get(0).getPosici
on().getPosicionX() &&
grafo.getSegmentos().get(s).getInicial().getPosicion().getPosicionY()==vehiculos.get(0).getPosicio
n().getPosicionY()))){
                        System.out.print("El vehiculo paso por el segmento: "
+grafo.getSegmentos().get(s).getId() + " con coordenadas: "
+grafo.getSegmentos().get(s).getInicial().getPosicion().getPosicionX()+ ",
"+grafo.getSegmentos().get(s).getInicial().getPosicion().getPosicionY()+ "\n");
                    }
                    if(s<8 && di[s]==0){
                        x2[s]+=delta2[s];
                    }
                    else if(s<8 && di[s]==1){
                        y2[s]+=delta2[s];
                    }

                    if(x1[s] + (10*2) > 1240){
                        x1[s] = 1240 - (10*2);
                        deltax[s]*=-1;
                    }
                    else if(x1[s]<80){
                        x1[s] = 80;
                        deltax[s]*= -1;
                    }
                    if(y1[s] + (10*2) > 560){
                        y1[s] = 560 - (10*2);
                        deltay[s]*= -1;
                    }
                    else if(y1[s] < 80){
                        y1[s] = 80;
                        deltay[s]*= -1;
                    }
                    if(s<8 && di[s]==0 && x2[s] + (10*2) >
grafo.getSegmentos().get(s).getFinal().getPosicion().getPosicionX()){
                        x2[s] =
grafo.getSegmentos().get(s).getFinal().getPosicion().getPosicionX() - (10*2);
                        delta2[s]*= -1;
                    }
                    else if(s<8 && di[s]==0 && x2[s] <
grafo.getSegmentos().get(s).getInicial().getPosicion().getPosicionX()){
                        x2[s] =
grafo.getSegmentos().get(Integer.valueOf(s)).getInicial().getPosicion().getPosicionX();
                        delta2[s]*= -1;
                    }
                    if(s<8 && di[s]==1 && y2[s] + (10*2) >
grafo.getSegmentos().get(s).getFinal().getPosicion().getPosicionY()){
```

```
                        y2[s]                                                   =
grafo.getSegmentos().get(s).getFinal().getPosicion().getPosicionY() - (10*2);
                        delta2[s]*= -1;
                }
                else    if(s<8      &&      di[s]==1      &&      y2[s]      <
grafo.getSegmentos().get(s).getInicial().getPosicion().getPosicionY()){
                        y2[s]                                                   =
grafo.getSegmentos().get(s).getInicial().getPosicion().getPosicionY();
                        delta2[s]*= -1;
                }
                usuarios.get(s).setPosicion(new Posicion(x1[s], y1[s]));
                if(s<8){
                        vehiculos.get(s).setPosicion(new Posicion(x2[s], y2[s]));
                }
                repaint();
                }
  }

  //Se calculan las posiciones iniciales de los usuarios, tanto personas como vehiculos
  public void valoresIniciales(){

        //Variables que se utilizan para el calculo de los limites
        int mx = 0, my = 5000, mex = 5000, mey = 0, mh = 0, meh = 0, mv = 0, mev = 0;

        //Se obtienen los limites de movimiento
        for(int f = 0; f<grafo.getSegmentos().size(); f++){
                if(grafo.getSegmentos().get(f).getInicial().getPosicion().getPosicionX() > mx){
                        mx                                                      =
grafo.getSegmentos().get(f).getInicial().getPosicion().getPosicionX();
                }
                if(grafo.getSegmentos().get(f).getInicial().getPosicion().getPosicionY() < my){
                        my                                                      =
grafo.getSegmentos().get(f).getInicial().getPosicion().getPosicionY();
                        System.out.println("El mayor en y es: " +my);
                }
                if(grafo.getSegmentos().get(f).getFinal().getPosicion().getPosicionX() < mex){
                        mex                                                     =
grafo.getSegmentos().get(f).getFinal().getPosicion().getPosicionX();
                }
                if(grafo.getSegmentos().get(f).getFinal().getPosicion().getPosicionY() > mey){
                        mey                                                     =
grafo.getSegmentos().get(f).getFinal().getPosicion().getPosicionY();
                }
                if(mx > mex){
                        mh = mx;
                        meh = mex;
                }
                else{
                        mh = mex;
                        meh = mx;
                }
```

```
                if(my > mey){
                        mv = my;
                        mev = mey;
                }
                else{
                        mv = mey;
                        mev = my;
                }
        }

        //Se inicializan las posiciones iniciales y velocidades iniciales de cada PERSONA
                        for(index=0; index< um; index++){
                                deltax[index] = 1 + ran.nextInt(5);
                                deltay[index] = 1 + ran.nextInt(5);
                                usuarios.add(new                Persona(Color.YELLOW,
String.valueOf(index)));
                                x1[index] = (meh-30) + ran.nextInt(mh-70);
                                y1[index] = (mev -30) + ran.nextInt(mv);
                                usuarios.get(index).setPosicion(new         Posicion(x1[index],
y1[index]));

  //System.out.println(usuarios.get(index).getPosicion().getPosicionX());
                        }

                        //Se inicializan las posiciones y velocidades de cada VEHICULO
                        for(index =0; index<8; index++){

  if(grafo.getSegmentos().get(index).getInicial().getPosicion().getPosicionY()                ==
grafo.getSegmentos().get(index).getFinal().getPosicion().getPosicionY()){
                                        vehiculos.add(new
Vehiculo(String.valueOf(um+index)));
                                        x2[index]                                            =
grafo.getSegmentos().get(index).getInicial().getPosicion().getPosicionX()                   +
ran.nextInt(grafo.getSegmentos().get(index).getFinal().getPosicion().getPosicionX());
                                        y2[index]                                            =
grafo.getSegmentos().get(index).getInicial().getPosicion().getPosicionY();
                                        di[index] = 0;
                                        vehiculos.get(index).setPosicion(new
Posicion(x2[index], y2[index]));

  grafo.getSegmentos().get(index).agregarVehiculo(vehiculos.get(index));

                                }


  if(grafo.getSegmentos().get(index).getInicial().getPosicion().getPosicionX()                ==
grafo.getSegmentos().get(index).getFinal().getPosicion().getPosicionX())
                                {
                                        vehiculos.add(new
Vehiculo(String.valueOf(um+index)));
```

```
                                        x2[index]                                       =
grafo.getSegmentos().get(index).getInicial().getPosicion().getPosicionX();
                                        y2[index]                                       =
grafo.getSegmentos().get(index).getInicial().getPosicion().getPosicionY()              +
ran.nextInt(grafo.getSegmentos().get(index).getFinal().getPosicion().getPosicionY());
                                        di[index] = 1;
                                        vehiculos.get(index).setPosicion(new
Posicion(x2[index], y2[index]));

  grafo.getSegmentos().get(index).agregarVehiculo(vehiculos.get(index));
                                }
                                delta2[index] = 5 + ran.nextInt(20);
                        }
  }




public void cargarGrilla(){
        //Se generan y adaptan las probabilidades para la GRILLA
                        for(r=0; r<42; r++){
                                pu[r] = (double)Math.round(ran.nextDouble()*1000)/1000;
                                d2 += pu[r];
                        }

        //Se inicializa la grilla
                        double totalp = 0;
                        for(r=0; r<42; r++){
                                if(r==0){
                                        figura = new Rectangle2D.Float(xg, yg, width, width-
90);
                                }
                                else if(r%7!=0 && r!=0){
                                        xg+=width+4;
                                        figura = new Rectangle2D.Float(xg, yg, width, width-
90);
                                }
                                else if(r%7==0){
                                        yg+=width-90+4;
                                        xg=25;
                                        figura = new Rectangle2D.Float(xg, yg, width, width-
90);
                                }
                                cuadros.add(new                         CuadroGrilla(figura,
(double)(Math.round((pu[r]/d2)*1000))/1000, r));
                                pu[r] = cuadros.get(r).getProbabilidad();
                                totalp += pu[r];
                        }
  }
```

```
  public void convertirFraccion(double num){
        /*BigDecimal number = new BigDecimal(num);
        number.toString().split()*/
  }

  class nuevoComboBoxRenderer extends JLabel implements ListCellRenderer<Object>{
        private String titulo;

        public nuevoComboBoxRenderer(String titulo){
              this.titulo = titulo;
        }




         @Override
        public Component getListCellRendererComponent(JList<?> list, Object value, int index,
boolean isSelected, boolean hasFocus){
              if(index == -1 && value == null){
                    setText(this.titulo);
              }
                 else{
                    setText(value.toString());
                    this.setFocusable(true);
              }

              return this;
        }

  }
}
```

Clase Ventana:

```
package com.grafocaminos.estructura;

import java.awt.BasicStroke;
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.MenuItem;
import java.awt.Point;
import java.awt.SystemTray;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.geom.Ellipse2D;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.math.BigDecimal;
import java.math.BigInteger;
import java.util.ArrayList;
import java.util.Random;
import java.util.StringTokenizer;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.SwingUtilities;
import javax.swing.UIManager;

//Clase encargada de crear la ventana de inicio, cargar las posiciones iniciales e iniciar las demas
pantallas
@SuppressWarnings("serial")
public class Ventana extends JFrame{
  private Graphics g;
  private javax.swing.JComponent pantalla;
  private JTextArea areaText1;
  private JComboBox<String> comboBox1;
  private Dimension screenSize;
  private JLabel label1;
  private JPanel panel1;
  private JButton boton1;
```

```
private Grafo grafo;
static PintarGrafo personas;
private Segmentos segmento;
private Posicion posicionNodo;
private Nodo nodoInicial;
private Nodo nodoFinal;
private Vehiculo vehiculo;
private String temp = "";
private String contenido_archivo;
private String idVehiculo;
private String idSeg;
private int s = 0, m = 0, p=0, x = 0, y = 0, velocidad = 0;
private ArrayList<String> cargar_posiciones;
private ArrayList<String> cargar_nombres;
private File archivo;
private FileReader read;
private BufferedReader br;
private StringTokenizer tp;
private SystemTray tray;
private java.awt.TrayIcon trayIcon;

public Ventana(){
        super("Simulador gráfico tecnicas encubrimiento");
        initComponents();

        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
          } catch (Exception e) {}

        if (SystemTray.isSupported()) {
                tray = SystemTray.getSystemTray();
                java.awt.Image image = new
ImageIcon(getClass().getResource("/logoubb.png")).getImage();
                ActionListener exitListener = new ActionListener()
                {
                        public void actionPerformed(ActionEvent e) {
                                System.exit(0);
                                }
                        };

                        java.awt.PopupMenu popup = new java.awt.PopupMenu();
                        MenuItem defaultItem = new MenuItem("Cerrar");
                        defaultItem.addActionListener(exitListener);
                        popup.add(defaultItem);
                        defaultItem = new MenuItem("Maximizar");
                        defaultItem.addActionListener(new ActionListener()
                        {
                                public void actionPerformed(ActionEvent e) {
                                        setVisible(true);
                                        setExtendedState(0);
                                        }
```

```
                                        });
        popup.add(defaultItem);
        trayIcon = new java.awt.TrayIcon(image, "Generador consumos electricos", popup);
        trayIcon.setImageAutoSize(true);
      }
      addWindowStateListener(new java.awt.event.WindowStateListener() {
        public void windowStateChanged(WindowEvent e) {
          if (e.getNewState() == 1) {
            try {
              tray.add(trayIcon);
              setVisible(false);
            } catch (java.awt.AWTException ex) {}
          }
          if (e.getNewState() == 7) {
            try {
              tray.add(trayIcon);
              setVisible(false);
            } catch (java.awt.AWTException ex) {}
          }
          if (e.getNewState() == 6) {
            tray.remove(trayIcon);
            setVisible(true);
          }
          if (e.getNewState() == 0) {
            tray.remove(trayIcon);
            setVisible(true);
          }
        }
      });
      setIconImage(new ImageIcon(getClass().getResource("/logoubb.png")).getImage());
    }

    public void initComponents(){
            vehiculo = new Vehiculo(idVehiculo);
            //nodoInicial = grafo_caminos.getSegmentos().get(s).getInicial();
            //nodoFinal = grafo_caminos.getSegmentos().get(s).getFinal();
            segmento = new Segmentos(nodoInicial, nodoFinal, idSeg);
            segmento.agregarVehiculo(vehiculo);
            areaText1 = new JTextArea();
            comboBox1 = new JComboBox<String>();
            label1 = new JLabel();
            panel1 = new JPanel();
            boton1 = new JButton();
            label1.setText("<html>" + "Bienvenido al simulador grafico de la topologia del área de
Concepcion" + "</html>");
            label1.setFont(new Font("Monospaced", Font.BOLD, 16));
            label1.setBounds(480, 20, 480, 260);
            areaText1.setText("\n\nBienvenido al simulador grafico de la topologia del área de
Concepcion\n\n\n\n\n\n\n\n\n\n\n\n\n\n");
            areaText1.setSize(340, 340);
            areaText1.setEditable(false);
```

```
        comboBox1.addItem("Tecnica Ben Niu");
        comboBox1.addItem("Tecnica Feeling Based");
        comboBox1.addItem("Tecnica Road Networks");
        boton1.setText("Generar caminos");
        boton1.setBounds(620, 550, 120, 30);
        panel1.setName("PanelPrincipal");
        panel1.setBackground(getBackground());
        panel1.add(boton1);
        panel1.add(label1);
        //panel1.add(comboBox1);

        panel1.setSize(640, 520);
        panel1.setBackground(Color.RED);
        panel1.setLayout(null);

        boton1.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent evt) {
                        generarRutas(evt);
                }
        });

        screenSize = Toolkit.getDefaultToolkit().getScreenSize();

        setFocusable(false);
        setResizable(true);
        setSize(screenSize.width,screenSize.height);

        setLayout(new BorderLayout());
        add(panel1);
}

public void generarRutas(ActionEvent evt){
        panel1.setVisible(false);

        cargarPosiciones();

        personas = new PintarGrafo(grafo);

        //System.out.println(personas.getUsuarios().size());
        this.add(personas);
        setVisible(true);

}

public void cargarPosiciones(){
        grafo = new Grafo();
        archivo = new File("./posiciones.txt");
        cargar_posiciones = new ArrayList<String>();
        cargar_nombres = new ArrayList<String>();
        String temp2 = "";
```

```
try{
        read = new FileReader(archivo);
        br = new BufferedReader(read);
}catch(IOException ex){
        System.out.println("Error al leer posiciones de vertices");
}finally{

}

try {
        while((contenido_archivo = br.readLine())!=null){
                temp = temp + contenido_archivo;
        }
        temp2 = temp;
} catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
}

separar(temp2);

System.out.println(cargar_posiciones.size());

for(int i = 0; i < cargar_posiciones.size(); i++){
        if(i%2 != 0){
                y = Integer.valueOf(cargar_posiciones.get(i));
                s++;
        }

        else{
                x = Integer.valueOf(cargar_posiciones.get(i));
        }

        if(s%2 == 0 && i != 0 && s!=p){
                posicionNodo = new Posicion(x,y);
                //System.out.println(x);
                nodoFinal = new Nodo(String.valueOf(i), posicionNodo);
                m++;
                p=s;
        }

        else if(s%2 != 0 && i != 0 && s!=p){
                posicionNodo = new Posicion(x,y);
                //System.out.println(y);
                nodoInicial = new Nodo(String.valueOf(i), posicionNodo);
                m++;
                p=s;
        }

        if(m == 2){
```

```
                        segmento = new Segmentos(nodoInicial, nodoFinal,
String.valueOf(velocidad));
                        velocidad++;
                        grafo.setSegmentos(segmento);
                        m = 0;
                }
        }

        cargar_posiciones.clear();

  }

  public void separar(String bp){
        tp = new StringTokenizer(bp, ",");

        while(tp.hasMoreTokens()){
                cargar_posiciones.add(tp.nextToken());
        }

  }




  public static void main(String[] args){
        try { javax.swing.JFrame.setDefaultLookAndFeelDecorated(true);
     javax.swing.JDialog.setDefaultLookAndFeelDecorated(true);
     UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
    } catch (Exception e) { e.printStackTrace();
    }
        java.awt.EventQueue.invokeLater(new Runnable() {
                public void run() {
                        new Ventana().setVisible(true);


                        PintarGrafo figuras = new PintarGrafo();

                        /*System.out.println(figuras.getUsuarios().size());

                        new Thread(new MovimientoUsuarios(figuras)).start();*/
                }
        });

  }

}
```