Instituto Politécnico Nacional

Centro de Investigación en Computación

Maestría en Ciencias de la Computación

SERVICIOS DE LOCALIZACIÓN SOBRE UNA INFRAESTRUCTURA DE TELEFONÍA CELULAR

T E S I S

QUE PARA OBTENER EL GRADO DE MAESTRO EN CIENCIAS

PRESENTA EL ING. DANIEL EDUARDO PAREDES AGUIRRE

ASESOR: M. en C. ROLANDO MENCHACA MÉNDEZ

IPN - CIC
México D.F.
Septiembre 2003

Índice

Índice de Figuras

Capítulo 1. Protocolo de Aplicaciones Inalámbricas - WAP

Capítulo 2. Algoritmos de Búsqueda de Caminos Mínimos

Capítulo 3. Arquitectura y Análisis de la Aplicación

Capítulo 4. Diseño de la Aplicación

Capítulo 5. Implementación y Pruebas de la Aplicación

Servicios de Localización sobre una Infraestructura de Telefonía Celular

Resumen

En esta tesis se presenta la investigación, diseño e implementación de una aplicación de cómputo móvil que proporciona un conjunto de servicios basados en la localización sobre una infraestructura de telefonía celular. Esta aplicación permite a los usuarios encontrar una ruta para trasladarse de un lugar a otro dentro de una ciudad. La ruta que entrega el sistema tiene como característica ser la óptima en una relación de distancia, tiempo y condiciones viales en el momento en que realiza la consulta. Otro servicio que provee esta aplicación es el de realizar una búsqueda de un sitio de interés en particular o el más cercano a un lugar en específico.

Debido a que la aplicación está diseñada para implementarse sobre una infraestructura de telefonía celular, en el capítulo 1 se presenta un trabajo de investigación acerca de la tecnología WAP. WAP es un estándar abierto para el desarrollo de aplicaciones para dispositivos de comunicación móvil.

Como se dijo anteriormente, esta aplicación entrega rutas óptimas de traslado. Por esta razón en el capítulo 2 se presenta una investigación y análisis de los principales algoritmos de búsqueda de rutas óptimas en grafos. Así mismo se presenta la teoría básica de grafos.

En el siguiente capítulo se presenta la arquitectura para el desarrollo de esta aplicación, tomando en cuenta las tecnologías, algoritmos y dispositivos que son usados. Aquí mismo se presenta el análisis de los requerimientos de la aplicación y las características y funcionalidad que debe poseer en su fase terminal.

En el capítulo 4 se presenta la fase de diseño para el desarrollo de la aplicación mencionada. Además se presentan los Diagramas de Interacción y de Clases del UML. También se presenta el Diagrama de Estados de la implementación del algoritmo de Dijkstra para la búsqueda de la ruta óptima en grafos. En este mismo capítulo se realiza el diseño del Depósito de Datos así como el Modelo Lógico de la Base de Datos. Como parte final del capítulo se muestra una explicación para la generación del grafo a partir de la información almacenada en la base de datos.

La implementación de la aplicación es presentada en el capítulo 5, primeramente se describen de manera breve las herramientas que fueron utilizadas para llevar a cabo dicha implementación. Más adelante se muestran los detalles de implementación de la aplicación final, de la aplicación para dar mantenimiento a la información de la Estructura Vial y los Sitios de Interés y de los servicios de recuperación y modificación de información de la Base de Datos. Para demostrar la funcionalidad de la aplicación, dentro de este mismo capítulo se presentan las pruebas realizadas al sistema y los resultados obtenidos.

Por último, en el capítulo 6 se realiza una propuesta de líneas de investigación que pueden desarrollarse tomando como base esta tesis. Así mismo se presentan las conclusiones y comentarios finales referentes al término del desarrollo de la presente tesis.

Location-Based Services over a Mobile Telephony Infrastructure

Abstract

This thesis shows the research, design and development of a mobile computing application. This application is capable to provide a collection of location-based services over a mobile telephony infrastructure. The application allows users to find a route to go from one point to another one in a city. The route showed by the system is the very best in distance, time and road conditions terms at the moment. An additional service let users search for a particular interest place or the nearest one.

Because this application is designed to run over a mobile telephony medium, an introduction to the WAP (Wireless Application Protocol) technology is presented in chapter 1. WAP is an open, global specification that empowers mobile users with wireless devices to easily access and interact with information and services instantly.

As it was aforementioned, the application provides optimum routes. In order to achieve this, chapter 2 presents an analysis of the main shortest path algorithms and basic graph theory.

Based on previous chapters, in the next one it is presented the application architecture taking into account the technologies, algorithms and devices used. In addition, requirements analysis and final characteristics of the system are presented.

Chapter 4 presents the application design phase. Interaction, classes and Dijkstra's algorithm state diagrams are presented too. Further more, database design and logic model are shown. At the end of the chapter it is presented an explanation of how to generate the graph with the database information.

Application development phase is presented in chapter 5, initially the used software tools are described. The final user application and the utility to maintain interest places and road structure information are also explained in detail. In order to demonstrate the proper operation of the application developed, this chapter presents the results of the tests applied for the system.

In chapter 6 are suggested some research lines that could be based on this thesis. Final commentaries and conclusions are exposed too.

Introducción

En los últimos años, la telefonía celular ha estado presente en nuestra sociedad como un elemento de comunicación de total independencia y movilidad. Ahora tenemos la posibilidad de acceder a Internet desde nuestro dispositivo móvil (Teléfono celular, Palm, otros PDAs[1], etc.) desde cualquier lugar, teniendo a nuestra disposición un gran número de ventajas y servicios.

El cómputo móvil es una solución que permite a todos los sectores económicos ampliar sus fronteras de crecimiento y mejorar la operación de los negocios. En la administración de los procesos de venta, producción y distribución las empresas logran mejores controles y una sincronización de todos sus procesos. El desarrollo de estas tecnologías habilita una nueva etapa de desarrollo industrial.

El cómputo móvil permite sincronizar y controlar procesos y eventos donde el acceso a datos e información es prácticamente imposible por otros medios. Ejemplos:

- Acceso a expedientes clínicos en línea en casos de emergencia.
- Envío de fotografías de siniestros desde el lugar del accidente para su análisis, interpretación y devolución (ajustadores de seguros).
- Acceso a planos y datos para la administración de servicios públicos (drenajes, energía eléctrica, redes pluviales, etc.).

El cómputo móvil facilita los servicios de venta, facturación, cobranza y calidad en el servicio obteniendo y proporcionando datos e información en el lugar y en el momento donde se realiza la operación. Ejemplos:

- Proporcionando información sobre productos almacenados o en producción.
- Informando sobre el status o localización de un pedido ya realizado.
- Informando sobre el estado crediticio de un cliente.
- Confirmando cantidades y materiales recibidos por el cliente.
- Optimizando transacciones de venta y producción, capturando los datos del cliente y llevándolos directamente a los almacenes de envío o áreas de producción.

A través del uso de circuitos integrados (microlocalizadores) y ayuda satelital es posible rastrear productos, unidades móviles o bien personas. Ejemplos:

- Replanteo de rutas de servicio o transporte en casos de emergencia (accidentes, robos).
- Propuesta de mejores rutas de acceso.
- Localización de unidades para el control de paradas y consumo de combustible.

[1] Personal Digital Assistant.- Asistente Personal Digital

- Seguimiento de unidades de transporte bancario.
- Seguimiento de eventos deportivos especializados (maratonistas, ciclistas).

Las aplicaciones del cómputo móvil que se desarrollan en la actualidad pueden crecer enormemente con la prestación de servicios basados en la localización. Encontrar un restaurante según las preferencias del cliente, guiar a los vendedores en sus recorridos o ayudar en la logística serán algunas de las numerosas aplicaciones que podrán proporcionar servicios de este tipo. Y no solo eso, también podrán indicarles cuales rutas son las más cortas en ese instante de acuerdo al tráfico o cuales lugares, en los que están interesados, son los más cercanos.

La presente tesis aborda el problema de la búsqueda de rutas óptimas entre dos puntos y la búsqueda de los sitios de interés más cercanos a un punto en específico así como también la ruta óptima para llegar a éstos. Está organizada de la siguiente forma:

El capítulo 1 explica que es el Protocolo de Aplicaciones Inalámbricas (WAP[2]), su arquitectura, capas, servicios y funcionamiento general.

Una descripción de los algoritmos de búsqueda de caminos más cortos así como una pequeña lista de definiciones básicas referentes a los grafos son presentadas en el capítulo 2.

En el capítulo 3 se plantea la propuesta de solución al problema, su arquitectura y el análisis. El diseño del sistema se presenta en el capítulo 4.

La implementación de la solución planteada es descrita y documentada en el capítulo 5, haciendo una descripción de los programas obtenidos. Las pruebas de funcionamiento de la aplicación obtenida así como los resultados alcanzados son mostrados también en este capítulo.

En el capítulo 6 se presentan las conclusiones generales de la tesis y se proponen trabajos futuros en los que esta tesis sirva como base para su desarrollo.

Al final se dan las referencias y un glosario de términos importantes tratados a lo largo de la tesis. Los anexos presentan información necesaria para facilitar la comprensión del presente trabajo así como el código fuente de la aplicación desarrollada.

[2] Wireless Application Protocol

Trabajos Previos Relacionados

No existe ningún trabajo previo en específico que se haya tomado como base para el desarrollo de este trabajo. Aquellos trabajos, cuyo tema de estudio sea alguna de las tecnologías que se usarán para el desarrollo de esta tesis, serán los que se tomen como trabajos previos relacionados. No son muchas las aplicaciones se puedan tomar como antecedentes para la realización de esta tesis, la razón podría ser la falta de motivación de los programadores debido a las limitaciones funcionales que presentan los dispositivos móviles, algunas de éstas son: poca velocidad de procesamiento y capacidad de memoria, diminuta pantalla monocromática y de baja resolución, mayor dificultad para la introducción de caracteres alfabéticos debido a que la mayoría de los teléfonos celulares cuentan con teclado numérico, escaso ancho de banda, entre otras.

A continuación se presentan un par de servicios que ofrecen dos compañías de telefonía celular. Dichos servicios son similares en objetivo al que se desarrolla en esta tesis, pero diferentes en arquitectura y/o tecnología.

e-moción de Telefónica MoviStar®

e-moción es un servicio que ofrece Telefónica MoviStar en España, con el cual se pueden encontrar los lugares de interés más cercanos a un punto determinado: restaurantes, hoteles, cines, farmacias, gasolineras, etcétera. de los cuales se obtendría el teléfono, dirección o características del lugar [WWW12]. Esta consulta se realiza básicamente de dos formas:

- Por medio de un mensaje al número 404 indicando el tipo de lugar deseado, i.e. `cercaresit` indica que se está interesado en el restaurante de comida italiana más cercano.
- La otra forma es marcando el número 4040 en la que una grabación da las instrucciones necesarias.

Desventajas:

- Solo busca sitios de interés.
- No proporciona ruta alguna para llegar al sitio de interés deseado, solo regresa la dirección y teléfono del más cercano.

***RUTA** de Telcel®

*RUTA es un servicio que ofrece Telcel en México, a través del cual se obtiene una ruta para poder desplazarnos de un punto de la ciudad a otro [WWW13]. El funcionamiento de este servicio es el siguiente:

1. Marcar *RUTA (*7882)
2. Una operadora solicita el punto de partida, este punto tiene que ser dado por el nombre de dos calles que se intercepten.

3. Igual que el punto anterior solo que para el punto destino.
4. La operadora después de haber hecho la consulta en el sistema, da una explicación del trayecto que se debe seguir.

Desventajas:

- El horario de servicio es limitado, solo de 8:00 AM a 10:00 PM.
- No permite la búsqueda de sitios de interés ni de rutas hacía ellos.
- Para realizar una consulta es necesario una operadora.
- No permite que una dirección sea dada como calle y número, es forzoso proporcionar una esquina.
- Los datos recibidos deben ser memorizados ó escritos por el usuario.

Algunas otras aplicaciones del cómputo móvil son las siguientes:

Accitrade Móvil de Banamex®

AcciTrade Móvil es un servicio que proporciona toda la información financiera del mercado mexicano e internacional a través de teléfonos celulares disponible las 24 horas durante los 365 días del año. Accitrade Móvil provee de información acerca de los principales índices del mercado bursátil: el Índice de Precios y Cotizaciones, el Dow Jones, el Nasdaq, la cotización del dólar americano, el precio de las acciones de las empresas que cotizan en la Bolsa Mexicana de Valores, etc.

Planteamiento del Problema

Se planea diseñar y construir una arquitectura de cómputo móvil capaz de proveer un conjunto de servicios basados en la localización, sobre una infraestructura de telefonía celular.

Las características que debe ofrecer la arquitectura que dará solución al problema son las siguientes:

- Búsqueda de caminos óptimos entre dos puntos dados(origen y destino) dentro de una ciudad.
- Búsqueda de sitios de interés y las rutas óptimas para llegar a éstos.
- Búsqueda del sitio de interés más cercano a un punto de origen dado.
- Mantenimiento de información referente a la vialidad en una ciudad.
- Simulación de los módulos de telefonía celular que son parte de la arquitectura: el teléfono celular que cuente con un navegador para Internet Móvil y el WAP Gateway.

Objetivos

Objetivo General

 El objetivo principal de la presente tesis es diseñar e implementar una aplicación que suministre un conjunto de servicios de localización accesibles desde dispositivos de telefonía celular, en los cuales se proporcione la búsqueda de rutas óptimas para trasladarse de un punto a otro en una ciudad.

Objetivos Específicos

- Diseñar e implementar una aplicación para el mantenimiento de información del sistema, así como también de los módulos de consulta de rutas y sitios de interés vía WAP y, por último, del algoritmo de Dijkstra para encontrar la ruta más corta.

- Desarrollar una estructura de base de datos que permita almacenar un grafo que represente la estructura vial de una ciudad.

- Diseñar e implementar un mecanismo que permita al usuario continuar su búsqueda en el punto donde se quedó justo antes de alguna desconexión o problema con la señal.

- Diseñar y crear la base de datos así como de los servicios de recuperación y mantenimiento de información.

- Análisis del desarrollo de interfaces de usuario apropiadas para teléfonos celulares y sus limitaciones.

- Realizar un análisis de los algoritmos para búsqueda de caminos más cortos y los conceptos básicos de la teoría de grafos.

- Evaluar herramientas útiles para el desarrollo de aplicaciones para dispositivos de telefonía celular e investigar acerca de su uso (simuladores de teléfonos celulares y *Gateways* WAP).

Justificación

El número de usuarios de telefonía celular ha crecido considerablemente en los últimos años, así como la demanda servicios innovadores y sobre todo funcionales. Con la recién introducción al mercado de Internet móvil por parte de las principales compañías de telefonía celular en México se abre la posibilidad de desarrollar toda una gama de servicios similares a los que existen en el Internet tradicional, con la ventaja de que estos nuevos servicios podrán ser utilizados inalámbricamente por medio de dispositivos celulares.

Dado que con un teléfono celular se puede estar conectado a Internet sin necesidad de un cable, nuevas clases de servicios podrán ser desarrolladas. Una de estas clases será la integrada por los servicios basados en la localización (*Location-Based*), los cuales tendrán la característica de tomar en cuenta la posición del teléfono(y más que del teléfono, del usuario) para ofrecerle diferentes servicios.

Por otro lado, el número de personas que viven en las principales ciudades es excesivo, un gran número de estas personas cuentan con un automóvil. Esto hace que el congestionamiento vehicular en estas ciudades sea común. Por esta razón, es trascendental la creación de un servicio automatizado que indique cuál ruta es la mejor para trasladarse de un lugar a otro, tomando en cuenta diversos factores como son: las calles, la zona, un accidente, una manifestación, un bloqueo, la construcción de una obra y el tráfico en general.

Desde el punto de vista comercial, un sinnúmero de empresas podrán ofrecer a sus clientes información sobre cual de sus sucursales es la más cercana al punto donde se encuentren, o un usuario podrá consultar el sitio más cercano que sea de su interés, etc., y con ayuda el uso del servicio descrito en el párrafo anterior, cual ruta es la óptima para trasladarse hasta dicho lugar en el momento de la consulta.

Las áreas de estudio que involucra el desarrollo de esta tesis son las siguientes:

- **Algoritmos de búsqueda de caminos óptimos**
  Una de las partes fundamentales de la aplicación final de la tesis, es encontrar las rutas más cortas para trasladarse de un punto a otro.

- **Sistemas de Información**
  Aplicar una metodología de desarrollo de software para la creación de la aplicación.

- **Teoría de Grafos**
  Es necesario conocer la teoría de grafos para poder aplicar con más facilidad los algoritmos de búsqueda.

- **Bases de Datos y uso de un Manejador de Bases de Datos**
  Como todo sistema, la información que se maneja debe ser almacenada y poderse recuperar.

- **Diseño de interfaces de usuario para teléfonos celulares**
  Dadas las limitaciones funcionales que presentan los teléfonos funcionales, se deben de crear interfaces de usuario fáciles de manipular.

- **Redes de telefonía celular**
  Es útil conocer los servicios que proporcionan las tecnologías usadas en las redes de telefonía celular para crear mejores servicios.

- **Protocolo para Aplicaciones Inalámbricas (WAP)**
  La arquitectura de la aplicación por desarrollar estará basada en la que tiene el Protocolo de Aplicaciones Inalámbricas, por lo tanto es fundamental conocer está arquitectura y su funcionamiento.

- **Lenguajes de programación para teléfonos celulares**
  Es necesario conocer los lenguajes de programación que son soportados por los teléfonos celulares.

Además, el proyecto que se obtenga como resultado del desarrollo de esta tesis tiene la posibilidad de integrarse con otros proyectos del laboratorio que actualmente se están desarrollando. Ejemplo de éstos es uno que permitirá que, cuando una ambulancia esté atendiendo una emergencia médica, sería conveniente que pudiera saber cuál es el hospital más cercano que pueda recibir al paciente, así como la ruta más corta para llegar a tal.

Capítulo 1

Protocolo de Aplicaciones Inalámbricas - WAP

En este capítulo se detalla la especificación del Protocolo de Aplicaciones Inalámbricas (WAP) que define el *WAP Forum*. Se realiza un análisis de los componentes más importantes de esta arquitectura con la finalidad de aplicarlos para el desarrollo de la aplicación de esta tesis.

Primeramente se muestran los orígenes y la motivación para la creación del WAP. Más adelante se muestra el funcionamiento de una red y un *Gateway* WAP.

Para finalizar se presenta la arquitectura WAP y se explican cada una de las capas que la conforman.

1.1 Introducción

El Internet ha probado ser una fácil y eficiente forma de proporcionar servicios a millones de usuarios "conectados a una red". En 1997, Ericsson, Motorola, Nokia y Unwired Planet tuvieron la iniciativa de fundar el Foro WAP (WAP Forum), el cual tiene como misión brindar los beneficios que tiene Internet a la comunidad inalámbrica de la mejor forma posible. El WAP Forum se ha ganado un gran respeto en la industria inalámbrica de todo el mundo, más de 90 compañías líderes mundiales en el negocio de las telecomunicaciones inalámbricas son miembros de este foro. WAP es el resultado de la suma de esfuerzos del equipo de compañías que conforman el WAP Forum. Su objetivo es crear un estándar libre que proporcione información y servicios de telefonía a dispositivos inalámbricos. Para acceder a estos servicios WAP parte de una arquitectura basada en la definida para Internet: el World Wide Web (WWW), pero adaptada a los nuevos requisitos del sistema.

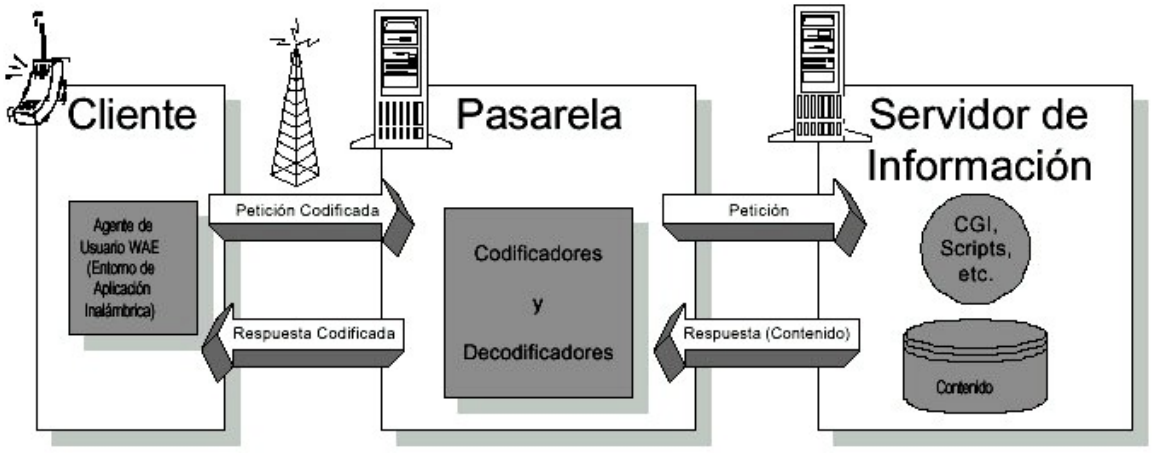


Figura 1.1 Modelo de funcionamiento del WAP

En la Figura 1.1 se muestra el modelo de funcionamiento del WAP. Primero, en el dispositivo inalámbrico(cliente) existiría un "mini navegador" encargado de la coordinación con el *Gateway* (pasarela), al cual le hace peticiones de información debidamente codificadas. El *Gateway* decodifica, procesa, convierte y redirige la petición al servidor de Información vía el Protocolo de Transferencia de HiperTexto (HTTP[1]). Una vez que el servidor procesa la petición de información por medio de un CGI, Servlet, etcétera; envía la respuesta de regreso al *Gateway*. La nueva función del *Gateway* es formatear y codificar la respuesta y enviarla al navegador del dispositivo móvil para ser desplegada [WWW14].

[1] HyperText Transfer Protocol

Para conseguir consistencia en la comunicación entre el dispositivo móvil y los servidores de red que proporcionan la información, WAP define un conjunto de componentes estándar:

- Un modelo de nombres estándar. Se utilizan los Identificadores Uniformes de Recursos (URI[2]) definidas en el WWW para identificar los recursos locales del dispositivo (tales como funciones de control de llamada) y las Direcciones Uniformes de Recursos (URL[3]) (también definidas en el WWW) para identificar el contenido WAP en los servidores de información.

- Un formato de contenido estándar, basado en la tecnología WWW.

- Unos protocolos de comunicación estándares, que permitan la comunicación del mini navegador del dispositivo móvil con el servidor Web en red.

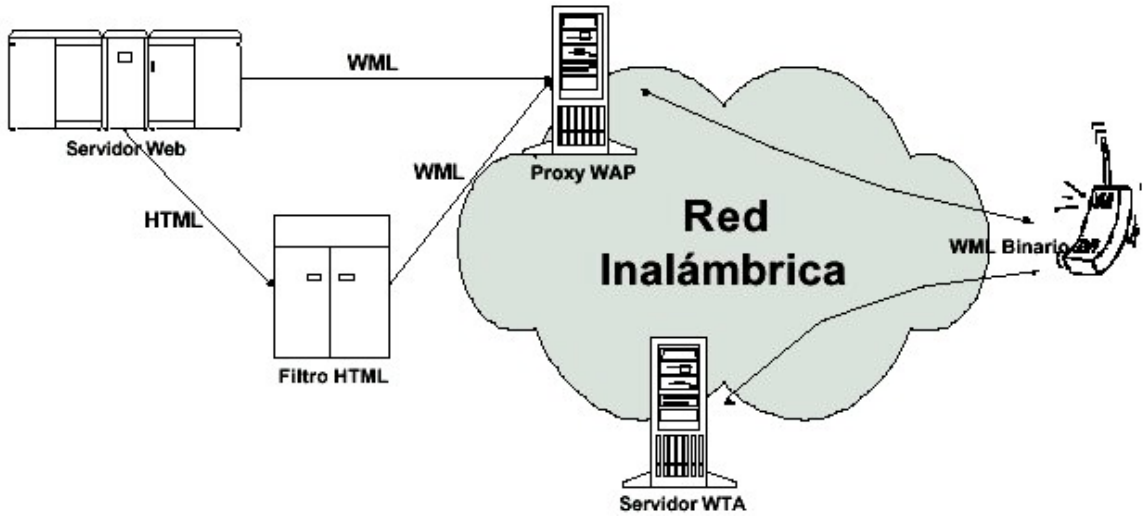


Figura 1.2 Ejemplo de una red WAP

Se presenta ahora un ejemplo de red WAP en la Figura 1.2. Se puede ver que el dispositivo móvil tiene dos posibilidades de conexión para poder acceder a servicios: Por medio de un *proxy*[4] WAP o de un servidor de Aplicaciones de Telefonía Inalámbrica (WTA[5]).

[2] Uniform Resource Identifier
[3] Uniform Resource Location
[4] Los términos *proxy*, *gateway* o pasarela se pueden intercambiar indistintamente.
[5] Wireless Telephony Applications

El *proxy* WAP tiene a su vez dos maneras de realizar peticiones al Servidor Web: De manera directa usando el Lenguaje de Marcado Inalámbrico (WML[6]) o con la ayuda de un Filtro de Lenguaje de Marcado de HiperTexto (HTML[7]) convertir la petición WML en HTML.

Por otra parte, el Servidor WTA permite acceder a las facilidades proporcionadas por la infraestructura de telecomunicaciones del proveedor de telefonía celular para desarrollar nuevas aplicaciones.

1.1.1 Tecnologías Alternas a WAP para el Desarrollo de Aplicaciones Inalámbricas

A continuación se presentan algunas tecnologías alternas al protocolo WAP, para el desarrollo de aplicaciones inalámbricas.

1.1.1.1 J2ME (Java 2 Micro Edition)

J2ME es a la vez un lenguaje de programación y una plataforma de ejecución incluida en teléfonos y otros dispositivos móviles para aporta un conjunto de nuevas capacidades para el usuario. Es especialmente conveniente para conseguir un mayor grado de interacción; para lograr capacidades gráficas; simplicidad en el manejo de los servicios e, incluso, funcionamiento sin conexión.

Se basa en pequeños programas (midlets) que se ejecutan en el teléfono móvil (inteligencia local) al estilo de un PC y que pueden acceder a Internet.

Por ello, las aplicaciones pueden ser más independientes, eficientes y rápidas en el uso de las conexiones (no siempre hay que conectarse y se evita transmitir mucha información), pueden almacenar localmente la información obtenida (no hay que volver a conectarse para mirarla) y la información de configuración (no hay que volver a teclearla).

Se incrementa enormemente la personalización del terminal al ser el usuario quien selecciona los programas que desea utilizar.

La cantidad de aplicaciones que podrá disponer el usuario es incalculable, pues el lenguaje es conocido por millones de programadores. Además, hay muchas librerías para que sean fáciles de diseñar, se reutilizará mucho trabajo ya realizado para Internet, y al ser estándar será soportado a la vez por muchos teléfonos y dispositivos.

Además, en J2ME las aplicaciones son pequeñas y pueden descubrirse y descargarse desde Internet directamente al teléfono o utilizando una computadora con conexión. Con estos métodos actualizar un programa o cambiar las aplicaciones se convierte en una tarea muy fácil, lo que facilita mucho su difusión y distribución

[6] Wireless Markup Language
[7] Hyper Text Markup Language

Los terminales que utilizan este sistema tienen unos recursos gráficos superiores (resolución, color, procesador, memoria, etc), incluyen capacidades GPRS de datos para una mayor eficiencia y velocidad de comunicación.

J2ME no es completamente una alternativa a WAP, más bien es una tecnología complementaria que añade mayor riqueza a los contenidos permitiendo crear aplicaciones inalámbricas más potentes.

Para el desarrollo de la aplicación móvil de esta tesis, fue elegido WAP sobre J2ME por las siguientes razones:

- Si bien es cierto que J2ME permite crear aplicaciones más complejas, también lo es que requiere dispositivos con mejores capacidades de procesamiento, memoria y pantalla
- En este momento existen en el mercado más teléfonos que soportan aplicaciones WAP que aplicaciones J2ME
- En general, se dispone de más información y herramientas para desarrollar aplicaciones con WAP que con J2ME
- Para el caso específico de esta tesis, las características que ofrece WAP son suficientes para el desarrollo de la aplicación propuesta, sin llegar a necesitar las que sólo J2ME permite.

1.1.1.2 I-MODE

I-mode es un servicio de Internet móvil con amplio éxito. Lanzado por NTT DoCoMo el 22 de febrero de 1999, en sus cinco primeros meses de vida alcanzó el primer millón de usuarios. Actualmente, i-mode tiene más de 30 millones de usuarios sólo en Japón, y cada día se suscriben a él más de 30.000 clientes en ese país.

El servicio i-mode, en Japón, utiliza la conmutación de paquetes para trasladar los datos desde su red W-CDMA[8] a los usuarios, a una velocidad de descarga de 9,6 kb/s, similar a la velocidad a la que circulan los datos en el sistema GSM. Con el GPRS ya se han alcanzado en Europa velocidades de transmisión de los datos de hasta 40 kb/s. FOMA, el servicio 3G de NTT DoCoMo, ofrece en la actualidad en Japón una velocidad de descarga de 384 kb/s.

El lenguaje que utiliza i-mode es una versión simplificada de HTML, en concreto Compact HTML (cHTML), frente al WML del protocolo WAP, por lo que es relativamente más fácil de utilizar para los desarrolladores de contenidos que ya tienen páginas web en Internet.

Los usuarios de i-mode pueden acceder desde sus dispositivos móviles, en una conexión IP permanente las 24 horas del día, a información, reservación de boletos de

[8] Wide band-Code Division Multiple Access

avión, realización de operaciones bancarias, envío y recepción de correo electrónico, etcétera. Alrededor de 30.000 proveedores tienen adaptados sus contenidos a i-mode.

Gran parte del éxito de este servicio en Japón se debe a los dispositivos móviles, fabricados para NTT DoCoMo. Provistos de grandes pantallas de hasta 120x130 pixeles -que permiten de 5 a 10 líneas de texto-, desde que se lanzó el servicio contaban con pantallas que permitían hasta 256 colores.

Aunque i-mode ofrece características similares a WAP y en este caso si representa una alternativa a WAP, no fue elegido para el desarrollo de la aplicación móvil de esta tesis por las siguientes razones:

- A pesar de que su uso va en ascenso, solo ha sido difundido en países asiáticos
- Mismo caso que con J2ME, en este momento hay más modelos de dispositivos móviles que soportan WAP que a esta tecnología

1.2 Orígenes del WAP

Desde 1995 diferentes compañías de telecomunicaciones intentaron estandarizar protocolos que ofrecieran más servicios dentro de la telefonía móvil.

La compañía Ericsson en 1995 presentó el protocolo ITTP (Intelligent Terminal Transfer Protocol), su viaje a la estandarización fracasó ya que las compañías Nokia y Unwired Planet (actualmente Phone.com) presentaron durante los dos años siguientes diferentes protocolos que ofrecerían diversos servicios a los usuarios.

Unwired Planet, una empresa de software para dispositivos inalámbricos, presentó el lenguaje HDML (Handheld Device Markup Language) y el protocolo HDTP (Handheld Device Transport Protocol), estos estándares eran similares al HTML y al HTTP respectivamente y estaban preparados para ser mostrados en pequeñas pantallas. A su vez Nokia presentaba el Smart Messaging una conexión entre Internet y los aparatos GSM con la ayuda del SMS y el lenguaje TTML (Tagged Text Markup Language).

Estos proyectos fracasaron por la no estandarización de ninguno de ellos debido al gran número de protocolos que aparecieron y el hecho de que fuesen siempre propiedad de alguna compañía. Esto hizo que la integración de estos nuevos servicios se viese retrasada, motivo por el cual se reunieron las principales compañías de telefonía móvil (Motorola, Nokia, Ericsson y Unwired Planet) para conseguir estandarizar un protocolo único y estable.

En 1997, el WAP Forum fue creado para especificar e implementar los objetivos de esta nueva tecnología. El esfuerzo de este organismo no consistió únicamente en definir un estándar abierto de funcionalidad de aplicaciones inalámbricas, sino que también se tuvieron en cuenta las limitaciones físicas de las transmisiones inalámbricas, así como las necesidades de los usuarios móviles. La especificación WAP ofrece numerosos beneficios

y oportunidades a distribuidores, usuarios, fabricantes y desarrolladores. La arquitectura WAP se construye sobre tecnologías basadas en los actuales modelos de redes y protocolos de Internet, mientras que al mismo tiempo integra características normalmente asociadas con telefonía y servicios móviles.

WAP nace cuando un grupo de industrias de aplicaciones y dispositivos móviles crean un consorcio para el desarrollo de un estándar abierto y ampliamente aceptado para los dispositivos de comunicaciones móviles. Las primeras organizaciones que se sentaron sobre la mesa en el verano de 1997 fueron Unwired Planet (posteriormente Phone.com) y fabricantes de móviles como Ericsson, Nokia y Motorola. Estas compañías crearon el WAP Forum, una asociación de industrias que sería responsable del desarrollo de un poderoso y ampliamente aceptado estándar de protocolo inalámbrico, a través de la integración de todos los segmentos de las comunicaciones móviles. Desde su creación el WAP Forum ha crecido rápidamente y hoy incluye a prácticamente toda la industria de las comunicaciones móviles. Con más de 200 compañías, WAP Forum representa más del 95 por ciento del mercado mundial de fabricantes, distribuidores y desarrolladores de tecnologías para dispositivos móviles. Antes de la introducción de la especificación WAP, los representantes de las comunicaciones móviles globales se dieron cuenta de la necesidad de nuevas características que antes no se habían tenido en cuenta en las comunicaciones de voz tradicionales. La especificación WAP comenzó por atacar dos de estas características:

- Las aplicaciones y protocolos de red que actualmente se aplican en Internet necesitan un gran ancho de banda, característica que, hoy en día, no ofrecen las comunicaciones inalámbricas.

- Las necesidades de los usuarios de dispositivos móviles en muchos casos no se ajustan a las de los usuarios de PCs.

Las limitaciones de las transmisiones inalámbricas demandaban una especificación que aportara la máxima eficiencia, ya que los dispositivos inalámbricos representan al máximo las limitaciones en la computación, CPU limitada, memoria limitada, baja duración de la batería e interfaz de usuario extremadamente sencilla. Las redes inalámbricas han sido ajustadas para este bajo ancho de banda e impredecible disponibilidad y estabilidad. Sin embargo, el reto consistía precisamente en hacer llegar a los nuevos usuarios móviles todo el valor y utilidad que la arquitectura WWW proporciona a los usuarios de PC actuales. La especificación WAP extiende e incrementa las tecnologías existentes, tales como el Protocolo Internet (IP[9]), HTTP, el Lenguaje de Marcado eXtensible (XML[10]) o la Capa de Socket de Seguridad (SSL[11]), por mencionar algunos.

Pero no sólo estas limitaciones en la infraestructura fueron las impulsoras de la creación de este nuevo estándar, las diferentes necesidades de los usuarios inalámbricos marcaron también esta especificación. Se ha intentado crear un estándar que permita el desarrollo de productos y servicios fáciles de usar y de disponibilidad instantánea. Debido

[9] Internet Protocol
[10] eXtensible Markup Language
[11] Socket Security Layer

a estas dos limitaciones se ha realizado un sacrificio de muchos de los componentes gráficos y dinámicos del WWW actual por una apuesta de mayor pragmatismo y funcionalidad.

1.3 Motivaciones para la creación del WAP

Durante años recientes, tanto Internet como las comunicaciones de voz inalámbricas han experimentado una amplia y rápida aceptación. Por el contrario, la unificación de estas dos tecnologías, Internet Móvil, no ha tenido el mismo nivel de desarrollo aun cuando Internet provee un medio para el desarrollo rápido de servicios, fácil uso y conveniente manejabilidad.

Una pregunta que surge entonces es por qué el uso de las capacidades de datos inalámbricos, en el contexto del acceso a Internet Inalámbrico, no ha seguido las mismas tendencias de Internet ni de las comunicaciones de voz inalámbricas.

Las expectativas son las que tienen mucho que ver con esto. Cualquiera que haya intentado acceder a Internet usando una computadora portátil y un teléfono celular, sabe que las expectativas que nos hemos creado por acceder a Internet desde la oficina o en la casa no son llenadas completamente; de hecho es común que esto se convierta en una experiencia aburrida y cansada.

WAP se enfocó en este problema siendo diseñado para hacer frente a las restricciones y limitaciones del entorno inalámbrico. Tanto las limitaciones de la red como de los dispositivos móviles fueron consideradas.

A continuación se da un resumen acerca de las motivaciones para la creación del WAP:

1.3.1 Limitaciones de la Red Inalámbrica

Los problemas más importantes en la red que son atacados por el WAP son:

Reducido Ancho de Banda: Las redes inalámbricas poseen un ancho de banda pequeño, esto conduce a que cuando una cantidad masiva de usuarios estén haciendo uso del medio de comunicación, los tiempos de respuesta se verán afectados aun más debido a la repartición del poco ancho de banda entre todos ellos. WAP ataca este problema minimizando el tráfico sobre el medio aéreo. El WML y el WMLScript son binariamente codificados en forma compacta con la finalidad de minimizar el número de bits y bytes que finalmente son enviados hacia la red.

Por otro lado, también soporta sesiones de larga duración además de permitir que sean suspendidas y activadas nuevamente, ahorrando un valioso ancho de banda al no necesitar que se establezcan con frecuencia nuevas sesiones.

Alta Latencia: Las redes inalámbricas tienen una gran latencia comparada con las redes comunes. Esta característica es importante en las redes inalámbricas actuales, aun para aquellas que proveen un ancho de banda amplio. Este problema es atacado en WAP minimizando la cantidad de información enviada del dispositivo y la red móviles. Un modelo asíncrono de petición / respuesta es también usado.

En las redes comunes, la baja latencia implica que las peticiones y respuestas sean manejadas de forma síncrona debido a que el tiempo entre éstas regularmente no afectan la interacción con el usuario. En las redes inalámbricas con grandes latencias no es viable hacer esto debido a que los tiempos de respuesta son muy lentos, por lo que el usuario perdería interacción con el sistema.

Baja Estabilidad de Conexión / Impredecible Disponibilidad de Portador: El acceso a redes conectadas físicamente por un cable provee una conexión a la red un tanto más confiable. Este no es el caso de las redes inalámbricas en las cuales los portadores pueden estar inaccesibles por periodos de tiempo pequeños y grandes debido a las constantes caídas del servicio, perdida de cobertura o deficiente capacidad.

Como ya se mencionó, las sesiones soportadas por la capa de sesión en WAP, son asumidas como de larga vida. De esta manera el problema es resuelto al permitir que las sesiones viejas sean nuevamente activadas.

La capa de transacción en WAP, ha sido comparada con el TCP[12] usado en Internet. Debido a que no se requiere conexión los efectos de la pérdida del protocolo portador y los tiempos de inactividad son minimizados.

La naturaleza de una conexión inalámbrica implica que pequeños segmentos de un mensaje frecuentemente se pierdan. La capa de transacción soporta la retransmisión selectiva de datos, y no el mensaje completo como lo hace TCP.

1.3.2 Limitaciones de los Dispositivos Móviles

WAP está dirigido a dispositivos de mano de varios tipos. Los servicios deben estar disponibles adecuadamente tanto en una PC de bolsillo como en un pequeño teléfono celular. La Figura 1.3 muestra las limitaciones de un dispositivo móvil, entre las cuales están los siguientes:

Pequeña Pantalla: La mayoría de los teléfonos celulares poseen una pantalla pequeña. No importa que tan buenas llegaran a ser estas pantallas en el futuro, dado que el tamaño de la mano humana siempre limitará el tamaño de los celulares, y por consiguiente de las pantallas. No sería buena la experiencia que se tendría accediendo a un servicio diseñado para un dispositivo con una pantalla grande desde un pequeño teléfono celular. Los resultados en la mayoría de los casos serían servicios con un pésimo desempeño lejos de aquello para los que originalmente fueron creados. La información que el usuario realmente necesitaría estaría inundada en información no deseada debido a las pequeñas dimensiones de la pantalla del celular.

[12] Transfer Control Protocol

Figura 1.3 Limitaciones de un dispositivo móvil

En vez de usar la estructura de documento tan plana que ofrece el HTML, WML estructura los documentos en barajas (deck) y tarjetas (card). Una tarjeta es una simple unidad de interacción con el usuario final., por ejemplo una pantalla de texto, una lista de selección, un campo de entrada de texto, o una combinación de ellos. Típicamente, una tarjeta es lo suficientemente pequeña para que sea desplegada, aun sobre una pequeña pantalla. Cuando un servicio es ejecutado el usuario navega a través de series de tarjetas. Las series de tarjetas usadas para conformar un servicio son agrupadas en una baraja.

Facilidades de Tecleo Limitadas: Los dispositivos inalámbricos frecuentemente no tienen las mismas facilidades para la entrada de texto como los dispositivos equivalentes en las redes comunes. Además de no contar con un ratón como interfaz.

WML aborda este problema de una manera satisfactoria. Los elementos que son usados en WML pueden ser implementados para que sea razonable la necesidad del uso del teclado. El uso de barajas y tarjetas provee un modelo de navegación que necesite un mínimo de exploración entre páginas debido a que el usuario es guiado a través de series de tarjetas en vez de tener que desplazar de arriba a abajo en grandes páginas.

Adicionalmente, los botones 'inteligentes' son soportados por WML para elegir entre dos opciones simultaneas que pueden ser programadas para realizar acciones especificas que cambien en la aplicación cuando el programador lo necesite.

Poca Memoria y Velocidad de CPU: Los dispositivos móviles no son equipados con cantidades de memoria y poder de procesamiento (CPU[13]) en comparación con las computadoras de escritorio. La restricción de memoria es tanto en Memoria de Acceso Aleatorio (RAM[14]) como en Memoria de Solo Lectura (ROM[15]). Aun cuando la tendencia

[13] Unidad Central de Procesamiento
[14] Random Access Memory
[15] Read Only Memory

indique que se tendrá disponible más memoria y poder de procesamiento en los dispositivos en el corto plazo, la proporción en la diferencia probablemente permanecerá.

WAP ataca estas restricciones definiendo una ligera pila de protocolos adaptados para este propósito. El limitado conjunto de funcionalidad proporcionado por el WML y el WMLScript hace posible que se implementen navegadores que hagan una menor demanda de recursos tanto de memoria ROM como de poder computacional. En cuanto a memoria RAM se refiere, la codificación binaria de WML y WMScript ayuda a mantenerla tan pequeña como se pueda.

Batería de Baja Duración: El gran tropiezo en los dispositivos de comunicación inalámbrica de hoy en día es el tiempo de operación. p.e. la batería restringe su uso. A pesar de que las baterías cada vez son mejores, y las interfaces de radiofrecuencia están puestas a punto para que consuman menos batería, todavía hay mucho que conseguir en esta área.

El acceso a servicios inalámbricos incrementara la utilización de la señal del portador (interfaz de radiofrecuencia), y por lo tanto el consumo de batería también incrementará. Este problema es un tanto resuelto minimizando el ancho de banda y por consiguiente la utilización de la señal del portador tan baja como sea posible.

1.4 Funcionamiento de una red WAP

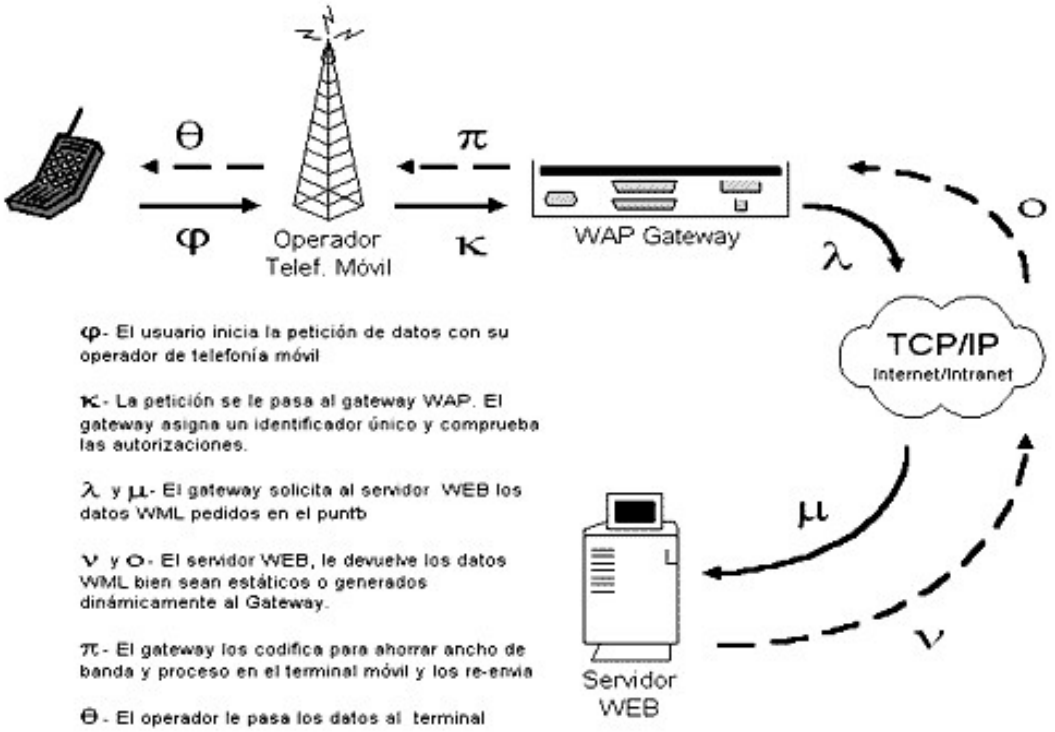


Figura 1.4 Funcionamiento de una red WAP

El funcionamiento de una red WAP se describe en la Figura 1.4, a continuación se da una breve explicación:

1. El usuario selecciona a través del dispositivo móvil un URL como si de una página web se tratara.

2. La petición es trasladada al *Gateway* WAP, utilizando para ello el protocolo WAP, el cual es independiente del servicio portador[16].

3. Cuando la petición WAP del usuario llega al *Gateway*, éste genera y envía automáticamente una petición HTTP al servidor web.

4. El servidor web procesa la petición HTTP que podría ser una página ASP[17], un CGI[18], un servlet o cualquier otra forma válida de URL estándar.

5. Como resultado del proceso anterior, el servidor web devolverá al *Gateway* WAP un archivo en formato WML.

6. El *Gateway* WAP codifica el archivo WML y, de forma binaria, lo transmite al dispositivo móvil que lo solicitó.

7. Finalmente, el dispositivo WAP interpreta el archivo WML y lo presenta en pantalla.

1.4.1 *Gateway* WAP

El *Gateway* WAP es un software que se conecta a la red de telefonía móvil y a Internet, gracias a esta doble conexión podemos acceder a todos los servicios y ventajas de la red, desde nuestro equipo móvil o desde otro dispositivo WAP. Para que esto sea posible es necesario que un operador GSM[19] administre los distintos tipos de conexiones para cada uno de los portadores que se pueden utilizar: SMS Center, CSD conexión IP, etc.

Tras realizar la conexión, podremos navegar por Internet, por una Intranet o por un WAP *Gateway* que actúa como un servidor WAP. En este último caso los contenidos a los que accedamos no estarán en la red sino en el propio WAP *Gateway*/servidor.

Actualmente existen varios *Gateway*/servidor WAP con contenidos de este tipo, Nokia, APiON, CMG, Ericsson, Phone.com o Mr. Materna son algunas de las compañías que ofrecen estos servicios. En ellos encontraremos diversa información, pasando desde

[16] Se refiere a la tecnología de telefonía móvil usada para la comunicación entre el dispositivo celular y el *Gateway* WAP.
[17] Active Server Page.- Página Activa de Servidor
[18] Common Gateway Interface.- Interfaz de Entrada Común.
[19] Global System for Mobile Communications.- Sistema Global para Comunicaciones Móviles es una tecnología de telefonía celular para la comunicación entre el dispositivo móvil y la central telefónica.

la típica información meteorológica, bolsa, noticias, etc. hasta servicio de hosting para aplicaciones, conversión HTML - WML, estructura abierta, push, etc.

1.5 Arquitectura WAP

La arquitectura WAP está pensada para proporcionar un "entorno escalable y extensible para el desarrollo de aplicaciones para dispositivos de comunicación móvil". Para ello, se define una estructura en capas, en la cual cada capa es accesible por la capa superior así como por otros servicios y aplicaciones a través de un conjunto de interfaces muy bien definidos y especificados. Este esquema de capas de la arquitectura WAP lo podemos ver en la Figura 1.5.

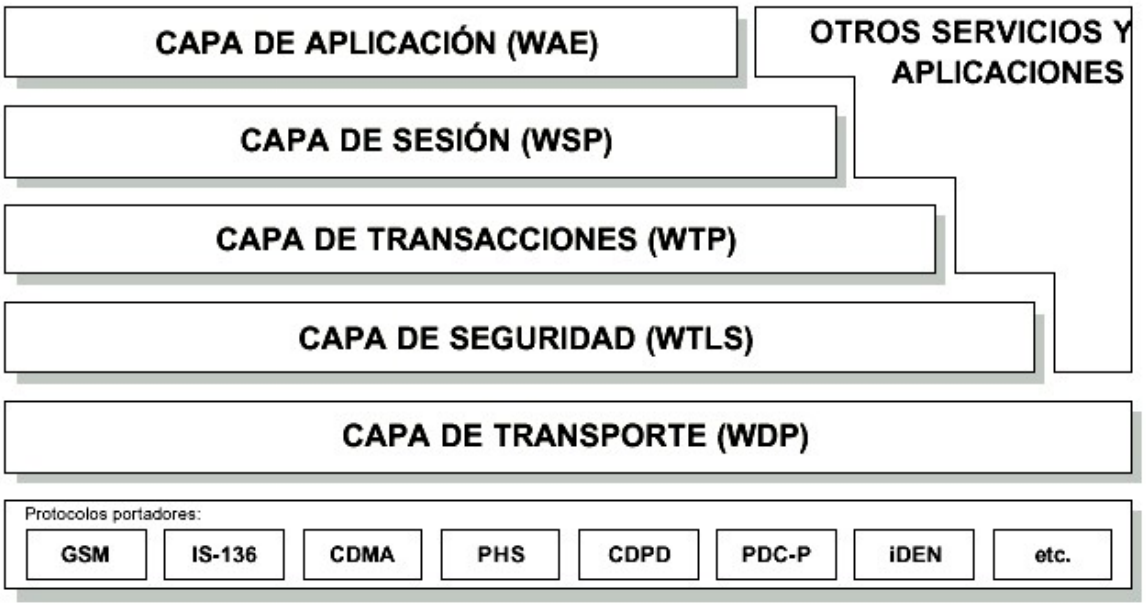


Figura 1.5 Arquitectura WAP

1.5.1 Capa de Aplicación

El Entorno Inalámbrico de Aplicación (WAE[20]) es un entorno de aplicación de propósito general basado en la combinación del World Wide Web y tecnologías de comunicación Móvil.

Este entorno incluye un mini-navegador, que posee las siguientes funcionalidades:

[20] Wireless Application Environment

- Un lenguaje denominado WML(Wireless Markup Language) similar al HTML, pero optimizado para su uso en terminales móviles.

- Un lenguaje denominado WMLScript (Wireless Markup Language Script), similar al JavaScript que permite realizar cálculos, validar datos introducidos por el usuario, acceder a funciones básicas del teléfono, entre otras.

- Un conjunto de formatos de contenido bien definidos entre los que se encuentran imágenes, entradas en la agenda de teléfonos e información de calendario.

1.5.1.1 Entorno Inalámbrico de Aplicaciones

El objetivo del Entorno Inalámbrico de Aplicaciones es construir un entorno de aplicación de propósito general, basado fundamentalmente en la filosofía y tecnología del WWW. Principalmente, se pretende establecer un entorno que permita a los operadores y proveedores de servicios construir aplicaciones y servicios que puedan utilizarse en una amplia variedad de plataformas inalámbricas de forma útil y eficiente.

De esta forma, la arquitectura del WAE está enfocada principalmente sobre los aspectos del cliente del sistema WAP, es decir, en los puntos relacionados con los agentes de usuario. Esto es debido a que la parte que más interesa de la arquitectura es aquella que afecta principalmente a las terminales móviles (aquellos puntos en los cuales van a ejecutarse los diversos agentes de usuario).

Si volvemos sobre la Figura 1.1, vemos que entre los agentes de usuario localizados en el cliente y los servidores de información se define un nuevo elemento: Las Pasarelas. Su función es codificar y decodificar la información intercambiada con el cliente, para así reducir la cantidad de datos radiados, así como minimizar el proceso de la información por parte del cliente.

Basándonos en esta arquitectura, vamos a profundizar un poco más en los componentes de este entorno. Tal y como podemos observar en la Figura 1.6, se divide en dos capas lógicas:

- Los Agentes de Usuario, que incluyen aquellos elementos como navegadores, agendas telefónicas, editores de mensajes, etc.

- Los Servicios y Formatos, que incluyen todos aquellos elementos y formatos comunes, accesibles a los Agentes de Usuario, tales como WML, WMLScript, formatos de imagen, etc.

Como se puede ver en la Figura 1.6, dentro del WAE se separan los servicios de los Agentes de Usuario, lo que proporciona flexibilidad para combinar varios servicios dentro de un único Agente de Usuario, o para distribuir los servicios entre varios Agentes de Usuario.

Los dos Agentes de Usuario más importantes son el Agente de Usuario para WML y el Agente de Usuario para WTA.

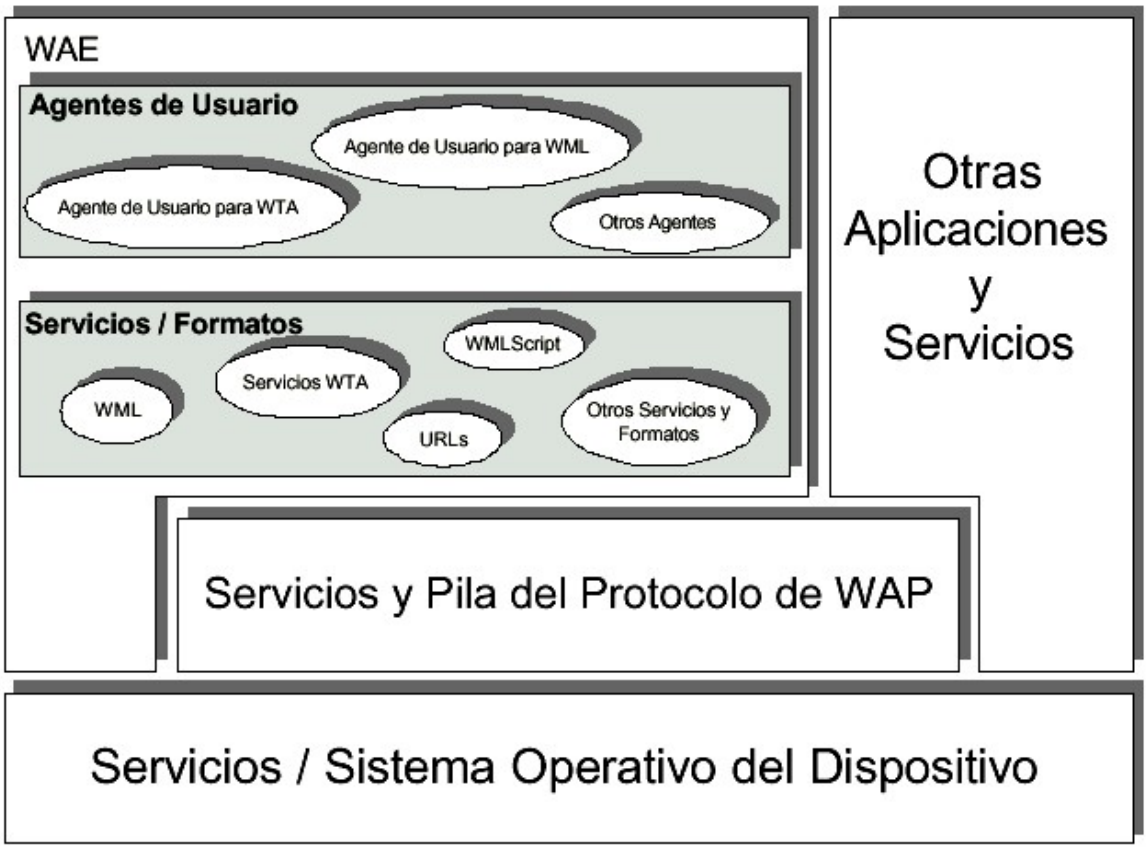


Figura 1.6 Componentes del Cliente del WAE

El Agente de Usuario para WML es el Agente de Usuario fundamental en la arquitectura del Entorno Inalámbrico de Aplicación. A pesar de su importancia, este Agente de Usuario no está definido formalmente dentro de esta arquitectura, ya que sus características y capacidades se dejan en manos de los encargados de su implementación. El único requisito de funcionalidad que debe cumplir este Agente de Usuario, es el proporcionar un sistema intérprete a los lenguajes WML y WMLScript, de forma que se permita la navegación desde el terminal móvil.

Por otra parte, el Agente de Usuario para WTA permite a los autores acceder e interactuar con las características de los teléfonos móviles (p.e. Control de Llamada), así como otras aplicaciones supuestas en los teléfonos, tales como agendas de teléfono y aplicaciones de calendario.

Los componentes más importantes del WAE son:

- Modelo de direccionamiento
- WML y WMLScript
- WTA y WTAI

A continuación se da una breve explicación de cada uno de ellos.

Modelo de direccionamiento

Como se había dicho anteriormente, WAP utiliza el mismo modelo de direccionamiento que el usado en Internet, los URLs. Un URL identifica inequívocamente un recurso. p.e. un documento WML en un servidor puede ser recuperado usando protocolos bien-conocidos.

Además de los URLs, WAP también usa los URIs. Un URI es usado para direccionar recursos que no son necesariamente accedidos usando protocolos bien-conocidos. Un ejemplo de uso de un URI es el acceso local a las funciones telefónicas del dispositivo inalámbrico.

Lenguaje de Marcado Inalámbrico (WML)

El Lenguaje de Marcado Inalámbrico es la analogía que existe en WAP con respecto al HTML del HTTP. WML está basado en el XML.

WML hace uso de un par de elementos para especificar un servicio: Baraja (Deck) y Tarjeta (Card). Una Tarjeta es una unidad de interacción con el usuario, puede ser la presentación o petición de información por parte del usuario. Una colección de tarjetas es llamada una baraja, la cual usualmente constituye un servicio. Este enfoque asegura que una cantidad adecuada de información es mostrada simultáneamente al usuario.

Algunas de las características del WML son:

- Variables
- Formateo de texto de entrada
- Soporte para imágenes
- Control de navegación
- Control del historial de navegación (history)
- Soporte para manejo de eventos (p.e. Servicios telefónicos)
- Diferentes elementos de interacción con el usuario (listas de selección, campos de entrada de texto, etc.)

WML puede ser codificado en binario por el WAP *Gateway* para ahorrar ancho de banda en el dominio inalámbrico.

WMLScript

Al igual que el WML con el HTML, el WMLScript es muy parecido a JavaScript. Puede ser usado para mejorar los servicios escritos en WML. Agrega a los servicios la lógica procedural, ciclos, expresiones condicionales y funciones computacionales.

El WMLScript puede ser usado para validar los datos tecleados por el usuario. O para acceder a las funciones del propio teléfono como son la agenda telefónica, realizar una llamada, etc.

WMLScript también soporta el uso de librerías. Estas librerías contienen funciones que extienden la funcionalidad básica del WMLScript. Esto provee medio para futuras expansiones de funciones sin tener que cambiar el núcleo del WMLScript.

Tal y como lo hace el WML, el WMLScript permite que sea codificado por el WAP *Gateway* para minimizar la cantidad de datos que son enviados por el medio inalámbrico.

Aplicaciones Telefónicas Inalámbricas (WTA)

El ambiente WTA proporciona un medio para crear servicios de telefonía usando el WAP. Como se ha mencionado con anterioridad, WTA utiliza un agente de usuario diferente al Agente de Usuario para WML. El Agente de Usuario para WTA se basa en el Agente de Usuario para WML, pero es extendido con la funcionalidad que proporcionan los servicios de telefonía. Esta funcionalidad incluye:

- **Interfaz para Aplicaciones de Telefonía Inalámbrica (WTAI).-** Una interfaz para un conjunto de funciones relacionadas con telefonía en un teléfono celular que pueden ser invocadas desde el WML o el WMLScript. Algunas de estas funciones son: manipulación de llamada, manejo de mensajes de texto y control de la libreta de direcciones.

  WTAI está dividido en tres categorías: Funciones comunes de red, funciones específicas de red y las funciones públicas. Las funciones comunes están disponibles en cualquier tipo de red, mientras que las específicas son únicas para un cierto tipo de red. Las funciones públicas son las que pueden ser invocadas desde el Agente de Usuario para WML

- **Repositorio.-** Muchos servicios WTA establecen requerimientos de manejo en tiempo real, con la implicación de que no es posible entregar contenido desde el servidor sin involucrar un cierto retardo. El Repositorio hace posible almacenar servicios WTA persistentemente en el dispositivo para ser accedidos sin necesidad de usar la red.

- **Manejo de eventos.-** Los eventos típicos en una red inalámbrica son: llamada entrante, desconexión de llamada y llamada contestada. De modo que para la creación de servicios telefónicos es posible que se tengan que manejar estos eventos. El manejo de eventos del WTA permite que servicios WTA almacenados en el repositorio sean ejecutados en respuesta a tales eventos. En WML, los eventos pueden ser ligados con una cierta acción para hacer posible el manejo de eventos dentro de un servicio.

- **Indicación de Servicio WTA.-** Un tipo de contenido que permite que el usuario sea notificado acerca de diferentes tipos de eventos(p.e. nuevo correo de voz) y darle la posibilidad de ejecutar un servicio que manipule el

evento adecuadamente. En su forma más básica, la Indicación de Servicio permite enviar un URL y un mensaje al dispositivo móvil. El mensaje es mostrado al usuario y le es cuestionado acerca de que comenzar el servicio, indicado por el URL, inmediatamente o posponerlo para un tratamiento posterior.

1.5.2 Capa de Sesión

El Protocolo Inalámbrico de Sesión (WSP[21]) proporciona a la Capa de Aplicación una interfaz con dos servicios de sesión: Un servicio orientado a conexión que funciona por encima de la Capa de Transacciones y un servicio no orientado a conexión que funciona por encima de la Capa de Transporte (y que proporciona un servicio de datagramas seguro y no seguro).

Actualmente, esta capa consiste en una gama de servicios adaptados a aplicaciones basadas en la navegación Web, proporcionando las siguientes funcionalidades:

- Semántica y funcionalidades del HTTP/1.1 en una codificación compacta.
- Negociación de las características del Protocolo.
- Suspensión de la Sesión y reanudación de la misma con cambio de sesión.

1.5.2.1 Protocolo Inalámbrico de Sesión

El Protocolo Inalámbrico de Sesión constituye la capa que se sitúa por debajo de la capa de Aplicación, proporcionando la capacidad necesaria para:

- Establecer una conexión fiable entre el cliente y el servidor, y liberar esta conexión de una forma ordenada.
- Ponerse de acuerdo en un nivel común de funcionalidades del protocolo, a través de la negociación de las posibilidades.
- Intercambiar contenido entre el cliente y el servidor utilizando codificación compacta.
- Suspender y recuperar la sesión.

Hoy por hoy, este protocolo ha sido definido únicamente para el caso de la navegación, definiéndose como WSP/B. Esta implementación está realizada para el establecimiento de una conexión sobre la base de un protocolo compatible con HTTP1.1.

De esta forma, se han definido un conjunto de primitivas de servicio para permitir la comunicación entre la capa de sesión integrada dentro del equipo cliente y la capa de

[21] Wireless Session Protocol

sesión integrada en el equipo servidor. Estas primitivas, junto con una pequeña descripción de las mismas, pueden verse en la siguiente tabla:

Tabla 1.1 Primitivas de Servicio de Sesión

Primitiva	Descripción
S-Connect	Esta primitiva se utiliza para iniciar el establecimiento de la conexión, y para la notificación de su éxito.
S-Disconnect	Esta primitiva se utiliza para desconectar una sesión, y para notificar al usuario de una sesión que esa sesión no se puede establecer, que ha sido desconectada.
S-Suspend	Esta primitiva se utiliza para solicitar la suspensión de la sesión.
S-Resume	Esta primitiva se utiliza para solicitar que se recupere la sesión utilizando para las direcciones el nuevo identificador de punto de acceso de servicio.
S-Exception	Esta primitiva se utiliza para notificar aquellos eventos que no están asignados a una transacción en particular, ni provocan la desconexión o suspensión de la sesión.
S-MethodInvoke	Esta primitiva se utiliza para solicitar una operación que deba ser ejecutada en el servidor.
S-MethodResult	Esta primitiva se utiliza para devolver una respuesta a una petición de operación.
S-MethodAbort	Esta primitiva se utiliza para abortar una solicitud de ejecución de operación, que no haya sido aún completada.
S-Push	Esta primitiva se utiliza para enviar información no solicitada desde el servidor, dentro del contexto de una sesión de forma y sin confirmación.
S-ConfirmedPush	Esta primitiva realiza las mismas funciones que la anterior, pero con confirmación.
S-PushAbort	Esta primitiva se utiliza para anular una primitiva anterior del tipo S-Push o S-ConfirmedPush.

Adicionalmente, existen cuatro tipos de cada una de las primitivas anteriores, éstos son:

Tabla 1.2 Tipos de Primitivas de Sesión

Tipo	Abreviación	Descripción
Request	Req	Se utiliza cuando una capa superior solicita un servicio de la capa inmediatamente inferior.
Indication	Ind	Una capa que solicita un servicio utiliza este tipo de primitiva para notificar a la capa inmediatamente superior de las actividades relacionadas con su par, o con el proveedor del servicio.
Response	Res	Este tipo de primitiva se utiliza para reconocer la recepción de la primitiva de tipo Indication de la capa inmediatamente inferior.
Confirm	Cnf	La capa que proporciona el servicio requerido utiliza este tipo de primitiva para notificar que la actividad ha sido completada satisfactoriamente.

Por último, resumir que cada una de estas primitivas está perfectamente definida dentro de la especificación, tanto desde el punto de vista del diagrama de tiempos en el que se tienen que invocar las primitivas, como desde el punto de vista de los parámetros intercambiados.

1.5.3 Capa de Transacciones

El Protocolo Inalámbrico de Transacción (WTP[22]) funciona por encima de un servicio de datagramas, tanto seguros como no seguros, proporcionando las siguientes funcionalidades:

- Tres clases de servicio de transacciones:
  - Peticiones inseguras de un solo camino.
  - Peticiones seguras de un solo camino.
  - Transacciones seguras de dos caminos (petición-respuesta)
- Seguridad usuario-a-usuario opcional.
- Transacciones asíncronas.

1.5.3.1 Protocolo Inalámbrico de Transacción

[22] Wireless Transaction Protocol

El Protocolo Inalámbrico de Transacción se establece para proporcionar los servicios necesarios que soporten aplicaciones de "navegación" (del tipo petición/respuesta). Es a este dúo petición/respuesta, lo que vamos a denominar como transacción. Este protocolo se sitúa por encima del Protocolo Inalámbrico de Datagramas y, de forma opcional, de la Capa Inalámbrica de Seguridad de Transporte, que serán estudiados posteriormente.

Las características de este protocolo son:

- Proporciona tres clases de servicios de transacción:
  Clase 0: mensaje de solicitud no seguro, sin mensaje de resultado.
  Clase 1: mensaje de solicitud seguro, sin mensaje de resultado.
  Clase 2: mensaje de solicitud seguro, con, exactamente, un mensaje de resultado seguro.

- La seguridad se consigue a través del uso de identificadores únicos de transacción, confirmación (Ack), eliminación de duplicados y retransmisiones.

- Seguridad opcional usuario a usuario.

- De forma opcional, la última confirmación de la transacción puede contener algún tipo de información adicional relacionada con la transacción, como medidas de prestaciones, etc.

- Se proporcionan mecanismos para minimizar el número de transacciones que se reenvían como resultado de paquetes duplicados.

- Se permiten las transacciones asíncronas.

Al igual que en el protocolo anterior (el protocolo inalámbrico de sección), en la Tabla 1.3 se verán las primitivas de servicio que sustentan la comunicación entre dos capas de transacciones situadas en dos equipos distintos:

Tabla 1.3 Primitivas de Servicio de Transacción

Primitiva	Descripción
TR-Invoke	Esta primitiva se utiliza para iniciar una nueva transacción.
TR-Result	Esta primitiva se utiliza para devolver el resultado de transacción iniciada anteriormente.
TR-Abort	Esta primitiva se utiliza para abortar una transacción existente.

Para finalizar, se nombran las principales características de este protocolo así como una breve explicación:

- **Transferencia de Mensajes.-** Dentro de este protocolo se distinguen dos tipos de mensajes: mensajes de datos y mensajes de control. Los mensajes de datos transportan únicamente datos de usuario, mientras que los mensajes de control se utilizan para los mensajes de confirmación, informes de error, etc. pero sin transportar datos de usuario.

- **Retransmisión hasta la confirmación.-** Esta característica se utiliza para la transferencia fiable de datos desde un proveedor WTP a otro, en caso que haya pérdida de paquetes. A modo de comentario, dejar claro que para reducir lo máximo posible el número de paquetes que se transmiten, este protocolo utiliza un mensaje de confirmación explícito siempre que sea posible.

- **Confirmación de Usuario.-** La Confirmación del Usuario permite al usuario confirmar cada mensaje recibido por el proveedor WTP.

- **Información en la última confirmación.-** Se permite enviar información en el último, y únicamente en el último, mensaje de confirmación de una transacción. De esta forma, se puede enviar, por ejemplo, información del rendimiento proporcionado por el sistema durante la transacción realizada, etc.

- **Concatenación y Separación.-** Podemos definir concatenación como el proceso de trasmitir múltiples Unidades de Datos del Protocolo (PDU[23]) de WTP en una Unidad de Datos del Servicio (SDU[24]) de la red portadora.

  Por el contrario, separación es el proceso de separar múltiples PDUs de un único SDU (esto es, el proceso inverso al anterior).

  El objetivo de estos sistemas es proveer eficiencia en la transmisión inalámbrica, al requerirse un menor número de transmisiones.

- **Transacciones Asíncronas.-** Para un correcto funcionamiento del protocolo, múltiples transacciones deben ser procesadas de forma asíncrona, debe ser capaz de iniciar múltiples transacciones antes que reciba la respuesta a la primera transacción.

- **Identificador de la Transacción.-** Cada transacción está identificada de forma única por los pares de direcciones de los sockets (Dirección fuente, puerto fuente, dirección destino y puerto destino) y por el Identificador de Transacción (TID[25]), el cual se incrementa para cada una de las

[23] Protocol Data Unit
[24] Service Data Unit
[25] Transaction IDentifier

transacciones iniciadas. Este número es de 16 bits, utilizándose el bit de mayor orden para indicar la dirección.

- **Segmentación y re-ensamblado (opcional).-** Si la longitud del mensaje supera la Unidad Máxima de Transferencia (MTU[26]), el mensaje puede ser segmentado por el WTP y enviado en múltiples paquetes. Cuando esta operación se realiza, estos paquetes pueden ser enviados y confirmados en grupos. De esta forma, el emisor puede realizar control de flujo cambiando el tamaño de los grupos de mensajes dependiendo de las características de la red.

1.5.4 Capa de Seguridad

La Capa de Transporte Inalámbrico Seguro (WTLS[27]) es un protocolo basado en el estándar SSL, utilizado en el entorno Web para proporcionar seguridad en la realización de transferencias de datos. Este protocolo ha sido especialmente diseñado para los protocolos de transporte de WAP y optimizado para ser utilizado en canales de comunicación de banda estrecha. Para este protocolo se han definido las siguientes características:

- **Integridad de los datos.-** Este protocolo asegura que los datos intercambiados entre el terminal y un servidor de aplicaciones no han sido modificados y no se ha convertido en información corrupta.

- **Privacidad de los datos.-** Este protocolo asegura que la información intercambiada entre el terminal y un servidor de aplicaciones no puede ser entendida por terceras partes que puedan interceptar el flujo de datos.

- **Autenticación.-** Este protocolo contiene servicios para establecer la autenticidad del terminal y del servidor de aplicaciones.

Adicionalmente, el WTLS puede ser utilizado para la realización de comunicación segura entre terminales, por ejemplo en el caso de operaciones de comercio electrónico entre terminales móviles.

1.5.4.1 Capa de Transporte Inalámbrico Seguro

La Capa de Transporte Inalámbrico Seguro (en adelante WTLS), constituye una capa modular, que depende del nivel de seguridad requerido por una determinada aplicación. Esta capa proporciona a las capas de nivel superior de WAP de una interfaz de servicio de transporte seguro, que lo resguarde de una interfaz de transporte inferior.

[26] Maximum Transfer Unit
[27] Wireless Transport Layer Security

El principal objetivo de esta capa es proporcionar privacidad, integridad de datos y autentificación entre dos aplicaciones que se comuniquen. Adicionalmente, la WTLS proporciona una interfaz para el manejo de conexiones seguras.

Al igual que hemos hecho en los protocolos anteriores, en la Tabla 1.4 se presentan las primitivas de servicio que sustentan la comunicación entre dos capas situadas en dos equipos distintos:

Tabla 1.4 Primitivas de Servicio de la Capa de Transporte Seguro

Primitiva	Descripción
SEC-Unitdata	Esta primitiva se utiliza para intercambiar datos de usuario entre los dos participantes. Sólo puede ser invocada cuando existe previamente una conexión segura entre las direcciones de transporte de los dos participantes.
SEC-Create	Esta primitiva se utiliza para iniciar el establecimiento de una conexión segura.
SEC-Exchange	Esta primitiva se utiliza en la creación de una conexión segura si el servidor desea utilizar autentificación de clave pública o intercambio de claves con el cliente.
SEC-Commit	Esta primitiva se inicia cuando el handshake (handshake es un término utilizado para denominar el intercambio de primitivas entre cliente y servidor con el objetivo de establecer una sesión segura) se completa y cualquiera de los equipos participantes solicita cambiar a un nuevo estado de conexión negociado.
SEC-Terminate	Esta primitiva se utiliza para finalizar la conexión.
SEC-Exception	Esta primitiva se utiliza para informar al otro extremo sobre las alertas de nivel de aviso.
SEC-Create-Request	Esta primitiva se utiliza por el servidor para solicitar al cliente que inicie un nuevo handshake.

Hemos hablado anteriormente del proceso de establecimiento de una sesión segura o *handshake*.

1.5.5 Capa de Transporte

El Protocolo Inalámbrico de Datagramas (WDP[28]) proporciona un servicio fiable a los protocolos de las capas superiores de WAP y permite la comunicación de forma transparente sobre los protocolos portadores válidos.

Debido a que este protocolo proporciona un interfaz común a los protocolos de las capas superiores, las capas de Seguridad, Sesión y Aplicación pueden trabajar independientemente de la red inalámbrica que dé soporte al sistema.

1.5.5.1 Protocolo Inalámbrico de Datagramas

La base de la pila de protocolos WAP es la capa de transporte (datagramas).

El Protocolo Inalámbrico de Datagramas ofrece un servicio consistente al protocolo (Seguridad, Transacción y Sesión) de la capa superior de WAP, comunicándose de forma transparente sobre uno de los servicios portadores disponibles.

Si el WAP es usado sobre un portador que soporte el Protocolo de Datagramas de Usuario (UDP[29]), la capa WDP no se necesita. Sobre otros portadores, como GSM SMS, la funcionalidad de datagrama es proporcionada por WDP. Esto significa que si WAP usa UDP o WDP, proporciona un servicio de datagramas que oculta las características de los diferentes portadores y provee de una funcionalidad de número de puerto, si es necesario WDP puede ser extendido con la funcionalidad para la segmentación y reensamble de datagramas que son demasiado grandes para el portador que se está utilizando.

Este protocolo ofrece servicios a los protocolos superiores del estilo de direccionamiento por numero de puerto, segmentación y re-ensamblado opcional y detección de errores opcional, de forma que se permite a las aplicaciones de usuario funcionar de forma transparente sobre distintos servicios portadores disponibles. Para ello, se plantea una arquitectura de protocolo como el que se muestra en la Figura 1.7:

[28] Wireless Datagram Protocol
[29] User Datagram Protocol

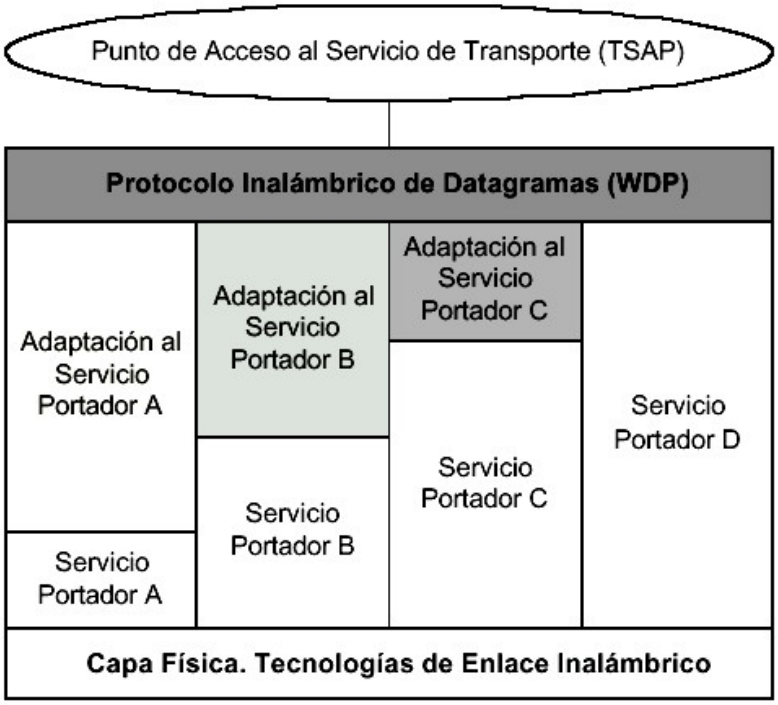


Figura 1.7 Arquitectura del WDP

En la Tabla 1.5 se presentan las primitivas de servicio que se utilizan en este protocolo:

Tabla 1.5 Primitivas de Servicio de la Capa de Datagramas

Primitiva	Descripción
T-DUnitdata	Esta primitiva es la utilizada para transmitir datos como datagramas. No requiere que exista una conexión para establecerse.
T-DError	Esta primitiva se utiliza para proporcionar información a la capa superior cuando ocurre un error que pueda influenciar en el servicio requerido.

1.5.6 Protocolos Portadores

El protocolo portador es el encargado de transmitir los datos desde el dispositivo móvil hasta la operadora de telefonía celular. WAP es totalmente independiente al portador; el cual se encarga simplemente de traer la información al equipo móvil.

El WDP se encarga de adaptar la información a uno de los portadores disponibles. Los principales portadores son:

- **Servicio de Mensajes Cortos (SMS[30]):** Su longitud es de 160 caracteres por mensaje. Teniendo en cuenta que un pequeño programa en WML puede ocupar alrededor de 1000 caracteres, necesitando por lo tanto enviar varios mensajes SMS para hacer llegar la información, utilizando a su vez muchos recursos y tiempo. El SMS no suele ser utilizado como portador WAP ya que no es el candidato perfecto como tal.

- **Datos Conmutados por Circuito (CSD[31]):** A pesar de su lentitud a la hora de realizar las conexiones, el CSD es uno de los portadores más utilizados. El CSD necesita hacer una llamada cada vez que elegimos una opción, ya que no permite mantener una conexión permanente, por lo tanto tampoco es un portador ejemplar. Hasta que no se termine de implantar el GPRS seguirá siendo el más utilizado.

- **Servicio General de Paquetes de Radio (GPRS[32]):** El portador WAP ideal. El GPRS tiene una gran capacidad como portador; realiza conexiones inmediatas y a una velocidad de transferencia rápida. El portador más utilizado en cuanto esté totalmente integrado y disponible.

1.6 Integración con la tesis

Dentro de los objetivos de esta tesis se encuentra el desarrollar una aplicación WAP por medio de la cual se proporcione al usuario final un conjunto de servicios de localización. Con el uso del lenguaje definido por WAP, WML, se puede crear la interfaz para que el usuario interactúe con la aplicación final. Por ejemplo, la aplicación debe solicitar al usuario que se autentique para poder accesar a los servicios de localización. En la Figura 1.8 se muestra esta pantalla.

Figura 1.8 Pantalla de Registro para accesar a los servicios de localización

La aplicación debe ser intuitiva y sencilla de usar, para conseguir esto se deben crear pantallas que presenten un mínimo de información a la vez. Esto se logra con el uso de Barajas (*Decks*) y Tarjetas (*Cards*) definidas en WML. En la Figura 1.9 se muestra una serie de tarjetas donde el usuario puede navegar con relativa facilidad.

[30] Short Message Service
[31] Circuit Switched Data
[32] General Packet Radio Service

La aplicación también debe tomar en cuenta las limitaciones que presentan los dispositivos celulares (explicados anteriormente). Por ejemplo, el sistema debe considerar el escaso teclado que posee un teléfono celular, por lo cual no le debe pedir al usuario que teclee el nombre de algún elemento, sino presentarle una lista donde pueda seleccionarlo fácilmente y con un mínimo de teclas presionadas. En la Figura 1.10 se muestra una pantalla donde el usuario puede elegir una delegación simplemente seleccionándola y no teniendo que teclear su nombre.

Figura 1.9 WAP define, dentro de WML, barajas y tarjetas para facilitar la interacción con el usuario

Figura 1.10 WAP mitiga las limitaciones de un celular proporcionando controles que minimizan la introducción de caracteres

1.7 Resumen

En este capítulo se presentó un trabajo de investigación acerca del Protocolo de Aplicaciones Inalámbricas (WAP). Como primer punto se tiene una breve introducción al protocolo WAP así como a las tecnologías alternas mostrando sus ventajas y desventajas entre éstas. También se llevó a cabo una revisión de los orígenes del WAP así como de la motivación que dio lugar a la creación del WAP.

Posteriormente se presentó el funcionamiento de una red WAP y se estudió la arquitectura WAP y de cada una de sus capas. Como último punto se mostró la forma en como el protocolo WAP va a ser utilizado para el desarrollo de la aplicación de esta tesis.

En el próximo capítulo se presenta de manera breve la teoría de Grafos, así como los principales conceptos que permitirán mejorar la comprensión de los algoritmos de búsqueda de caminos mínimos. Más adelante se verán las definiciones y características de los algoritmos de búsqueda de caminos mínimos.

Capítulo 2

Algoritmos de Búsqueda de Caminos Mínimos

En este capítulo se describen los algoritmos de Búsqueda de Caminos Mínimos sobre grafos.

Para esto se presenta, de manera breve, la Teoría de Grafos y algunas definiciones que permiten una mejor comprensión de los algoritmos más adelante.

Posteriormente se estudia el Encaminamiento mínimo en grafos y, por último, se presentan las definiciones y características de los Algoritmos de Búsqueda de Camino mínimo.

2.1 Introducción

El problema de encontrar la ruta más corta es un componente esencial para muchas aplicaciones incluyendo avanzados sistemas de información para viajeros, redes de computadoras discutidas por Dimitri Bertzekas [BZK98], redes de energía eléctrica en las ciudades, etcétera.

Dada la problemática que aborda esta tesis, La búsqueda de la ruta óptima es también una de las partes esenciales de la misma. En este capítulo se hace una descripción de varios de los algoritmos de búsqueda de caminos mínimos, análisis de complejidad de éstos, así como su validez (teorema).

Debido a que estos algoritmos básicamente actúan sobre grafos, también se presenta un esbozo de la Teoría de Grafos, Encaminamiento en grafos y definiciones relacionadas.

2.2 Definiciones Básicas

2.2.1 Grafos

Un grafo consiste de un conjunto V de vértices (o nodos) y un conjunto E de arcos que conectan a esos vértices. Un grafo está completamente definido por sus conjuntos de vértices y aristas [LOO85]. Existen dos tipos de grafos: Dirigidos y no dirigidos. La Figura 2.1 muestra un ejemplo de cada uno de ellos.

Un grafo no dirigido es conexo si todos sus pares de vértices distintos están conectados por un camino en el grafo [KEN90].

Un grafo dirigido o digrafo es el conjunto finito y no vacío V (de vértices) junto con un conjunto E (disjunto de V) de pares ordenados de distintos elementos de V. En este caso, se refiere a los elementos de E como arcos [BEH81].

Además de esto, los grafos pueden ser extendidos mediante la adición de rótulos a los arcos. Estos rótulos pueden representar costos, longitudes, distancias, pesos, etc.

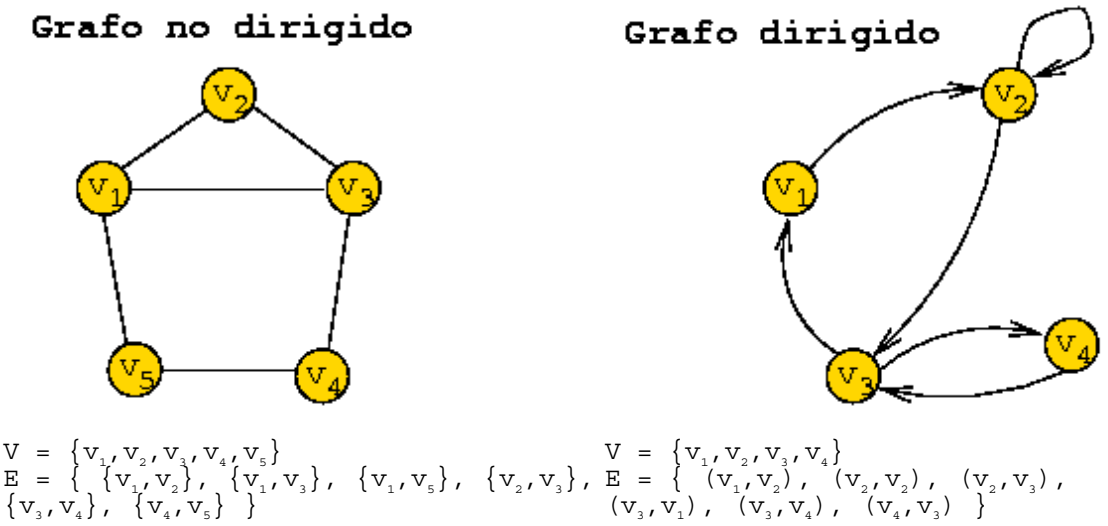


Figura 2.1 Ejemplo de un grafo no dirigido y de un dirigido

2.2.2 Caminos, Ciclos y Árboles

Un camino es una secuencia de arcos en que el extremo final de cada arco coincide con el extremo inicial del siguiente en la secuencia [KEN90].

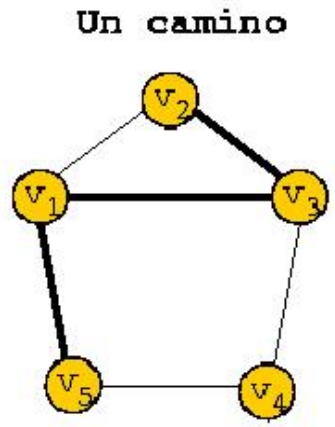


Figura 2.2 Ejemplo de camino (línea gruesa)

Un camino es simple si no se repiten vértices, excepto posiblemente el primero y el último [WWW2]. Un ejemplo de camino se muestra en la Figura 2.2.

Un ciclo es un camino simple y cerrado [WWW2]. La Figura 2.3 muestra un ejemplo de un ciclo en un grafo.

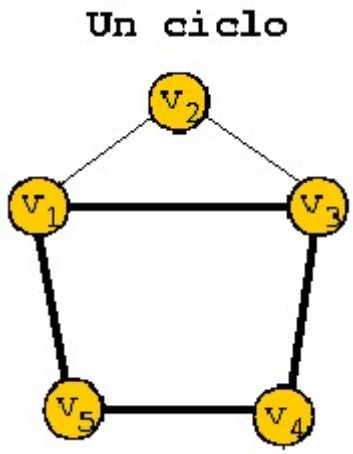


Figura 2.3 Ejemplo de ciclo (línea gruesa)

Un grafo es conexo si desde cualquier vértice existe un camino hasta cualquier otro vértice del grafo.

Se dice que un grafo no dirigido es un árbol si es conexo y acíclico [BON76]. La Figura 2.4 muestra un ejemplo de un árbol.

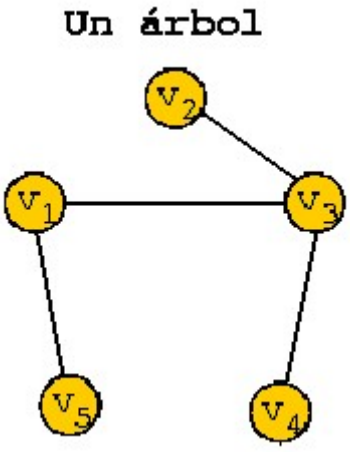


Figura 2.4 Ejemplo de árbol

2.3 Caminos Mínimos en Grafos

2.3.1 Distancia en un grafo

Sean G=(V,A) un grafo ponderado, u, v vértices de G. Se llama **distancia** de u a v, **d(u,v)**, a la mínima longitud de los caminos que unen u con v. Si no existe camino de u a v se dice que d(u,v)=∞

Propiedades: Si las aristas (o arcos, en el caso dirigido) no reciben pesos negativos entonces,

1. d(x,y)≥0 y d(x,y)=0 si y sólo si x=y
2. d(x,y)=d(y,x)
3. d(x,y)+d(y,z) ≥d(x,z)

2.3.2 Nociones relacionadas con distancia

La **excentricidad** de un vértice v es **e(v)** = máx{d(v,z)/ z∈ V(G)}

El **radio** de un grafo es **rad(G)** = mín{e(v)/ v∈ V(G)}

El **diámetro** de un grafo es **diam(G)** = máx{e(v)/ v∈ V(G)}

El **centro** de un grafo G es el subgrafo inducido por el conjunto de vértices de excentricidad mínima.

La **distancia** total de un vértice v es $dt(v) = \sum_{z \in V(G)} d(v,z)$

La **mediana** de un grafo G es el subgrafo inducido por el conjunto de vértices de distancia total mínima, un ejemplo se muestra en la Figura 2.5.

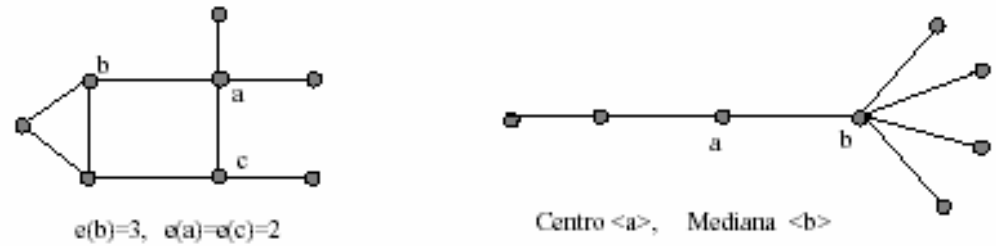


Figura 2.5 Mediana de un grafo

Propiedades:

1. Si G es un grafo conexo entonces rad(G)≤diam(G) ≤2rad(G)
2. Todo grafo es el centro de un grafo conexo
3. El centro de un árbol T está formado por uno o dos vértices de T

2.4 Encaminamiento en Grafos

El **encaminamiento** puede ser definido como un proceso mediante el cual se trata de encontrar un camino entre dos puntos de la red: el nodo origen y el nodo destino [WWW1]. En esta definición se tiene que matizar el hecho de que cuando se está hablando de *un camino* se está refiriendo a varios, el mejor o el mejor para llegar de 1 a N puntos.

El objetivo que se persigue es encontrar las mejores rutas entre pares de nodos j-k. Para ello se tiene que establecer lo que se entiende por mejor ruta y la métrica que se emplea para determinarla:

a) Mejor Ruta

Por Mejor Ruta (ó Ruta Óptima) se entiende aquella que cumple alguna de estas condiciones:

- menor retardo de tránsito,
- camino más corto (en función de una cierta métrica en función de retardo, coste de los enlaces, ...),
- máxima utilización de la capacidad de la red,
- mínima saturación máxima de la red

a) Métrica de la Red

Se presenta un par de ellas:

- Número de saltos (canales) necesarios para ir de un nodo a otro. No se comporta de forma óptima, pero si ofrece buenos resultados, y es empleada con bastante frecuencia.. La distancia (valor que se asocia a cada canal) es igual a 1 para todos los canales.

- Retardo de Transito entre nodos vecinos. En este caso la distancia se expresa en unidades de tiempo (p.e. ms), y no es constante a lo largo del tiempo sino que depende del tráfico que soporta el canal.

2.5 Algoritmos de Búsqueda del Camino Mínimo

Dado un grafo ponderado y dos vértices s y t se quiere hallar d(s,t) y el camino con dicha longitud.

Los primeros algoritmos que se presentan obtienen todos los caminos de longitud mínima desde un vértice dado s al resto de vértices del grafo. El último algoritmo resuelve el problema para un par cualquiera de vértices de G.

Si el vértice u se encuentra en un camino C de longitud mínima entre los vértices s y z entonces, la parte de C comprendida entre los vértices s y u es un camino de longitud mínima entre s y u. Por tanto, el conjunto de caminos mínimos desde s a los restantes vértices del grafo G es un árbol, llamado el **árbol de caminos mínimos** desde s.

2.5.1 Algoritmo de Dijkstra

El algoritmo de Dijkstra encuentra los caminos más cortos desde el nodo origen s a todos los demás nodos de la red, siempre y cuando no haya enlaces de longitud negativa [DJK 59] [DJK 80]. El algoritmo de Dijkstra mantiene una etiqueta de distancia d(i) para cada nodo i, que es una medida superior de la distancia del camino al nodo i. En todos los pasos intermedios el algoritmo divide a los nodos en dos grupos :

- aquellos que designa como etiquetados en forma permanente (o permanentes)
- aquellos que designa como etiquetados en forma transitoria (o temporales)

La etiqueta de distancia a un nodo permanente representa la distancia más corta desde la fuente al nodo. Para un nodo temporal, la etiqueta de distancia representa una cota superior a la distancia del camino más corto a ese nodo.

La idea básica del algoritmo es partir del nodo s y permanentemente etiquetar nodos de acuerdo a su distancia a s. Inicialmente d(s)=0 y d(i)=μ para todo i, s. En cada iteración d(i) es la distancia más corta desde la fuente al nodo a través de un camino que utilice sólo nodos permanentes (además del nodo i y el nodo s). El algoritmo selecciona un nodo i con etiqueta temporaria mínima, lo hace permanente y actualiza las etiquetas de sus nodos adyacentes que aún sean temporarios. El algoritmo termina cuando todos los nodos fueron designados permanentes.

Se describe el funcionamiento del algoritmo en otras palabras: Si P es un camino de longitud mínima s--z y P contiene al vértice v, entonces la parte s--v de P es también camino de longitud mínima de s a v. Esto sugiere que si se desea determinar el camino óptimo de s a cada vértice z de G, se puede hacer en orden creciente de la distancia d(s,z).

Descripción del algoritmo

Entrada: Un grafo ponderado, un vértice $s \in V$. El peso de la arista uv se indica por w(uv), poniendo $w(uv)=\infty$ si uv no es arista. (Las aristas tienen pesos no negativos)

Clave: Mantener el conjunto T de vértices para el que se conoce el camino más corto y ampliar T hasta que T=V. Para ello se etiqueta cada vértice z con t(z) que es la longitud del camino más corto ya encontrado.

Inicialización: Sea $T=\{s\}$, $t(s)=d(s,s)=0$, $t(z)=w(sz)$ para $z \neq s$.

Iteración: Elegir el vértice $v \in T$ con etiqueta mínima. Añadir v a T
Analizar cada arista vz con $z \in T$ y actualizar la etiqueta de z a
$\min\{t(z), t(v)+w(vz)\}$
La iteración continua hasta que T=V(G) o hasta que $t(z)=\infty$
para cada vértice $z \notin T$

En cualquier caso la etiqueta de cada vértice z en T será la distancia de s a z. En el segundo caso los vértices que no están en T no son accesibles desde s.

Análisis de la complejidad

En cada iteración se añade un vértice a T, luego el número de iteraciones es n. En cada una se elige una etiqueta mínima, la primera vez entre n-1, la segunda entre n-2, ..., luego la complejidad total de estas elecciones es $O(n^2)$. Por otra parte cada arista da lugar a una actualización de etiqueta, que se puede hacer en tiempo constante O(1), en total pues O(q). Por tanto la complejidad del algoritmo es $O(n^2)$.

Teorema (Validez del algoritmo)

El algoritmo de Dijkstra calcula d(s,z) para cada vértice $z \in V(G)$

Demostración

Se debe probar que la etiqueta definitiva t(z) es d(s,z). Sean $x_1, x_2, ..., x_n$ los vértices de G ordenados por su incorporación al conjunto T. Así $x_1=s$. Se demuestra el resultado por inducción sobre i.

Primer paso.- El resultado es cierto para i=1, pues x1=s y se sabe que d(s,s)=0=t(s)

Paso de i a i+1.- La hipótesis de inducción es que $t(x_1)=d(s,x_1)$, ..., $t(x_i)=d(s,x_i)$. Se debe probar que $t(x_{i+1})=d(s,x_{i+1})$

Llamemos $S=\{x_1, x_2, ..., x_i\}$, La etiqueta $t(x_{i+1})$ es, por la construcción del algoritmo, la longitud de un camino Q s,..., u, x_{i+1}, donde u es el último vértice en S y $e=ux_{i+1}$ es una arista de G. Si hay otro camino Q' de s a x_{i+1} debemos probar que $long(Q) \leq long(Q')$.

Sea z el primer vértice de Q' fuera de S, vz la primera arista y Q'' el resto del camino de z a x_{i+1}

$long(Q') = d(s,v) + w(vz) + long(Q'') \geq t(z) + long(Q'')$

y como x_{i+1} se elige como vértice de menor etiqueta será $t(z) \geq t(x_{i+1})$ y así $long(Q') \geq t(x_{i+1}) + long(Q'') \geq t(x_{i+1})$ por ser todas las aristas de peso no negativo. Por tanto $t(x_{i+1}) = long(Q) = d(s,x_{i+1})$

En la Figura 2.6 se muestra un ejemplo de la demostración del Algoritmo.

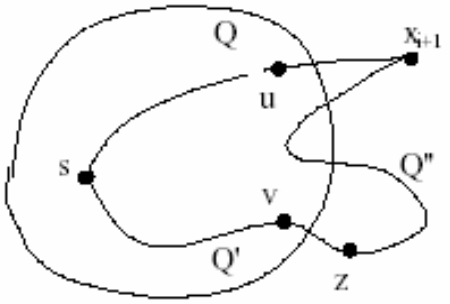


Figura 2.6 Grafo de la demostración del Algoritmo de Dijkstra

Ejemplo

A continuación se presenta un ejemplo para mostrar el funcionamiento del algoritmo de Dijkstra. Se tiene el siguiente grafo:

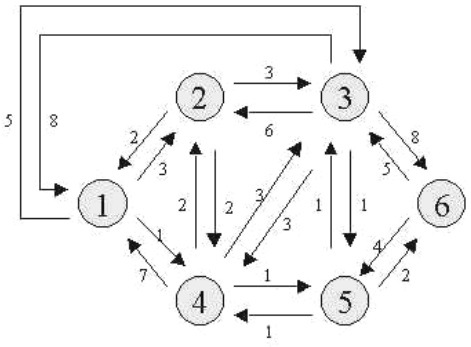


Figura 2.7 Algoritmo de Dijkstra (Ejemplo) - Grafo inicial

Para el nodo origen s = 1, se tiene que el nodo con la arista de menor peso es 4 (v=4), se agrega este nodo a T.

Como T no contiene a todos los nodos se calculan las siguientes distancias:

Nodo 3: *min*{5, 1+3} = 4
Nodo 6: *min*{∞, ∞} = ∞
Nodo 5: *min*{∞, 1+1} = 2

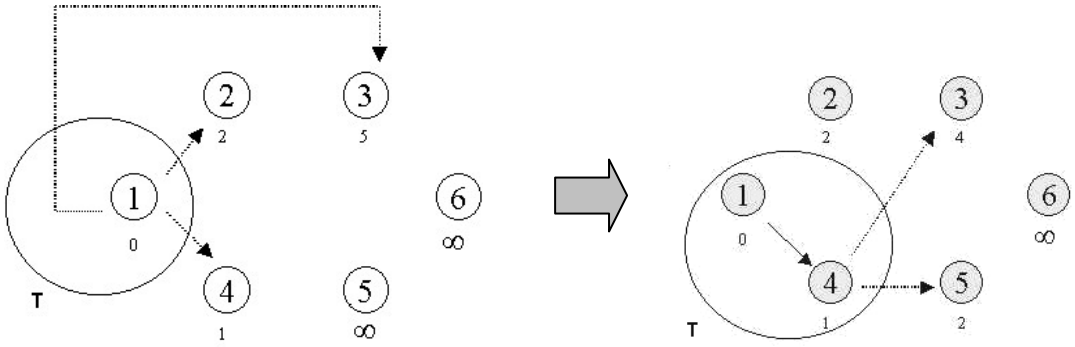


Figura 2.8 Algoritmo de Dijkstra (Ejemplo) – Elección el nodo con la arista de menor peso (4)

Se obtiene un empate entre los nodos 2 y 5, se elige el nodo 2 al azar (v=2) y se agrega a T, posteriormente se hace lo mismo con v=5 y v=3 (Figura 2.9).

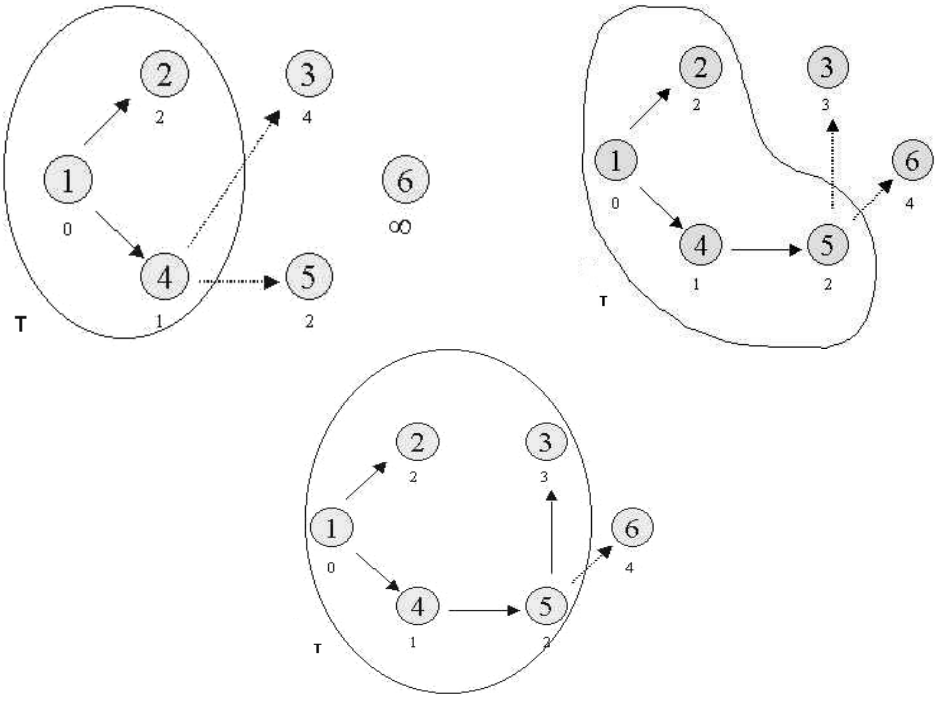


Figura 2.9 Algoritmo de Dijkstra (Ejemplo) – Se agregan los nodos 2, 5 y 3 al árbol de caminos mínimos

La Figura 2.10 se muestra el árbol final de encaminamiento a partir del nodo 1.

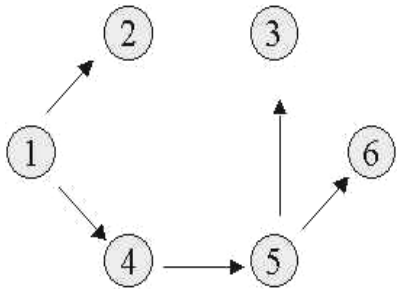


Figura 2.10 Algoritmo de Dijkstra (Ejemplo) – Árbol final de encaminamiento del nodo 1 al resto de los nodos de la red

2.5.2 Algoritmo de Bellman – Ford

Es una variante del algoritmo de Dijkstra que admite la asignación de pesos negativos en los arcos, aunque no permite la existencia en el grafo de ciclos de peso negativo.

Encuentra la mínima distancia de un nodo dado al resto de los nodos, y si se lleva información adicional, proporciona las tablas de encaminamiento.

Itera sobre el número de saltos, es decir, se busca el mejor camino, el de distancia más corta, con la restricción de llegar al destino en un número de saltos (número de la iteración). No encuentra las mejores rutas hasta que el algoritmo no se ha ejecutado por completo.

Es decir, consiste fundamentalmente en calcular la distancia de un nodo (aquel para el cual aplicamos el algoritmo) a los demás sin dar ningún salto, luego dando uno, dos, tres, etcétera.

Descripción del algoritmo

Entrada: Un grafo ponderado con pesos no negativos en los arcos, un vértice $s \in V$. El peso del arco uv se indica por w(uv), poniendo $w(uv)=\infty$ si uv no es arco.

Salida: La distancia desde s a cada vértice del grafo

Clave: Mejorar en cada paso las etiquetas de los vértices, t(u)

Inicialización: Sea T={s}, t(s)=d(s,s)=0, t(z)=w(sz) para $z \neq s$.

Iteración: Mientras existan arcos e=xz para los que t(z)>t(x) + w(e) actualizar la etiqueta de z a min{t(z), t(x)+w(xz)}

Análisis de la complejidad

En primer lugar se debe observar que cada arco puede considerarse varias veces. Se comienza ordenando los arcos del grafo D siendo éste el orden en que se considerarán los arcos en el algoritmo. Después de la primera pasada se repite el proceso hasta que en una pasada completa no se produzca ningún cambio de etiquetas. Si D no contiene ciclos negativos puede demostrarse que, si el camino mínimo s → u contiene k arcos entonces, después de k pasadas se alcanza la etiqueta definitiva para u. Como k≤n y el número de arcos es q, resulta que la complejidad del algoritmo de Bellman - Ford es O(qn). Además podemos detectar un ciclo negativo si se produce una mejora en las etiquetas en la pasada número n.

Ejemplo

A continuación se presenta un ejemplo para mostrar el funcionamiento del Algoritmo de Bellman – Ford. Se tiene el siguiente grafo:

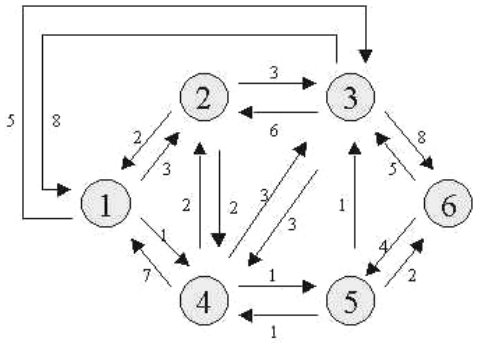


Figura 2.11 Algoritmo de Bellman - Ford (Ejemplo) - Grafo inicial

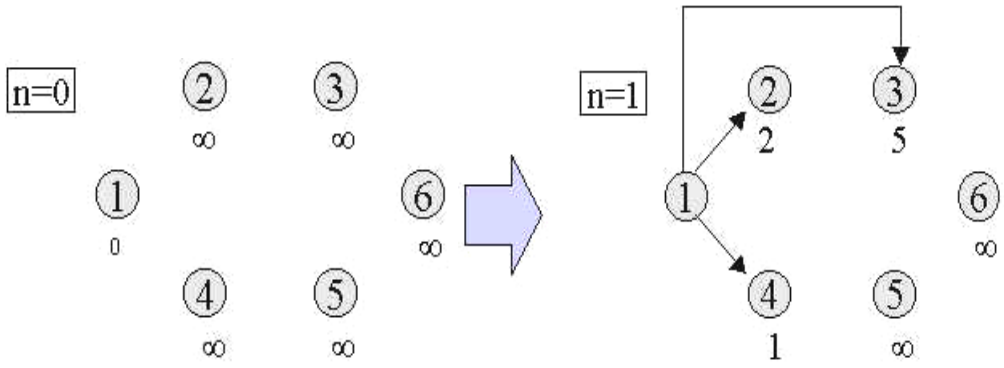


Figura 2.12 Algoritmo de Bellman - Ford (Ejemplo) – Grafo sin dar saltos y dando un salto

Se puede ver en las figuras 2.12 y 2.13 que cuesta más llegar al nodo 3 con un salto (distancia = 5) que con dos saltos (distancia = 4).

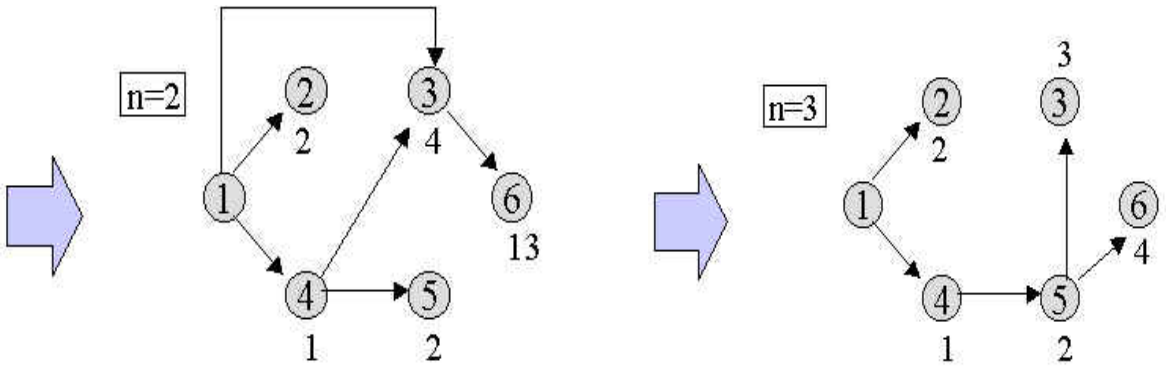


Figura 2.13 Algoritmo de Bellman - Ford (Ejemplo) – Grafo dando dos y tres saltos

No se realizan más saltos ya que no aportan mejora alguna. Se prueba dando un salto más y verificando que las distancias que se tenían antes resultan mejores. Entonces para n=3 se tiene el árbol final de encaminamiento del nodo 1 a los demás.

2.5.3 Algoritmo de Floyd - Warshall

A veces no es suficiente calcular las distancias con respecto a un vértice s, si no que necesitamos conocer la distancia entre cada par de vértices. Para ello se puede aplicar reiteradamente alguno de los algoritmos anteriores, variando el vértice s de partida. Así tendríamos algoritmos de complejidad $O(n^3)$ (si usamos el algoritmo de Dijkstra) u $O(n^2q)$ (si usamos el algoritmo de Bellman - Ford). A continuación se describe un algoritmo, creado por Floyd y Warshall, con una estructura sencilla, que permite la presencia de arcos de peso negativo y que resuelve el mismo problema. (Naturalmente los ciclos de peso negativo siguen estando prohibidos).

La idea básica del algoritmo es la construcción de una sucesión de matrices W_0, W_1, ..., W_n, donde el elemento ij de la matriz W^k nos indique la longitud del camino mínimo $i \rightarrow j$ utilizando como vértices interiores del camino los del conjunto $\{v_1, v_2, ..., v_k\}$. La matriz W^0 es la matriz de pesos del grafo, con $w^0_{ij}=w(ij)$ si existe el arco $i \rightarrow j$, $w^0_{ii}=0$ y $w^0_{ij}=\infty$ si no existe el arco $i \rightarrow j$.

Descripción del algoritmo

Entrada: Un grafo ponderado sin ciclos de peso negativos. El peso del arco uv se indica por w(uv), poniendo $w(uv)=\infty$ si uv no es arco.

Salida: La distancia entre dos vértices cualesquiera del grafo

Clave: Construimos la matriz W^k a partir de la matriz W^{k-1} observando que

$$w_{ij}^{k} = min\{w_{ij}^{k-1}, w_{ik}^{k-1} + w_{kj}^{k-1}\}$$

Iteración: Para cada k =1, ..., n, hacer

$$w_{ij}^{k} = min\{w_{ij}^{k-1}, w_{ik}^{k-1} + w_{kj}^{k-1}\} \quad \forall i,j=1,...,n$$

El elemento ij de la matriz W^n nos da la longitud de un camino mínimo entre los vértices i y j.

Análisis de la complejidad

Se deben construir n matrices de tamaño nxn y cada elemento se halla en tiempo constante. Por tanto, la complejidad del algoritmo es $O(n^3)$

Teorema (Validez del algoritmo)

La indicación dada en la clave se puede demostrar por inducción sobre k, con lo que tendríamos demostrada la validez del algoritmo.

Observaciones:

1. Si existe un ciclo negativo entonces algún elemento de la diagonal principal se hará negativo en alguna de las matrices W^k .

2. Si además de las longitudes de los caminos mínimos se desean obtener dichos caminos basta construir una sucesión auxiliar de matrices P^0, P^1,..., P^n de la siguiente forma:

- El elemento ij de la matriz P0 es j
- Si en el elemento ij de la matriz Wk no se produce cambio entonces pkij coincide con el elemento correspondiente de la matriz Pk-1. Si hay cambio entonces pkij = pk-1ik
- Así el elemento ij de la matriz Pn indica el primer vértice después de vi en el camino de longitud mínima que conecta los vértices vi y vj. Y con esto resulta fácil reconstruir todo el camino.

Ejemplo

A continuación se presenta un ejemplo para mostrar el funcionamiento del Algoritmo de Floyd – Warshall. Se tiene el siguiente grafo:

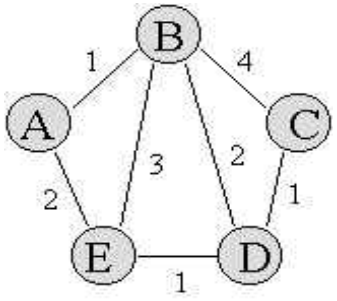


Figura 2.14 Algoritmo de Floyd - Warshall (Ejemplo) - Grafo inicial

Entonces se tiene la secuencia de matrices de la Figura 2.15.

$$\begin{bmatrix} 0 & 1 & - & - & 2 \\ 1 & 0 & 4 & 2 & 3 \\ - & 4 & 0 & 1 & - \\ - & 2 & 1 & 0 & 1 \\ 2 & 3 & - & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & - & - & 2 \\ 1 & 0 & 4 & 2 & 3 \\ - & 4 & 0 & 1 & - \\ - & 2 & 1 & 3 & 1 \\ 2 & 3 & - & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 5 & 3 & 2 \\ 1 & 0 & 4 & 2 & 3 \\ 5 & 4 & 0 & 1 & 7 \\ 3 & 2 & 1 & 0 & 1 \\ 2 & 3 & 7 & 1 & 0 \end{bmatrix} \rightarrow$$

$$\rightarrow \begin{bmatrix} 0 & 1 & 5 & 3 & 2 \\ 1 & 0 & 4 & 2 & 3 \\ 5 & 4 & 0 & 1 & 7 \\ 3 & 2 & 1 & 0 & 1 \\ 2 & 3 & 7 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 4 & 3 & 2 \\ 1 & 0 & 3 & 2 & 3 \\ 4 & 3 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 2 & 3 & 2 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 4 & 3 & 2 \\ 1 & 0 & 3 & 2 & 3 \\ 4 & 3 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 2 & 3 & 2 & 1 & 0 \end{bmatrix}$$

Figura 2.15 Algoritmo de Floyd - Warshall (Ejemplo) – Secuencia de matrices

La segunda matriz tiene el nodo A como nodo intermedio, la tercera matriz tiene el nodo B como nodo intermedio, ... y así hasta la última matriz que tiene el nodo E como nodo intermedio y da como resultado la matriz de distancias mínimas buscada.

La última matriz es la matriz de distancias buscada, ya que se han probado todos los nodos intermedios.

2.6 Integración con la tesis

Como se planteó en los objetivos de esta tesis, dentro de los servicios de localización que proporcionará la aplicación está la búsqueda de rutas más cortas para trasladarse de un punto a otro utilizando el algoritmo de Dijkstra. A continuación se presenta la forma en como será aplicado este algoritmo en el desarrollo de la aplicación.

A partir de la red vial de una ciudad con sus diferentes elementos como son: delegaciones, colonias, calles, avenidas, ejes, etcétera. se obtienen los elementos para crear el grafo al cual se le aplica el algoritmo de Dijkstra para encontrar la ruta más corta para llegar de un punto origen a un punto destino. Por ejemplo, para el fragmento de red vial de la Ciudad de México que se muestra en la Figura 2.16 se obtendría el grafo que se muestra en la Figura 2.17.

Figura 2.16 Fragmento de la red vial de la Ciudad de México

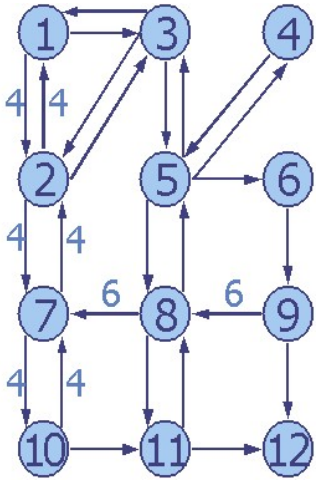


Figura 2.17 Grafo correspondiente a la red vial de la Figura 2.16

Se pueden ver en la Figura 2.16, los nodos que contendrá el grafo (numerados), estos nodos están definidos por las intersecciones de las principales avenidas y calles ingresadas en el sistema (en capítulos posteriores se verá más a detalle el ingreso de elementos al sistema así como la generación del grafo a partir de éstos). En la Figura 2.17 se pueden ver los grafos numerados de acuerdo a la Figura 2.16, así como los sentidos de las calles y de los pesos o costos para llegar de un nodo a otro. En este caso el costo se refiere a la cantidad de carriles que tiene el fragmento de esa calle.

El grafo de la Figura 2.17 contiene todos los elementos necesarios para que el algoritmo de Dijkstra pueda ser aplicado y así obtener las rutas más cortas para llegar de un nodo a otro.

2.7 Resumen

En este capítulo se realizó un análisis de los algoritmos de búsqueda de caminos mínimos. Inicialmente fue presentada la teoría de grafos así como algunas definiciones relacionadas con el tema. Más adelante se estudió el encaminamiento en grafos que se de buscar la mejor ruta entre dos puntos.

Posteriormente fueron expuestos los principales algoritmos de búsqueda de caminos mínimos en grafos. Para cada uno de estos algoritmos se presentó una breve introducción, descripción, análisis de complejidad, teorema, demostración y un ejemplo.

Para el final del capítulo se presenta cual algoritmo fue elegido y la forma en que será implementado para el desarrollo de la aplicación de esta tesis.

En el siguiente capítulo se presenta una descripción del problema planteado, así como la arquitectura y análisis de la aplicación de esta tesis.

Capítulo 3

Arquitectura y Análisis de la Aplicación

En este capítulo se realiza una descripción del problema planteado y la forma en cómo es resuelto.

En primer lugar se da una descripción general de la aplicación desarrollada, en la cual se detallan los requerimientos y funcionalidad que debe cumplir.

Después se explica la arquitectura del sistema a manera de bloques, explicando cada uno de los elementos que lo componen.

Posteriormente se presentan los casos de uso que definen la funcionalidad que debe tener la aplicación desarrollada.

3.1 Introducción

En este capítulo se realiza una descripción detallada del problema que se plantea en el presente trabajo, así como las necesidades y requerimientos que tiene que satisfacer la solución propuesta.

A partir de la descripción hecha al inicio del capítulo se propone una Arquitectura general. En dicha arquitectura se definen cada uno de los módulos que conforman al sistema que da solución al problema planteado.

Paso seguido se presenta la fase de Análisis. En dicha fase se lleva a cabo el razonamiento y estudio de cada uno de los requerimientos del sistema con la finalidad de que posteriormente, en la siguiente fase, se realice un buen diseño del mismo.

3.2 Descripción General

Se construye una aplicación computacional que sea capaz de proporcionar una gama se servicios basados en la localización geográfica haciendo uso de una infraestructura de telefonía celular.

Estos servicios podrán ser accedidos desde cualquier teléfono celular que se encuentre dentro de una zona con cobertura celular digital y que además cuente con tecnología WAP.

Dicha aplicación consta de dos módulos principales: Uno de ellos es la aplicación WAP (*frontend*), a ésta es a la acceden los usuarios de telefonía celular a través de Internet Móvil, y es la que proporciona los servicios de localización que más adelante se verán a detalle. Por otro lado está una aplicación (*backend*) que es usada para dar de alta información al sistema así como mantenimiento a la misma. Es decir que, esta aplicación alimentará de información al *frontend* para que pueda prestar los servicios de localización al usuario final.

Posteriormente se dará una explicación a detalle de las funciones que deben realizar cada uno de estos módulos. A continuación se presenta la arquitectura del sistema a dos niveles de profundidad.

3.3 Arquitectura

En la Figura 3.1 se presenta la arquitectura de nivel 0 del sistema que se desarrolla en esta tesis. Los módulos del *frontend* principalmente están implementados dentro del Servidor Web o de Aplicaciones, aunque son transmitidos por Internet, el gateway WAP y la red celular. Los módulos del *backend* están implementados en la PC. Ambos grupos de módulos acceden a los servicios implementados en el Depósito de Datos principalmente para la recuperación y mantenimiento de la información.

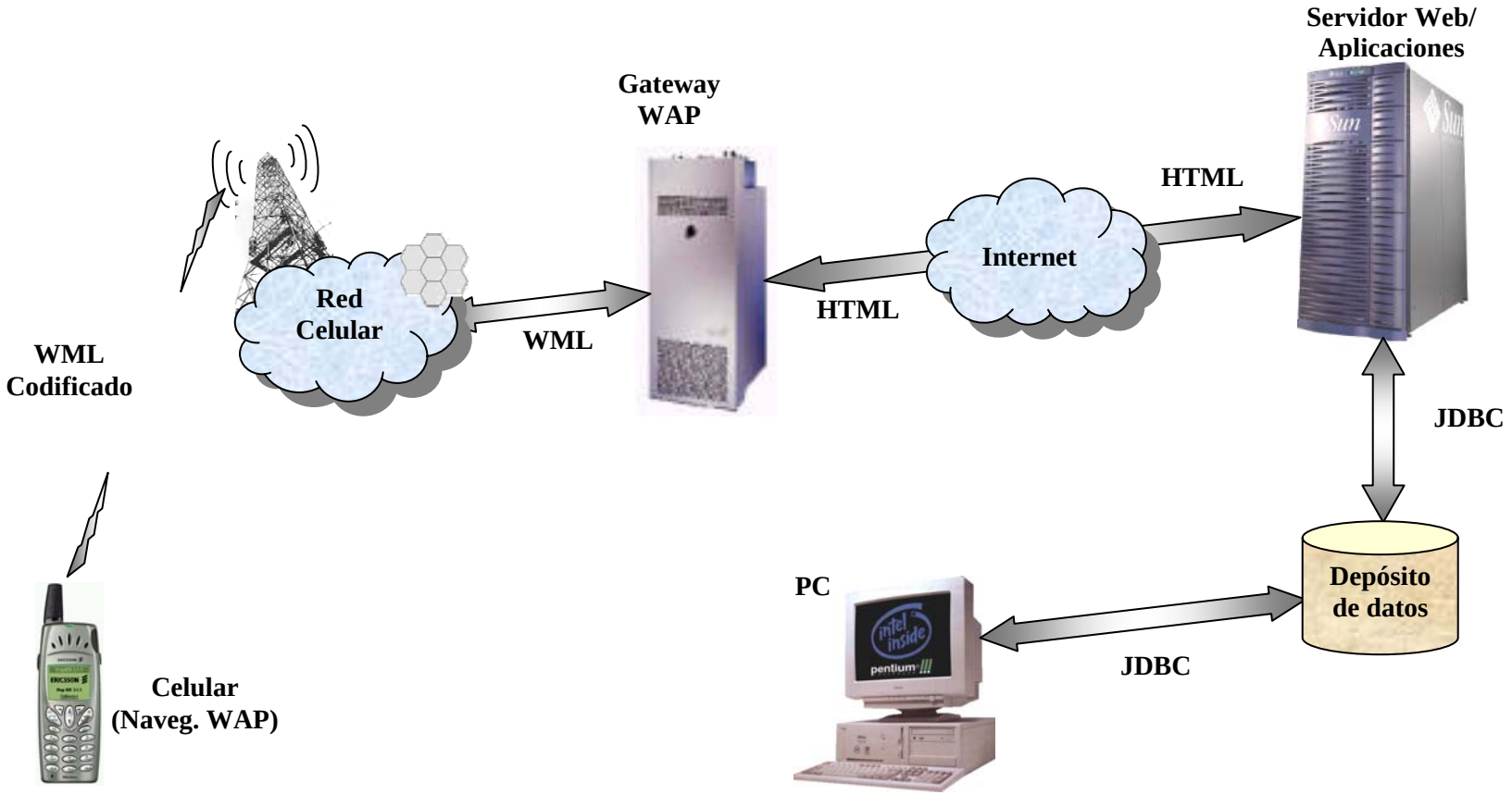


Figura 3.1 Arquitectura del sistema (Nivel 0)

La arquitectura de nivel 0 del sistema representa la manera en como los usuarios, tanto del *Frontend* como del *Backend*, acceden a un conjunto de servicios con el objetivo de consultar y/o modificar la información del sistema. En el caso del *frontend*, los servicios son accedidos a través de un URL[1] (Localizador Uniforme de Recursos).

La arquitectura general del sistema propuesto está basada en el modelo WAP. El modelo WAP hace uso del paradigma de Internet para proveer una plataforma de servicio más flexible, una plataforma que permita adaptar el acceso inalámbrico (celular) al espacio de información que ofrece el WWW. WAP está basado en una tecnología de Internet bien conocida que ha sido optimizada para atacar las limitaciones de los ambientes inalámbricos.

Volviendo a la Figura 3.1 se puede ver que existen dos maneras en que los usuarios pueden interactuar con el sistema:

En el caso de que el usuario final requiera hacer una consulta tiene que teclear, en el navegador del dispositivo móvil, un URL solicitando algún servicio, dicha petición viaja a través de la red celular hasta que un *gateway* WAP la recibe.

Es el *gateway* WAP el que se encarga de decodificar y convertir la petición WAP a HTTP y dirigirla hacía el servidor Web por medio de Internet, el servidor Web es el encargado de dar respuesta y entregar la información solicitada. Él se encarga de ejecutar servicios de BD, acceder a tablas y demás información con el fin de generar las páginas que serán enviadas de nuevo al dispositivo móvil. En este caso, todas las conexiones a BD las realiza a través de JDBC.

Una vez que ya se tienen las páginas formadas con la información solicitada, el servidor Web manda una respuesta HTTP a través de Internet, dichos objetos llegan hasta el gateway WAP. Éste último se encarga de codificar esta información y enviarla por la red celular hacía el dispositivo que solicitó el servicio. El navegador WAP del dispositivo despliega la información y es de esta manera como el usuario ve los resultados.

La otra forma de interacción con el sistema es para dar mantenimiento a la información que se usa para prestar los servicios de localización. Esto se hace a través de una PC que pueda acceder directamente a la BD, la conexión a BD también se realiza por JDBC. El usuario-operador tiene varias opciones para dar mantenimiento a la estructura vial del sistema en general, así como también a los sitios de interés.

[1] Uniform Resource Locator

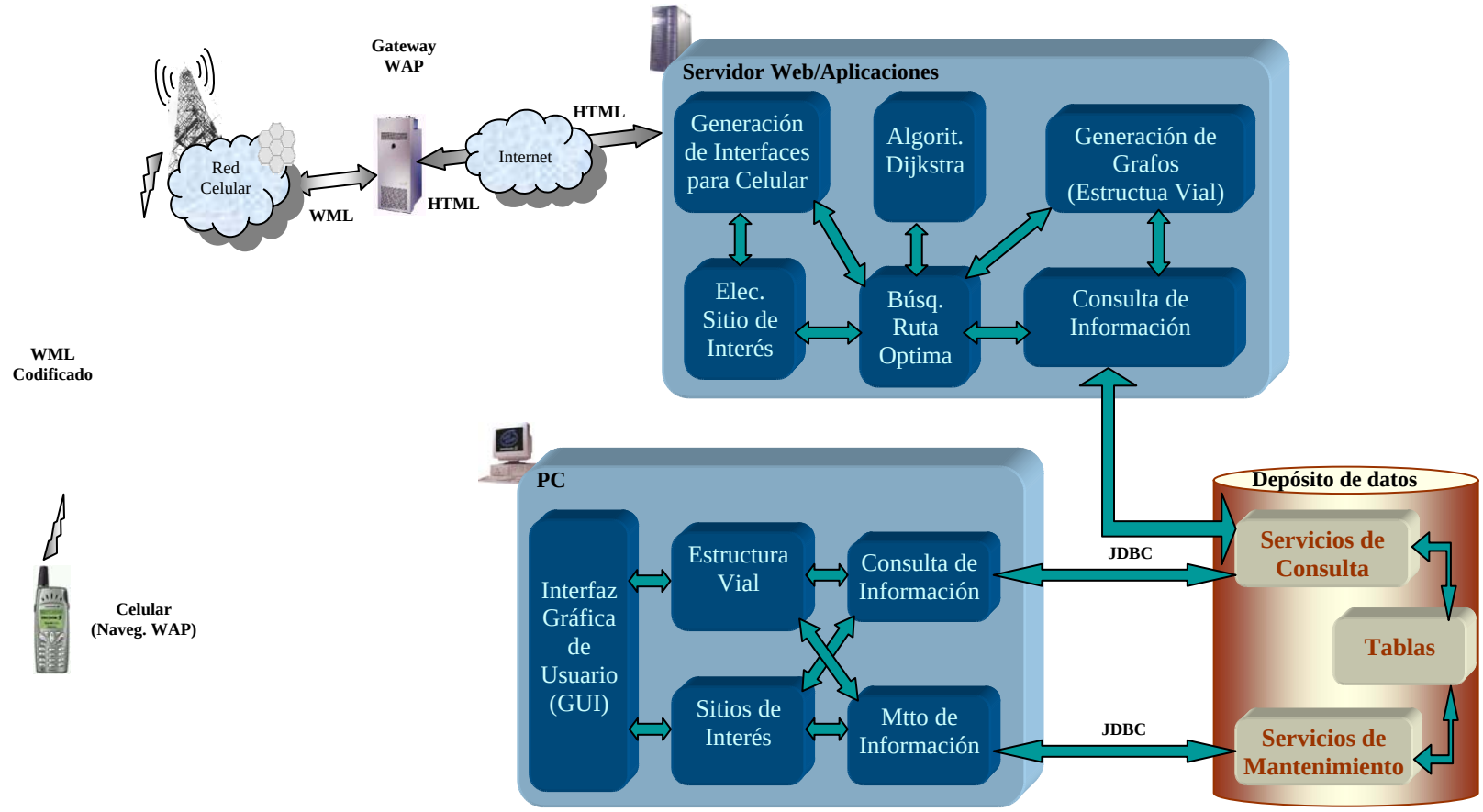


Figura 3.2 Arquitectura del sistema (Nivel 1)

En la figura 3.2 se muestra un siguiente nivel de la arquitectura del sistema, el nivel 1. En ésta se presentan cada uno de los módulos (componentes, servicios de BD, tablas, procesos, algoritmos, etc.) que constituyen al sistema. Estos módulos se encuentran principalmente en tres diferentes sitios dentro de la arquitectura de nivel 0.

3.3.1 Módulos del Servidor Web

El primer conjunto de módulos los encontramos en el **Servidor Web** (o Servidor de Aplicaciones). A continuación se muestra un listado de cada uno de éstos, seguido de una breve descripción:

3.3.1.1 Generación de Interfaces para Celulares

Este módulo es el que provee toda la funcionalidad para que el usuario interactúe con el sistema de una forma amigable y eficiente. Es decir que, dicha interacción fue pensada tomando en cuenta las características y limitaciones que tienen los dispositivos celulares (presentadas en el capítulo 1). Este módulo incluye un mecanismo que permita continuar una búsqueda pendiente debido a una desconexión del medio de comunicación.

3.3.1.2 Elección de Sitio de Interés

Este módulo permite al usuario realizar una búsqueda de uno o más sitios de interés, por medio de una serie de consultas. Este módulo es utilizado cuando el usuario desea conocer la ruta óptima de un punto de partida (u origen) hacia un sitio de interés del que desconoce su dirección. Para lograr lo anterior, este módulo proporciona un conjunto de opciones para realizar la búsqueda de un sitio en particular y así obtener la dirección destino requerida.

3.3.1.3 Búsqueda de Rutas Óptimas

Este módulo es el encargado de integrar al resto de los módulos que se encuentran en el Servidor Web, su función es la de ejecutar y sincronizar todas las etapas necesarias para proporcionar, al usuario del servicio, la ruta óptima entre dos puntos dados o hacía el sitio de interés más cercano.

3.3.1.4 Algoritmo de Dijkstra

Es la implementación computacional del Algoritmo de Dijkstra para encontrar las rutas más cortas (óptimas) dentro de un grafo ponderado. Pero en general puede representar la estructura vial de una ciudad (calles, avenidas, ejes, etc.). Este modulo es el que se tendrá que aplicar dentro del sistema y que nos arrojará como resultado una lista ordenada de calles que conformará la ruta óptima en ese momento, ya que depende de las condiciones de tráfico y el estado actual de la vialidad.

3.3.1.5 Generación de Grafos (Estructura vial)

El sistema realiza una manipulación de los datos recuperados de la base de datos para construir un conjunto de estructuras de datos que representan al grafo. El sistema realiza una serie de cálculos para la ponderación del grafo, esto en base a una serie de condiciones y características de la calles, como son: Longitud de la calle, si existe un bloqueo, algún accidente vial, manifestación y condiciones de tráfico en general.

3.3.1.6 Consulta de Información (Acceso a BD)

Provee de una interfaz de comunicación con la base de datos, en específico con los servicios de consulta de información. Este módulo proporciona un mecanismo para establecer conexiones con BD, crear sentencias de SQL[2], ejecutar servicios de recuperación de información, liberar recursos, etcétera.

3.3.2 Módulos de la Computadora (PC)

Otro conjunto de módulos los encontramos del lado de la **Computadora**, en la cual se tiene la aplicación que empleará el usuario operador para manipular la información de la estructura vial y los sitios de interés. A continuación se presenta una breve descripción de cada módulo de este conjunto:

3.3.2.1 Interfaz Gráfica de Usuario (GUI[3])

La Interfaz Gráfica de Usuario proporciona un medio de interacción y comunicación del usuario operador con esta parte del sistema. La GUI está implementada con el lenguaje Java. Por las mismas características que ofrece este lenguaje, el sistema proporciona cierta flexibilidad al ser totalmente independiente del sistema operativo sobre el que se desee ejecutar. A través la GUI, el operador tiene acceso a un conjunto de opciones que le permite mantener y/o consultar la información tanto de los sitios de interés como de la estructura vial.

3.3.2.2 Estructura Vial

Este módulo es útil para realizar una consulta de los datos que constituyen a la estructura vial. La estructura vial se refiere a una serie de elementos que conforman la vialidad de una ciudad, estos elementos son: delegaciones, colonias, calles, cuadras, cruceros, etc., todos estos elementos son almacenados en la BD. También en este módulo es donde se realiza el mantenimiento de esta información, se pueden dar de alta nuevas colonias, calles, etc. así como realizar una actualización de sus características como son: nombre, número de carriles, sentido, tráfico, entre otras.

[2] Structured Query Language.- Lenguaje Estructurado de Consultas
[3] Graphic User Interface

3.3.2.3 Sitios de Interés

De manera similar al módulo anterior, en este módulo se puede consultar la información referente a los sitios de interés: nombre, categoría, dirección, número telefónico, etc. También es posible actualizar cada uno de estos datos, así como dar de alta nuevos sitios.

3.3.2.4 Consulta de Información

Este módulo es muy similar al que está implementado del lado del servidor web. Permite la interacción con los servicios de consulta de BD para la recuperación de la información.

3.3.2.5 Mantenimiento de Información

Es encargado de proporcionar una interfaz con los servicios de mantenimiento de información que se encuentran en la BD. A través de este módulo se ejecutan los procedimientos almacenados (SP[4]) encargados de actualizar, insertar o eliminar la información almacenada en la BD. Un procedimiento almacenado es un conjunto de sentencias en lenguaje SQL[5] precompiladas que realizan una operación sobre los datos de la BD.

3.3.3 Módulos del Depósito de Datos (BD)

Por último se tienen los módulos de BD. A continuación se presenta una breve descripción de los módulos de esa parte:

3.3.3.1 Servicios de Consulta

En este módulo se encuentran todos los servicios desarrollados para consultar la información almacenada en BD.

3.3.3.2 Servicios de Mantenimiento

Todos los SPs que realizan el mantenimiento de la información de BD se encuentran en este módulo. Principalmente se tienen los servicios de inserción, actualización y eliminación de registros en general, aunque se tienen algunos más que realizan otros procesos de actualización.

[4] Stored Procedures
[5] Structured Query Language.- Lenguaje de Consulta Estructurada

3.4 Funcionamiento General

El usuario WAP teclea un URL raíz para acceder a la pantalla de registro, en la cual debe introducir su usuario y password que le hayan sido asignados. Si el registro se realiza exitosamente, la aplicación determina si ese usuario tiene una búsqueda pendiente, si es así le muestra la pantalla en donde se había quedado la última ocasión. Por el contrario, si no tiene una búsqueda pendiente, el sistema muestra un menú con los servicios de localización implementados. En este menú el usuario puede elegir entre que el sistema le busque una ruta óptima entre dos puntos dados por él mismo o encontrar una ruta hacía un sitio de interés en particular.

Si la opción elegida es la primera, el sistema solicita, mediante una serie de sencillos pasos, la introducción de la dirección del punto origen y del punto destino. A continuación el sistema presenta la opción de buscar la ruta óptima. Cuando ésta es seleccionada, después de ejecutar un algoritmo de búsqueda de caminos óptimos del lado del servidor, de verificar las condiciones de tráfico, la existencia de obras viales, etc. y si es que existe al menos una ruta entre estos dos puntos, el sistema regresa la ruta óptima representada por medio de una lista ordenada de calles cada una con un sentido cardinal, que de recorrerlas desde el punto origen en el orden presentado nos lleva por la ruta que en ese momento es la óptima hacia el punto destino.

Como se había comentado anteriormente, el sistema también da la opción de encontrar la ruta óptima desde un punto origen hacia un sitio de interés en particular o hacía el sitio de interés dentro de una categoría (restaurantes, escuelas, cines, bancos, etc.) más cercano al punto origen. De igual manera que como en el primer caso, la ruta es representada por medio de una sucesión de calles orientadas que van desde el punto origen hacia la dirección del sitio de interés buscado.

Todos los datos a los que se acaba de hacer referencia son alimentados por medio del sistema *standalone* encontrado del lado de la PC. El usuario operador es el encargado de realizar esta tarea con ayuda de este sistema.

3.5 Modelado de Procesos

Para el desarrollo de este apartado se hará uso de la metodología IDEF0[6]. IDEF0 consiste en una serie de normas que definen la metodología para la representación de funciones modeladas [WWW3].

Estos modelos consisten en una serie de diagramas jerárquicos junto con unos textos y referencias cruzadas entre ambos que se representan mediante unos rectángulos o cajas y una serie de flechas. Uno de los aspectos de IDEF0 más importantes es que como concepto de modelado va introduciendo gradualmente más y más niveles de detalle a través de la estructura del modelo. De esta manera, se produce un tema bien definido con una cantidad de información detallada disponible para profundizar en el modelo.

[6] Integration Definition for Function Modeling.- Definición de Integración para el Modelado de Funciones

Los elementos del modelo deben ser de uno de los siguientes: Las entradas (*inputs*) representan los datos u objetos que son transformados por la actividad. Las salidas (*outputs*) representan los datos u objetos producidos. Los controles (*controls*) representan las condiciones requeridas para producir una salida correcta. Finalmente, los mecanismos (*mechanisms*) representan los medios usados por la actividad para transformar la entrada [BAL00].

A continuación se presenta el modelado de procesos del sistema desarrollado en este trabajo.

3.5.1 Modelo de Contexto

Todo modelo debe tener un diagrama de contexto en el que se representa el sistema en su totalidad con una caja única con sus correspondientes flechas. A este diagrama se le denomina diagrama A-0 (Figura 3.3). Las flechas de este diagrama interconectan con funciones fuera del sistema.

Entradas

- Dirección Origen.- Representa la dirección del punto de origen a partir del cual se quiere encontrar la ruta óptima hacia otro punto (puede ser el de un sitio de interés).

- Dirección Destino.- Similar al caso anterior, ésta representa la dirección del punto destino al que se desea llegar y hacia donde se requiere que la ruta óptima llegue.

- Sitio de Interés.- Es el sitio de interés al que se desea llegar.

Salidas

- Ruta Óptima.- Es la ruta óptima resultado de la ejecución del proceso general. Se representa dentro del sistema como una sucesión de calles que van desde el punto origen al destino o al sitio de interés según sea haya elegido.

Control

- Instrucciones del usuario.- El usuario es el encargado de manipular al sistema. De él dependerá en gran medida el correcto funcionamiento de la aplicación y la obtención de resultados coherentes.

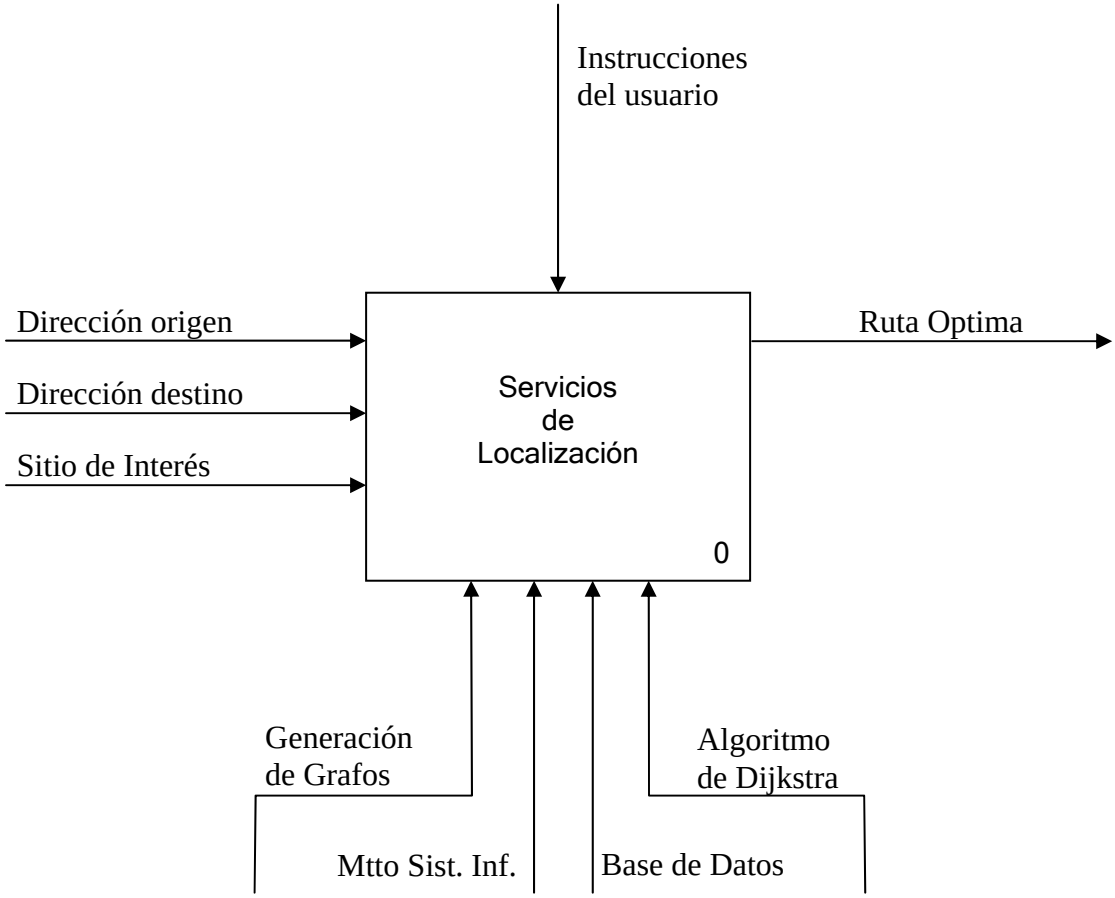


Figura 3.3. Diagrama A-0 del sistema

Mecanismos

- Generación de Grafos.- Se refiere a la generación de grafos a partir de los datos que están almacenados en BD. Este mecanismo crea varias estructuras de datos que representan al grafo, y a su vez a la vialidad de una ciudad, con la información recuperada de la BD.

- Mantenimiento del Sistema de Información.- Es sumamente importante ya que por medio de éste, el usuario ingresará al sistema toda la información de la estructura vial y hacerla persistente. Esta información es necesaria para generar posteriormente el grafo.

- Base de Datos.- A través de los servicios (SP) que fueron creados, se puede tanto consultar como manipular la información que es alimentada al sistema para la generación de grafos, sitios de interés, etc.

- Algoritmo de Dijkstra.- Una vez que se tiene en las estructuras de datos al grafo, así como las direcciones de origen y destino (o el sitio de interés si es el caso), la implementación computacional del Algoritmo de Dijkstra arroja como resultado la ruta óptima resultante, misma que es regresada hasta el teléfono celular del usuario que realizó la consulta.

3.6 Casos de Uso

Los Casos de Uso son una técnica para capturar información de cómo un sistema o negocio trabaja, o de cómo se desea que trabaje. No pertenece estrictamente al enfoque orientado a objeto, es una técnica para captura de requisitos. Un caso de uso es *una secuencia de transacciones que son desarrolladas por un sistema en respuesta a un evento que inicia un actor sobre el propio sistema*. Los diagramas de casos de uso sirven para especificar la funcionalidad y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas. O lo que es igual, un diagrama que muestra la relación entre los actores y los casos de uso en un sistema. Una relación es una conexión entre los elementos del modelo, por ejemplo la relación y la generalización son relaciones.

Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema al mostrar como reacciona una respuesta a eventos que se producen en el mismo. En este tipo de diagrama intervienen algunos conceptos nuevos: un *actor* es una entidad externa al sistema que se modela y que puede interactuar con él; un ejemplo de actor podría ser un usuario o cualquier otro sistema. Las relaciones entre casos de uso y actores pueden ser las siguientes:

- Un actor se comunica con un caso de uso.
- Un caso de uso extiende otro caso de uso.
- Un caso de uso usa otro caso de uso.

Cada Caso de Uso puede estar definido por [WWW4]:

- texto que lo describe,
- secuencia de pasos (flujo de eventos) ejecutados dentro del caso de uso,
- precondiciones y poscondiciones para que el caso de uso comience o termine, y
- mezclando las anteriores

Para el desarrollo del presente trabajo, y continuando con la definición de los procesos del mismo, se procede a definir los casos de uso identificados para este sistema.

3.6.1 Definición de Casos de Uso del Sistema

3.6.1.1 *PC: Sistema de Información – Estructura vial*

Como ya se ha dicho anteriormente, uno de los objetivos del Sistema de Información es el de crear y mantener la estructura vial.

Una ciudad o parte de ella consta de un gran número de elementos: casas, calles, edificios, fábricas, locales comerciales, restaurantes, etc. A su vez, cada uno de estos elementos contiene un número, todavía mucho mayor, de atributos que los distinguen. Por lo que es obvio que se necesita de una buena abstracción para descartar la mayor cantidad de estos atributos y elementos. Además, dada la naturaleza de esta aplicación, lo que más interesa es el manejo de las direcciones de la ciudad que el sistema esté modelando, y de rutas dentro de la misma.

Se define ahora lo que es la estructura vial de una ciudad. La estructura vial se refiere a la red que forman las calles, avenidas, delegaciones, ejes, circuitos, colonias, etc. de dicha ciudad para la circulación de vehículos automotores.

Para el caso particular del desarrollo de este sistema, se planeó la definición de los siguientes elementos:

- **Delegación.-** Ya que las ciudades se dividen en delegaciones o municipios. Además este elemento permitiría una búsqueda inicial más eficiente, en vez de seleccionar una calle de una lista inmensa, una preselección de la delegación arrojaría un número de calles mucho menor (serían las contenidas en esta delegación).

- **Colonia.-** Este elemento tiene una doble función, la primera de ellas es igual a la anterior, y la segunda es que existen casos en que dos calles tienen el mismo nombre, solo que están situadas en diferente colonia. En este caso sería este elemento el que las diferenciaría.

- **Calle.-** Serán usadas para proporcionar la ruta óptima, ésta se desplegará como una sucesión de calles por las que el usuario debería transitar para recorrer la ruta óptima.

- **Nodo.-** Equivalente a un crucero o esquina entre dos o más calles. Este elemento definirá los puntos en los que puede haber un cambio de dirección o sentido.

- **Cuadra.-** Dado que muchas de las calles tiene una longitud grande y no sería eficiente tomar de ellas sus dos extremos, es necesario dividirlas en varias cuadras. Tener varias cuadras con longitudes más pequeñas permitirá proporcionar rutas más exactas y eficientes, aunque por otro lado tendrá la desventaja de que la ejecución de los algoritmos de búsqueda de caminos óptimos puede tomar mas tiempo.

Estos son los cinco elementos que definirán la estructura vial. A continuación se definen los casos de uso para el mantenimiento de cada uno de ellos.

Caso de uso	Alta de Delegación
Actor	Operador (Usuario)
Descripción	Ingresar al sistema una Delegación para que forme parte de la estructura vial.
Precondiciones	No encontrarse en la ejecución de alguna otra opción.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona la pestaña Delegación del menú principal.
	2. El sistema muestra una lista con todas las delegaciones dadas de alta, además de una caja de texto deshabilitada, donde se podrá capturar el nombre de la delegación que se desee dar de alta.
	3. El operador presiona el botón Insertar
	4. El sistema habilita la caja de texto. Así como también proporciona un par de botones: Aceptar y Cancelar.
	5. El operador teclea el nombre de la delegación.
	6. El operador presiona el botón Aceptar para que la delegación sea guardada en base de datos.
	7. El sistema guarda en BD la nueva delegación.
Excepciones	El operador tiene además la opción Cancelar, ésta opción es útil cuando él necesita terminar el flujo sin que la delegación sea almacenada en base de datos.
Postcondiciones	Ninguna

Caso de uso	Modificación de Delegación
Actor	Operador (Usuario)
Descripción	Modificar una delegación del sistema.
Precondiciones	No encontrarse en la ejecución de alguna otra opción y que la delegación a modificar esté dada de alta.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona la pestaña Delegación del menú principal.
	2. El sistema muestra una lista con todas las delegaciones dadas de alta, además de una caja de texto deshabilitada, donde se podrá capturar la nueva delegación.
	3. El operador selecciona de la lista la delegación que desea modificar. Después presiona el botón de Modificar.
	4. El sistema habilita la caja de texto. Así como también proporciona un par de botones: Aceptar y Cancelar.
	5. El operador teclea el nuevo nombre de la delegación.
	6. El operador presiona el botón Aceptar para que la delegación sea actualizada en base de datos.
	7. El sistema actualiza en BD la información de la delegación.
Excepciones	El operador tiene además la opción Cancelar, ésta opción es útil cuando necesita terminar el flujo sin que la delegación sea modificada en base de datos.
Postcondiciones	Ninguna

Caso de uso	Baja de Delegación
Actor	Operador (Usuario)
Descripción	Eliminar una delegación del sistema.
Precondiciones	No encontrarse en la ejecución de alguna otra opción y que la delegación a eliminar esté dada de alta.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona la pestaña Delegación del menú principal.
	2. El sistema muestra una lista con todas las delegaciones dadas de alta.
	3. El operador selecciona de la lista, la delegación que desea eliminar. Después presiona el botón de Eliminar.
	4. El sistema Elimina la delegación en BD y actualiza la lista en la aplicación.
Excepciones	El operador tiene además la opción Cancelar, ésta opción es útil cuando necesita terminar el flujo sin que la delegación sea eliminada de la base de datos.
Postcondiciones	Ninguna

Los casos de uso para el alta, baja y modificación de una calle o nodo, son iguales que los anteriores a excepción de los nombres en la aplicación, y tablas y servicios en la BD.

A continuación se definen los casos de uso para el mantenimiento de las colonias.

Caso de uso	Alta de Colonia
Actor	Operador (Usuario)
Descripción	Ingresar al sistema una Colonia que pertenezca a una delegación en específico.
Precondiciones	No encontrarse en la ejecución de alguna otra opción. Que la delegación a la que pertenece ya esté ingresada en el sistema.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona la pestaña Colonia del menú principal.
	2. El sistema muestra una lista con todas las colonias que están ingresadas en el sistema y la delegación a la que pertenecen, también una caja de texto deshabilitada, donde se podrá capturar el nombre de la colonia que se desee dar de alta, además de un combo también deshabilitado donde se encontrarán todas las delegaciones registradas en el sistema.
	3. El operador presiona el botón Insertar
	4. El sistema habilita la caja de texto y el combo. Así como también proporciona un par de botones: Aceptar y Cancelar.
	5. El operador teclea el nombre de la colonia y selecciona en el combo la delegación a la que pertenece.
	6. El operador presiona el botón Aceptar para que la colonia sea almacenada en base de datos.
	7. El sistema guarda en BD los datos de la colonia.
Excepciones	El operador tiene además la opción Cancelar, ésta opción es útil cuando él necesita terminar el flujo sin que la Colonia sea dada de alta.
Postcondiciones	Ninguna

Caso de uso	Modificación de Colonia
Actor	Operador (Usuario)
Descripción	Modificar los datos de una colonia del sistema.
Precondiciones	No encontrarse en la ejecución de alguna otra opción y que la colonia a modificar esté dada de alta.
Flujo normal de eventos	1. Este caso de uso comienza cuando el operador selecciona la pestaña Colonia del menú principal.
	2. El sistema muestra una lista con todas las colonias dadas de alta, además de una caja de texto y un combo deshabilitados, donde capturará el nuevo nombre y/o seleccionará otra delegación de la colonia.
	3. El operador selecciona de la lista la colonia que desea modificar. Después presiona el botón de Modificar.
	4. El sistema habilita la caja de texto y el combo. Así como también proporciona un par de botones: Aceptar y Cancelar.
	5. El operador teclea un nuevo nombre de la colonia o sobre el combo cambia la delegación a la que pertenece.
	6. El operador presiona el botón Aceptar para que la colonia sea actualizada en base de datos.
	7. El sistema actualiza en BD la información de la colonia.
Excepciones	El operador tiene además la opción Cancelar, ésta opción es útil cuando necesita terminar el flujo sin que la colonia sea modificada en BD.
Postcondiciones	Ninguna

Caso de uso	Baja de Colonia
Actor	Operador (Usuario)
Descripción	Eliminar una colonia del sistema.
Precondiciones	No encontrarse en la ejecución de alguna otra opción y que la colonia a eliminar esté dada de alta.
Flujo normal de eventos	1. Este caso de uso comienza cuando el operador selecciona la pestaña Colonia del menú principal.
	2. El sistema muestra una lista con todas las colonias insertadas en BD.
	3. El operador selecciona de la lista, la colonia que desea eliminar. Después presiona el botón de Eliminar.
	4. El sistema Elimina la colonia en BD y actualiza la lista en la aplicación.
Excepciones	El operador tiene además la opción Cancelar, ésta se usará cuando se necesite salir del flujo sin que la colonia sea eliminada de la base de datos.
Postcondiciones	Ninguna

Como último elemento de esta sección se presentan los casos de uso para llevar a cabo el mantenimiento de cuadras de la red vial dentro de este sistema.

Caso de uso	Alta de Cuadra
Actor	Operador (Usuario)
Descripción	Agregar una Cuadra al sistema vial.
Precondiciones	No encontrarse en la ejecución de alguna otra opción.
Flujo normal de eventos	1. Este caso de uso comienza cuando el operador selecciona la pestaña Cuadra del menú principal.

	2. El sistema muestra una lista con todas las cuadras registradas en el sistema, además de un conjunto de controles inicialmente deshabilitados, donde se podrá capturar los datos de la cuadra,
	3. El operador presiona el botón Insertar
	4. El sistema habilita los controles, así como también proporciona un par de botones: Aceptar y Cancelar.
	5. El operador teclea el nombre de la cuadra y todos los datos de la misma.
	6. El operador presiona el botón Aceptar para que la cuadra sea almacenada en base de datos.
	7. El sistema guarda en BD los datos de la cuadra.
Excepciones	El operador tiene además la opción Cancelar, ésta opción es útil cuando él necesita terminar el flujo sin que la cuadra sea insertada.
Postcondiciones	Ninguna

Caso de uso	Modificación de Cuadra
Actor	Operador (Usuario)
Descripción	Modificar los datos de una cuadra dada de alta en el sistema.
Precondiciones	No encontrarse en la ejecución de alguna otra opción.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona la pestaña Cuadra del menú principal.
	2. El sistema muestra una lista con todas las cuadras almacenadas, además de un conjunto de controles capturará de la cuadra.
	3. El operador selecciona de la lista la cuadra que desea modificar. Después presiona el botón Modificar.
	4. El sistema habilita los controles y proporciona los botones: Aceptar y Cancelar.
	5. El operador introduce los nuevos datos de la cuadra.
	6. El operador presiona el botón Aceptar para que la cuadra sea actualizada en base de datos.
	7. El sistema actualiza en BD la información de la cuadra.
Excepciones	El operador tiene además la opción Cancelar, ésta opción es útil cuando necesita terminar el flujo sin que la cuadra sea modificada en BD.
Postcondiciones	Ninguna

Caso de uso	Baja de Cuadra
Actor	Operador (Usuario)
Descripción	Eliminar una cuadra del sistema.
Precondiciones	No encontrarse en la ejecución de alguna otra opción.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona la pestaña Cuadra del menú principal.
	2. El sistema muestra una lista con todas las cuadras insertadas en BD.
	3. El operador selecciona de la lista, la cuadra que desea eliminar. Después presiona el botón de Eliminar.
	4. El sistema elimina la cuadra en BD y actualiza la lista en la aplicación.
Excepciones	El operador tiene además la opción Cancelar, ésta se usará cuando se necesite salir del flujo sin que la cuadra sea eliminada de la base de datos.
Postcondiciones	Ninguna

3.6.1.2 *PC: Sistema de Información – Sitios de Interés*

El sistema debe tener como opción la búsqueda de rutas dirigidas hacia sitios de interés, por lo que es lógica la creación de un mantenimiento para los sitios de interés que manejará este sistema. A continuación se definen los casos que definen la funcionalidad de este módulo del sistema.

Caso de uso	Alta de Sitio de Interés
Actor	Operador (Usuario)
Descripción	Insertar en el sistema un sitio de interés.
Precondiciones	No encontrarse en la ejecución de alguna otra opción.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona, del menú principal, la pestaña Sitio.
	2. El sistema muestra una lista con todos los sitios de interés registrados en el sistema. Además de un conjunto de controles que permiten capturar un nuevo sitio.
	3. El operador presiona el botón Insertar
	4. El sistema habilita los controles y proporciona un par de botones adicionales: Aceptar y Cancelar.
	5. El operador ingresa los datos del nuevo sitio: nombre, dirección (calle, número y colonia), número telefónico, entre otros.
	6. Posteriormente, el operador presiona Aceptar para que el sitio sea almacenado en BD.
	7. El sistema guarda en BD
Excepciones	El operador tiene también la opción Cancelar, ésta opción le sirve para cuando desea terminar el flujo sin que el sitio se almacene en BD.
Postcondiciones	Ninguna

Caso de uso	Modificación de Sitio de Interés
Actor	Operador (Usuario)
Descripción	Modificar un sitio de interés ingresado en el sistema.
Precondiciones	No encontrarse en la ejecución de alguna otra opción.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona la pestaña Sitio del menú principal.
	2. El sistema muestra una lista con todos los sitios de interés, además de los controles para capturar datos.
	3. El operador selecciona de la lista el sitio que desea modificar. Para posteriormente presionar el botón Modificar.
	4. El sistema habilita los controles y muestra un par de botones: Aceptar y Cancelar.
	5. El operador ingresa los nuevos datos para el sitio seleccionado.
	6. El operador presiona el botón Aceptar para que el sitio sea actualizado en la base de datos.
	7. El sistema actualiza en BD la información del sitio elegido.
Excepciones	El operador tiene además la opción Cancelar, ésta opción sirve para cuando requiera descartar el flujo sin que los datos del sitio sean cambiados en BD.
Postcondiciones	Ninguna

Caso de uso	Eliminación de Sitio de Interés.
Actor	Operador (Usuario)
Descripción	Eliminar un sitio de interés del sistema.
Precondiciones	No encontrarse en la ejecución de alguna otra opción.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona la pestaña Sitio del menú principal.
	2. El sistema muestra una lista con todos los sitios de interés registrados en BD.
	3. El operador selecciona de la lista, el sitio que desea eliminar. Después presiona el botón de Eliminar.
	4. El sistema Elimina el sitio de interés y actualiza la lista de sitios dentro de la aplicación.
Excepciones	El operador tiene además la opción Cancelar, ésta opción para cuando él requiere terminar el flujo sin eliminar el sitio de BD.
Postcondiciones	Ninguna

3.6.1.3 *Servidor Web (Frontend)*

Caso de uso	Registro y Búsqueda Pendiente.
Actor	Usuario
Descripción	Ingresar a la aplicación y continuar una búsqueda pendiente.
Precondiciones	Ninguna
Flujo normal de eventos	1. Este caso de uso comienza cuando el usuario teclea el URL de la aplicación móvil.
	2. El sistema muestra la pantalla de registro.
	3. El usuario teclea su Nombre de Usuario y Password.
	4. El sistema valida los datos ingresados, de ser correctos determina si el usuario tiene una búsqueda pendiente. Si la tuviera lo lleva hasta el punto en donde se quedó. De lo contrario le muestra el menú principal de la aplicación de la aplicación móvil.
Excepciones	Ninguna
Postcondiciones	Ninguna

Caso de uso	Buscar Ruta Óptima entre dos puntos.
Actor	Usuario
Descripción	Obtener la ruta óptima entre dos puntos.
Precondiciones	Ninguna
Flujo normal de eventos	1. Este caso de uso comienza cuando el usuario selecciona del menú principal de la aplicación móvil la opción Dos Puntos.
	2. El sistema devuelve el menú para esta modalidad de búsqueda.
	3. El usuario selecciona la dirección origen y destino.
	4. El sistema presenta la opción de buscar la ruta óptima entre esos dos puntos.
	5. El usuario selecciona la opción mencionada en el punto anterior.
	6. El sistema busca la ruta óptima entre estos dos puntos. A menos que no existiera una ruta disponible, el sistema presenta un listado ordenado de calles que van desde un punto al otro.
Excepciones	Ninguna

Postcondiciones	Ninguna

Caso de uso	Buscar Ruta Óptima hacia sitio de interés.
Actor	Usuario
Descripción	Obtener la ruta óptima entre un punto origen hasta un sitio de interés en particular.
Precondiciones	Ninguna
Flujo normal de eventos	1. Este caso de uso comienza cuando el usuario selecciona del menú principal de la aplicación móvil, la opción Sitio de Interés.
	2. El sistema devuelve el menú para esta modalidad de búsqueda.
	3. El usuario selecciona la dirección origen y un sitio de interés.
	4. El sistema presenta la opción de buscar la ruta óptima entre el punto origen y el sitio de interés.
	5. El usuario selecciona la opción anterior.
	6. El sistema busca la ruta óptima entre estos dos puntos. A menos que no existiera una ruta disponible, el sistema presenta un listado ordenado de calles que van desde el punto origen hasta el sitio de interés elegido.
Excepciones	Ninguna
Postcondiciones	Ninguna

Caso de uso	Seleccionar dirección.
Actor	Usuario
Descripción	Subflujo que permite introducir una dirección, ya sea de origen o destino.
Precondiciones	Ninguna
Flujo normal de eventos	1. Este caso de uso comienza cuando el usuario ha elegido la opción de introducir una dirección, ya sea la de origen o destino.
	2. El sistema regresa un listado de todas las delegaciones del sistema y lo muestra en el navegador del celular.
	3. El usuario elige la delegación donde se encuentra la dirección que desea introducir.
	4. El sistema regresa un listado de las colonias que pertenecen a la delegación previamente seleccionada, y le permite que elija una de ellas.
	5. El usuario elige la colonia.
	6. El sistema retorna ahora un listado de las calles que se encuentran en la colonia seleccionada en el punto anterior.
	7. El usuario elige la calle.
	8. El sistema solicita al usuario una pantalla donde debe capturar el número de lote o casa de la dirección que esté ingresando.
	9. El usuario captura el número solicitado en el punto anterior. Si los datos que ha ingresado son correctos, él debe seleccionar la opción Aceptar, la cual grabará la dirección completa y regresará al menú en donde se encontraba al momento de elegir la opción que modela este caso de uso.
Excepciones	Ninguna
Postcondiciones	Ninguna

Caso de uso	Seleccionar sitio de interés.
Actor	Usuario
Descripción	Subflujo que permite seleccionar un sitio de interés.

Precondiciones	Ninguna
Flujo normal de eventos	1. Este caso de uso comienza cuando el usuario ha elegido la opción de elegir un sitio de interés.
	2. El sistema regresa un listado de todas las categorías a las que pertenece un sitio de interés.
	3. El usuario elige la categoría del sitio de su interés.
	4. El sistema regresa un listado de todos los sitios de interés que pertenecen a la categoría que eligió en el punto anterior.
	5. El usuario elige el sitio de interés que desee.
	6. El sistema retorna el control al flujo que dio inicio a éste (seleccionar sitio de interés).
Excepciones	Ninguna
Postcondiciones	Ninguna

Se presenta en la Figura 3.4 el Diagrama de Casos de Uso del sistema, para la aplicación *backend*.

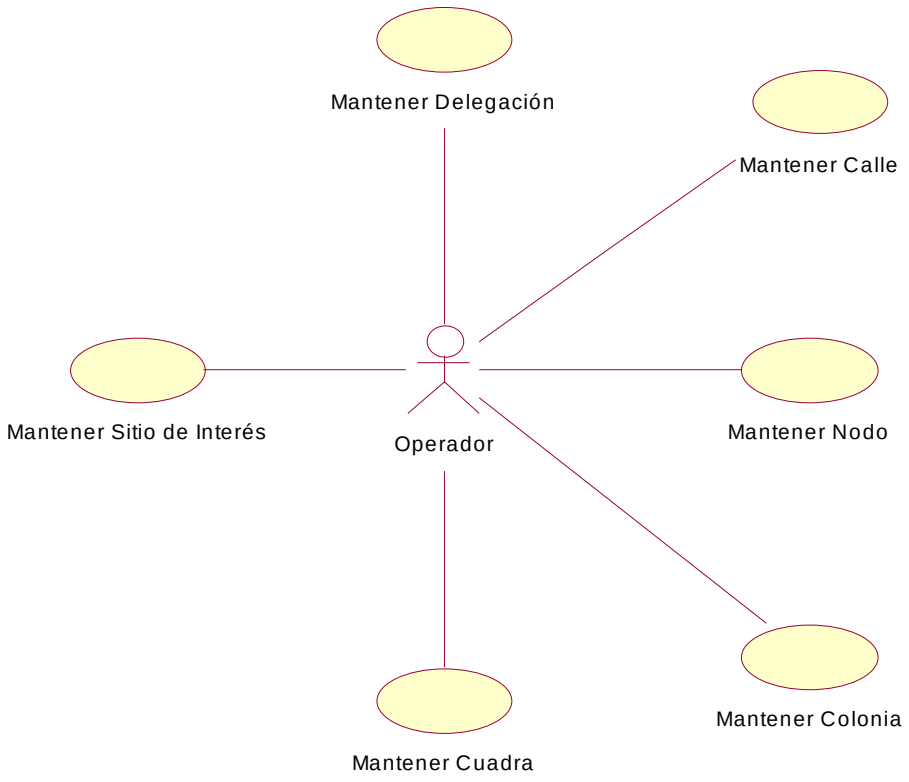


Figura 3.4 Diagrama de Casos de Uso del *backend* de la aplicación

El Diagrama de Casos de Uso para el *frontend* es presentado en la Figura 3.5.

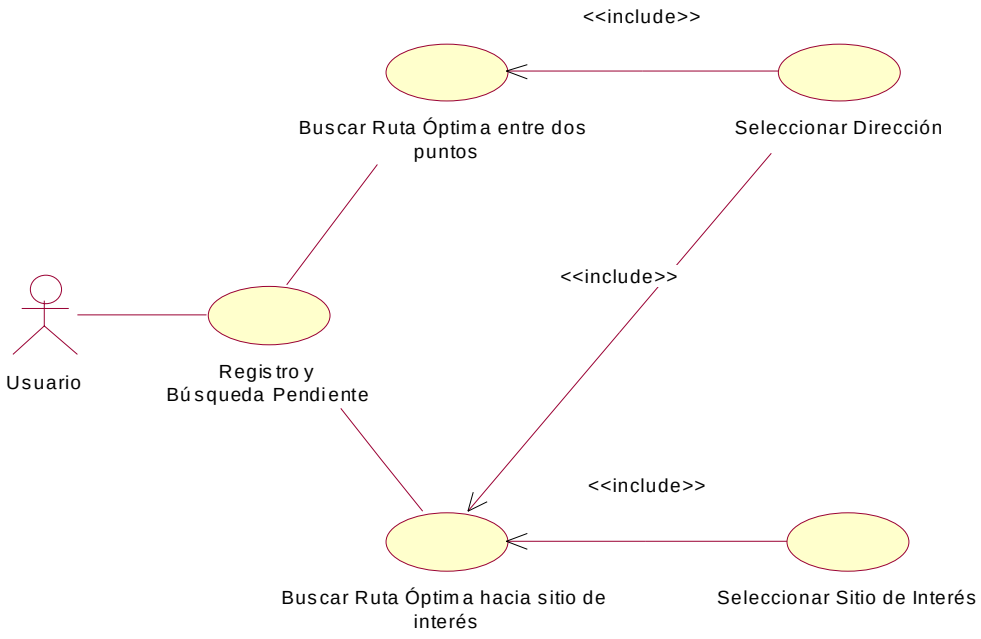


Figura 3.5 Diagrama de Casos de Uso del *frontend* de la aplicación

3.7 Resumen

En este capítulo se muestra la arquitectura general de la aplicación. Esta arquitectura está basada en el modelo WAP, el cual define elementos y protocolos que atacan las limitaciones y restricciones que presentan los ambientes inalámbricos. Posteriormente son definidas las dos principales aplicaciones que conforman el sistema desarrollado en esta tesis: El *frontend* y el *backend*, así como cada uno de los módulos que las integran.

Posteriormente, como parte del análisis de la aplicación, se presentan los casos de uso que definen la funcionalidad de la aplicación, es decir, para cada uno de los módulos del *frontend* y del *backend*. También son presentados los respectivos Diagramas de Casos de Uso.

En el siguiente capítulo se presenta la etapa de diseño del sistema, dentro del desarrollo de software. Se describen además cada uno de los elementos que conforman y participan en esta etapa, así como los diagramas que define UML[7]. También se da una explicación del diseño de la base de datos y se presenta el modelo entidad-relación.

[7] Unified Modeling Language.- Lenguaje de Modelado Unificado

Capítulo 4

Diseño de la Aplicación

En este capítulo se presentan cada uno de los elementos que conforman la etapa de Diseño dentro del desarrollo de sistemas con UML, aplicados al presente trabajo. En específico, se trata de los diagramas de Interacción, los diagramas de Clases y los diagramas de Estados, principalmente.

Por otro lado, se presenta una descripción del Depósito de Datos y se presenta el Modelo Lógico de la base de datos. Más adelante se da una explicación referente a la generación del grafo que representa a la Estructura Vial, a partir de la información contenida en base de datos.

4.1 Introducción

El diseño de un sistema produce los detalles que establecen la forma en cómo éste cumplirá con los requerimientos identificados durante la fase de análisis. Durante el diseño se completan las definiciones de las clases y las asociaciones que figuran en el análisis, se añaden clases nuevas para facilitar o mejorar la implementación, y se definen las interfaces y algoritmos de las operaciones, con lo que el sistema queda listo para su implementación.

Esta etapa se centra más en conceptos computacionales que en el dominio de aplicación. Los objetos descubiertos durante el análisis son el esqueleto del sistema, pero en esta etapa se debe escoger entre diversas opciones para implementarlos teniendo en cuenta criterios de eficiencia (en tiempo de ejecución y memoria) y costo.

Los elementos del **modelo estructural** (diagramas de clases, en los que aparecen clases y relaciones) determinan la estructura (estática) de la implementación del sistema, por lo que se incluyen directamente en el diseño. A veces, alguna de las clases no aparece explícitamente puesto que su funcionalidad se ha dividido entre otras clases por motivos de eficiencia. Más frecuentemente se añaden durante el diseño clases y relaciones y asociaciones redundantes también con vistas a mejorar la eficiencia. Es necesario además diseñar cómo se va a implementar cada una de las relaciones que se detectaron durante el análisis [WWW5].

Los elementos del **modelo de comportamiento** (diagramas de estados y de interacción), que describe cómo reacciona el sistema ante los eventos, determinan el control de la implementación. Es necesario diseñar estos aspectos dinámicos del sistema antes de proceder a su implementación [WWW5].

Este capítulo se divide en dos principales secciones: En la primera se presentan los diagramas pertenecientes a la etapa de diseño, definidos en UML, para el desarrollo de sistemas orientado a objetos. En la segunda se muestra el diseño de la base de datos, conforme al análisis realizado en el capítulo anterior, así como a las clases identificadas en esta etapa.

Básicamente los elementos que son presentados como parte de la primera sección son:

- Los Diagramas de Interacción,
- Los Diagramas de Clases, y
- Los Diagramas de Estados

En este orden se presentan y explican los **diagramas de interacción** de los casos de uso identificados en el capítulo anterior[1]. En UML, estos diagramas se dividen en dos: los **diagramas de secuencia** y los **diagramas de colaboración**.

[1] Por razones de espacio, en el contenido de este capítulo sólo se presentan los diagramas más representativos, dejando el resto para un apéndice en la parte final de la tesis.

Los diagramas de secuencia son utilizados para modelar las interacciones entre los objetos organizadas en una secuencia temporal para cumplir con el objetivo del caso de uso del que se originan. Representa una interacción, un conjunto de comunicaciones (intercambio de mensajes) entre objetos organizados visualmente por orden temporal. [RAM00].

Por otro lado, los diagramas de colaboración presentan una alternativa a los de secuencia para modelar interacciones entre objetos del sistema, sólo que no se centra en la secuencia cronológica del escenario que se está modelando, sino en estudiar todos los efectos de un objeto dado durante un escenario.

Posteriormente se presentan los **diagramas de clases**. Estos diagramas son útiles para tener en cuenta los detalles concretos que llevará la implementación del sistema.

Como última parte de esta primera etapa se presenta el **diagrama de estados** de una de las clases más importantes de este sistema, es la clase que implementa el algoritmo de Dijkstra para encontrar las rutas más cortas dentro de un grafo.

Los elementos mostrados como parte de la segunda sección son:

- El Modelo Entidad-Relación, y
- El Modelo Relacional

Los cuales representan el modelo de datos definido en el capítulo anterior y muestran de una manera más tangible la estructura de la base de datos del sistema.

4.2 Diagramas de Interacción (Secuencia y Colaboración)

Como se dijo en la introducción a este capítulo, se comenzará por analizar los diagramas de interacción para los casos de uso identificados anteriormente. En el capítulo pasado se comentó que la aplicación consta de dos principales módulos: el *backend* (Aplicación de Mantenimiento de Información) y el *frontend* (Aplicación WAP), en este mismo orden son presentados los diagramas más representativos de cada módulo.

4.2.1 Mantenimiento de un Sitio de Interés (*Backend*)

En la Figura 4.1, se muestra el diagrama de interacción (secuencia) para realizar un alta de un sitio de interés en el sistema. Este diagrama forma parte del caso de uso *Mantener Sitio de Interés*. Los pasos en el diagrama son:

1. En el momento que el operador acepta que el sitio de interés sea registrado, se crea una instancia de la clase Sitio.
2. Posteriormente, los atributos del objeto son llenados con la información que capturó el operador en la interfaz.
3. Es creado un objeto de la clase MantSitio, ésta se encarga de darle mantenimiento a un sitio de interés.
4. Le es pasado como parámetro el objeto de la clase Sitio (creado en el paso 1), con la información que se desea dar de alta.
5. Se invoca al método insertar() de la clase MantSitio, ésta se encarga de realizar el proceso de alta en BD.
6. Se crean dos objetos de las clases que proporciona el API de java: Connection y Statement. La primera es necesaria para almacenar una conexión con la BD y posteriormente crear una sentencia de SQL. La segunda es la sentencia SQL, esta es la que permite ejecutar una sentencia SQL para realizar una operación en la BD (en este caso una inserción, aunque ésta es realizada a través de un procedimiento almacenado[2]).
7. Del objeto de la clase Connection es ejecutado el método createStatement(), este método es el encargado de crear la sentencia SQL. La sentencia que retorna el método es asignada al objeto que se había creado en el paso 6.
8. Del objeto de tipo Statement, es ejecutado el método executeUpdate(), este método sirve para ejecutar una sentencia SQL que es pasada como un parámetro de tipo String. En este caso es ejecutado el SP con nombre **LOC_spi_sitio**, al cual le son pasados como parámetros todos los atributos de la clase Sitio.
9. Le es retornado al Operador un mensaje: De éxito si el sitio fue dado de alta en BD correctamente o de error si por algún motivo no pudo registrarse el sitio.

El diagrama de colaboración para esta acción del mismo caso de uso es presentado en la Figura 4.2.

[2] Stored Procedure (SP)

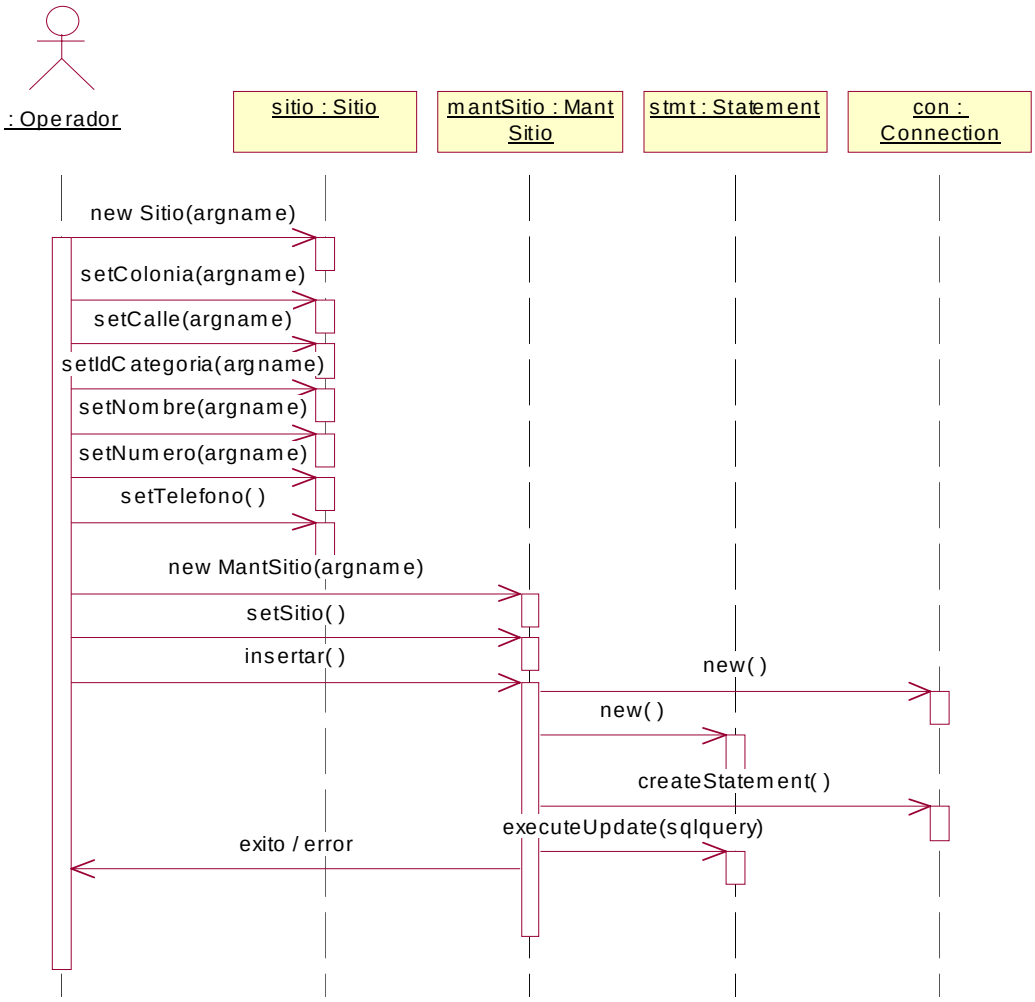


Figura 4.1 Diagrama de secuencia para cuando se inserta un sitio de interés

Ahora se presenta el diagrama de secuencia para realizar una baja de un sitio de interés del sistema. Al igual que el anterior, este diagrama forma parte del caso de uso Mantener Sitio de Interés. El diagrama es mostrado en la Figura 4.3, y los pasos son los siguientes:

1. El operador acepta que el sitio que seleccionó sea eliminado del sistema, se instancia un objeto de la clase Sitio.
2. En este caso, el único atributo que requiere ser llenado es el idSitio, ya que existe un identificador único para cada sitio de interés.
3. Es creado un objeto de la clase MantSitio, ésta se encarga de darle mantenimiento a un sitio de interés.
4. Le es pasado como parámetro el objeto de la clase Sitio.

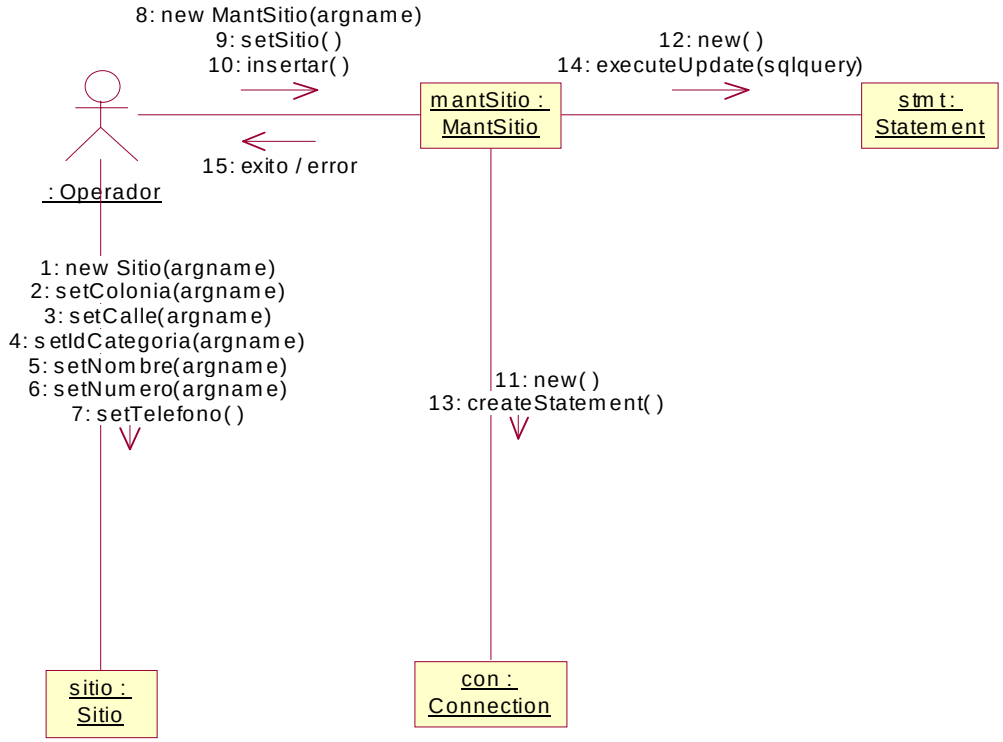


Figura 4.2 Diagrama de colaboración[3] para cuando se inserta un sitio de interés

5. Se invoca al método eliminar() de la clase MantSitio, ésta se encarga de realizar una baja en BD.
6. Se crean dos objetos de las clases que proporciona el API de java: Connection y Statement. La primera es necesaria para almacenar una conexión con la BD y posteriormente crear una sentencia de SQL. La segunda es la sentencia SQL, ésta es la que permite ejecutar una sentencia SQL para realizar una operación en BD.
7. Del objeto de la clase Connection es ejecutado el método createStatement(), este método es el encargado de crear la sentencia SQL. La sentencia que retorna el método es asignada al objeto que se había creado en el paso 6.
8. Del objeto de tipo Statement, es ejecutado el método executeUpdate(), este método sirve para ejecutar una sentencia SQL que es pasada como un parámetro de tipo String. En este caso es ejecutado el SP con nombre **LOC_spd_sitio**, al cual le es pasado como parámetro el atributo idSitio de la clase Sitio.
9. Le es retornado al Operador un mensaje: De éxito si fue eliminado el sitio de BD o de error si por alguna razón no se realizó.

[3] Los diagramas de colaboración restantes se incluyen en un apéndice posteriormente.

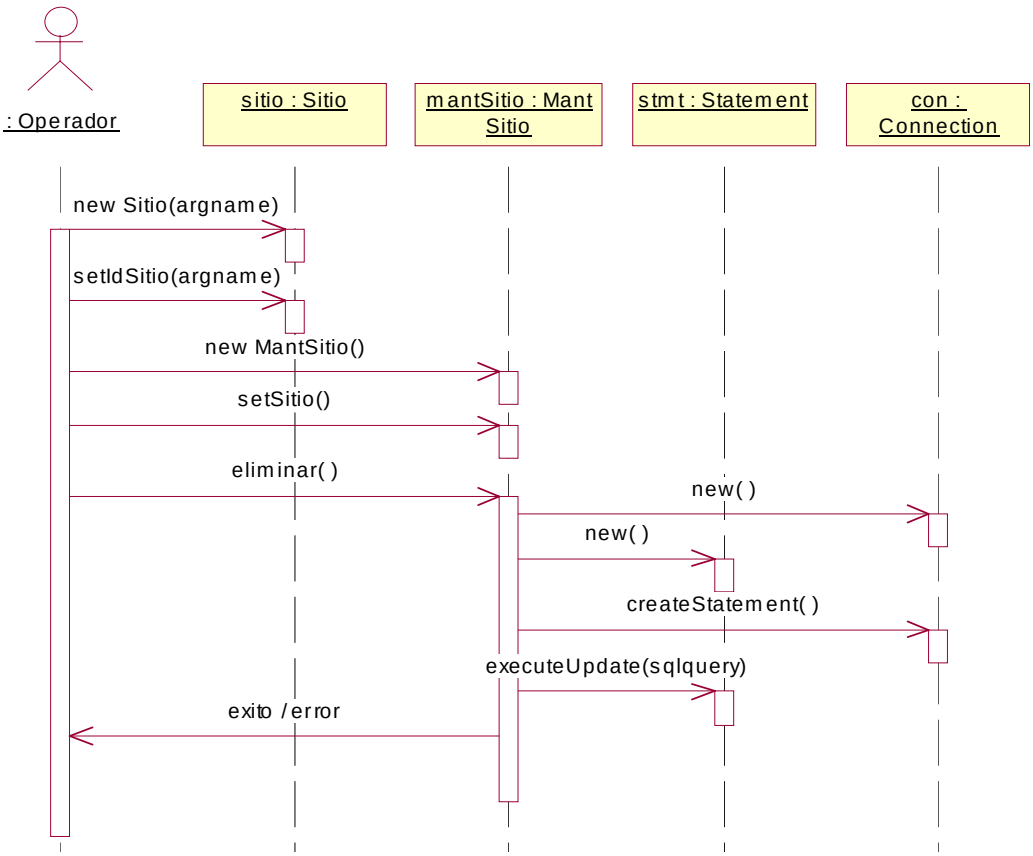


Figura 4.3 Diagrama de secuencia para cuando se elimina un sitio de interés

Como última parte se tiene la actualización de un sitio de interés. En la Figura 4.4 se muestra el diagrama de secuencia para cuando se realiza esta operación. Los pasos que se llevan a cabo son:

1. El operador, una vez que ha cambiado los datos del sitio de interés, acepta que sea actualizado, un objeto de la clase Sitio es instanciado.
2. Los atributos del objeto son llenados con la información que capturó el operador en la interfaz, recuperando el idSitio ya que será la referencia del sitio que se actualizará.
3. Es creado un objeto de la clase MantSitio, ésta se encarga de darle mantenimiento a un sitio de interés.
4. Le es pasado como parámetro el objeto de la clase Sitio (creado en el paso 1), con la información con la que se modificará.
5. Se invoca al método actualizar() de la clase MantSitio, ésta se encarga de realizar la actualización del sitio en BD.
6. Se crean dos objetos de las clases que proporciona el API[4] de java: Connection y Statement. La primera es necesaria para almacenar una conexión con la BD y

[4] Application Programming Interface.- Interfaz de Programación de Aplicaciones.

posteriormente crear una sentencia de SQL. La segunda es la sentencia SQL, esta es la que permite ejecutar una sentencia SQL para realizar la operación deseada en la BD.

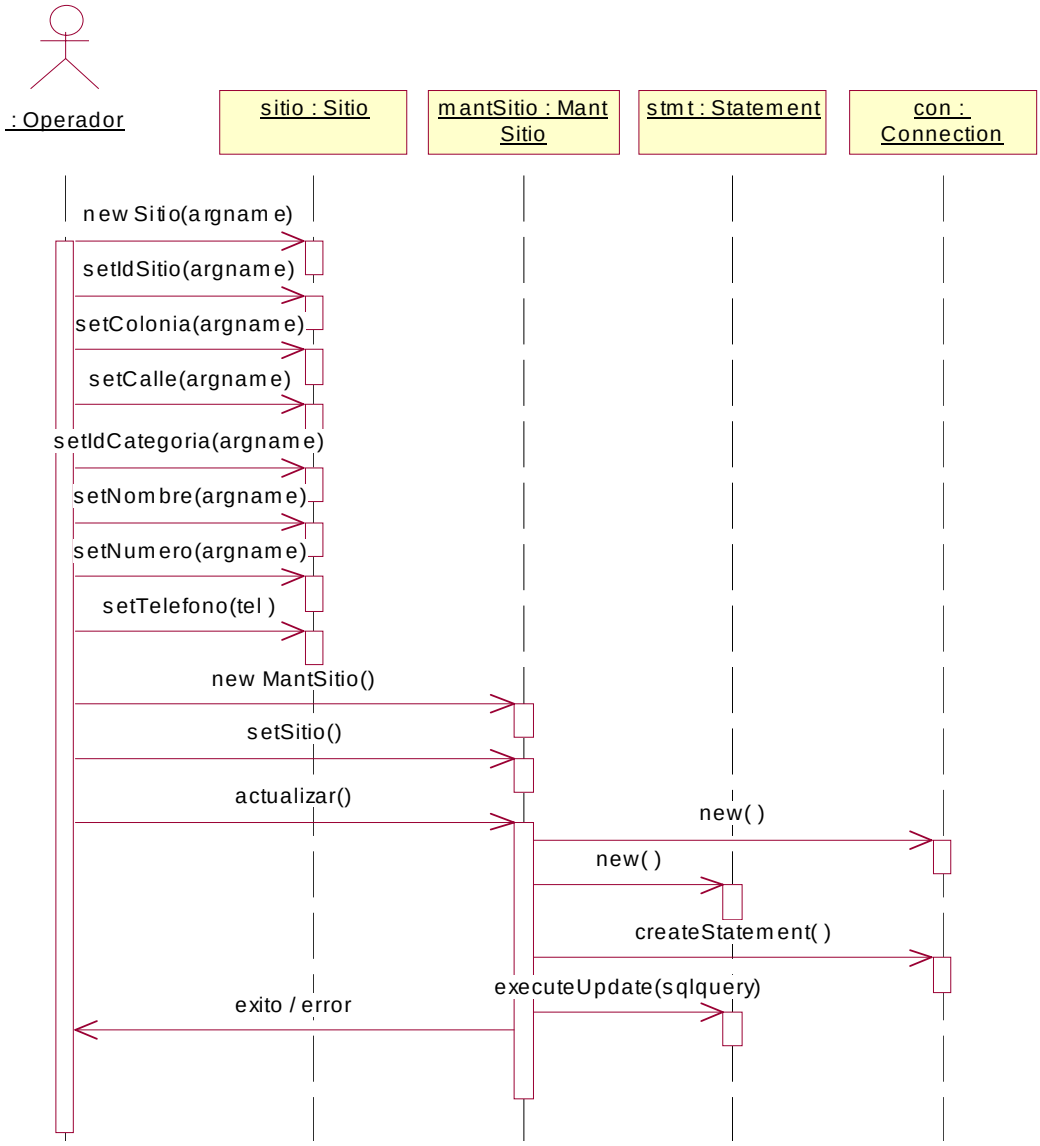


Figura 4.4 Diagrama de secuencia para cuando se actualiza un sitio de interés

7. Del objeto de la clase Connection es ejecutado el método createStatement(), este método es el encargado de crear la sentencia SQL. La sentencia que retorna el método es asignada al objeto que se había creado en el paso 6.
8. Del objeto de tipo Statement, es ejecutado el método executeUpdate(), este método sirve para ejecutar una sentencia SQL que es pasada como un parámetro de tipo String. En este caso es ejecutado el SP con nombre **LOC_spu_sitio**, al cual le son pasados como parámetros todos los atributos de la clase Sitio.

9. Le es retornado al Operador un mensaje: De éxito si el sitio fue actualizado en BD correctamente o de error si por algún motivo no pudo realizarse el cambio.

Éstos son los diagramas de interacción para este caso de uso. Los diagramas de interacción para los casos de uso restantes, en lo que al *backend* se refiere, solo serán mostrados en un apéndice posterior a este capítulo debido a que tienen un funcionamiento similar a los anteriores.

A continuación se muestran los diagramas de interacción del *frontend* de la aplicación de esta tesis. Los primeros diagramas que son presentados son los del caso de uso Seleccionar Dirección.

4.2.2 Registro y Búsqueda Pendiente (*Frontend*)

Este procedimiento se lleva a cabo cuando el usuario desea ingresar a la aplicación móvil además de tener la posibilidad de continuar una búsqueda que tuviera pendiente debido a una desconexión producto de la inestabilidad que caracteriza a un medio de comunicación celular.

En la Figura 4.5 se muestra el diagrama de secuencia para este caso de uso. A continuación se describen los pasos que se siguen en dicho diagrama:

1. El usuario teclea el URL para ingresar a la aplicación. El primer objeto al que se accede es el servlet LocSrvPrincipal.
2. La aplicación le devuelve una pantalla de registro en donde se capturan el usuario y password.
3. El usuario ingresa el usuario y password que le hayan sido asignados.
4. La aplicación valida que el usuario y password sean correctos por medio del método validarUsr().
5. Si son correctos estos datos, este mismo servlet verifica si el usuario tiene una búsqueda pendiente.
6. Si se tiene una búsqueda pendiente, la aplicación redirecciona la petición hacia el objeto que presenta la pantalla en donde se quedó pendiente la búsqueda. De lo contrario la aplicación muestra al usuario el menú principal de la aplicación.

4.2.3 Seleccionar Dirección (*Frontend*)

Este procedimiento se lleva a cabo cuando el usuario desea conocer la ruta óptima, ya sea entre dos puntos dados o entre un punto dado y un sitio de interés. Un punto se refiere a una dirección dentro de un sistema de calles ordinario de cualquier ciudad del mundo. Entonces estas direcciones deben poder ser introducidas al sistema de una manera eficiente, tomando en cuenta las limitaciones de un teléfono celular explicadas en el capítulo 1.

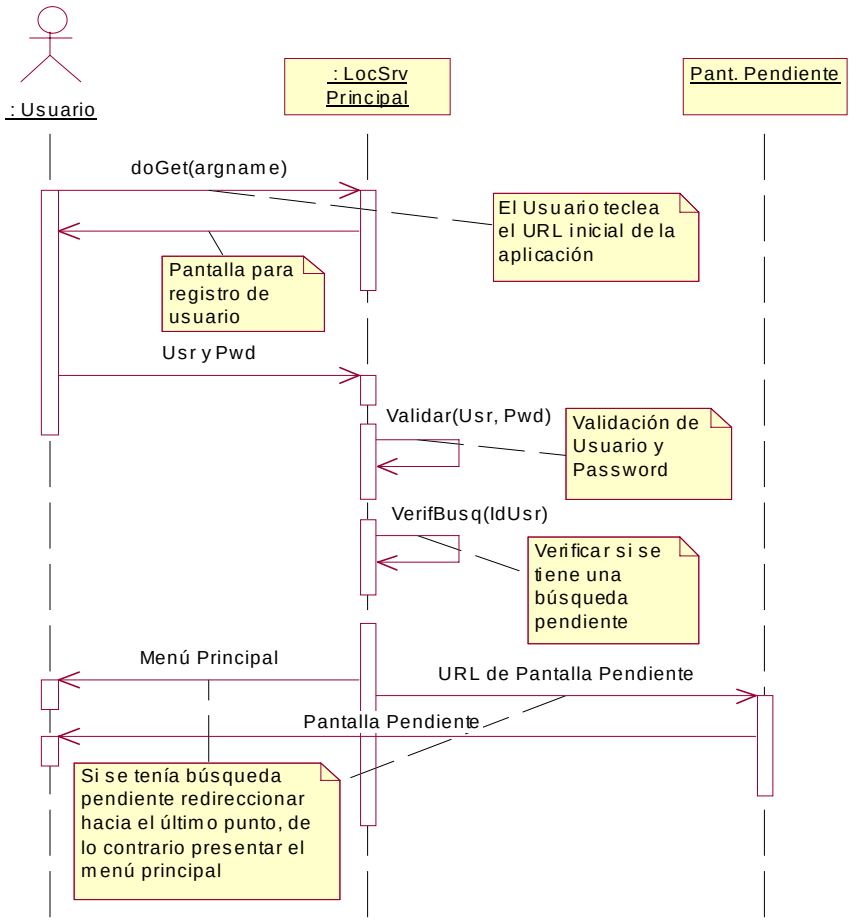


Figura 4.5 Diagrama de secuencia para el módulo de Registro y Búsqueda Pendiente

Por ejemplo, debido al escaso teclado con el que cuentan la mayoría de los celulares (teclado numérico), la captura de texto alfanumérico a través de éste se vuelve muy tardada y tediosa. Por esta razón podemos imaginar lo poco eficiente que sería que, para la captura de la dirección, el usuario tuviera que teclear el nombre de la calle. Además de que se prestaría demasiado a ambigüedades debido a que existen varias calles con el mismo nombre pero en diferentes colonias. Y así como la calle, también se tendría que capturar el nombre de la colonia y de la delegación, y esto solo para la entrada de una dirección (punto).

Otra opción sería que el usuario pudiera elegir la calle de entre una lista que el sistema le presentara. De esta manera se evitaría que el usuario tuviera que teclear la calle. El problema aquí es que la lista de calles a desplegar sería inmensa y no le sería fácil encontrar de manera rápida la que le interesara, debido a que otra de las limitantes de los teléfonos celulares es que poseen una pantalla (*display*) pequeña y solo le permitiría observar pocas calles a la vez. Además de que no se estaría tomando en cuenta

una limitante más al estar enviando grandes cantidades de información: El reducido ancho de banda que maneja la infraestructura celular.

Tomando en cuenta estas limitantes, es necesario idear la manera en como hacer más eficiente el proceso de captura de una dirección. La solución planteada e implementada en el sistema fue la siguiente: tomando en cuenta que el usuario no tuviera que teclear nombre alguno, se optó por mostrarle una lista de opciones de las cuales podría elegir las que fueran de su interés sólo que el proceso tendría una variante.

Para hacer un mejor uso del ancho de banda y que la lista de opciones fuera más específica, la selección de la dirección se realizaría de forma gradual: Inicialmente se le entregaría una lista con las delegaciones registradas en el sistema, fácilmente podría elegir la delegación de la dirección a través de las teclas direccionales del celular. Una vez elegida, el sistema sólo mostraría una lista de las colonias que pertenecen a esa delegación. Al tener una lista de colonias reducida se habría utilizado menos ancho de banda y el usuario localizaría con mayor rapidez la colonia de su interés.

El mismo caso se presentaría ahora al introducir la calle: al momento que el usuario eligiera la colonia, el sistema le presentaría una lista reducida de calles, al solo mostrarle aquellas que pertenecieran a dicha colonia. Por último, el usuario introduciría el número de la dirección a través del teclado dado que, en este caso, sería la forma más eficiente.

La Figura 4.6 muestra el diagrama de secuencia de este caso de uso. Los pasos en concreto que se llevan a cabo para seleccionar una dirección, ya sea de origen o destino, son:

7. El usuario ha elegido introducir al sistema una dirección, sea la dirección origen o destino el procedimiento de captura es el mismo. El servlet LocSrvDireccion es el encargado de atender la petición.
8. El método frameDelegacion() devuelve al usuario una página WML con la lista de delegaciones registradas en el sistema.
9. El usuario elige la delegación en donde se encuentra la dirección que desea ingresar, la petición se dirige nuevamente al servlet LocSrvDireccion.
10. El método frameColonia() devuelve al usuario una página WML con la lista de colonias que pertenecen a la delegación seleccionada.
11. El usuario elige la colonia de la dirección que desea ingresar, la petición se dirige al LocSrvDireccion
12. Ahora el método frameCalle() retorna una página WML con la lista de calles que pertenecen a la colonia que seleccionó previamente.
13. En esta ocasión, el usuario selecciona la calle de la dirección que requiere introducir, la petición se redirige al servlet.
14. Es el método frameNum() el que ahora envía una página en la que el usuario debe introducir el número de la calle de la dirección que requiere ingresar.
15. El usuario teclea el número. Después el sistema muestra la opción de Aceptar.
16. Una vez que el usuario elige la opción anterior, la petición regresa al servlet LocSrvDireccion, el cual crea un objeto de la clase Dirección. En los atributos de este objeto son almacenados los valores introducidos para la dirección elegida.

LocSrvDireccion manda llamar al servlet que generó el menú WML en el cual se había elegido introducir la dirección.

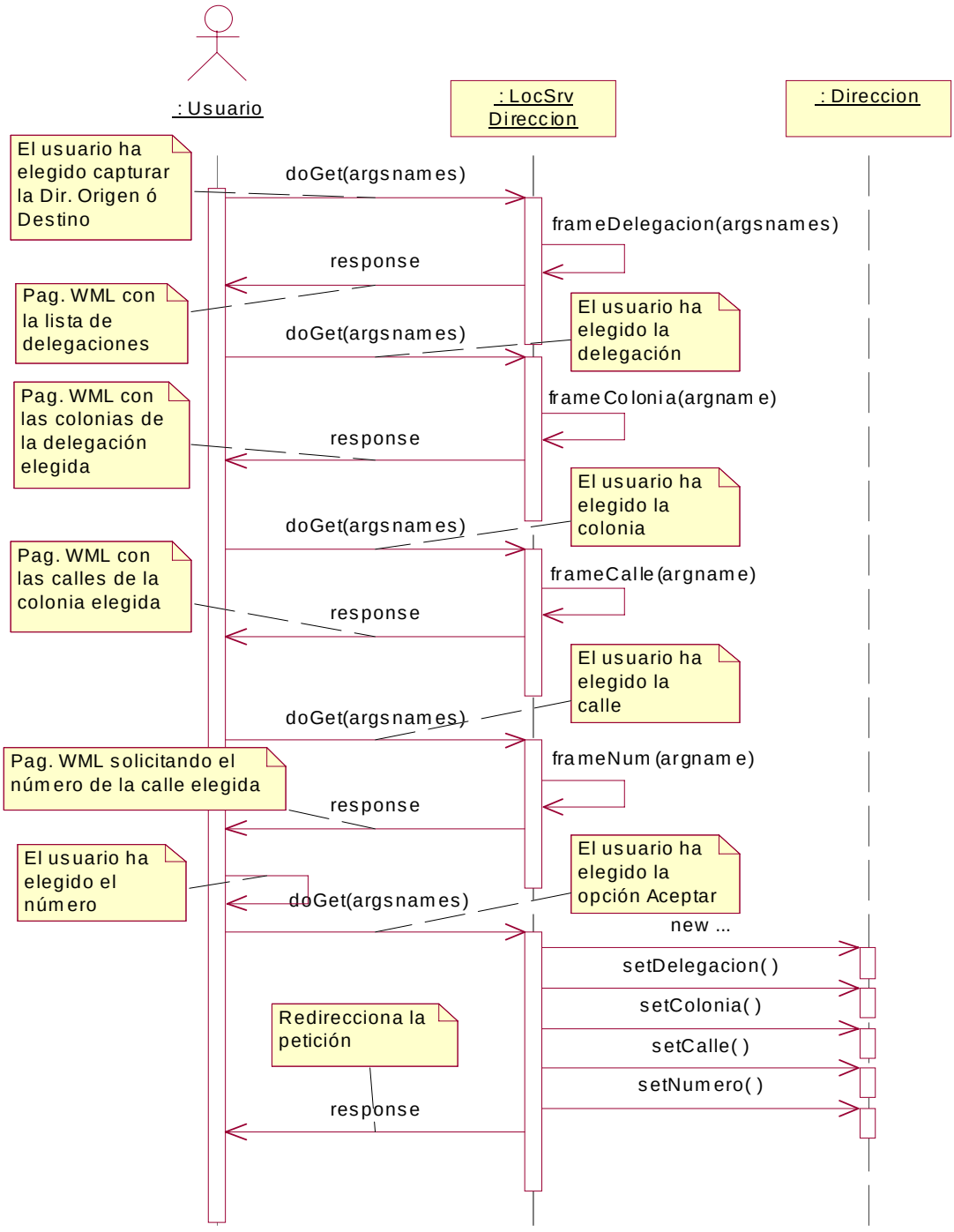


Figura 4.6 Diagrama de secuencia para cuando se selecciona una dirección

4.2.4 Seleccionar Sitio de Interés (*Frontend*)

Este procedimiento se aplica cuando el usuario desea conocer la ruta óptima desde un punto dado hacia un sitio de interés. Al igual que con las direcciones, en este caso también es necesario que el procedimiento de elección del sitio de interés sea eficiente, teniendo en cuenta las limitaciones que la infraestructura celular presenta.

Similar a la solución propuesta para la introducción de direcciones, la elección del sitio de interés se realiza mediante una búsqueda gradual. Como primer paso, el usuario tiene la opción de elegir el sitio a través de diferentes criterios, por ejemplo, de acuerdo a la categoría, delegación, colonia, etc.

El diagrama de secuencia de este caso de uso es presentado en la Figura 4.7. Los pasos que se llevan a cabo para seleccionar un sitio de interés son:

1. El usuario está interesado en encontrar la ruta hacía un sitio de interés, por lo cual ha elegido indicar el sitio en el que está interesado. El servlet LocSrvSitio es el encargado de atender la petición.
2. El método frameBuscarSitioPor() devuelve al usuario una página WML en la que el usuario debe elegir la manera en la que desea seleccionar el sitio.
3. En este punto, el usuario elige la opción de buscar el sitio de interés por medio de la categoría a la que pertenece.
4. El método frameBuscarSitioPorCateg() del servlet LocSrvSitio devuelve una página WML con una lista de todas las categorías a las que un sitio puede pertenecer.
5. El usuario elige la categoría del sitio de interés en el que está interesado.
6. El mismo método del punto 4 retorna ahora una página WML con la lista de todos los sitios de interés que pertenecen a la categoría previamente seleccionada.
7. El usuario tiene ahora dos opciones: elegir un sitio que le interese en particular o que el sistema busque el más cercano a la dirección origen dada.
8. El servlet LocSrvSitio se encarga ahora de almacenar en variables de sesión la dirección del sitio de interés elegido o una bandera que indique que el usuario desea encontrar el sitio de interés más cercano. El control es retornado al servlet que genera el menú para esta opción.

4.2.5 Buscar Ruta Óptima entre dos puntos (*Frontend*)

El procedimiento mostrado a continuación se utiliza para encontrar la ruta óptima que tenga como extremos dos puntos (direcciones) dados por el usuario. En la Figura 4.8 se presenta el diagrama de secuencia para el caso de uso **Buscar Ruta Óptima entre dos puntos**.

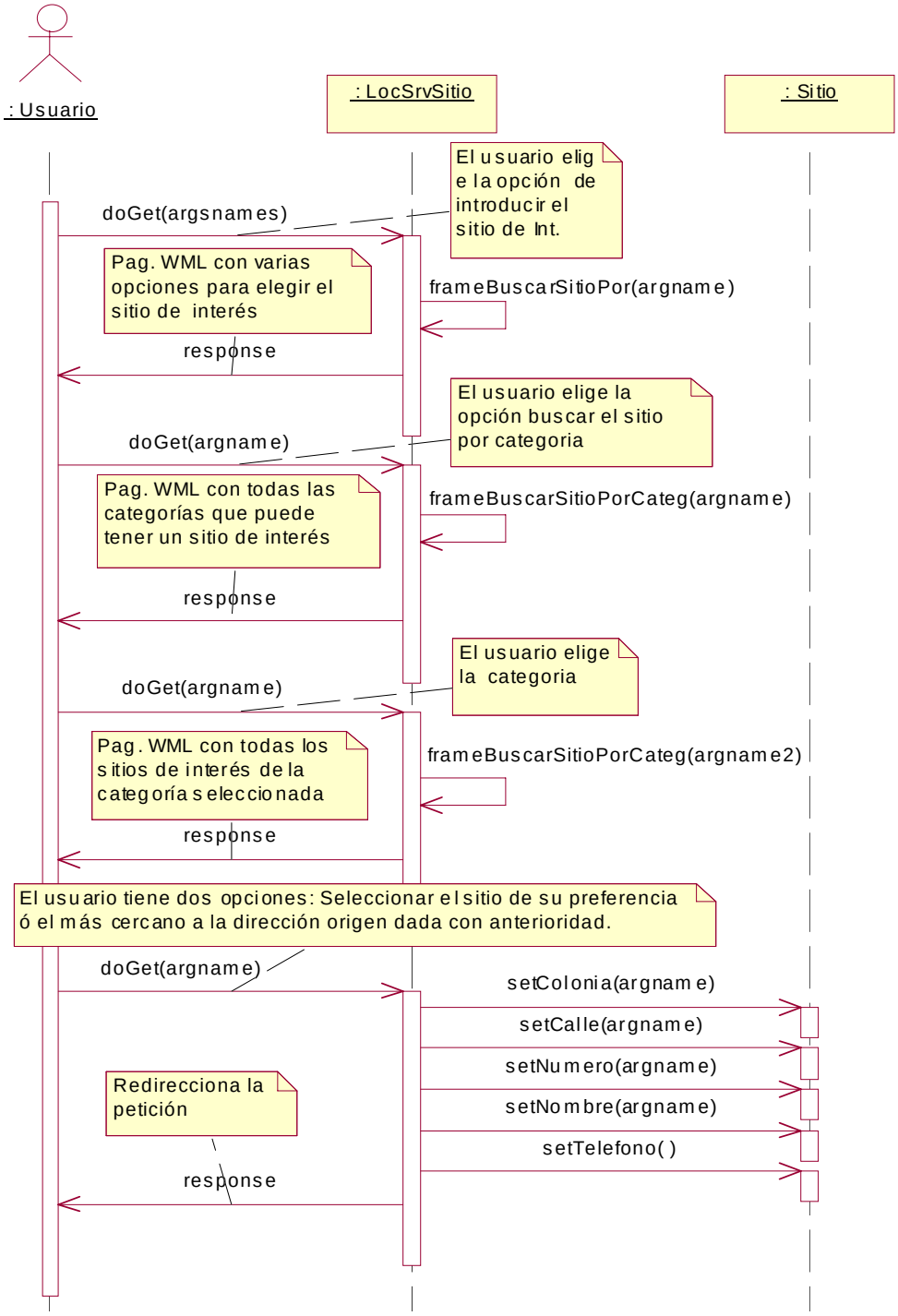


Figura 4.7 Diagrama de secuencia para cuando se selecciona un sitio de interés

Los pasos que se llevan a cabo son:

1. El usuario ha elegido esta opción de búsqueda de la ruta óptima desde el menú inicial.
2. El servlet LocSrvRutaDosPuntos presenta al usuario un menú con las opciones necesarias para realizar esta acción:

    - Introducir (seleccionar) la dirección origen
    - Introducir la dirección destino, y
    - Encontrar la ruta óptima entre estas direcciones

3. El usuario selecciona la opción de introducción de la dirección origen (se lleva a cabo el proceso descrito anteriormente: Seleccionar Dirección, en este caso la del punto origen).
4. La petición regresa al LocSrvRutaDosPuntos, el cual vuelve a generar y presentar el menú descrito en el punto 2.
5. El usuario ahora debe seleccionar introducir la dirección destino (el mismo proceso se vuelve a utilizar, solo que se crea un objeto Dirección adicional en donde es almacenada la dirección destino electa.)
6. Vuelve a retornar la petición al LocSrvRutaDosPuntos, el cual vuelve a generar el menú del punto 2.
7. En este punto, el usuario selecciona la opción: Encontrar Ruta Óptima.
8. Al elegir esta opción, se ejecuta el servlet LocSrvBuscarRutaDosPuntos, el cual se encarga de realizar lo necesario para encontrar la ruta óptima y retornársela al usuario en una página WML.
9. Es ejecutado el algoritmo de Dijkstra para encontrar la ruta óptima (de menor costo) entre dos puntos. La clase en donde está implementado el algoritmo lleva el nombre de Dijkstra.
10. Esta clase cuenta con un método llamado getNumNodosRutaOptima() el cual retorna un valor entero que indica el número de nodos que abarca la ruta óptima, incluyendo los nodos de origen y destino. Este valor se utiliza para dimensionar el arreglo en donde se almacenará la lista de calles que componen dicha ruta óptima.
11. Una vez que el arreglo ha sido dimensionado, el método getNombresCalles() retorna y asigna al arreglo la lista de calles que forman la ruta óptima obtenida.
12. Como se había comentado en el punto 8, LocSrvBuscarRutaDosPuntos crea una página WML con la lista progresiva de calles que forman la ruta óptima del punto origen al destino proporcionados.

4.2.6 Buscar Ruta Óptima hacía un sitio de interés (*Frontend*)

En la Figura 4.9 se muestra el diagrama de secuencia para cuando se desea obtener la ruta óptima desde un punto origen dado (dirección origen de la que iniciará la ruta óptima encontrada) y un sitio de interés. El sitio de interés puede ser el que el usuario elija mediante una serie de opciones o el más cercano si así lo decidiera.

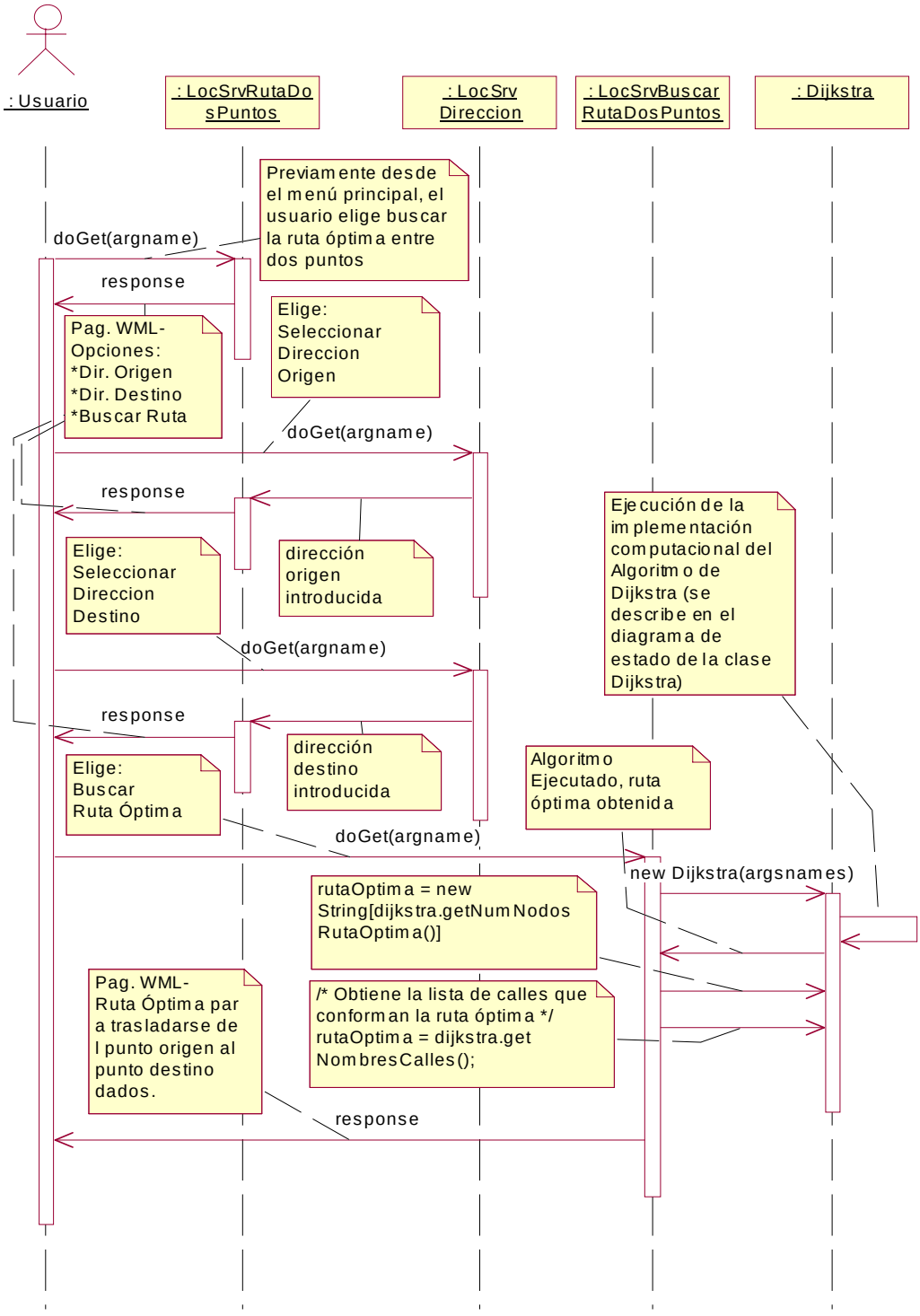


Figura 4.8 Diagrama de secuencia para cuando se busca la ruta óptima entre dos puntos

Los pasos que sigue el diagrama son:

1. El usuario ha elegido la opción de búsqueda de la ruta óptima de un punto origen a un sitio de interés desde el menú inicial.
2. El servlet LocSrvRutaSitio presenta al usuario un menú con las opciones necesarias para realizar esta acción:

    - Introducir la dirección origen
    - Elegir sitio de interés o el más cercano, y
    - Encontrar la ruta óptima entre estas direcciones

3. El usuario selecciona la opción de introducción de la dirección origen (se lleva a cabo el proceso descrito anteriormente: Seleccionar Dirección, en este caso la del punto origen).
4. La petición regresa al LocSrvRutaSitio, el cual vuelve a generar y presentar el menú descrito en el punto 2.
5. El usuario ahora debe seleccionar la opción de elegir el sitio de interés (en este caso se ejecuta el proceso: Seleccionar Sitio de Interés.)
6. Vuelve a retornar la petición al LocSrvRutaSitio, el cual vuelve a generar el menú del punto 2.
7. En este punto, el usuario selecciona la opción: Encontrar Ruta Óptima.
8. Al elegir esta opción, se ejecuta el servlet LocSrvBuscarRutaSitio, el cual se encarga de realizar lo necesario para encontrar la ruta optima y retornársela al usuario en una página WML.
9. Es ejecutado el algoritmo de Dijkstra para encontrar la ruta óptima (de menor costo) entre dos puntos. La clase en donde está implementado el algoritmo lleva el nombre de Dijkstra.
10. Esta clase cuenta con un método llamado getNumNodosRutaOptima() el cual retorna un valor entero que indica el número de nodos que abarca la ruta óptima, incluyendo los nodos de origen y el más cercano al sitio de interés elegido. Este valor se utiliza para redimensionar el arreglo que contendrá las calles que componen la ruta óptima.
11. Una vez que el arreglo ha sido dimensionado, el método getNombresCalles() retorna y asigna al arreglo la lista de calles que forman la ruta óptima obtenida.
12. Como se había comentado en el punto 8, LocSrvBuscarRutaSitio crea una página WML con la lista secuencial de calles que forman la ruta óptima solicitada.

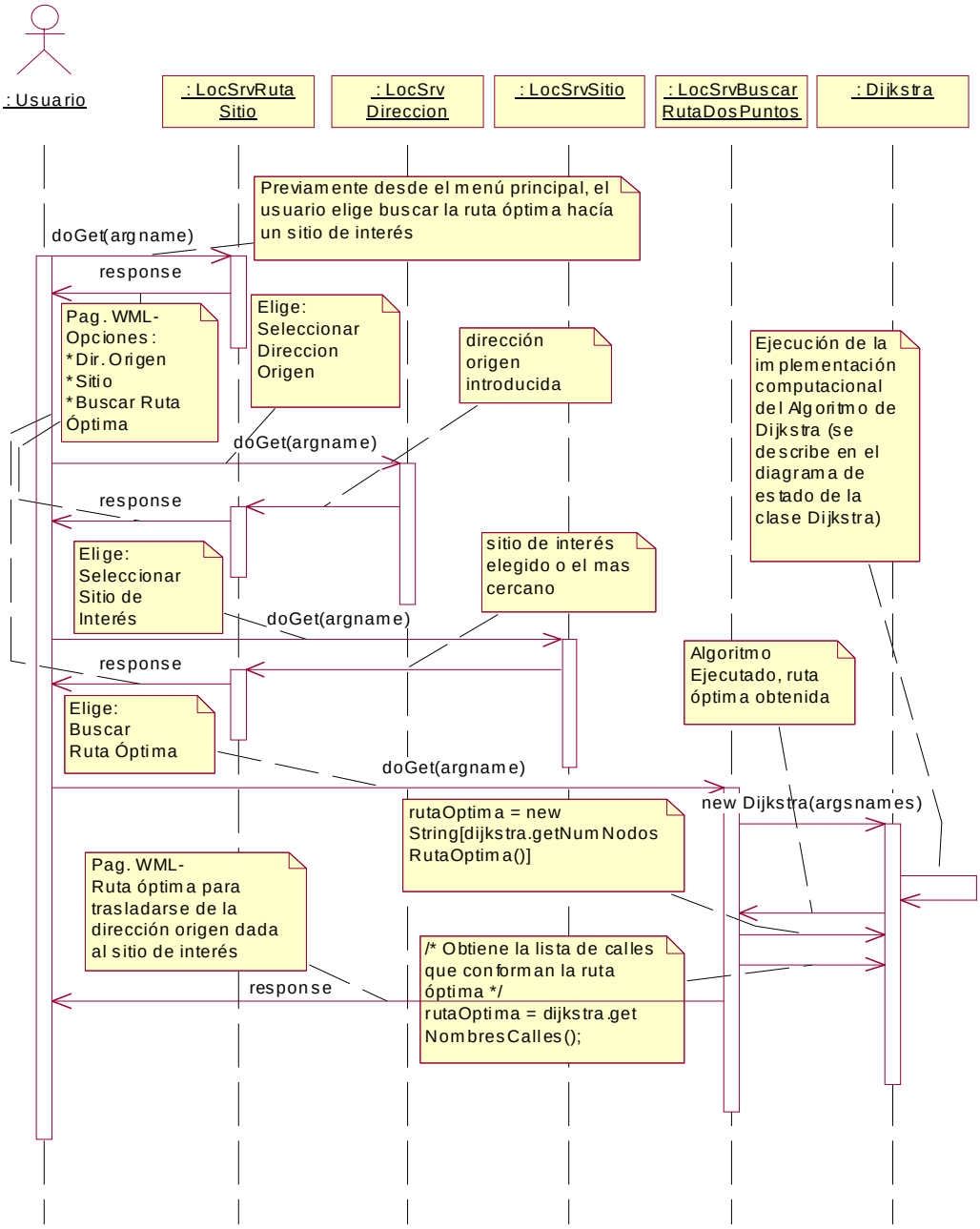


Figura 4.9 Diagrama de secuencia para cuando se busca la ruta óptima entre una dirección origen dada y un sitio de interés

4.3 Diagrama de Clases

En la sección anterior se mostraron los diagramas de interacción (principalmente diagramas de secuencia), en los cuales se pudo observar la manera en cómo los objetos cooperan entre si para realizar una tarea. A partir de los objetos que contienen estos diagramas, es posible obtener las clases que serán implementadas, así como los atributos y métodos (operaciones) que incluirán. Se comenzará por presentar y explicar las clases que serán parte del *backend*, posteriormente tocará turno para las del *frontend*.

Dando continuidad a los objetos participantes en los diagramas de secuencia del mantenimiento de sitios de interés, se explican las clases del sistema que ayudarán en esta labor. En la Figura 4.10 se muestra la definición de la clase Sitio, la cual servirá para almacenar los valores de un sitio de interés recuperados de BD, así como los que serán guardados en BD también.

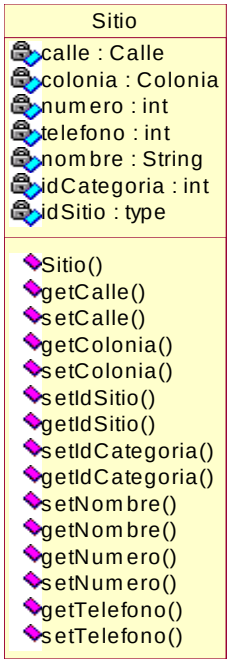


Figura 4.10 Definición de la clase Sitio perteneciente al backend

Esta clase pertenece al estereotipo entidad (entity). El estereotipo entidad es usado para los objetos que guarden información sobre el estado interno del sistema, a corto y largo plazo, correspondiente al dominio del problema. Todo comportamiento naturalmente acoplado con esta información también se incluye en los objetos entidad [WWW6].

Los atributos de la clase Sitio corresponden con los campos de la BD en donde se almacenarán los datos de un sitio de interés. Las operaciones son utilizadas para obtener o colocar valores en dichos atributos. Los métodos getXXX() se utilizan para obtener el valor de un atributo mientras que los setXXX() son usados para poner un valor a un atributo.

Como se puede observar, ninguno de los métodos de esta clase se encarga de consultar y manipular la información de BD de los sitios de interés, para esta labor se creó la clase MantSitio. En la Figura 4.11 se muestra la definición de esta clase.

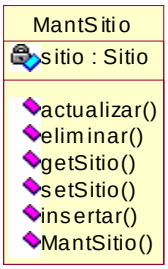


Figura 4.11 Definición de la clase MantSitio

El único atributo que posee esta clase en un objeto de la clase Sitio. Todos los métodos de la clase MantSitio actúan sobre este atributo. Primeramente tenemos un par de métodos que se encargan de asignar u obtener el atributo sitio para que la clase opere sobre él. El método insertar() se encarga de dar de alta en BD un nuevo sitio de interés, cada uno de los campos a insertar se obtienen del objeto sitio. De igual forma el método eliminar() borra de BD el sitio que tenga los mismos valores que el objeto sitio. Como último método tenemos actualizar(), el cual se encarga de modificar los valores del sitio de interés con los valores almacenados en el atributo sitio.

El resto de las clases del *backend* tienen un funcionamiento similar al de las dos clases anteriores, razón por la cual será omitida su explicación.

Ahora se presenta completo, en la Figura 4.12, el diagrama de clases correspondiente a la aplicación *backend* del sistema de localización desarrollado para esta tesis.

En el diagrama se puede observar una clase más, la clase LocAppMtto, la cual es una aplicación *standalone* de Java. Esta clase se encarga de toda la interfaz gráfica de usuario (GUI[5]) y la interacción del Operador con el sistema *backend*. Recordando que el backend es el sistema de mantenimiento de la estructura vial descrita en el capítulo anterior, así como el mantenimiento también de los sitios de interés del sistema.

[5] Graphic User Interface

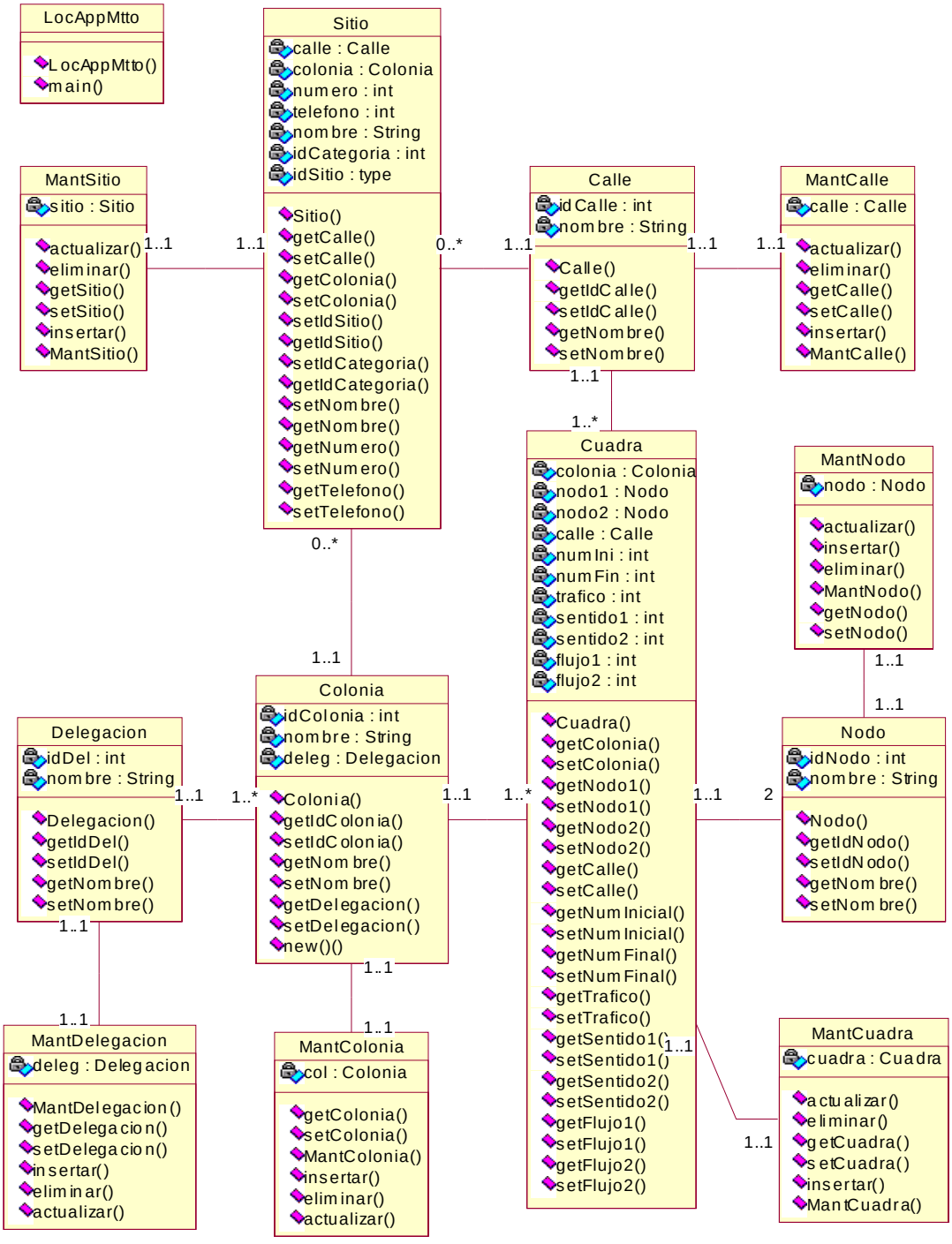


Figura 4.12 Diagrama de Clases de los objetos que componen el *backend*

Ahora se explicarán las clases del sistema correspondientes al *frontend*. La clase LocSrvPrincipal es un servlet que se encarga de presentar al usuario el menú principal de la aplicación WAP. En dicho menú se presentan dos opciones principales: La primera se implementa en el servlet LocSrvRutaDosPuntos, el cual nos permite encontrar una ruta entre dos puntos introducidos por el usuario y la segunda se implementa en el servlet LocSrvRutaSitio, el cual nos permite obtener una ruta de un punto introducido hacía un sitio de interés, ya sea elegido por el usuario o el más cercano al punto dado.

A su vez, LocSrvRutaDosPuntos hace uso de dos servlets más: LocSrvDireccion y LocSrvBuscarRutaDosPuntos. El primero de ellos permite introducir una dirección al sistema, como ya se había comentado con anterioridad, esto se realiza de forma gradual. La dirección introducida por el usuario es almacenada en un objeto de la clase Direccion. El segundo servlet se utiliza para conocer la ruta óptima que va desde el punto origen al punto destino dados. Para lograr esto utiliza la clase Dijkstra, la cual implementa el algoritmo de Dijkstra para encontrar rutas óptimas (también nombradas rutas mínimas, caminos mínimos, rutas de menor costo). Por último, presenta al usuario la lista de calles que conforman la ruta óptima obtenida.

Por otro lado, LocSrvRutaSitio utiliza tres servlets para lograr su cometido: LocSrvDireccionSitio, LocSrvSitio y LocSrvBuscarRutaSitio. El primero de ellos sirve para capturar la dirección del sitio origen, funciona de manera muy parecida al servlet LocSrvDirección explicado anteriormente. También almacena la información del punto en un objeto de la clase Direccion. El segundo servlet es utilizado para seleccionar un sitio de interés o, en su caso, indicarle al sistema que se desea encontrar la ruta óptima al sitio de interés, de la categoría elegida, más cercano al punto origen dado. Los datos del sitio de interés seleccionado se almacenan en un objeto de la clase Sitio perteneciente al *backend*. El tercer servlet se utiliza para conocer la ruta óptima que va del punto origen dado al sitio de interés elegido. También utiliza la clase Dijkstra para lograr encontrar la ruta de menor costo y, al igual que el anterior, presenta al usuario una lista con la calles que trazan la ruta óptima obtenida.

En la Figura 4.13 se muestra el diagrama de clases para el *frontend* de la aplicación WAP. Debido a que la clase Dijkstra realiza una de las tareas más complejas de la aplicación, en la Figura 4.14 se muestra el diagrama de estados de esta clase, en la cual se puede ver un poco más a detalle su funcionamiento.

Como se comentó anteriormente, los servlet LocSrvBuscarRutaDosPuntos y LocSrvBuscarRutaSitio son los encargados de obtener la ruta óptima. En el proceso de encontrar dicha ruta, cualquiera de los servlets anteriores crean una instancia de la clase Dijkstra pasándole, a través de su constructor, las direcciones de ambos puntos (ya sea de un punto origen y otro de destino, o uno de origen y el de un sitio de interés). En este momento se crean e inicializan muchas de las variables globales y objetos que participarán en el proceso del algoritmo. Otra tarea que se realiza en este punto es la de redimensionar los arreglos en donde se almacenará el grafo y los resultados de la ejecución del algoritmo.

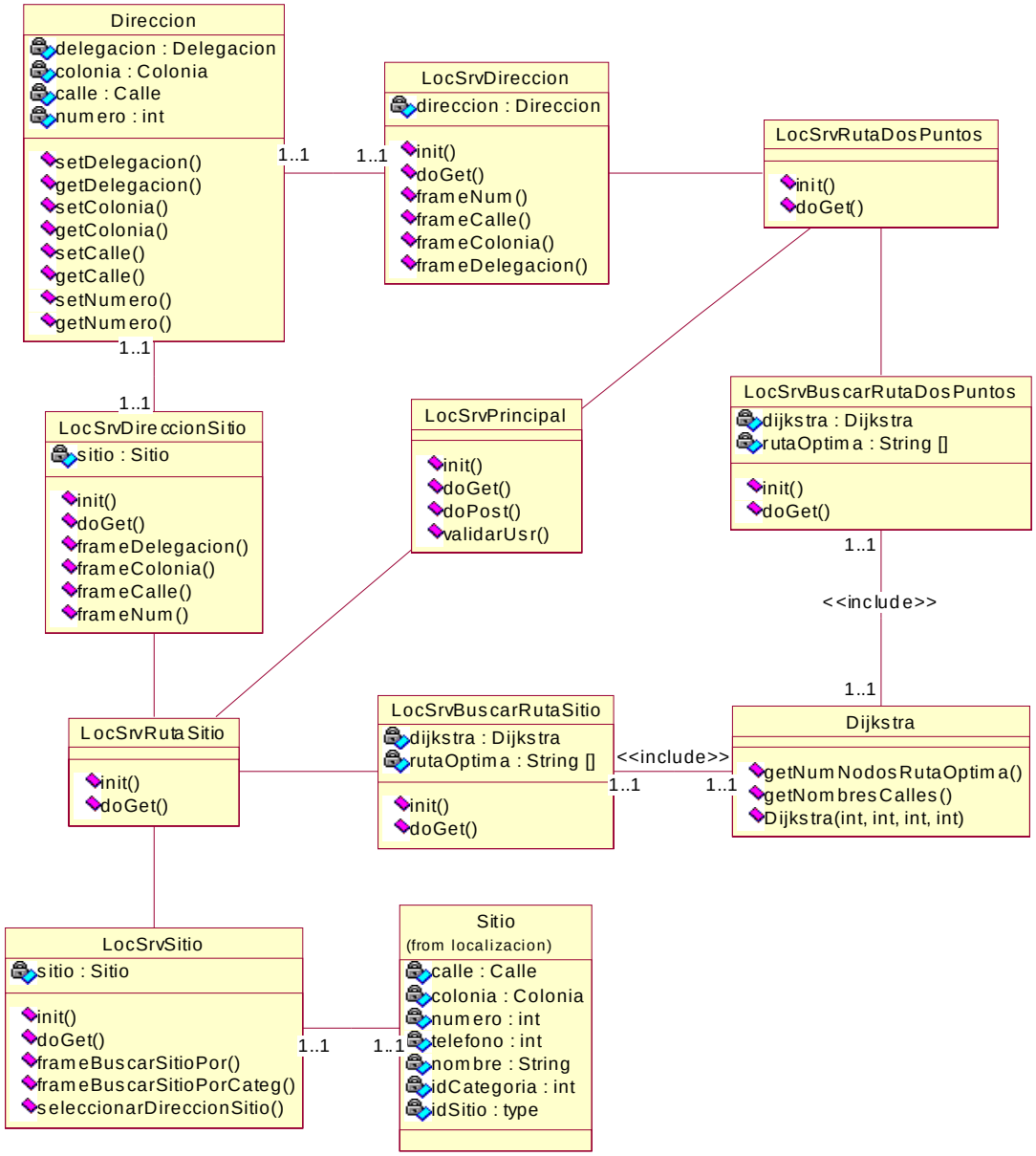


Figura 4.13 Diagrama de Clases de los objetos del *frontend*

El siguiente paso es construir (cargar) el grafo a partir de los datos guardados en BD, para esto se realiza una serie de consultas y los valores obtenidos se almacenan en

los arreglos inicializados. Además de esto se realiza un cálculo[6] para obtener el peso de trasladarse de nodo[7] a otro.

Con las direcciones (origen y destino) enviadas a través del constructor, el próximo paso es situarlas dentro del grafo y encontrar los nodos más cercanos a cada una de ellas, estos nodos se denominarán origen y destino. Con estos nodos ubicados se verifica que no se encuentren bloqueados: un nodo se encuentra bloqueado cuando no cuenta con un camino libre de algún tipo de obstáculo, que lo conecte con cualquier otro nodo del grafo. En el caso de que cualquiera de los nodos origen y destino se encontrara bloqueada, se finaliza el proceso de búsqueda de la ruta óptima, mandando un mensaje al usuario del bloqueo de alguno de los puntos o en su caso de ambos.

Si el nodo origen y el destino se encuentran desbloqueados, inicia la ejecución del algoritmo de Dijkstra inicializando objetos y variables propias del algoritmo, el nodo actual es igualado al nodo origen, todas las distancias son puestas a cero. Se ejecuta el primer paso del algoritmo en el que, dentro de los nodos alcanzables[8], solo se encuentra el nodo origen.

El siguiente paso es agregar el nodo actual a la ruta óptima y calcular las distancias a los nodos próximos. Es en este paso (paso A) donde se pregunta sí el nodo destino ya forma parte de los nodos alcanzables: Si lo anterior ya se ha cumplido significa que ya se tiene la ruta óptima del nodo origen al destino, por lo que se procede a obtener de BD los nombres de las calles que forman la ruta óptima obtenida.

Si por el contrario, dentro de los nodos alcanzables no está aún el nodo destino se procede a realizar un nuevo cálculo de los costos a los siguientes nodos accesibles desde los que hasta el momento conforman la ruta óptima. Si la distancia de alguno de los nodos próximos es mayor que la distancia del nodo actual más la arista del nodo actual a ese nodo próximo, esto indica que se ha obtenido un nodo con un costo menor. Por lo anterior se lleva a cabo una actualización de los caminos de costo mínimo además de igualar el nodo actual al nodo recién obtenido. El algoritmo continúa con la ejecución del paso A nuevamente hasta que se alcance el nodo destino.

Se puede presentar el caso en el que ya no se obtengan nodos alcanzables nuevos, en cuyo caso la ejecución del algoritmo finaliza, enviando al usuario un mensaje de que no se encontraron rutas disponibles del nodo origen al destino.

[6] El detalle de este cálculo se muestra más adelante en la parte de la implementación.
[7] Un nodo de manera física, para esta aplicación, se refiere a la intersección de dos o más cuadras.
[8] Nodos alcanzables se refiere a aquellos nodos a los que se puede llegar desde el nodo origen, conforme se va ejecutando el algoritmo de Dijkstra.

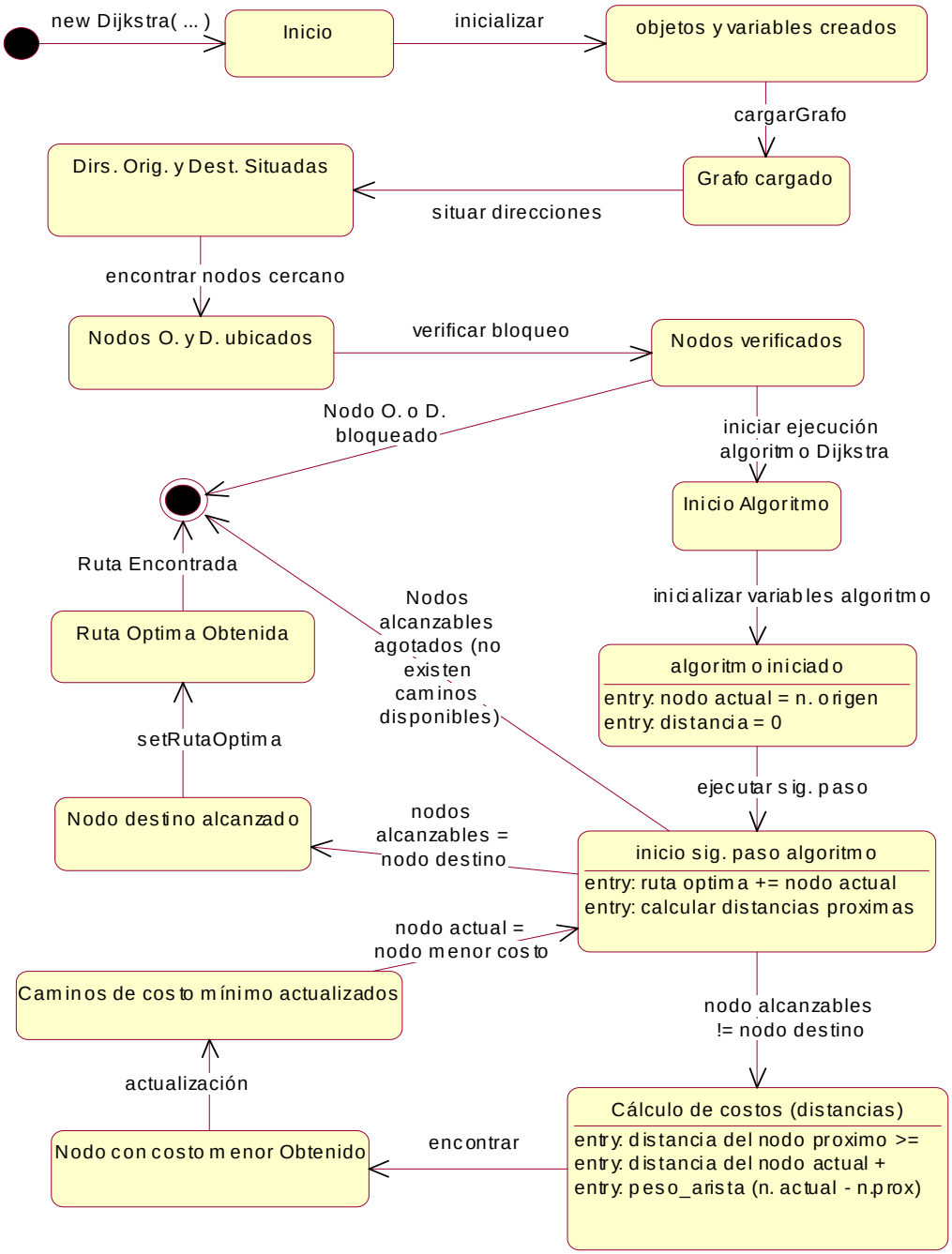


Figura 4.14 Diagrama de estados de la clase Dijkstra (Implementación del Algoritmo de Dijkstra)

4.4 Depósito de Datos

El depósito de datos proporciona los medios y herramientas para el almacenamiento y manipulación de la información que el sistema maneja. Está constituido por una base de datos relacional que se encarga de la persistencia de la información y por un conjunto de servicios que permiten consultar y modificar dicha información.

El depósito de datos realiza las siguientes funciones:

- **Control sobre la redundancia de datos.** Los sistemas de archivos almacenan varias copias de los mismos datos en archivos distintos. Esto hace que se desperdicie espacio de almacenamiento, además de provocar la falta de consistencia de datos. En los sistemas de bases de datos todos estos archivos están integrados, por lo que no se almacenan varias copias de los mismos datos. Sin embargo, en una base de datos no se puede eliminar la redundancia completamente, ya que en ocasiones es necesaria para modelar las relaciones entre los datos, o bien es necesaria para mejorar las prestaciones.

- **Consistencia de datos.** Eliminando o controlando las redundancias de datos se reduce en gran medida el riesgo de que haya inconsistencias. Si un dato está almacenado una sola vez, cualquier actualización se debe realizar sólo una vez, y está disponible para todos los usuarios inmediatamente. Si un dato está duplicado y el sistema conoce esta redundancia, el propio sistema puede encargarse de garantizar que todas las copias se mantienen consistentes. Desgraciadamente, no todos los DBMS de hoy en día se encargan de mantener automáticamente la consistencia.

- **Más información sobre la misma cantidad de datos.** Al estar todos los datos integrados, se puede extraer información adicional sobre los mismos.

- **Compartición de datos.** En los sistemas de archivos, los archivos pertenecen a las personas o a los departamentos que los utilizan. Pero en los sistemas de bases de datos, la base de datos pertenece a la empresa y puede ser compartida por todos los usuarios que estén autorizados. Además, las nuevas aplicaciones que se vayan creando pueden utilizar los datos de la base de datos existente.

- **Mantenimiento de estándares.** Gracias a la integración es más fácil respetar los estándares necesarios, tanto los establecidos a nivel de la empresa como los nacionales e internacionales. Estos estándares pueden establecerse sobre el formato de los datos para facilitar su intercambio, pueden ser estándares de documentación, procedimientos de actualización y también reglas de acceso.

Para la implantación de la base de datos se hará uso de un Sistema Manejador de Base de Datos (DBMS[9]), las principales ventajas de utilizar un DBMS son las siguientes:

[9] DataBase Manager System

- **Mejora en la integridad de datos.** La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el DBMS quien se debe encargar de mantenerlas.

- **Mejora en la seguridad.** La seguridad de la base de datos es la protección de la misma frente a usuarios no autorizados. Sin unas buenas medidas de seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de archivos. Sin embargo, los DBMS permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la base de datos. Las autorizaciones se pueden realizar a nivel de operaciones, de modo que un usuario puede estar autorizado a consultar ciertos datos pero no a actualizarlos, por ejemplo.

- **Mejora en la accesibilidad a los datos.** Muchos DBMS proporcionan lenguajes de consultas o generadores de informes que permiten al usuario hacer cualquier tipo de consulta sobre los datos, sin que sea necesario que un programador escriba una aplicación que realice tal tarea.

- **Mejora en la productividad.** El DBMS proporciona muchas de las funciones estándar que el programador necesita escribir en un sistema de archivos. A nivel básico, el DBMS proporciona todas las rutinas de manejo de archivos típicas de los programas de aplicación. El hecho de disponer de estas funciones permite al programador centrarse mejor en la función específica requerida por los usuarios, sin tener que preocuparse de los detalles de implementación de bajo nivel. Muchos DBMS también proporcionan un entorno de cuarta generación consistente en un conjunto de herramientas que simplifican, en gran medida, el desarrollo de las aplicaciones que acceden a la base de datos. Gracias a estas herramientas, el programador puede ofrecer una mayor productividad en un tiempo menor.

- **Aumento de la concurrencia.** En algunos sistemas de archivos, si hay varios usuarios que pueden acceder simultáneamente a un mismo archivo, es posible que el acceso interfiera entre ellos de modo que se pierda información o, incluso, que se pierda la integridad. La mayoría de los DBMS gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.

Posteriormente, en el siguiente capítulo, se dan más detalles acerca del DBMS elegido para esta aplicación y de las funciones que realiza. Ahora se presenta el diagrama entidad relación para el diseño de la base de datos de la aplicación desarrollada.

4.4.1 Modelo Lógico

En la Figura 4.15 se presenta el modelo lógico de la base de datos del sistema.

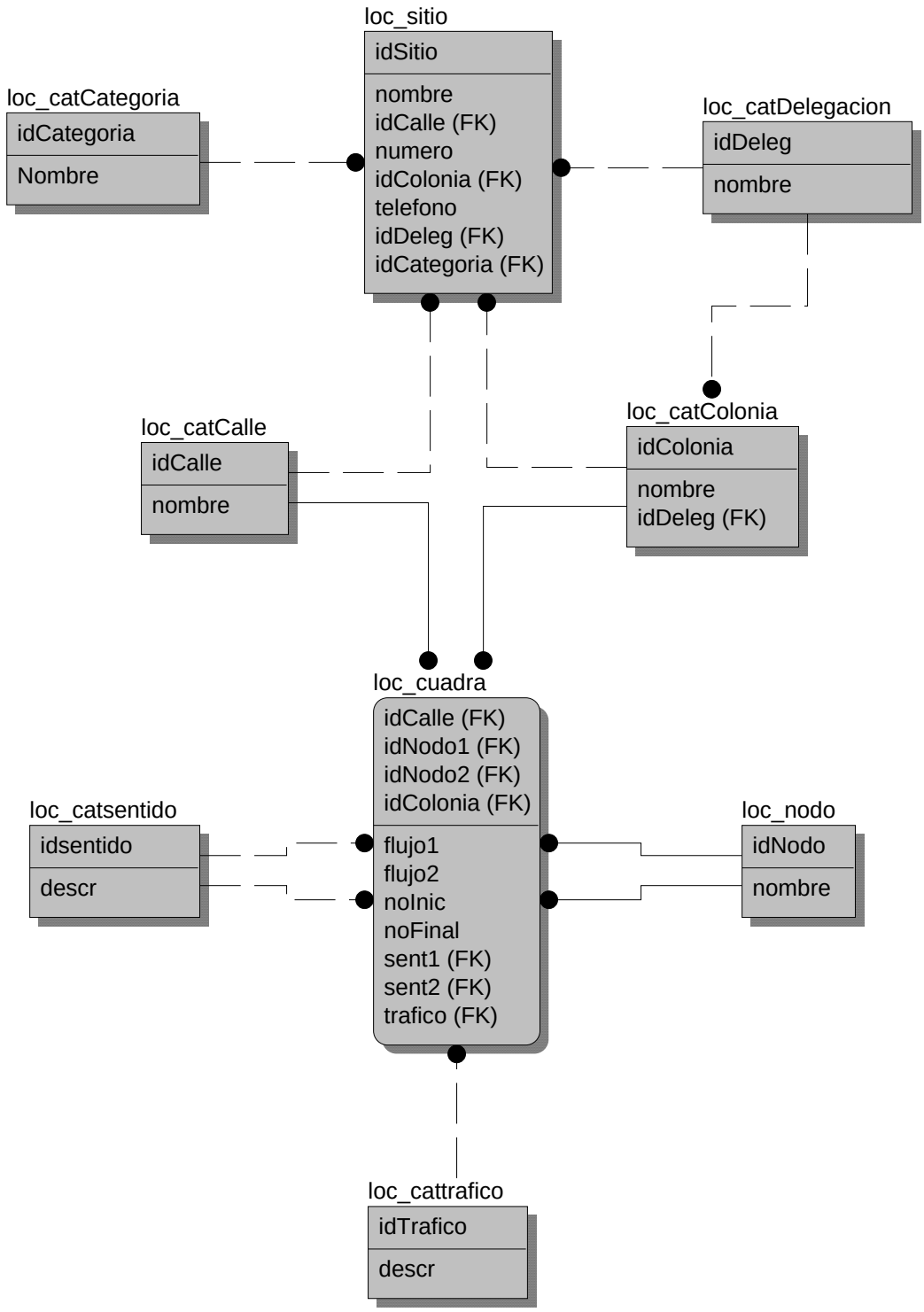


Figura 4.15 Modelo lógico de la base de datos del sistema

4.5 Generación del Grafo (Estructura Vial) a partir de BD

En este punto se explica una parte importante en la concepción y desarrollo de este trabajo: La generación del grafo a partir de la información almacenada en la base de datos. Como se dijo en el capítulo anterior, el grafo representa a la red de calles, avenidas, ejes, circuitos, etc., pertenecientes a una ciudad, por las que un automóvil puede circular para trasladarse de un lugar a otro. A continuación se detalla la forma en como la información guardada en BD se puede mapear a esta red vial (grafo).

Se comenzará por definir el elemento vial más básico e importante, a partir del cual se conformarán el resto de los elementos (calles, avenidas, etc.). A tal elemento se le nombrará **cuadra**. En la Figura 4.15 que muestra el Modelo lógico de la base de datos, se puede ver que como resultado de esta relación es creada una nueva entidad con el nombre **loc_cuadra**. Dicha entidad contiene los atributos necesarios para construir el grafo a partir de la información que en ella se almacene.

Tomando en cuenta esta información decimos que, una cuadra puede ser vista como una arista en un grafo. Los nodos, de donde parte y hacia donde se dirige dicha arista, están definidos por los cruces que forman la cuadra en sí, y las calles que la delimitan. Los pesos o costos que llevará una arista se obtienen de realizar un cálculo a partir de los atributos: flujo1, flujo2 y tráfico (el detalle de dicho cálculo se presenta y explica en el capítulo de la implementación). De esta manera es como se forma el grafo a partir de cada registro contenido en la tabla **loc_cuadra**, auxiliándose de las tablas con las que está relacionada. En la Figura 4.16 se describen cada uno de los atributos que definen a una cuadra dentro del sistema y que son representados en la tabla **loc_cuadra**.

A continuación se presenta un ejemplo de como es formado el grafo a partir de la información contenida en **loc_cuadra**. En la Figura 4.17 se presenta un fragmento de red vial de la zona norte de la Ciudad de México, aquí se puede ver que las intersecciones de las calles son convertidas en los nodos que formarán al grafo, y que las aristas serán representadas por la calle intermedia entre dos nodos. En la Figura 4.18 se muestra el grafo obtenido del fragmento de red vial mostrada en la Figura 4.17, los nodos llevan como etiqueta el número que se le asignó desde esta última figura. También se ven representados las calles a través de aristas. En el caso de aristas que van de un nodo a otro y viceversa, se trata de una calle con dos direcciones, en el caso de que solo la arista vaya de un nodo a otro, se trata de una calle unidireccional. En el ejemplo, la etiqueta de la arista indica el número de carriles que contiene la calle representada por dicha arista.

En la Figura 4.19, se muestran las tablas y catálogos que intervienen en la creación del grafo y de como representado y almacenado un grafo en la base de datos. En la tabla **loc_cuadra** de muestra la información que define las calles Instituto Politécnico Nacional y Montevideo.

Atributo	Descripción
idCalle	Es el identificador de la calle a la cual corresponde la cuadra que se está definiendo. Este identificador se relaciona con el de la tabla loc_catCalle, la cual contiene un registro de todas las calles dadas de alta en el sistema.
idColonia	Es el identificador de la colonia en la que esta localizada la calle a la que se le está definiendo una cuadra. Corresponde con las colonias almacenadas en la tabla
idNodo1, idNodo2	Se refieren a los extremos que definen la cuadra, nodo 1 y nodo 2 respectivamente. Como ya se había dicho, un nodo se encuentra definido por la intersección de dos o más calles.
flujo1	Define el número de carriles disponibles para que un automóvil circule del nodo 2 hacia el nodo 1. Un valor igual a cero para este atributo indica que no se tiene vialidad en ese sentido para el tramo que define esta cuadra.
flujo2	El mismo caso que en el atributo anterior solo que en sentido inverso, del nodo 1 hacia el nodo 2.
noInic, noFinal	Estos atributos definen el rango de números de la calle que abarca esta cuadra.
sent1	Se refiere a la dirección cardinal cuando un automóvil se dirige del nodo 2 hacia el nodo 1. Los sentidos disponibles son tomados del catálogo loc_catSentido.
sent2	Aplica lo mismo que para el atributo anterior, solo que este se emplea para cuando el automóvil se desplaza del nodo 1 hacia el nodo 2.
trafico	Se refiere a las condiciones de tráfico actuales que afectan a esa cuadra. Estas condiciones están basadas en la experiencia del operador y van desde un valor de tráfico *normal* hasta *muy pesado*. Si por alguna razón esta cuadra se encuentra bloqueada, por ejemplo, debido a una obra pública, manifestación, accidente, etc., el operador también lo puede indicar por medio de este atributo.

Figura 4.16 Descripción de atributos que definen a una cuadra (tabla loc_cuadra)

Figura 4.17 Fragmento de red vial de la Ciudad de México

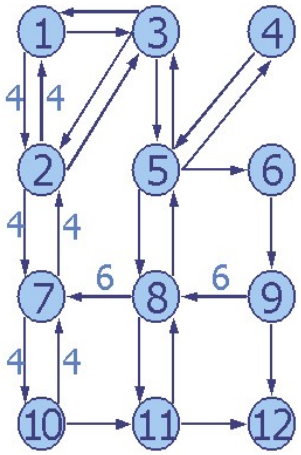


Figura 4.18 Grafo correspondiente al fragmento de la red vial (Figura 4.17)

loc_catDelegación

idDeleg	nombre
7	Gustavo A Madero

loc_catColonia

idColonia	nombre	idDeleg
23	Lindavista	7

loc_catCalle

idCalle	nombre
35	Ticomán
36	IPN
37	Insurgentes
38	Montevideo

loc_cuadra

idCalle	idNodo1	idNodo2	idColonia	flujo1	flujo2	noInic	noFinal	sent1	sent2	tráfico
36	1	2	23	4	4	22	89	NE	SO	1
36	2	7	23	4	4	90	295	NE	SO	1
36	7	10	23	4	4	296	531	NE	SO	1
38	7	8	23	6	0	121	452	NO	SE	1
38	8	9	23	6	0	120	1	NO	SE	1

Figura 4.19 Información contenida en la base de datos que representa una parte del grafo (Figura 4.18)

4.6 Resumen

En este capítulo se presentó la etapa de diseño de la aplicación de esta tesis. Como primera parte se muestran los diagramas de Interacción, los cuales se conforman de los diagramas de Secuencia y diagramas de Colaboración. Posteriormente se presentan el diagrama de Clases de la aplicación. Del Algoritmo de Dijkstra se presenta su diagrama de Estados, el cual muestra la ejecución del algoritmo y de como logra encontrar una ruta óptima.

Por la parte del Depósito de Datos se presenta el Modelo Lógico de la base de datos, en la cual se observa como están formadas las tablas y como es la relación entre ellas. Posteriormente se explica como se lleva a cabo la generación del grafo a partir de la información contenida en la base de datos.

En el siguiente capítulo se presentan los detalles de la implantación de la base de datos y la aplicación del sistema desarrollado, así como también se presentan las herramientas utilizadas en dicha etapa (como son el DBMS, servidor *Web*, *WAP Gateway*, etc.). Posteriormente, se presentan los resultados de las pruebas realizadas a la aplicación.

Capítulo 5

Implementación y Pruebas de la Aplicación

En este capítulo se describe la manera en cómo se implementaron cada uno de los componentes del sistema, así como las pruebas que comprueban su correcto funcionamiento.

Primeramente se describen, de manera breve, cada una de las herramientas utilizadas en esta etapa. Después se realiza una explicación sobre los detalles de implementación de cada uno de los módulos de la aplicación. Se da una explicación más detallada de los procesos más importantes como son la construcción del grafo, la programación del Algoritmo de Dijkstra y de la Generación de Páginas WML, además se presenta el Diagrama de Despliegue.

Por último, se presentan las pruebas realizadas al sistema y los resultados obtenidos.

5.1 Introducción

En este capítulo se presentan tanto los detalles de implementación de la aplicación desarrollada en esta tesis, como de las pruebas realizadas para verificar el correcto funcionamiento del mismo y así verificar que se cumplen los objetivos planteados en el capítulo inicial del presente documento.

Una de las tareas que se realiza en la etapa de implementación es la generación del código fuente final, de acuerdo a los elementos obtenidos de la etapa de diseño (diagramas, modelos, esquemas, etc.). En esta etapa también se lleva a cabo la elección del lenguaje de programación y del manejador de base de datos que mejor se adapten a lo especificado en la etapa de diseño. Aunque el diseño de objetos sea independiente del lenguaje utilizado, todos estos lenguajes tienen sus particularidades, las cuales deberán adecuarse durante la implementación final.

Como primer punto de este capítulo se presenta una breve descripción de cada una de las herramientas empleadas para el desarrollo de la implementación del proyecto. Posteriormente se lleva a cabo una explicación de la ejecución de la fase de implementación para los principales módulos tanto del *backend* como del *frontend*. Más adelante se presentan los detalles de implementación del proceso de construcción del grafo a partir de datos almacenados en la BD. Continuando con el desarrollo del capítulo, se realiza una explicación de cómo fue desarrollado y programado el algoritmo de Dijkstra para la obtención de rutas óptimas en grafos.

Otra parte fundamental para el desarrollo del *frontend* es la generación de páginas WML. Estas páginas son usadas para generar la interfaz que permita al usuario interactuar con el *frontend*. Además de que también, a través de este medio, se le presenta información y proporciona una respuesta al usuario cuando realiza una petición.

Continuando con el proceso de desarrollo de software con UML para esta etapa, es presentado el diagrama de despliegue para esta aplicación.

Finalmente se presenta la etapa de pruebas de la aplicación realizada, en la cual se verifica que ésta funcione de acuerdo a los casos de uso presentados y analizados en capítulos previos.

5.2 Herramientas Utilizadas

Para llevar a cabo la implementación de la aplicación, hubo la necesidad de evaluar un conjunto de herramientas de software que se utilizarían para cada uno de los módulos de la arquitectura del sistema planteada con anterioridad. En la Figura 5.1 se muestra un listado de dichas aplicaciones.

Nombre	Marca	Descripción
WapIDE v3.1.1	Ericsson Corporation	Ambiente para Desarrollo WAP (Navegador WAP y Editor WML).
Mobile Gateway v3.00	Realwow Ltd.	Gateway WAP
Resin v2.1.0	Caucho Technology	Servidor Web
Visual Café 4.1 EE	WebGain Corporation	Ambiente de Desarrollo Java.
SQLServer 2000	Microsoft Corporation	Sistema Manejador de Base de Datos

Figura 5.1 Herramientas de software utilizadas en la etapa de implementación

A continuación se describen con más detalle estas herramientas, presentando sus características principales.

5.2.1 Ericsson - WapIDE v3.1.1

WapIDE es un Kit de Desarrollo de Software (SDK) que permite a los desarrolladores crear y probar aplicaciones WAP. Es un Ambiente de Desarrollo Integrado (IDE[1]) para WAP muy completo, el cual incluye tres tipos de clientes: WAP, HTTP y de archivo. Esto significa que el usuario puede navegar y probar aplicaciones localizadas en un servidor HTTP de manera directa, de manera indirecta a través de un gateway WAP o desde el disco.

Los principales elementos del WapIDE son el Navegador WAP (*WAP Browser*) y el Diseñador de Aplicaciones (*Application Designer*), ambos descritos a continuación.

5.2.1.1 Navegador WapIDE

El navegador WapIDE permite al desarrollador visualizar y probar contenido WAP en diferentes modelos de teléfonos Ericsson (dispositivos Ericsson soportados actualmente: R320s, R380s, R520m y T39). Esto le permite al usuario examinar el comportamiento de una determinada aplicación sobre navegadores con diferentes características, tales como el soporte de gráficos, el tamaño de la pantalla, etc. Esto le dará al usuario una retroalimentación de cómo se percibiría la aplicación.

El navegador puede obtener información desde un Servidor Web a través de un Gateway WAP, o también desde un disco duro local, así como se muestra en la Figura 5.2.

En la Figura 5.3 se muestra la ventana principal del navegador WapIDE, la cual provee un conjunto de opciones que permiten la configuración de diversos elementos. Por ejemplo, es posible indicar el *gateway* WAP que se quiere sea utilizado para conectarse a un servidor Web del cual obtenga información. Para mejorar el desempeño, el WapIDE mantiene una memoria de caché no persistente donde los elementos descargados son almacenados (WML, WMLScript e imágenes).

[1] Integrated Development Environment

Figura 5.2 Formas de obtención de contenido del navegador WapIDE

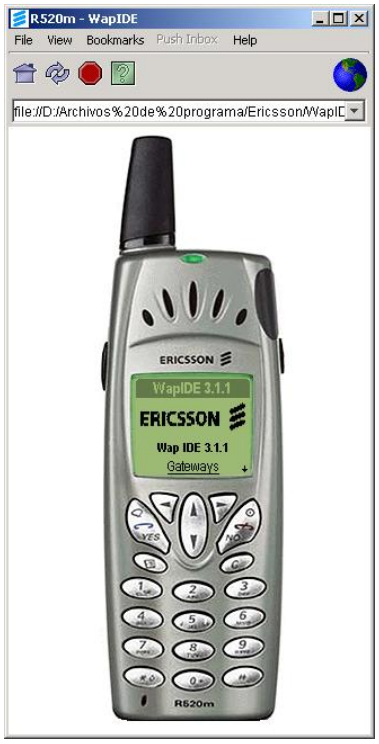


Figura 5.3 Ventana principal del navegador WapIDE

5.2.1.2 Diseñador de Aplicaciones WapIDE

El Diseñador de Aplicaciones WapIDE permite al usuario desarrollar aplicaciones con WML y WMLScript en modo WYSIWYG[2] Proporciona un completo ambiente de desarrollo de aplicaciones WAP que facilitan la edición y creación del código fuente, ya

[2] What You See Is What You Get.- "Lo que ves es lo que obtienes"

que se puede hacer de manera automática al insertar de forma visual un elemento WML. Entre las características más importantes están las siguientes:

- Incorpora un editor de código, así como un navegador integrado dentro del entorno.
- Proporciona una barra con diversos botones que insertan las correspondientes etiquetas en el código sin necesidad de ser escritas por el programador. Da la posibilidad de incluir las etiquetas más usuales de una manera muy sencilla.
- Todos los desarrollos que se realizan se almacenan como proyectos, de manera que un proyecto es una agrupación de uno o más archivos WML, esta forma de trabajo es mucho más lógica y proporciona una visión más estructurada de los desarrollos.
- Compila y verifica el código fuente WML.
- Cuando se compila una aplicación se tiene en la ventana de salida (Output) toda la información sobre el proceso, lo que será de una gran ayuda cuando se tengan problemas.

En la Figura 5.4 se tiene la pantalla principal del Diseñador de Aplicaciones WapIDE. Se puede ver que cuenta en su parte superior con una serie de íconos con los diferentes elementos WML. Éstos son usados para insertar elementos de forma visual y que automáticamente se inserte el código WML correspondiente.

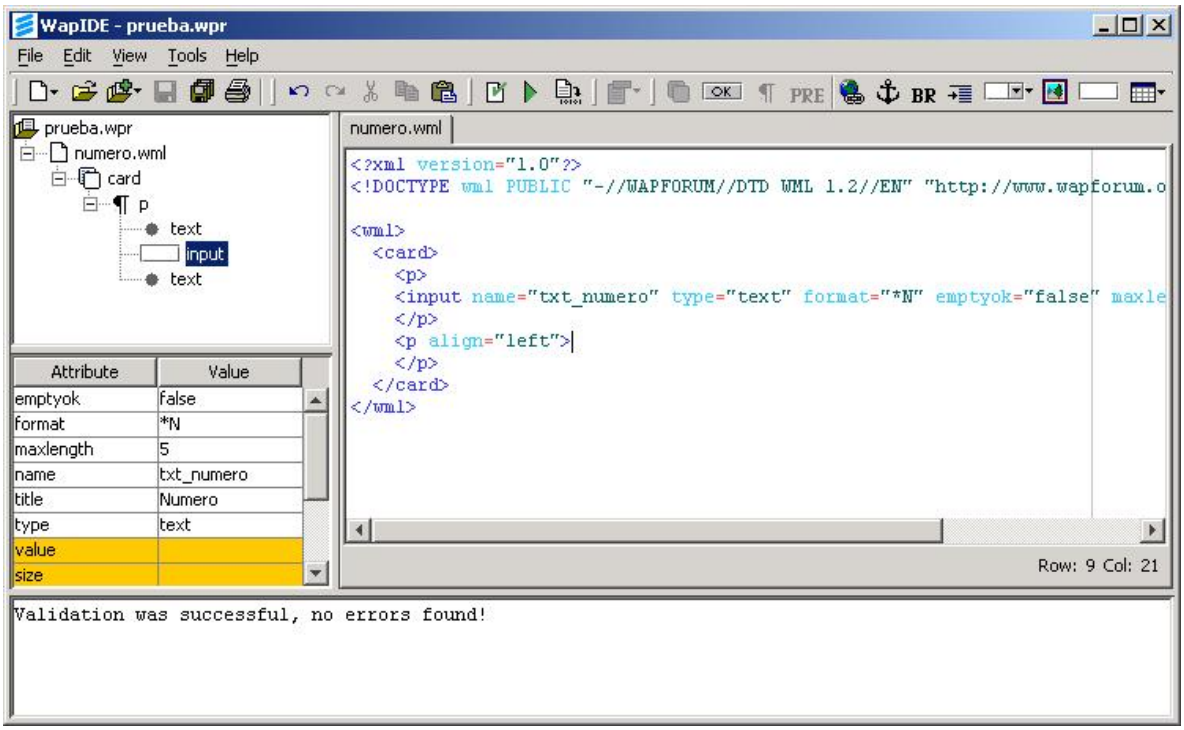


Figura 5.4 Pantalla principal del Diseñador de Aplicaciones WapIDE

5.2.2 Realwow - Mobile Gateway v3.0

Es un *gateway* WAP relativamente fácil de instalar y utilizar. Permite la configuración de un gran número de atributos y elementos propios de un *gateway* WAP.

El Mobile Gateway permite que teléfonos GSM/GPRS accedan a diversos servicios WAP y SMS[3], es decir, con esta aplicación es posible usar cualquier servidor Web estándar para acceder a servicios y contenido WAP. A través de la ventana de configuración, el administrador puede fácilmente configurar las conexiones, los *plugins*, la lista de servidores confiables, los atributos HTTP y del *proxy*, generar estadísticas, entre otros. En la Figura 5.5 se muestra dicha ventana.

El Mobile Gateway provee una conexión entre los servidores Web y los dispositivos móviles que soportan WAP, permitiendo a los usuarios navegar en páginas suministradas por servidores Web situados tanto en las Intranets corporativas como en Internet. Para reducir el tamaño de los datos transmitidos, las peticiones y respuestas son codificadas en un formato binario compacto.

Las principales características del Mobile Gateway son:

- Fácil instalación además de que trabaja con cualquier servidor Web estándar.
- Pequeño, eficiente, robusto y escalable.
- Maneja HTTPS y es compatible con GPRS y UMTS.
- Soporta WAP 1.2.1 y compresión de WMLScript.
- Proporciona un registro del gateway (Información en tiempo real acerca de lo que está pasando).
- Soporta tanto el modo orientado a conexión como el no orientado a conexión.
- Maneja sesión no permanente (HTTP cookies).

Los requerimientos del sistema son:

- Windows 9x / Me / NT 4.0 / 2000, XP o posterior.
- 128 MB de RAM mínimo.
- Un servidor HTTP accesible al Mobile Gateway.

[3] Short Message Service - Servicio de Mensajes Cortos

5.2.3 Caucho - Resin v2.1.0

Resin es un servidor Web gratuito (para fines educacionales), que implementa un poderoso motor de Servlets y JSPs[4] que soporta el balance de cargas que incrementa la confiabilidad. Por esta razón es que mantiene tiempos de respuesta bajos. Las principales tecnologías y servicios que maneja Resin se muestran a continuación.

5.2.3.1 HTTP / 1.1

Resin incluye un servidor web HTTP / 1.1 muy completo, dedicado a entregar contenido dinámico Java de forma rápida. Dado que algunos sitios necesitan características especiales de otros servidores, Resin puede trabajar conjuntamente con la mayoría de los servidores comunes, entre los que destacan: Apache, IIS, iPlanet, Zeus, y cualquiera que provea de las interfaces NSAPI y ISAPI.

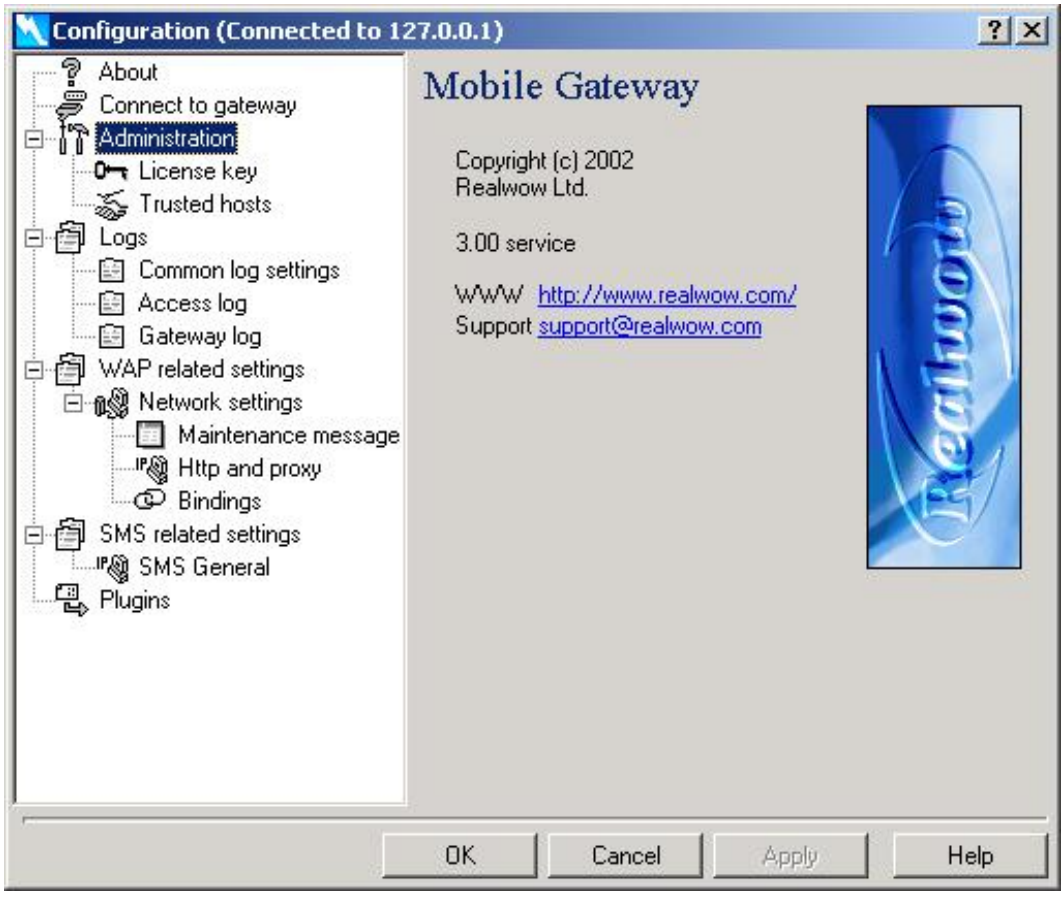


Figura 5.5 Ventana de configuración del Mobile Gateway

[4] Java Server Pages

5.2.3.2 Servlets

Resin soporta la última especificación Servlet 2.3 de Sun. Los servlets son clases Java que toman el control de la requisición HTTP.

5.2.3.3 Balanceo de Cargas

El balanceo de cargas incrementa el desempeño y la confiabilidad. Los servlets pueden tomar tiempo realizando una consulta en base de datos, leer archivos, realizar cálculos, etc, razón por la cual el balanceo de carga le permite agregar nuevos servidores si la demanda se incrementa en ese momento.

Resin implementa el balanceo de cargas para incrementar la confiabilidad, debido a que si un servidor llegara a fallar, automáticamente usará otro servidor. De esta forma, si un servidor tiene un porcentaje de falla promedio de 1%, entonces dos servidores balanceados a través de Resin tendrán en conjunto un porcentaje de falla de tan sólo el 0.01%.

5.2.4 Microsoft - SQLServer 2000

SQL Server es un manejador de bases de datos relacionales (RDBMS), bajo la arquitectura Cliente/Servidor, que utiliza el Transact-SQL para enviar peticiones entre el cliente y el servidor SQL. [GAM01]

SQL Server proporciona soporte para un conjunto de características que aportan las siguientes ventajas:

- Facilidad de instalación, distribución y utilización
- Gran escalabilidad
- Almacenamiento y recuperación de datos
- Integración del sistema con otro servidor

5.3 Implementación

Al igual que en los capítulos anteriores, esta etapa del desarrollo de software es presentada también a través de sus dos principales módulos: el **aplicación de mantenimiento de información** ***(backend)*** y la **aplicación WAP de localización** ***(frontend)***. Para cada uno de estos módulos se presentan sus principales componentes, así como también se describen las partes esenciales de su funcionamiento.

5.3.1 Sistema de Mantenimiento de Información (*Backend*)

Como se comentó con anterioridad, el *backend* está conformado por un sistema de información que es utilizado para dar mantenimiento a los sitios de interés y a la estructura vial dentro del sistema, por ejemplo, registrar una nueva colonia, eliminar una calle, modificar alguna propiedad de una cuadra, etc.

Dentro de la aplicación *backend*, se tiene la sección de mantenimiento a los sitios de interés del sistema. En la Figura 5.6 se muestra esta aplicación, en específico se trata de la sección de mantenimiento de sitios de interés (pestaña `Sitio`).

Figura 5.6 Pantalla de mantenimiento de sitios de interés (*Backend*)

En la sección A se tienen las pestañas que habilitan cada uno de los mantenimientos existentes: Calle, Colonia, Delegación, Nodo, Cuadra y Sitio. Debido a

que esta última pestaña es la que se encuentra activa, en la sección B se muestra la consulta de sitios de interés dados de alta en el sistema, en la cual es posible elegir un sitio de interés a la vez, para que los botones de `Modificar` y `Eliminar` actúen sobre él.

La sección C contiene una agrupación de controles que permiten capturar o introducir la información de un sitio de interés que se desee modificar o dar de alta. Por último en la sección D se tienen los botones que activan o desactivan las operaciones de mantenimiento de la aplicación.

De esta manera está conformado el mantenimiento de sitios de interés dentro del *backend*. De las mismas secciones se dispone para llevar a cabo el mantenimiento de información para el resto de los elementos que conforman la estructura vial del sistema, en las cuales solo varían los atributos para cada elemento y su distribución.

La consulta a BD para obtener la información de los sitios de interés se realiza por medio del Procedimiento Almacenado (SP)[5] **loc_spr_sitio**. El código de la Figura 5.7 muestra el SP que realiza la consulta de sitios de interés.

```
CREATE PROCEDURE dbo.loc_spr_sitio
/************************************
Consulta de sitios de interes
************************************/
       @idSitio       INT = null,
       @idcateg       INT = null
AS
BEGIN
       SELECT  A.idSitio,
               A.nombre,
               A.idCalle,
               B.nombre 'nomCalle',
               A.numero,
               A.idcolonia,
               C.nombre 'nomColonia',
               A.telefono,
               A.iddeleg,
               D.nombre 'nomDelegacion',
               A.idcategoria,
               E.nombre 'nomcategoria'
       FROM    localizacion..loc_sitio A (NOLOCK),
               localizacion..loc_catcalle B (NOLOCK),
               localizacion..loc_catcolonia C (NOLOCK),
               localizacion..loc_catdelegacion D (NOLOCK),
               localizacion..loc_catcategoria E (NOLOCK)
       WHERE   A.idcalle = B.idcalle
       AND     A.idcolonia = C.idcolonia
       AND     A.iddeleg = D.iddeleg
       AND     A.idcategoria = E.idcategoria
       AND     (A.idSitio = @idSitio OR @idSitio IS NULL)
       AND     (A.idcategoria = @idcateg OR @idcateg IS NULL)
END
```

Figura 5.7 Procedimiento almacenado para la consulta de sitios de interés: loc_spr_sitio

[5] Un Procedimiento Almacenado (Stored Procedure) es un conjunto de sentencias SQL precompiladas que son útiles para recuperar ó modificar información guardada en la BD.

Existen en la BD los correspondientes SP para realizar las operaciones de mantenimiento de información para los sitios de interés: Insertar (**loc_spi_sitio**), Modificar (**loc_spu_sitio**) y Eliminar (**loc_spd_sitio**). Para la consulta y mantenimiento del resto de los elementos que forman la estructura vial del sistema se poseen sus respectivos SP.

5.3.2 Aplicación WAP de Localización (Frontend)

La función primordial de la aplicación *frontend* es la de proporcionar un conjunto de servicios de localización sobre una infraestructura de telefonía celular; principalmente son la búsqueda de rutas óptimas para trasladarse de un lugar a otro en una ciudad y la búsqueda del sitio de interés más cercano a un punto elegido por el usuario.

La primer pantalla que despliega la aplicación es la pantalla de Registro mostrada en la Figura 5.8 (a), en ésta se solicita al usuario que ingrese su **`Usuario`** y **`Password`** asignados. Una vez que son ingresados estos datos, el usuario selecciona la opción **`Enviar`** para que la aplicación valide esta información. Si los datos proporcionados son inválidos es desplegado un mensaje que indica al usuario esta condición, mostrado en la Figura 5.8 (b), en esta pantalla con la opción Aceptar el sistema presenta nuevamente la pantalla de Registro.

Si los datos ingresados son correctos, el sistema verifica si el usuario tiene una búsqueda pendiente en cuyo caso lo redirecciona a la última pantalla en la que se estuvo antes del momento de la desconexión.

Figura 5.8 (a) Pantalla de Registro, (b) Mensaje de Usuario y/o Password erróneos

Por el contrario, si el usuario no tuviera una búsqueda pendiente, el sistema presenta el menú principal de la aplicación (Figura 5.9) con las opciones que proporcionan los servicios de localización descritos anteriormente. Estas opciones son:

a) **`Dos Puntos`**.- Búsqueda de la ruta óptima entre dos lugares (direcciones) ingresados por el usuario.

b) **`Sitio de Interés`**.- Búsqueda de la ruta óptima entre un lugar ingresado por el usuario y un sitio de interés. El sitio de interés puede ser proporcionado de dos maneras: Que sea elegido por el usuario o que sea el más cercano al lugar seleccionado y pertenezca a la categoría deseada (Restaurantes, Escuelas, Cines, etc.).

Figura 5.9 Menú principal de la aplicación *frontend*

Cada una de las opciones anteriores tiene a su vez su propio menú que permite sean introducidos cada uno de los datos requeridos por el sistema para proporcionar cada servicio. En la Figura 5.10 es presentado el menú que se despliega cuando es elegida la opción `Dos Puntos`, en la cual se puede observar que dicho menú cuenta con tres opciones principales:

a) **`Dir. Origen`**.- Esta opción se utiliza para llamar a la pantalla que permite la captura de la dirección origen a partir de la cual iniciará la búsqueda de la ruta óptima hacia la dirección destino.

b) **`Dir. Destino`**.- Se utiliza para ingresar a la aplicación la dirección destino.

c) **`Ruta Optima`**.- Cuando ya se han ingresado ambas direcciones (origen y destino), esta opción inicia el proceso de búsqueda de la ruta óptima de traslado de la dirección origen a la destino.

Figura 5.10 Menú de la opción Dos Puntos del menú principal

Por otro lado, en la Figura 5.11 se presenta el menú que se muestra cuando es elegida la opción `Sitio de Interés`.

Figura 5.11 Menú de la opción Sitio de Interés del menú principal

De manera similar que en el caso anterior, ente menú consta de tres principales opciones:

a) **`Dir. Origen`**.- Esta opción se encarga de llamar a la pantalla de captura de la dirección origen. Esta dirección es tomada como base para encontrar el sitio de interés más cercano ó algún sitio de interés en particular.

b) **`Sitio`**.- Se utiliza para elegir un sitio de interés o para indicar que se desea encontrar el sitio más cercano de una categoría en específico en la que el usuario está interesado.

c) **`Ruta Optima`**.- Esta opción inicia el proceso de búsqueda de la ruta óptima para el traslado de la dirección origen al sitio de interés, sea el elegido o el más cercano a dicha dirección.

5.3.2.1 Construcción del Grafo

En el capítulo anterior se explicó el proceso de generación del grafo a partir de la información almacenada en BD. A continuación se expone la manera en cómo se realizó la implementación de este proceso.

Como paso inicial, es necesario obtener el número de nodos registrados en BD a través del método `getNumeroNodos()` de la clase `Dijkstra`. Este método ejecuta el servicio (SP) de BD llamado **loc_sp_numeronodos**, el cual retorna el número de nodos que están dados de alta en el sistema. Ahora se define, con el nombre `peso`, una matriz bidimensional de tamaño igual al número de nodos del sistema.

Esta matriz representa al grafo que simboliza los elementos de la estructura vial de una ciudad; específicamente hablando, cada valor de la matriz representa a una arista del grafo que va del nodo con id igual al valor de su primera dimensión hacia el nodo con id igual al valor de su segunda dimensión. Esto se puede ver de manera más clara con el siguiente ejemplo. Si esta matriz tiene el siguiente valor:

```
peso[5][13] = 8.1
```

Significa que existe una arista que va del nodo 5 al nodo 13 con un costo de 8.1. Esto a su vez representaría a una cuadra de la ciudad en la que se puede circular del cruce X, que forman las calles A y B, hacia el cruce Y, que forman las calles C y D (Regularmente una de las calles de un cruce es la misma que una de las del otro cruce), con un costo de traslado de 8.1. El costo de traslado es obtenido a través de un cálculo en el que intervienen un conjunto de variables[6] que son recuperadas de la tabla **loc_cuadra**: flujo1, flujo2 y trafico. En la Figura 5.12 se presenta un fragmento de código que realiza la asignación de valores a la matriz que representa al grafo.

[6] El significado y función de estas variables fue explicado en el capítulo de diseño.

De esta forma es como se genera y carga el grafo en el sistema, el algoritmo de búsqueda de Dijkstra será el que opere sobre este grafo (matriz) para obtener la ruta óptima. En el siguiente apartado se presenta la implementación de este algoritmo.

5.3.2.2 Algoritmo de Dijkstra

En el capítulo anterior se definió la creación de una clase que implementara el algoritmo de Dijkstra para la búsqueda de rutas mínimas en un grafo. También se presentó el diagrama de estados de esta clase, en el que se mostraba como deberían ser, y de acuerdo a que, las transiciones de un estado a otro para implementar dicho algoritmo y al final obtener la ruta óptima para trasladarse desde un punto a otro del grafo. A continuación se presentan los detalles más relevantes de la implementación de esta clase.

Como primer paso, en el constructor de la clase son inicializados un conjunto de variables y arreglos que se usan a lo largo de la ejecución del algoritmo. Posteriormente es cargado el grafo con los valores almacenados en la base de datos, tal y como se explicó en el apartado anterior.

Con las calles y números de las direcciones de origen y destino, se establecen los nodos origen y destino a través del método encontrarNodoCercano(). Si los nodos fueron encontrados satisfactoriamente y su acceso no se encuentran bloqueado, se inicia la ejecución del algoritmo por medio de la función ejecutarAlg().

```
try{
	ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_flujonodos");
	int n1, n2;
	int f1; f2;
	int t;

	if(rs1!=null){
		while(rs1.next()){
			n1 = rs1.getInt(1);
			n2 = rs1.getInt(2);
			f1 = rs1.getInt(3);
			f2 = rs1.getInt(4);
			t = rs1.getInt(5);

			if((f2 == 0) || (t == 0))
				peso[n1][n2] = 0;
			else
				peso[n1][n2] = (int)((9*(11-f2)) + (25*t));

			if((f1 == 0) || (t == 0))
				peso[n2][n1] = 0;
			else
				peso[n2][n1] = (int)((9*(11-f1)) + (25*t));
		}
	}
}catch(Exception e){
	debug("Error:" + e.toString() + "_Msg=" + e.getMessage());
	e.printStackTrace(System.out);
}
```

Figura 5.12 Código para generar el grafo a partir de la información de BD

Lo primero que realiza esta función es inicializar todo lo necesario para que el algoritmo comience su ejecución. Posteriormente se realiza un ciclo por cada uno de los nodos del grafo, en el que se ejecuta el método ejecutarSiguientePasoAlg(). A su vez, este método es encargado de mandar llamar a los métodos restantes que implementan el algoritmo de Dijkstra.

El algoritmo de Dijkstra para esta aplicación se encuentra implementada por un par de métodos: iniciarPasoDijkstra() y finalizarPasoDijkstra(), su código se presenta en en las Figuras 5.13 y 5.14 respectivamente. Primero, el método iniciarPasoDijkstra(i,j) realiza las siguientes funciones:

- Verifica que no exista una ruta más optima, tomando en cuenta la arista formada del nodo i al nodo j.
- En caso de que exista, calcula el nuevo peso que adquiere este nodo con la nueva ruta.
- Actualiza las variables que determinan los nodos de inicio y final de la ruta óptima hasta ese momento.

```
public void iniciarPasoDijkstra(int i, int j) {
       if ( (distFinal[i] != -1) && (distFinal[j] == -1) ) {
              if ( (dist[j]==-1) || (dist[j]>=(dist[i]+peso[i][j])) ) {
                     if ( (dist[i]+peso[i][j])<dist[j] ) {
                            cambioDist[j]=true;
                            numcambioDist++;
                     }
                     dist[j] = dist[i]+peso[i][j];
                     if ( (mindist==0) || (dist[j]<mindist) ) {
                            mindist=dist[j];
                            minstart=i;
                            minend=j;
                     }
              }
       }
}
```

Figura 5.13 Método que inicia un paso del algoritmo de Dijkstra

El método finalizarPasoDijkstra() se encarga de lo siguiente:

- Identifica los nodos alcanzables desde el nodo inicial.
- Indica si no hay nodos alcanzables desde el nodo inicial
- Si no se encontraron caminos desde el nodo fuente (inicial) hacia el nodo destino, manda un mensaje de que no se encontraron rutas disponibles.

```
public void finalizarPasoDijkstra() {
        if ((ejecutando) && (mindist==0))
        {
                int nalcanzable = 0;
                for (int i=0; i<numnodos; i++)
                        if (distFinal[i] > 0)
                                nalcanzable++;

                if (nalcanzable == 0){
                        debug("El algoritmo ha terminado, no hay nodos
                                alcanzables desde el nodo inicial.");
                }
                else if (nalcanzable< (numnodos-1)){
                        debug("El algoritmo ha terminado, No hay caminos del
                                nodo_inicial a ningun otro nodo.");
                        callesRutaOptima = new String[1];
                        callesRutaOptima[0] = "No se encontraron rutas
                                disponibles.";
                        numNodosRutaOptima = 1;
                        }else{
                        debug("El algoritmo ha terminado.");
                }
        }
}
```

Figura 5.14 Método que finaliza un paso de la implementación del algoritmo de Dijkstra

5.3.2.3 Generación de Páginas WML

Una parte importante del *frontend* es la generación de páginas WML que son utilizadas para presentar información en los teléfonos celulares. La mayoría de los teléfonos celulares actuales tienen la capacidad de interpretar páginas creadas con este lenguaje. El contenido para estas páginas puede ser estático o dinámico: cuando una página es estática significa que la información que presenta no cambia bajo ninguna circunstancia. Por el contrario, si una página es dinámica significa que los datos que presenta cambian constantemente.

Para la implementación del *frontend* se desarrolló una combinación de páginas estáticas y dinámicas. Entre las estáticas quedaron aquellos menús iniciales que no requieren de cambios constantes. Las páginas dinámicas se crearon utilizando la tecnología *Java Servlet*. Esta tecnología provee a los desarrolladores Web un mecanismo simple y consistente para extender la funcionalidad de un servidor Web. Los *servlets* proveen un método para construir aplicaciones Web basado en componentes e independiente de la plataforma, sin las limitaciones de desempeño que tienen los programas CGI[7], además de ser independientes del servidor, todo esto da la libertad de realizar una buena estrategia de elección para servidores, plataformas y herramientas [WWW9].

En la Figura 5.15 se muestra el código fuente del método doGet() del *servlet* llamado LocSrvPrincipal, el cual se encarga de generar el menú principal del *frontend* presentado anteriormente en la Figura 5.9. Como se puede apreciar, este método recibe un par de parámetros: Un objeto de tipo *HttpServletRequest* y otro del tipo *HttpServletResponse*. A través del primero de ellos es posible controlar y manipular la

[7] Common Gateway Interface

petición que realiza el cliente, por ejemplo se puede obtener su dirección IP, los datos de sesión, un parámetro que sea enviado, etc. Por otro lado, el segundo parámetro (*HttpServletResponse*) es útil para darle un tratamiento y retornar la respuesta al cliente; por ejemplo, es posible indicar que tipo de contenido será retornado al cliente (HTML, XML, MIME, WML en nuestro caso, etc.), obtener el flujo por el que se responderá al cliente, etc.

La primera línea de código del método doGet() es la siguiente:

```
PrintWriter out = res.getWriter();
```

En ésta se define un objeto *PrintWriter* que es inicializado invocando al método getWriter() del parámetro res. Este método devuelve un flujo de salida que es usado para enviar datos en caracteres al cliente. En la siguiente línea se indica que tipo de contenido se esta retornando al cliente, en el caso del método doGet() del LocSrvPrincipal es uno que indica que es contenido WML el que se está devolviendo:

```
res.setContentType("text/vnd.wap.wml");
```

```
public void doGet(HttpServletRequest req, HttpServletResponse res) throws
    ServletException, IOException
{
    PrintWriter out = res.getWriter();
    res.setContentType("text/vnd.wap.wml");
    out.println("  <?xml version='1.0'?>");
    out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
      'http://www.wapforum.org/DTD/wml12.dtd'>");
    out.println("<wml>");
    out.println("  <card id='cardP' title='Localización' newcontext='true'>");
    out.println("    <p align='left'>");
    out.println("     Buscar ruta de:");
    out.println("    </p>");
    out.println("    <p align='left'>");
    out.println("     <a
      href='http://148.204.211.215:8080/servlet/LocSrvRutaDosPuntos?Rand="+
      Math.random() +"' title='R2P'>Dos Puntos</a>");
    out.println("    </p>");
    out.println("    <p align='left'>");
    out.println("     <a
      href='http://148.204.211.215:8080/servlet/LocSrvRutaSitio?Rand="+
      Math.random() +"' title='RS'>Sitio de Interes</a>");
    out.println("    </p>");
    out.println("    <p align = 'left'>");
    out.println("    </p>");
    out.println("  </card>");
    out.println("</wml>");
}//doGet()
```

Figura 5.15 Método doGet() del servlet que genera el menú principal del frontend (LocSrvPrincipal)

Con el resto de las líneas out.println() se está formando las páginas WML con la información que será devuelta al cliente. El método println() de la clase *PrintWriter* escribe en el flujo una cadena de caracteres agregándole al final un salto de línea.

De está forma es como son creadas y configuradas las páginas WML para que sean correctamente desplegadas e interpretadas por los dispositivos de telefonía celular.

5.4 Diagrama de Despliegue

Los Diagramas de Despliegue muestran la disposición física de los distintos nodos que componen un sistema [RAM00]. Un nodo es un recurso de ejecución tal como una computadora, un dispositivo o unidad de memoria. Los estereotipos permiten precisar la naturaleza del equipo:

- Dispositivos
- Procesadores
- Memoria

En la Figura 5.16 se muestra el diagrama de despliegue de la aplicación desarrollada. Está conformado por dos dispositivos (Teléfono Celular y el *Gateway* WAP) y por tres equipos de procesamiento (Servidor Web, servidor de base de datos y la computadora del operador).

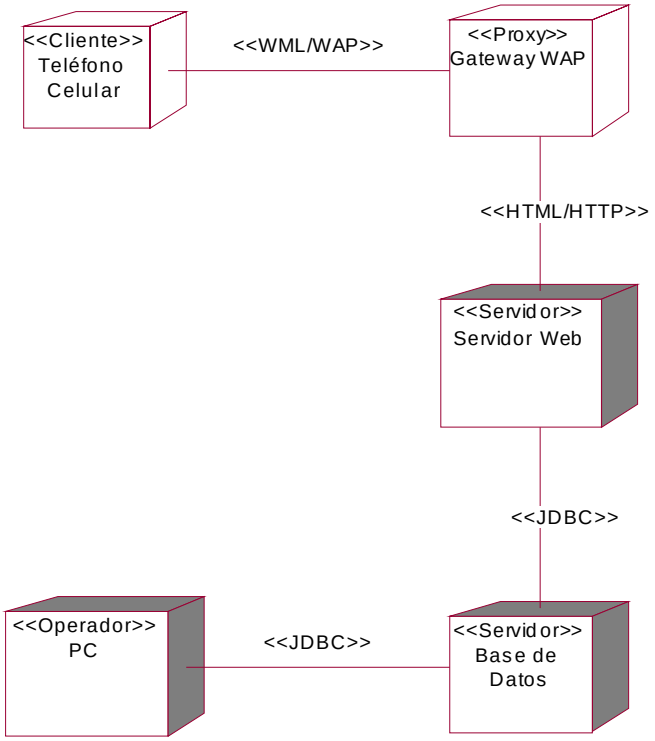


Figura 5.16 Diagrama de despliegue del sistema

5.5 Pruebas de la Aplicación

5.5.1 Pruebas de Funcionalidad

Con la finalización de la etapa de implementación se tienen totalmente construidas las aplicaciones *backend* y *frontend*. El siguiente paso es verificar la funcionalidad del sistema por medio de la etapa de pruebas.

Probar un sistema es relativamente independiente de la metodología de desarrollo utilizada para construirlo. Existen diversos tipos de pruebas aplicados durante las diferentes actividades del proceso de desarrollo. Estas pruebas requieren de tiempo y presupuesto adicional, pudiendo llegar a significar entre un 30% y un 50% del costo total de desarrollo. Por tal motivo, el modelo de pruebas debe ser planificado con anticipación y de manera integral junto con el propio desarrollo del sistema. Es un error pensar que las pruebas son la última actividad del desarrollo ya que no se puede lograr software de alta calidad sólo mediante pruebas finales y depuraciones. Las pruebas deben hacerse en paralelo al desarrollo del sistema, teniendo pruebas finales únicamente como certificación final de la calidad del producto y no como la oportunidad para encontrar errores. Encontrar errores al final del desarrollo es bastante problemático dado que requerirá regresar a etapas anteriores para resolverlos. Se considera que "evitar defectos" es más poderoso que "corregir defectos" [WWW10].

En lo que respecta a la etapa de pruebas para la aplicación desarrollada en esta tesis, se limitará a verificar el sistema de acuerdo a los casos de uso del mismo. Es decir que, como objetivo de las pruebas se revisará que la funcionalidad implementada corresponda a los casos de uso correspondientes especificados durante la etapa de análisis. A continuación se revisan los casos de uso principales y/o más significativos tanto del *backend*, como del *frontend*. El primer caso de uso que se verifica es el de **Alta de Sitio de Interés**, el cual pertenece al *backend*. Este caso de uso fue presentado en la fase de análisis del sistema.

Caso de uso	Alta de Sitio de Interés
Actor	Operador (Usuario)
Descripción	Insertar en el sistema un sitio de interés.
Precondiciones	No encontrarse en la ejecución de alguna otra opción.
Flujo normal de eventos	**1.** Este caso de uso comienza cuando el operador selecciona, del menú principal, la pestaña Sitio.
	2. El sistema muestra una lista con todos los sitios de interés registrados en el sistema y un conjunto de controles que permiten modificarlo.
	3. El operador presiona el botón Insertar
	4. El sistema habilita los controles para modificar los datos.
	5. El operador ingresa los datos del nuevo sitio: nombre, dirección (calle, número y colonia), número telefónico, entre otros.
	6. Posteriormente, el operador presiona Aceptar para que el sitio sea almacenado en BD.
	7. El sistema guarda en BD
Excepciones	El operador tiene también la opción Cancelar, ésta opción le sirve para cuando desea terminar el flujo sin que el sitio se almacene en BD.
Postcondiciones	Ninguna

Este caso de uso presenta la secuencia para dar de alta un sitio de interés en el sistema, la cual se puede ver en el Flujo normal de eventos. En las Figuras 5.17, 5.18 y 5.19 se muestra la pantalla de mantenimiento de la Estructura Vial y los Sitios de Interés (*backend*), en el proceso de dar de alta un sitio de interés en el sistema. Los números que se muestran en cada una de las figuras corresponden con el número de evento del Flujo normal de eventos del caso de uso.

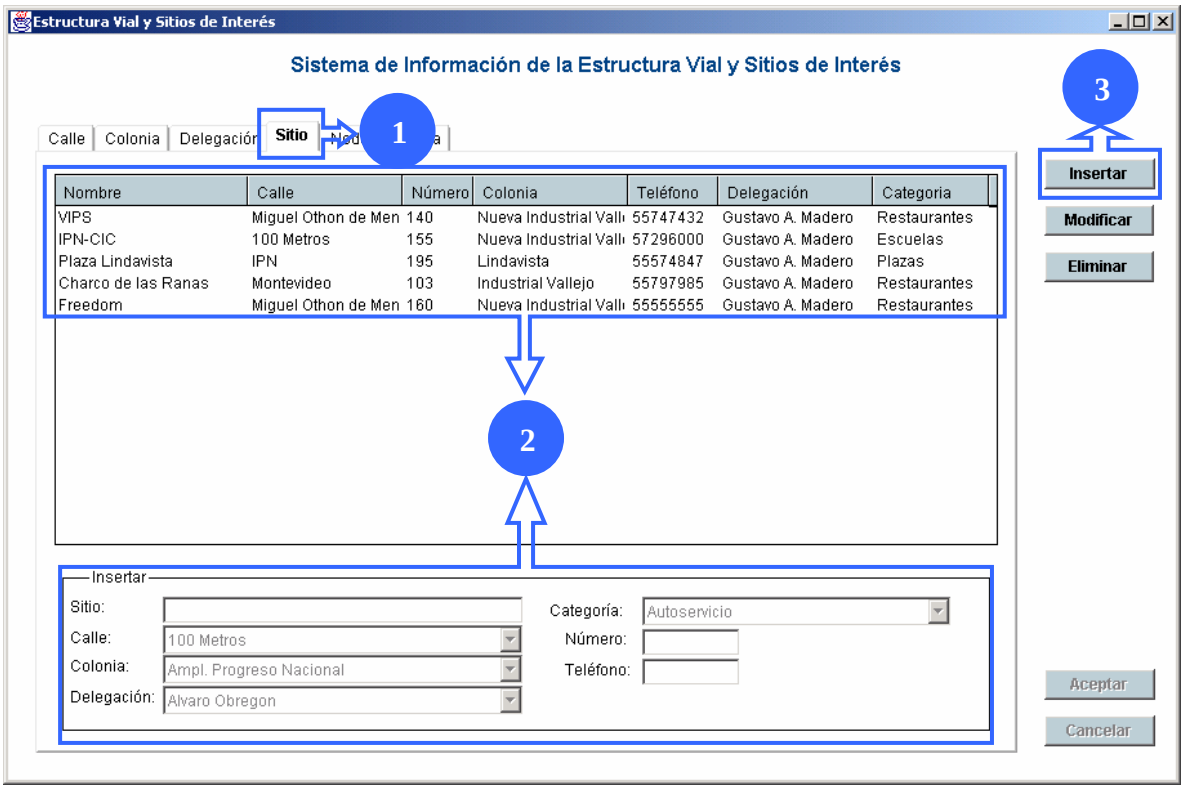


Figura 5.17 Eventos 1, 2 y 3 del Flujo normal de eventos del caso de uso Alta de Sitio de Interés

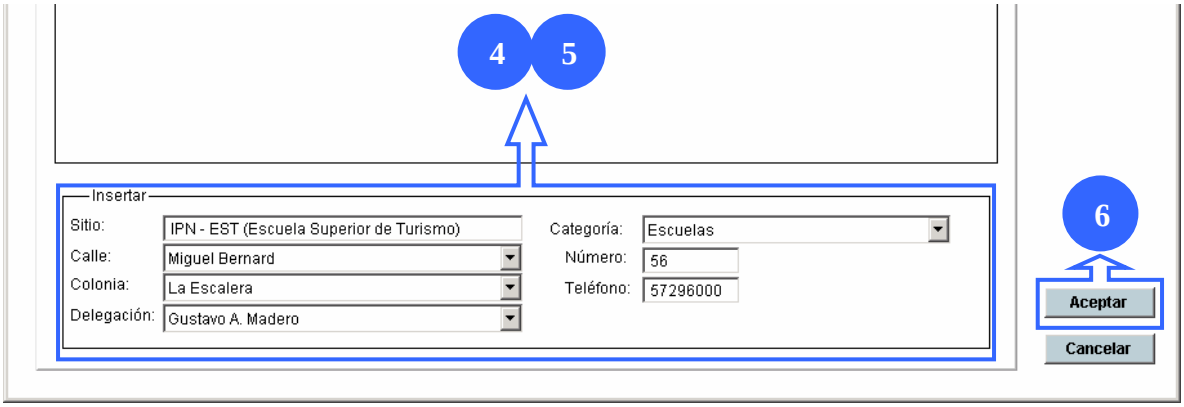


Figura 5.18 Eventos 4, 5 y 6 del Flujo normal de eventos del caso de uso Alta de Sitio de Interés

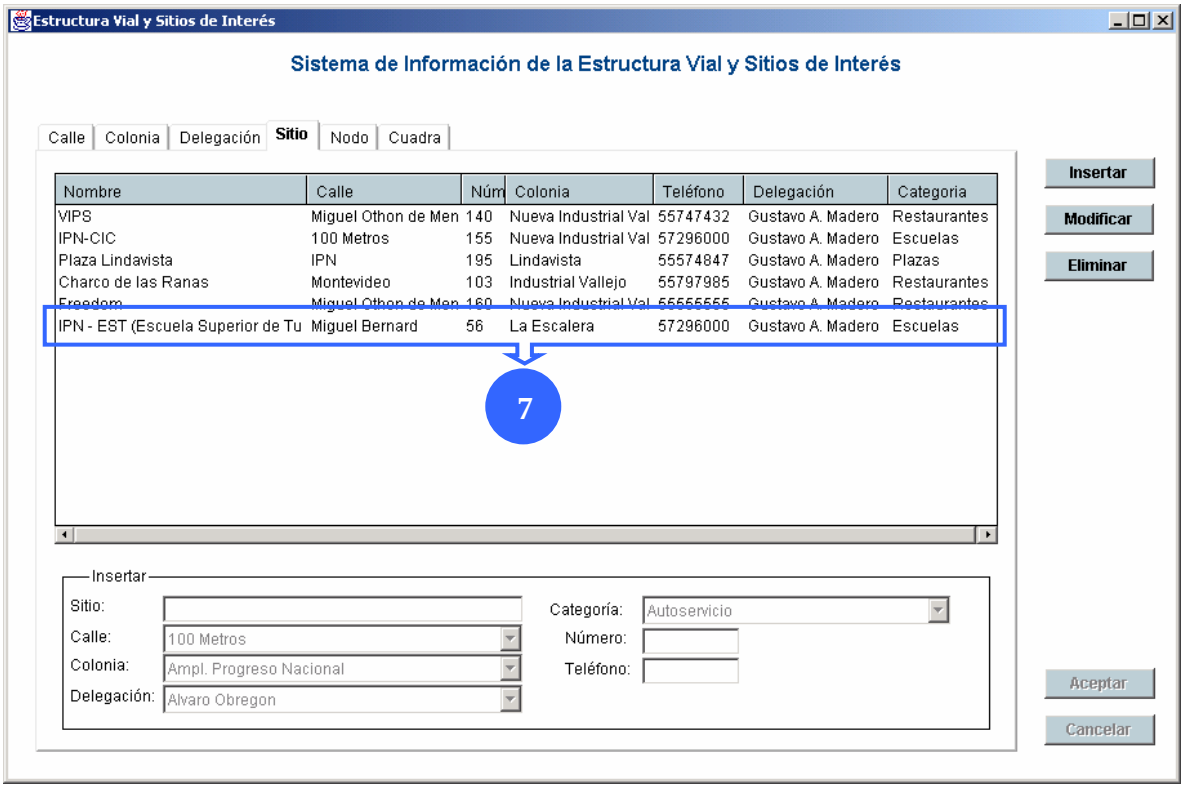


Figura 5.19 Evento 7 del Flujo normal de eventos del caso de uso Alta de Sitio de Interés

En el ejemplo anterior se realizó el alta de la Escuela Superior de Turismo (IPN - EST) como sitio de interés. Con esto se prueba que la funcionalidad y comportamiento de la aplicación están apegados a la definición que proporciona el caso de uso probado.

Por otra parte, la aplicación *frontend* será probada con el siguiente ejemplo. Se desea conocer la ruta óptima para llegar desde la dirección: **Poniente 112 No. 190 Col. Panamericana Del. Gustavo A. Madero** (dirección origen), hasta la **Escuela Superior de Turismo (IPN - EST)** (sitio de interés).

Para lograr esto, el siguiente caso de uso que será probado es el de: **Buscar Ruta Óptima hacia sitio de interés**. Este caso de uso utiliza un par de casos de uso adicionales: **Seleccionar dirección** y **Seleccionar Sitio de Interés,** razón por lo cual no se describen de manera independiente a éste.

De igual forma que en el caso anterior, primero se presentan las definiciones de los casos de uso de la fase de análisis, para después mostrar y probar la aplicación desarrollada.

Estos son los casos de uso que serán probados:

Caso de uso	Buscar Ruta Óptima hacia sitio de interés
Actor	Usuario
Descripción	Obtener la ruta óptima entre un punto origen hasta un sitio de interés en particular.
Precondiciones	Ninguna
Flujo normal de eventos	1. Este caso de uso comienza cuando el usuario selecciona del menú principal de la aplicación móvil, la opción Sitio de Interés.
	2. El sistema devuelve el menú para esta modalidad de búsqueda.
	3. El usuario selecciona la dirección origen y un sitio de interés.
	4. El sistema presenta la opción de buscar la ruta óptima entre el punto origen y el sitio de interés.
	5. El usuario selecciona la opción anterior.
	6. El sistema busca la ruta óptima entre estos dos puntos. A menos que no existiera una ruta disponible, el sistema presenta un listado ordenado de calles que van desde el punto origen hasta el sitio de interés elegido.
Excepciones	Ninguna
Postcondiciones	Ninguna

Caso de uso	Seleccionar dirección
Actor	Usuario
Descripción	Subflujo que permite introducir una dirección, ya sea de origen o destino.
Precondiciones	Ninguna
Flujo normal de eventos	1. Este caso de uso comienza cuando el usuario ha elegido la opción de introducir una dirección, ya sea la de origen o destino.
	2. El sistema regresa un listado de todas las delegaciones del sistema y lo muestra en el navegador del celular.
	3. El usuario elige la delegación donde se encuentra la dirección que desea introducir.
	4. El sistema regresa un listado de las colonias que pertenecen a la delegación previamente seleccionada, y le permite que elija una de ellas.
	5. El usuario elige la colonia.
	6. El sistema retorna ahora un listado de las calles que se encuentran en la colonia seleccionada en el punto anterior.
	7. El usuario elige la calle.
	8. El sistema solicita al usuario una pantalla donde debe capturar el número de lote o casa de la dirección que esté ingresando.
	9. El usuario captura el número solicitado en el punto anterior. Si los datos que ha ingresado son correctos, él debe seleccionar la opción Aceptar, la cual grabará la dirección completa y regresará al menú en donde se encontraba al momento de elegir la opción que modela este caso de uso.
Excepciones	Ninguna
Postcondiciones	Ninguna

Caso de uso	Seleccionar sitio de interés
Actor	Usuario
Descripción	Subflujo que permite seleccionar un sitio de interés.
Precondiciones	Ninguna
Flujo normal de eventos	1. Este caso de uso comienza cuando el usuario ha elegido la opción de elegir un sitio de interés.
	2. El sistema regresa un listado de todas las categorías a las que pertenece un sitio de interés.
	3. El usuario elige la categoría del sitio de su interés.
	4. El sistema regresa un listado de todos los sitios de interés que pertenecen a la categoría que eligió en el punto anterior.
	5. El usuario elige el sitio de interés que desee.
	6. El sistema retorna el control al flujo que dio inicio a éste (seleccionar sitio de interés).
Excepciones	Ninguna
Postcondiciones	Ninguna

En la Figura 5.20, se muestra la secuencia de pantallas que define el caso de uso **Buscar Ruta Óptima hacia sitio de interés**. La primer pantalla muestra el menú principal de la aplicación móvil. Existen dos modalidades de búsqueda de la ruta óptima: `Dos Puntos` que indica que se desea conocer la ruta entre dos puntos dados por el usuario y `Sitio de Interés` para buscar una ruta de un punto especificado también por el usuario hacía un sitio de interés en particular o el más cercano al punto elegido.

Para efectos de la prueba se eligió la segunda opción (1), el sistema muestra entonces el submenú principal para esta opción, el cual se muestra en la siguiente pantalla de la misma figura (2). El siguiente paso es introducir la dirección origen y el sitio de interés (3), para realizar esto es necesario utilizar dos secuencias adicionales que definen los casos de uso: **Seleccionar Dirección** y **Seleccionar sitio de interés**. Estas pantallas se muestran en las Figuras 5.21 y 5.22 respectivamente.

Una vez que han sido ingresados estos datos, el sistema muestra de nueva cuenta el submenú (4) en el que se presenta de nuevo la opción `Ruta Optima` (5) la cual se encarga de devolver la ruta óptima (6).

En la Figura 5.21 se muestra, mediante flechas, la ruta óptima arrojada por la prueba realizada a la aplicación.

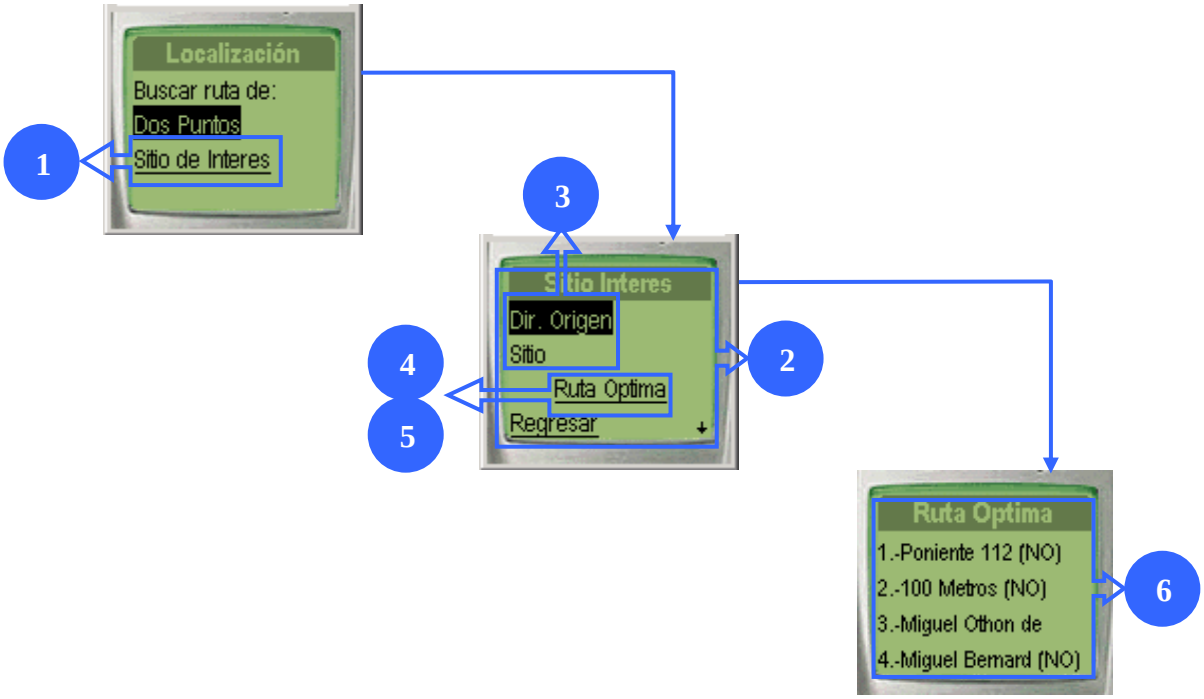


Figura 5.20 Secuencia de pantallas de prueba del *frontend*

Figura 5.21 Ruta Óptima obtenida como resultado de la prueba de la Figura 5.20

En la Figura 5.21, se muestra la secuencia de pantallas de la aplicación para introducir una dirección (punto) al sistema. Primeramente se elige la delegación de la dirección, en el caso de la prueba es **Gustavo A. Madero**. El sistema devuelve las colonias que pertenecen a esta delegación. En el siguiente paso se selecciona la colonia **Panamericana**, de nueva cuenta el sistema retorna las calles que pertenecen a esta colonia.

Es elegida la calle **Poniente 112** por lo que ahora la aplicación solicita sea introducido el número dentro de la calle previamente seleccionada, este valor es el **190**. Hasta este punto ha sido introducida por completo la dirección. Como última acción se debe dar clic en la opción `Aceptar` para que la aplicación retorne al menú del que fue seleccionada esta opción.

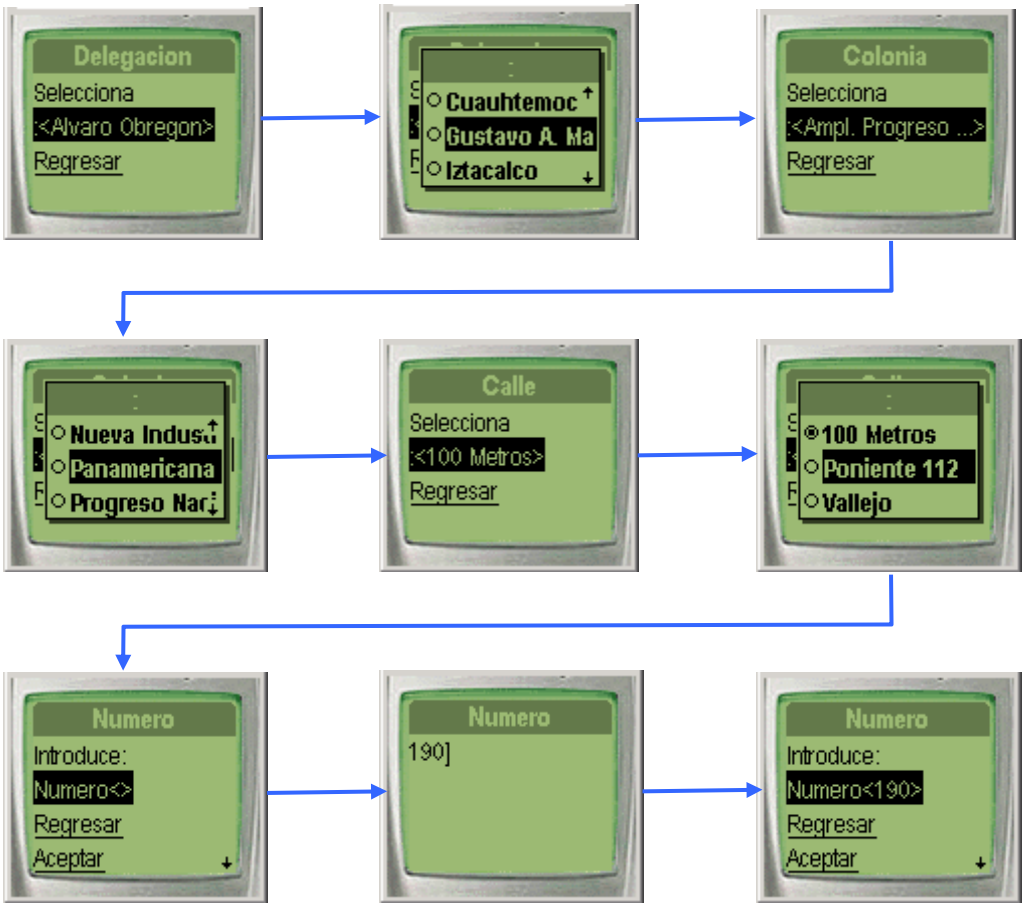


Figura 5.22 Secuencia de pantallas de prueba para ingresar una dirección (punto)

Para la elección del sitio de interés, se muestra en la Figura 5.22 la secuencia de pantallas de prueba. Como primer paso es elegido la forma en como se desea buscar el sitio de interés. Para el caso de la prueba se elige la opción `Categoría`. El sistema

regresa una lista con todas las categorías de los sitios de interés dados de alta en la aplicación. En la siguiente pantalla es elegida la categoría del sitio en el que se está interesado, para la prueba se eligió la categoría **Escuelas**. Ahora la aplicación muestra una lista con todos los sitios de interés pertenecientes a **Escuelas** incluyendo una opción más que se utiliza para indicarle al sistema que se desea llegar al sitio de interés más cercano al punto definido por la dirección origen, esta opción se llama `+ cercano`.

Continuando con la prueba, es elegido la escuela **IPN – EST (Escuela Superior de Turismo)**. Posteriormente el control de la aplicación retorna al menú presentado en la Figura 5.20. Con esto se completa el proceso de selección de un sitio de interés registrado en el sistema.

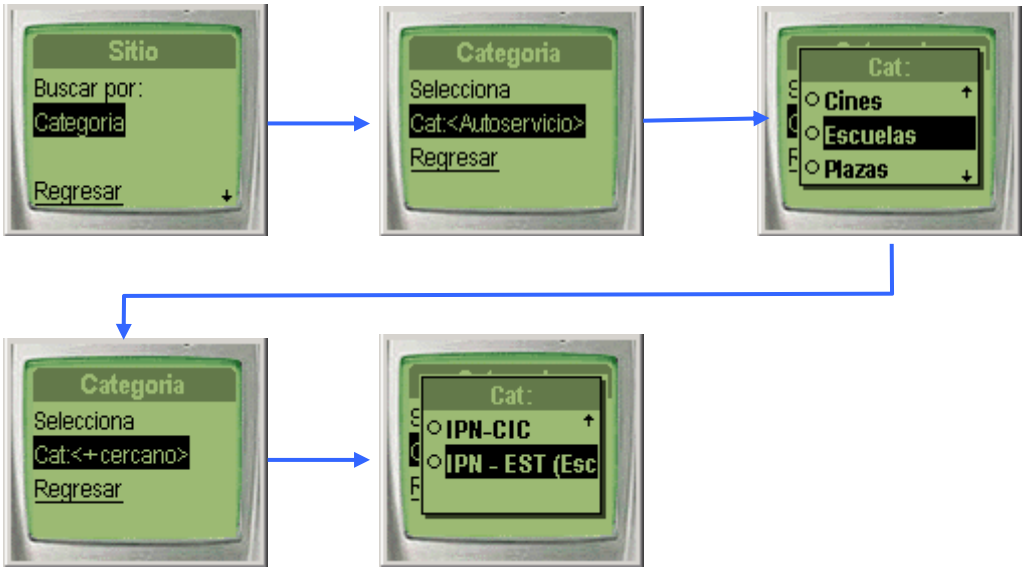


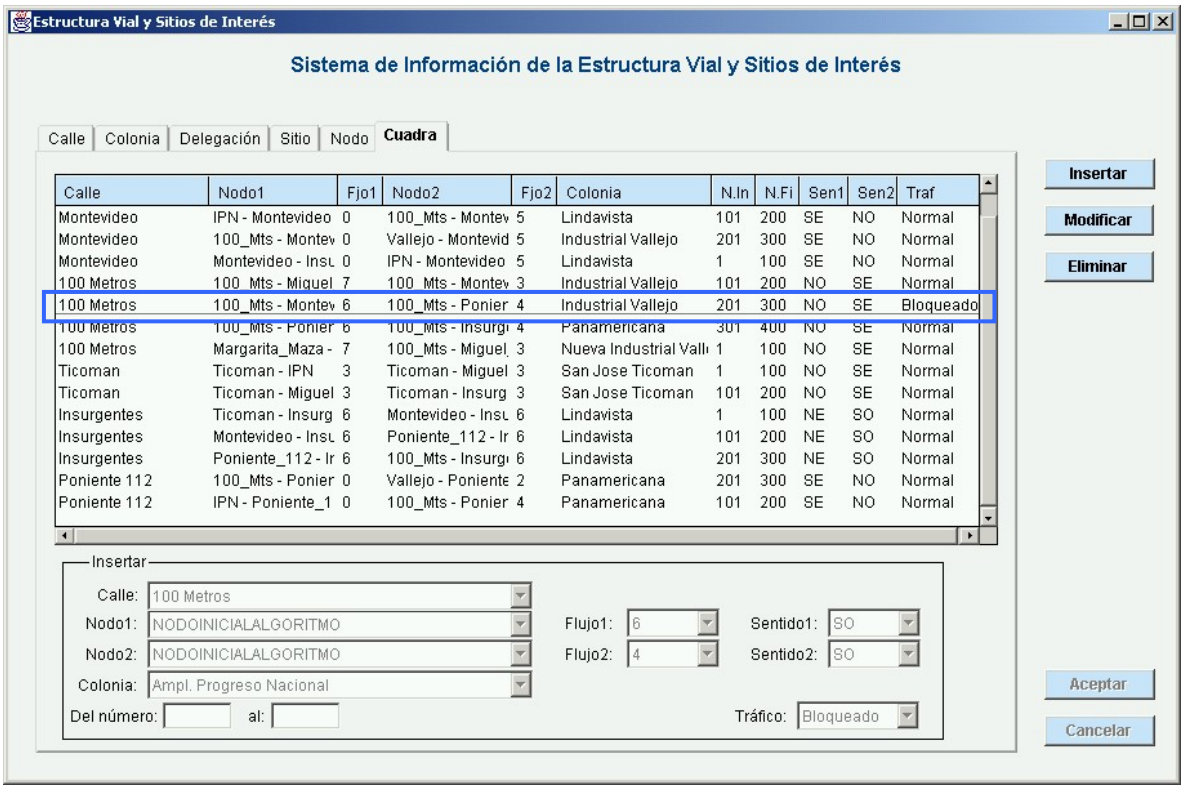
Figura 5.23 Secuencia de pantallas de prueba para la selección de un sitio de interés

Estos casos de uso en conjunto definen el proceso de búsqueda de la ruta óptima para llegar desde un punto, introducido a través de una dirección postal, a un sitio de interés que esté dado de alta en el sistema.

5.5.2 Prueba de Bloqueo

En capítulos anteriores se ha hablado que la aplicación proporcionaría una ruta óptima tomando en cuenta los distintos factores que podrían afectar la trayectoria para trasladarse del punto origen al punto destino, por ejemplo: una manifestación, un accidente, una obra pública, etcétera. Para probar esto se procederá a bloquear intencionalmente un tramo de calle que formaba parte de la ruta óptima en el ejemplo de la sección 5.5.1.

En la Figura 5.24 se muestra el bloqueo de un tramo de la calle 100Mts que formaba parte de la ruta óptima del ejemplo mencionado.



Calle	Nodo1	Fjo1	Nodo2	Fjo2	Colonia	N.In	N.Fi	Sen1	Sen2	Traf
Montevideo	IPN - Montevideo	0	100_Mts - Montev	5	Lindavista	101	200	SE	NO	Normal
Montevideo	100_Mts - Montev	0	Vallejo - Montevid	5	Industrial Vallejo	201	300	SE	NO	Normal
Montevideo	Montevideo - Insu	0	IPN - Montevideo	5	Lindavista	1	100	SE	NO	Normal
100 Metros	100_Mts - Miguel	7	100_Mts - Montev	3	Industrial Vallejo	101	200	NO	SE	Normal
100 Metros	100_Mts - Montev	6	100_Mts - Ponier	4	Industrial Vallejo	201	300	NO	SE	Bloqueado
100 Metros	100_Mts - Ponier	6	100_Mts - Insurg	4	Panamericana	301	400	NO	SE	Normal
100 Metros	Margarita_Maza -	7	100_Mts - Miguel	3	Nueva Industrial Vall	1	100	NO	SE	Normal
Ticoman	Ticoman - IPN	3	Ticoman - Miguel	3	San Jose Ticoman	1	100	NO	SE	Normal
Ticoman	Ticoman - Miguel	3	Ticoman - Insurg	3	San Jose Ticoman	101	200	NO	SE	Normal
Insurgentes	Ticoman - Insurg	6	Montevideo - Insu	6	Lindavista	1	100	NE	SO	Normal
Insurgentes	Montevideo - Insu	6	Poniente_112 - Ir	6	Lindavista	101	200	NE	SO	Normal
Insurgentes	Poniente_112 - Ir	6	100_Mts - Insurg	6	Lindavista	201	300	NE	SO	Normal
Poniente 112	100_Mts - Ponier	0	Vallejo - Poniente	2	Panamericana	201	300	SE	NO	Normal
Poniente 112	IPN - Poniente_1	0	100_Mts - Ponier	4	Panamericana	101	200	SE	NO	Normal

Figura 5.24 Bloqueo de un tramo de una calle

Ahora se ingresan nuevamente los datos de la dirección origen y los datos del sitio de interés al que se desea llegar. De nueva cuenta se solicita al sistema la ruta óptima arrojando la mostrada en la Figura 5.25. En ésta se puede ver que la ruta óptima no indica que se pueda volver a pasar por la sección de la calle que esta bloqueada.

Figura 5.25 Ruta óptima obtenida ante un bloqueo

En la Figura 5.26 se puede ver la ruta óptima que se obtiene al tener un bloqueo en la calle 100Mts entre Montevideo y Poniente 112.

Figura 5.26 Ruta óptima para el ejemplo de la sección 5.5.1 con un bloqueo en una calle

5.5.3 Prueba de Desconexión

En esta sección se da una demostración de un caso de desconexión. Inicialmente se realiza una consulta a la tabla **Loc_Busqueda** la cual se muestra en la Figura 5.27. En esta consulta se puede ver que no existe ningún registro en esta tabla, lo que indica que no existen búsquedas pendientes por ningún usuario.

```
SELECT * FROM LOC_Busqueda
```

idUsr	stsDir	URL1	URL2

Figura 5.27 Consulta de la tabla Loc_Busqueda en la parte inicial

Ahora se procede a ingresa al sistema, registrarse en la aplicación e iniciar una búsqueda. Las pantallas que muestran esto se presentan en la Figura 5.28. Inicialmente se teclea el URL de la aplicación, el sistema solicita el ingreso del nombre de usuario y su password. El usuario ingresa estos datos y el sistema muestra el menú principal. Posteriormente se van seleccionando los elementos que conformarán la búsqueda hasta antes de la desconexión.

Figura 5.28 Ingreso a la aplicación e inicio de la búsqueda hasta antes de la desconexión

Después de la última pantalla que muestra la Figura 5.28 se simula intencionalmente una desconexión cerrando y reiniciando el navegador WAP. Antes de volver a ingresar a la aplicación se verifica el estado de la tabla **Loc_Busqueda**, en la Figura 5.29 se muestra el contenido de ésta.

```
SELECT * FROM LOC_Busqueda
```

	idUsr	stsDir	URL1	URL2
1	1	1	http://148.204.211.192:8080/servlet/LocSrvDireccionSitio?dir=Ori&del=7&col=6	

Figura 5.29 Consulta de la tabla Loc_Busqueda después de la desconexión

Como se comentó con anterioridad, el campo **stsDir** con un valor igual a 1indica que el usuario no completo la dirección origen. El campo URL1 muestra el URL que indica en que parte de la búsqueda se quedó este usuario.

Con el navegador reiniciado se teclea el URL para ingresar a la aplicación móvil. Posteriormente se introducen el nombre de usuario y el password. El sistema valida la autenticidad del usuario y verifica si no tiene una búsqueda pendiente. En el caso de la prueba si se tiene una búsqueda pendiente, por lo que el sistema redirecciona a la última pantalla en la que se quedó este usuario. En la Figura 5.30 se muestra la secuencia de pantallas del proceso anterior.

Figura 5.30 Reingreso a la aplicación, registro de usuario y redirección según la búsqueda pendiente

A partir de la redirección a la pantalla pendiente, el usuario puede continuar su búsqueda con normalidad.

5.5.4 Cálculo de Costo de una Interacción con la Aplicación

A continuación se presenta un análisis de costo de una interacción con la aplicación. La interacción para la cual se calculó su costo es la búsqueda realizada en la sección 5.5.1.

La prueba se realizó desde un teléfono celular con servicio de Internet móvil proporcionado por Telcel, mediante una conexión GPRS en la que el Kb o fracción cuesta

$0.138 IVA incluido. En la Tabla 5.1 se presenta el cálculo del costo para la prueba mencionada:

Tabla 5.1 Cálculo de costo de una interacción con la aplicación

Pantalla	Costo (Kb)	Costo (pesos)
	355	0.49
	255	0.36
	374	0.52
	408	0.57
	1716	2.37
	981	1.36
	463	0.64

Tabla 5.1 Cálculo de costo de una interacción con la aplicación (Cont.)

Pantalla	Costo (Kb)	Costo (pesos)
	279	0.39
	181	0.25
	771	1.07
	467	0.65
	365	0.51

Haciendo la suma total del costo de cada pantalla se tiene que la interacción realizada tiene un costo de $9.18.

5.5.5 Comparación de Uso de la Aplicación versus Guía Roji

En este apartado se realiza una comparación del uso de la aplicación desarrollada con el uso del Guía Roji. Para esto se le pidió a 5 usuarios que realizaran una búsqueda de una ruta utilizando tanto la aplicación como el Guía Roji. La búsqueda de la ruta que se les solicitó encontrar es la misma de la sección 5.5.1, mostrando los resultados de la Tabla 5.2. Cabe hacer mención que a cada usuario se le dio previamente a su utilización, una demostración del funcionamiento de la aplicación.

Tabla 5.2 Comparación de tiempos entre el uso de la aplicación y del Guía Roji

Usuario	Edad (años)	Tiempo Aplicación (s)	Tiempo Guía Roji (s)
1	23	106	117
2	26	89	104
3	31	113	108
4	38	124	99
5	25	98	101

Los resultados muestran una ligera ventaja para la aplicación desarrollada haciendo notar que la ruta que los usuarios encontraron usando el Guía Roji pudo no haber sido la óptima, repercutiendo en el tiempo de traslado.

5.6 Resumen

En este capítulo se presentó un breve análisis de las herramientas utilizadas para realizar cada uno de los elementos de la fase de implementación de la aplicación. Posteriormente se da una explicación acerca de los detalles de implementación de cada módulo que conforma la aplicación, de los más importantes están la construcción del grafo, la programación del Algoritmo de Dijkstra y la generación de Páginas WML.

En la parte final se presentan las pruebas realizadas a la aplicación y los resultados obtenidos

En el siguiente capítulo se presentan las conclusiones finales y trabajos futuros que pueden mejorar y aumentar la funcionalidad y el desempeño que ofrece esta aplicación así como lo que aporta la tesis en general.

Capítulo 6

Conclusiones y Trabajos a Futuro

En este capítulo se presentan las conclusiones y comentarios finales a los que se llegaron con la realización de la presente tesis.

Además se proponen varias líneas de investigación que pueden continuarse a partir de lo investigado y desarrollado en este trabajo, así como las mejoras y optimizaciones que pudieran realizársele a la aplicación obtenida.

6.1 Logros Alcanzados

Los logros obtenidos con la finalización de esta tesis son descritos a partir de los objetivos planteados en el capítulo inicial de la misma. Estos logros son presentados en el orden de los capítulos que conforman la tesis, en donde se muestra la forma en cómo contribuyeron para el cumplimiento de dichos objetivos.

El objetivo principal de esta tesis, presentado en el primer capítulo, fue el de diseñar una arquitectura e implementar una infraestructura computacional que suministrara un conjunto de servicios que fueran accesibles desde un dispositivo de telefonía celular. Estos servicios permitirían la búsqueda de rutas óptimas para trasladarse de un punto a otro dentro de una ciudad tomando en cuenta su estructura vial, las condiciones de tráfico y el cierre de vialidades a consecuencia de manifestaciones, obras públicas, plantones, accidentes automovilísticos, entre otros. Otro servicio que estaría disponible sería el de permitir la búsqueda de los sitios de interés más cercanos a un punto dado por el usuario y la ruta para llegar hasta éste.

Dado el objetivo anterior, se plantea el desarrollo de una aplicación sobre una infraestructura de telefonía celular por lo tanto, en el capítulo 1, se presentó un trabajo de investigación acerca de la tecnología WAP, en el que se mostró su importancia y conveniencia de uso para desarrollar aplicaciones para dispositivos celulares (limitados en CPU, memoria, tamaño de pantalla y teclado) y para redes de comunicación celular (limitadas en ancho de banda y velocidades de transferencia de datos principalmente).

Adicionalmente se habla de la necesidad de obtener rutas óptimas en los grafos que representarían la estructura vial, por lo cual en el capítulo 2 se muestra una descripción de los algoritmos de búsqueda de caminos mínimos (óptimos) así como la información básica acerca de la Teoría de Grafos.

Al mencionar la creación de una aplicación computacional, implícitamente se habla de realizar cada una de las etapas que plantea la Ingeniería de Software para el desarrollo aplicaciones. Dado esto, en el capítulo 3 se presenta la etapa de Análisis. En esta etapa se propone una arquitectura para la infraestructura del sistema y de sus principales módulos. Además se detalla la funcionalidad que debe tener el producto terminado, así como los requerimientos que debe cumplir el sistema a través de los casos de uso de UML.

Continuando con el proceso de desarrollo de software, en el capítulo 4 se presenta la etapa de Diseño. En esta etapa se muestran los diagramas correspondientes con la especificación de UML, así como también se presenta el modelo lógico de la base de datos y una descripción del proceso de generación del grafo, que representa la vialidad, a partir de la información almacenada en la base de datos.

La siguiente etapa del proceso es la de implementación, la cual es presentada en el capítulo 5. Inicialmente se presenta una breve descripción de las herramientas utilizadas en cada uno de los módulos. Más adelante se presentan los detalles de implementación tanto de los servicios de base de datos que recuperan y dan

mantenimiento a la información del sistema como de las principales funciones del lado del lenguaje de programación. También se presentan los detalles técnicos para la generación del grafo a partir de la base de datos, del Algoritmo de Dijkstra para la búsqueda de rutas óptimas y de la generación de páginas en WML para los teléfonos celulares.

En este mismo capítulo se presenta la fase de Pruebas del sistema. La funcionalidad de la aplicación fue examinada a partir de las definiciones de los casos de uso especificados en la etapa de Análisis, en la que se corroboró que el sistema obtenido cumplía con los requerimientos y funcionalidad definidos al inicio del proceso de desarrollo. La aplicación *frontend* fue probada tanto en un emulador WAP desde una computadora como en un teléfono celular real, obteniendo un resultado satisfactorio en cada una de las pruebas cumpliendo de esta forma con el objetivo principal de esta tesis.

A lo largo de la tesis se han mencionado las ventajas y desventajas de la arquitectura planteada y de los detalles de implementación. A continuación se presenta un listado con los principales aportes que se realizaron en esta tesis:

- Diseño de una estructura de base de datos que permite almacenar información que representa la estructura vial de una ciudad.

- Desarrollo de un algoritmo para generar el grafo que representa la estructura vial con la información de BD.

- Creación de interfaces de usuario adecuadas para dispositivos celulares que permiten una fácil navegabilidad y rápida respuesta dadas las limitaciones que presentan tanto los dispositivos celulares como el medio de comunicación.

- Creación de un mecanismo de continuación de búsquedas ante posibles desconexiones que caracterizan al medio de comunicación celular.

- Implementación de un algoritmo de búsqueda de caminos más cortos para la resolución de un problema cotidiano como es evitar congestionamientos y bloqueos viales y obtener mejores tiempos de traslado.

Ahora se listan las principales desventajas y limitaciones que presenta la arquitectura planteada y su implementación:

- Para un adecuado funcionamiento de la aplicación se requiere de un operador que esté informado de las condiciones viales en tiempo real y en ese momento actualice la información del sistema y de esta manera se refleje en la búsqueda de rutas óptimas.

- Aún y cuando la navegabilidad de la aplicación móvil resulta sencilla y relativamente rápida, sería mucho mejor que la dirección inicial la pudiera obtener automáticamente de la ubicación física del dispositivo celular.

Por otro lado se tienen las ventajas que presenta la aplicación desarrollada con respecto a los desarrollos comerciales presentados al principio de la tesis en el apartado de Trabajos Previos Relacionados.

Con respecto al servicio **e-moción** que ofrece **Telefónica MoviStar** se tienen las siguientes ventajas:

- Permite la búsqueda del sitio de interés más cercano tomando en cuenta el giro o marca
- Proporciona la ruta óptima para llegar a un sitio de interés
- Permite buscar la ruta óptima para trasladarse de un punto de la ciudad a otro

Las ventajas que se tienen con respecto al servicio ***RUTA** que proporciona **Telcel** son:

- Horario de servicio continuo
- Permite la búsqueda de sitios de interés y la ruta óptima hacia ellos
- No es necesario tener una operadora
- El sistema despliega en pantalla la ruta óptima y datos del sitio de interés, con esto el usuario no necesita memorizarlos

6.2 Trabajos a Futuro

A continuación son descritos de manera breve algunos de los trabajos a futuro que podrían tomar como base lo conseguido con el desarrollo de esta tesis, así como también las líneas de investigación que ayudarían a extender y mejorar la aplicación e investigación realizadas.

6.2.1 Localización Automática del Usuario

La versión actual de la aplicación desarrollada en esta tesis requiere que el usuario introduzca la dirección origen y destino cuando aplique, en lo cual invierte un determinado tiempo y saldo. Esto es útil principalmente cuando el usuario desea introducir una dirección diferente a la que se encuentra en ese momento. Si, por el contrario, en un gran número de consultas se observa que el usuario comúnmente introduce como una de las dos direcciones su posición actual, que es la misma que la del dispositivo móvil, sería conveniente que el sistema le pudiera proporcionar una alternativa más eficiente para este proceso.

La funcionalidad de la aplicación podría ser mejorada de manera notable si ésta pudiera ajustar automáticamente la dirección origen o destino con la posición física del dispositivo celular. El sistema podría presentarle, al momento de empezar a introducir una dirección, una opción que especificara que el usuario desea que la tome automáticamente de la posición física del dispositivo celular.

La forma en que esto podría realizarse relativamente fácil y poco costosa, es la de explotar las celdas de red para conocer, gracias al identificador de celda (*cell ID*), la posición más o menos precisa del usuario dentro del área cubierta por una determinada celda. Obviamente, con GSM la precisión de tal tecnología es bastante reducida ya que, a menudo, el área de cobertura de una única celda de red es realmente muy amplia, con un radio de varios kilómetros, y el problema principal que se presenta es precisamente el cálculo de la posición exacta del usuario dentro de dicha área. La precisión debería mejorar mucho con el desarrollo del Sistema Universal de Telefonía Móvil (UMTS[1]) que prevé una cobertura mucho más densa, con áreas de cobertura de un radio de pocos metros en las zonas urbanas.

Una alternativa a las tecnologías citadas en el párrafo anterior sería el uso del Sistema de Posicionamiento Global (GPS[2]). Los servicios basados en el GPS, tecnología capaz de determinar la posición del usuario con una posición extrema dentro de una decena de metros, y que se sirven de las informaciones recogidas por los satélites, está revolucionando el sector de la telefonía móvil. Conviene tener en cuenta que, a diferencia de la tecnología *cell ID*, la principal desventaja del GPS reside en el hecho de que el usuario debe necesariamente disponer de un dispositivo móvil compatible con el sistema. Y, además, tal dispositivo móvil debe poder estar visible para tres satélites con el fin de que su posición se determine de forma eficaz. Condición muy difícil de obtener en ciudades urbanizadas y prácticamente imposible de satisfacer dentro de casas y otros edificios.

6.2.2 Integración en la aplicación de Algoritmos de Búsqueda de Caminos Mínimos Alternos y una mejor ponderación de los grafos.

En el capítulo 2 se realizó un trabajo de investigación acerca de los principales algoritmos de búsqueda de caminos mínimos. Para cada uno de ellos se proporcionó una descripción, un análisis de complejidad y el teorema que lo sustenta.

Como se pudo observar en capítulos posteriores, el Algoritmo de Dijkstra fue el elegido para ser estudiado con más detalle e implementado en la aplicación computacional. Este algoritmo fue elegido debido a que fue del que más información se obtuvo, además de ser el más difundido y analizado. Sin embargo, otros de los algoritmos estudiados en el capítulo 2 podrían ser más eficientes, arrojando mejores resultados en menor tiempo.

La ponderación del grafo es la clave para obtener resultados más confiables y que se apeguen más a la realidad. En la aplicación desarrollada fue utilizado un algoritmo de ponderación bastante precario que puede no dar los resultados esperados. Puede ser implementado un método de ponderación más eficiente basado en pruebas reales, que tome en cuenta factores adicionales que lo hagan más exacto y confiable.

[1] Universal Mobile Telephony System
[2] Global Positioning System

Algunos ejemplos de métricas que podrían ser tomadas en cuenta para que el algoritmo de búsqueda entregara mejores resultados son:

- Instalación de dispositivos para medir la carga de tráfico de una calle, dicho dispositivo detectaría la cantidad de automóviles que pasan por esta calle en un determinado tiempo y con esa información se actualizaría automáticamente la base de datos.

- Conteo del número de semáforos dado que, por lo regular, se puede circular más rápido por una calle donde haya menos.

- Consideración de las calles que en su cruce con otras se cuenten con algún puente, distribuidor vial, vueltas a la izquierda, etc.

6.2.3 Incorporación de otras zonas de la ciudad y calles secundarias alternas

Para las pruebas realizadas a la aplicación de esta tesis se consideró una parte de la delegación Gustavo A. Madero (norte de la Ciudad de México) y solo tomando en cuenta las principales arterias de esta zona. Como trabajo a futuro puede considerarse el incorporar otras zonas de la Ciudad de México que permitiera probar la eficacia, el desempeño y la rapidez del sistema al entregar los resultados. En caso de que alguno de los factores antes mencionados se viera afectado, se podrían buscar alternativas o identificar puntos del sistema que pudieran corregirle la falla detectada.

Por otro lado está el considerar calles secundarías que aunque pueden soportar un menor flujo de autos, representan una buena alternativa para circular por ellas y de esta forma distribuir el tráfico. Implementar esto ocasiona un problema al agregar calles secundarias a la aplicación dado que el grafo que represente a la estructura vial aumentará también en nodos y aristas con el consiguiente aumento de ejecución de los procesos que calculan las rutas óptimas entre otros. Este problema también tendría que ser mitigado o corregido para que el sistema siguiera siendo funcional, sin descuidar el desempeño y la rapidez.

6.2.4 Utilización de GIS[3] como Fuente de Información para generar el Grafo

Un GIS se define como un conjunto de métodos, herramientas y datos que están diseñados para actuar coordinada y lógicamente para capturar, almacenar, analizar, transformar y presentar toda la información geográfica y de sus atributos con el fin de satisfacer múltiples propósitos. Los GIS son sistemas que permiten administrar y analizar la información espacial y que surgió como resultado de la necesidad de disponer rápidamente de información para resolver problemas y contestar preguntas de modo inmediato.

Las soluciones para muchos problemas frecuentemente requieren acceso a varios tipos de información que sólo pueden ser relacionadas por geografía o distribución

[3] Geographic Information System.- Sistema de Información Geográfica

espacial. Los GIS permiten almacenar y manipular información usando geografía y para analizar patrones, relaciones, y tendencias en la información con la finalidad de mejorar en la toma de decisiones.

Como se explicó anteriormente, el grafo que representa la estructura vial es generado a partir de la información almacenada en la BD. Dicha información tiene que ser introducida al sistema de forma manual a través de la aplicación de mantenimiento desarrollada también de esta tesis. Una mejor alternativa a esto podría ser que un GIS alimentara la base de datos con información de la estructura vial de la ciudad donde se implantaría la aplicación móvil.

6.2.5 Computación Ubicua (*Ubiquitous Computing*)

La computación ubicua es consecuencia del auge del fenómeno de las telecomunicaciones y en especial de Internet y los dispositivos móviles, ya sean teléfonos o computadoras. Este fenómeno que es una realidad que comprende a millones de dispositivos de pequeño tamaño en una gran red distribuida que incluso será más amplia que la actual Internet.

Al día de hoy nadie discute que el futuro de las comunicaciones se basa es poder dar más servicios en dispositivos de menor tamaño.

Los teléfonos móviles con acceso a Internet y PDAs que se comunican inalámbricamente con otros dispositivos próximos, inician una nueva etapa en el campo de la informática y las telecomunicaciones. Poder tener acceso a información desde cualquier punto sin necesidad de estar ligado a un cable suscita preguntas como que en próximos años todos los aspectos cotidianos puedan estar ligados a las computadoras. Poder tener acceso a información mientras se está en movimiento o tomando café sin necesidad de tener una conexión por medio de un cable, parecía impensable hace unos años aunque actualmente ya es posible.

La computación ubicua marcará el futuro próximo en el campo de la informática y las telecomunicaciones [WWW11].

El término Computación Ubicua que denota esta visión, fue acuñado hace más de diez años por Mark Weiser [WEI91], un investigador del Centro de Investigación de Xerox Palo Alto. Weiser ve la tecnología solamente como un medio para un fin y como algo que debería quedar en segundo plano para permitir al usuario concentrarse completamente en la tarea que está realizando. En este sentido, considerar a la computadora personal como una herramienta universal para la tecnología de la información sería un enfoque equivocado, ya que su complejidad absorbería demasiado la atención del usuario.

Según Weiser, la computadora como dispositivo dedicado debería desaparecer, mientras que al mismo tiempo debería poner a disposición de todo lo que nos rodea sus capacidades de procesamiento de la información.

Weiser ve el término Computación Ubicua en un sentido más académico e idealista como una visión de tecnología discreta, centrada en la persona, mientras que la industria ha acuñado por eso el término Computación Pervasiva (*Pervasive Computing*) o ampliamente difundida con un enfoque ligeramente diferente: Aunque su visión siga siendo todavía integrar el procesamiento de la información en objetos cotidianos de forma casi invisible, su objetivo principal es utilizar tales objetos en un futuro próximo en el ámbito del comercio electrónico y para técnicas de negocios basados en web.

El potencial de las aplicaciones que utilizan objetos cotidianos inteligentes parece inmenso, especialmente si se asume que los objetos podrían utilizar, teóricamente, tecnología de conexión en red espontánea para funcionar de forma colaborativa entre sí, acceder a información almacenada en bases de datos en línea o en Internet, o utilizar cualquier servicio disponible en Internet.

Esta tesis representa un paso más en el camino de lograr el sueño de Weiser debido a que la aplicación desarrollada en ésta proporciona un conjunto de servicios e información accesibles aún mientras se está en movimiento y sin necesidad de tener una conexión a través algún cable.

6.3 Comentarios Finales

Con el paso de los años, el índice vehicular de las principales ciudades del mundo ha ido creciendo considerablemente y, por consiguiente, los problemas de tráfico y tiempos de traslado dentro de estas ciudades han aumentado en la misma proporción y forma. Nuevas y mejores vialidades en conjunto con aplicaciones del tipo como la desarrollada en esta tesis se vuelven cada vez más imprescindibles.

La aplicación desarrollada en la presente tesis ofrece tiempos de traslado menores, previniendo embotellamientos, manifestaciones, bloqueos, accidentes viales, etc. que pudieran surgir de imprevisto ó hasta los que hubieran sido planeados debido a una obra pública. Permite también ser una especie de balanceador de cargas de automóviles, de manera que podría repartir el tráfico cuando se tuvieran disponibles más de una ruta de traslado.

Por otro lado está el uso de esta aplicación para otros fines, al permitir encontrar sitios de interés cercanos a un punto deseado por el usuario, El sistema al encontrarle un lugar cercano al usuario le permitiría, si se trata de una emergencia, llegar lo más pronto posible; si su objetivo es de esparcimiento, le permitiría disponer de más tiempo para tal causa al encontrar lugares lo más cercano posible también. Al proporcionar estos servicios de localización, las compañías de telefonía celular estarían ofreciendo a sus usuarios un servicio de valor agregado. Adicionalmente estarían recibiendo ganancias por parte de los comercios, restaurantes y cualquier sitio de interés que quisiera ser incluido en la base de datos del sistema.

Un sistema de este tipo puede tener un sinfín de aplicaciones prácticamente disponibles en cualquier lugar siempre y cuando se tenga un dispositivo móvil con cobertura, todo esto a un precio razonablemente bajo debido a la gran difusión, popularidad y avances tecnológicos que tienen estos tipos de dispositivos.

Apéndice A

Código Fuente *(Backend)*

LocAppMtto.java

```
import java.awt.*;
import symantec.itools.awt.TabPanel;
import symantec.itools.awt.MultiList;
import symantec.itools.awt.BorderPanel;
import com.symantec.itools.awt.MaskedTextField;
import java.sql.*;

public class LocAppMtto extends Frame
{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

        static final int NINGUNA = 0;
        static final int INSERTAR = 1;
        static final int MODIFICAR = 2;
        static final int ELIMINAR = 3;
        int operacion = 0;

        public LocAppMtto()
        {
           //{{INIT_MENUS
                //}}

                //{{REGISTER_LISTENERS
                SymWindow aSymWindow = new SymWindow();
                this.addWindowListener(aSymWindow);
                SymAction lSymAction = new SymAction();
                btn_insertar.addActionListener(lSymAction);
                btn_aceptar.addActionListener(lSymAction);
                btn_modificar.addActionListener(lSymAction);
                btn_eliminar.addActionListener(lSymAction);
                btn_cancelar.addActionListener(lSymAction);
                //}}

                // This code is automatically generated by Visual Cafe when you add
                // components to the visual environment. It instantiates and initializes
                // the components. To modify the code, only use code syntax that matches
                // what Visual Cafe can generate, or Visual Cafe may be unable to back
                // parse your Java file into its visual environment.

                //{{INIT_CONTROLS
                setLayout(null);
                setSize(906,578);
                setVisible(false);
                try {
                        {
                                String[] tempString = new String[6];
                                tempString[0] = "Calle";
                                tempString[1] = "Colonia";
                                tempString[2] = "Delegación";
                                tempString[3] = "Sitio";
                                tempString[4] = "Nodo";
                                tempString[5] = "Cuadra";
                                tabPanel.setPanelLabels(tempString);
                        }
                }
```

```
catch(java.beans.PropertyVetoException e) { }
add(tabPanel);
tabPanel.setBounds(12,60,780,506);
panel0.setLayout(null);
tabPanel.add(panel0);
panel0.setBounds(12,33,756,462);
try {
        ml_calle.setHeadingBg(new java.awt.Color(185,206,208));
}
catch(java.beans.PropertyVetoException e) { }
try {
        {
                String[] tempString = new String[2];
                tempString[0] = "0";
                tempString[1] = "500";
                ml_calle.setColumnSizes(tempString);
        }
}
catch(java.beans.PropertyVetoException e) { }
try {
        {
                String[] tempString = new String[2];
                tempString[0] = "idCalle";
                tempString[1] = "Calle";
                ml_calle.setHeadings(tempString);
        }
}
catch(java.beans.PropertyVetoException e) { }
panel0.add(ml_calle);
ml_calle.setBackground(java.awt.Color.white);
ml_calle.setBounds(12,15,552,360);
try {
        bp_calle.setLabel("Insertar");
}
catch(java.beans.PropertyVetoException e) { }
try {
        bp_calle.setAlignStyle(symantec.itools.awt.BorderPanel.ALIGN_LEFT);
}
catch(java.beans.PropertyVetoException e) { }
bp_calle.setLayout(null);
panel0.add(bp_calle);
bp_calle.setBounds(12,375,492,84);
txt_nombre_calle.setEnabled(false);
bp_calle.add(txt_nombre_calle);
txt_nombre_calle.setBounds(62,9,396,21);
label1.setText("Calle:");
bp_calle.add(label1);
label1.setBounds(14,9,57,17);
panel1.setLayout(null);
tabPanel.add(panel1);
panel1.setBounds(12,33,756,462);
panel1.setVisible(false);
try {
        ml_colonia.setHeadingBg(new java.awt.Color(185,206,208));
}
catch(java.beans.PropertyVetoException e) { }
try {
        {
                String[] tempString = new String[4];
                tempString[0] = "0";
                tempString[1] = "280";
                tempString[2] = "0";
                tempString[3] = "250";
                ml_colonia.setColumnSizes(tempString);
```

```
		}
}
catch(java.beans.PropertyVetoException e) { }
try {
		{
			String[] tempString = new String[4];
			tempString[0] = "idColonia";
			tempString[1] = "Colonia";
			tempString[2] = "idDel";
			tempString[3] = "Delegación";
			ml_colonia.setHeadings(tempString);
		}
}
catch(java.beans.PropertyVetoException e) { }
panel1.add(ml_colonia);
ml_colonia.setBackground(java.awt.Color.white);
ml_colonia.setBounds(12,15,555,336);
try {
		bp_colonia.setLabel("Insertar");
}
catch(java.beans.PropertyVetoException e) { }
try {
		bp_colonia.setAlignStyle(symantec.itools.awt.BorderPanel.ALIGN_LEFT);
}
catch(java.beans.PropertyVetoException e) { }
bp_colonia.setLayout(null);
panel1.add(bp_colonia);
bp_colonia.setBounds(12,351,420,101);
txt_colonia.setEnabled(false);
bp_colonia.add(txt_colonia);
txt_colonia.setForeground(java.awt.Color.black);
txt_colonia.setBounds(98,0,276,21);
label2.setText("  Colonia:");
bp_colonia.add(label2);
label2.setBounds(26,0,60,24);
cbo_p1_delegacion.setEnabled(false);
bp_colonia.add(cbo_p1_delegacion);
cbo_p1_delegacion.setBounds(98,33,276,21);
label3.setText("Delegación:");
bp_colonia.add(label3);
label3.setBounds(14,33,72,24);
panel2.setLayout(null);
tabPanel.add(panel2);
panel2.setBounds(12,33,756,462);
panel2.setVisible(false);
try {
		ml_delegacion.setHeadingBg(new java.awt.Color(185,206,208));
}
catch(java.beans.PropertyVetoException e) { }
try {
		{
			String[] tempString = new String[2];
			tempString[0] = "0";
			tempString[1] = "320";
			ml_delegacion.setColumnSizes(tempString);
		}
}
catch(java.beans.PropertyVetoException e) { }
try {
		{
			String[] tempString = new String[2];
			tempString[0] = "idDelegacion";
			tempString[1] = "Delegación";
			ml_delegacion.setHeadings(tempString);
```

```
		}
}
catch(java.beans.PropertyVetoException e) { }
panel2.add(ml_delegacion);
ml_delegacion.setBackground(java.awt.Color.white);
ml_delegacion.setBounds(12,15,360,437);
try {
		bp_delegacion.setLabel("Insertar");
}
catch(java.beans.PropertyVetoException e) { }
try {
		bp_delegacion.setAlignStyle(symantec.itools.awt.BorderPanel.ALIGN_LEFT);
}
catch(java.beans.PropertyVetoException e) { }
bp_delegacion.setLayout(null);
panel2.add(bp_delegacion);
bp_delegacion.setBounds(384,15,360,84);
txt_delegacion.setEnabled(false);
bp_delegacion.add(txt_delegacion);
txt_delegacion.setBounds(74,9,252,21);
label4.setText("Delegación:");
bp_delegacion.add(label4);
label4.setBounds(2,9,72,24);
panel3.setLayout(null);
tabPanel.add(panel3);
panel3.setBounds(12,33,756,462);
panel3.setVisible(false);
try {
		ml_sitio.setHeadingBg(new java.awt.Color(185,206,208));
}
catch(java.beans.PropertyVetoException e) { }
try {
		{
			String[] tempString = new String[12];
			tempString[0] = "0";
			tempString[1] = "150";
			tempString[2] = "0";
			tempString[3] = "120";
			tempString[4] = "55";
			tempString[5] = "0";
			tempString[6] = "120";
			tempString[7] = "70";
			tempString[8] = "0";
			tempString[9] = "120";
			tempString[10] = "0";
			tempString[11] = "90";
			ml_sitio.setColumnSizes(tempString);
		}
}
catch(java.beans.PropertyVetoException e) { }
try {
		{
			String[] tempString = new String[12];
			tempString[0] = "idSitio";
			tempString[1] = "Nombre";
			tempString[2] = "idCalle";
			tempString[3] = "Calle";
			tempString[4] = "Número";
			tempString[5] = "idColonia";
			tempString[6] = "Colonia";
			tempString[7] = "Teléfono";
			tempString[8] = "idDel";
			tempString[9] = "Delegación";
			tempString[10] = "idCat";
```

```
                    tempString[11] = "Categoria";
                    ml_sitio.setHeadings(tempString);
            }
}
catch(java.beans.PropertyVetoException e) { }
panel3.add(ml_sitio);
ml_sitio.setBackground(java.awt.Color.white);
ml_sitio.setBounds(12,15,732,288);
try {
            borderPanel1.setLabel("Insertar");
}
catch(java.beans.PropertyVetoException e) { }
try {
            borderPanel1.setAlignStyle(symantec.itools.awt.BorderPanel.ALIGN_LEFT);
}
catch(java.beans.PropertyVetoException e) { }
borderPanel1.setLayout(null);
panel3.add(borderPanel1);
borderPanel1.setBounds(12,315,732,138);
txt_sitio.setEnabled(false);
borderPanel1.add(txt_sitio);
txt_sitio.setBounds(74,0,280,21);
label5.setText("Sitio:");
borderPanel1.add(label5);
label5.setBounds(2,-3,63,21);
cbo_p3_calle.setEnabled(false);
borderPanel1.add(cbo_p3_calle);
cbo_p3_calle.setBounds(74,22,280,21);
label6.setText("Calle:");
borderPanel1.add(label6);
label6.setBounds(2,21,58,22);
label7.setText("Colonia:");
borderPanel1.add(label7);
label7.setBounds(2,45,69,18);
cbo_p3_colonia.setEnabled(false);
borderPanel1.add(cbo_p3_colonia);
cbo_p3_colonia.setBounds(74,45,280,21);
label8.setText("Número:");
borderPanel1.add(label8);
label8.setBounds(386,21,48,24);
label9.setText("Teléfono:");
borderPanel1.add(label9);
label9.setBounds(386,45,60,24);
txtm_telefono.setCursor(java.awt.Cursor.getPredefinedCursor(java.awt.Cursor.TEXT_CURSOR));
txtm_telefono.setEnabled(false);
borderPanel1.add(txtm_telefono);
txtm_telefono.setBounds(446,48,76,21);
txtm_numero.setCursor(java.awt.Cursor.getPredefinedCursor(java.awt.Cursor.TEXT_CURSOR));
txtm_numero.setEnabled(false);
borderPanel1.add(txtm_numero);
txtm_numero.setBounds(446,25,76,21);
label20.setText("Delegación:");
borderPanel1.add(label20);
label20.setBounds(2,69,69,18);
cbo_p3_delegacion.setEnabled(false);
borderPanel1.add(cbo_p3_delegacion);
cbo_p3_delegacion.setBounds(74,69,280,21);
label21.setText("Categoría:");
borderPanel1.add(label21);
label21.setBounds(374,2,69,18);
cbo_p3_categoria.setEnabled(false);
borderPanel1.add(cbo_p3_categoria);
cbo_p3_categoria.setBounds(446,0,240,21);
panel4.setLayout(null);
```

```
tabPanel.add(panel4);
panel4.setBounds(12,33,756,462);
panel4.setVisible(false);
try {
	ml_nodo.setHeadingBg(new java.awt.Color(185,206,208));
}
catch(java.beans.PropertyVetoException e) { }
try {
	{
		String[] tempString = new String[2];
		tempString[0] = "0";
		tempString[1] = "500";
		ml_nodo.setColumnSizes(tempString);
	}
}
catch(java.beans.PropertyVetoException e) { }
try {
	{
		String[] tempString = new String[2];
		tempString[0] = "idNodo";
		tempString[1] = "Nodo";
		ml_nodo.setHeadings(tempString);
	}
}
catch(java.beans.PropertyVetoException e) { }
panel4.add(ml_nodo);
ml_nodo.setBackground(java.awt.Color.white);
ml_nodo.setBounds(12,15,720,360);
try {
	borderPanel2.setLabel("Insertar");
}
catch(java.beans.PropertyVetoException e) { }
try {
	borderPanel2.setAlignStyle(symantec.itools.awt.BorderPanel.ALIGN_LEFT);
}
catch(java.beans.PropertyVetoException e) { }
borderPanel2.setLayout(null);
panel4.add(borderPanel2);
borderPanel2.setBounds(12,375,516,80);
label10.setText("Cruce:");
borderPanel2.add(label10);
label10.setBounds(14,9,48,24);
borderPanel2.add(txt_nodo);
txt_nodo.setBounds(62,9,332,21);
panel5.setLayout(null);
tabPanel.add(panel5);
panel5.setBounds(12,33,756,462);
panel5.setVisible(false);
try {
	ml_cuadra.setMultipleMode(true);
}
catch(java.beans.PropertyVetoException e) { }
try {
	ml_cuadra.setHeadingBg(new java.awt.Color(185,206,208));
}
catch(java.beans.PropertyVetoException e) { }
try {
	{
		String[] tempString = new String[18];
		tempString[0] = "0";
		tempString[1] = "120";
		tempString[2] = "0";
		tempString[3] = "100";
		tempString[4] = "35";
```

```
                    tempString[5] = "0";
                    tempString[6] = "100";
                    tempString[7] = "35";
                    tempString[8] = "0";
                    tempString[9] = "120";
                    tempString[10] = "34";
                    tempString[11] = "34";
                    tempString[12] = "0";
                    tempString[13] = "38";
                    tempString[14] = "0";
                    tempString[15] = "38";
                    tempString[16] = "0";
                    tempString[17] = "70";
                    ml_cuadra.setColumnSizes(tempString);
            }
}
catch(java.beans.PropertyVetoException e) { }
try {
            {
                    String[] tempString = new String[18];
                    tempString[0] = "idCalle";
                    tempString[1] = "Calle";
                    tempString[2] = "idNodo1";
                    tempString[3] = "Nodo1";
                    tempString[4] = "Fjo1";
                    tempString[5] = "idNodo2";
                    tempString[6] = "Nodo2";
                    tempString[7] = "Fjo2";
                    tempString[8] = "idColonia";
                    tempString[9] = "Colonia";
                    tempString[10] = "N.In";
                    tempString[11] = "N.Fi";
                    tempString[12] = "idS1";
                    tempString[13] = "Sen1";
                    tempString[14] = "idS2";
                    tempString[15] = "Sen2";
                    tempString[16] = "idtraf";
                    tempString[17] = "Traf";
                    ml_cuadra.setHeadings(tempString);
            }
}
catch(java.beans.PropertyVetoException e) { }
panel5.add(ml_cuadra);
ml_cuadra.setBackground(java.awt.Color.white);
ml_cuadra.setBounds(12,15,732,288);
try {
            borderPanel3.setLabel("Insertar");
}
catch(java.beans.PropertyVetoException e) { }
try {
            borderPanel3.setAlignStyle(symantec.itools.awt.BorderPanel.ALIGN_LEFT);
}
catch(java.beans.PropertyVetoException e) { }
borderPanel3.setLayout(null);
panel5.add(borderPanel3);
borderPanel3.setBounds(12,303,696,157);
label11.setText("    Calle:");
borderPanel3.add(label11);
label11.setBounds(14,0,48,21);
cbo_p5_calle.setEnabled(false);
borderPanel3.add(cbo_p5_calle);
cbo_p5_calle.setBounds(62,0,300,21);
label12.setText("Nodo1:");
borderPanel3.add(label12);
```

```
label12.setBounds(14,21,48,24);
cbo_p5_nodo1.setEnabled(false);
borderPanel3.add(cbo_p5_nodo1);
cbo_p5_nodo1.setBounds(62,22,300,21);
label13.setText("Flujo1:");
borderPanel3.add(label13);
label13.setBounds(386,21,48,24);
cbo_p5_flujo1.addItem("0");
cbo_p5_flujo1.addItem("1");
cbo_p5_flujo1.addItem("2");
cbo_p5_flujo1.addItem("3");
cbo_p5_flujo1.addItem("4");
cbo_p5_flujo1.addItem("5");
cbo_p5_flujo1.addItem("6");
cbo_p5_flujo1.addItem("7");
cbo_p5_flujo1.addItem("8");
cbo_p5_flujo1.addItem("9");
try {
        cbo_p5_flujo1.select(0);
}
catch (IllegalArgumentException e) { }
cbo_p5_flujo1.setEnabled(false);
borderPanel3.add(cbo_p5_flujo1);
cbo_p5_flujo1.setBounds(434,21,73,21);
label14.setText("Nodo2:");
borderPanel3.add(label14);
label14.setBounds(14,45,48,24);
cbo_p5_nodo2.setEnabled(false);
borderPanel3.add(cbo_p5_nodo2);
cbo_p5_nodo2.setBounds(62,45,300,21);
label15.setText("Flujo2:");
borderPanel3.add(label15);
label15.setBounds(386,45,48,24);
cbo_p5_flujo2.addItem("0");
cbo_p5_flujo2.addItem("1");
cbo_p5_flujo2.addItem("2");
cbo_p5_flujo2.addItem("3");
cbo_p5_flujo2.addItem("4");
cbo_p5_flujo2.addItem("5");
cbo_p5_flujo2.addItem("6");
cbo_p5_flujo2.addItem("7");
cbo_p5_flujo2.addItem("8");
cbo_p5_flujo2.addItem("9");
try {
        cbo_p5_flujo2.select(0);
}
catch (IllegalArgumentException e) { }
cbo_p5_flujo2.setEnabled(false);
borderPanel3.add(cbo_p5_flujo2);
cbo_p5_flujo2.setBounds(434,45,73,21);
label16.setText("  Colonia:");
borderPanel3.add(label16);
label16.setBounds(2,69,60,24);
cbo_p5_colonia.setEnabled(false);
borderPanel3.add(cbo_p5_colonia);
cbo_p5_colonia.setBounds(62,69,300,21);
label17.setText("Del número:");
borderPanel3.add(label17);
label17.setBounds(2,93,72,24);
txtm_noinic.setCursor(java.awt.Cursor.getPredefinedCursor(java.awt.Cursor.TEXT_CURSOR));
txtm_noinic.setEnabled(false);
borderPanel3.add(txtm_noinic);
txtm_noinic.setBounds(74,94,54,21);
label18.setText("  al:");
```

```
borderPanel3.add(label18);
label18.setBounds(134,93,24,24);
txtm_nofinal.setCursor(java.awt.Cursor.getPredefinedCursor(java.awt.Cursor.TEXT_CURSOR));
txtm_nofinal.setEnabled(false);
borderPanel3.add(txtm_nofinal);
txtm_nofinal.setBounds(158,94,54,21);
label22.setText("Sentido2:");
borderPanel3.add(label22);
label22.setBounds(530,45,60,24);
cbo_p5_trafico.setEnabled(false);
borderPanel3.add(cbo_p5_trafico);
cbo_p5_trafico.setBounds(566,93,96,21);
label23.setText("Tráfico:");
borderPanel3.add(label23);
label23.setBounds(518,93,48,24);
cbo_p5_sentido1.setEnabled(false);
borderPanel3.add(cbo_p5_sentido1);
cbo_p5_sentido1.setBounds(590,21,73,21);
label19.setText("Sentido1:");
borderPanel3.add(label19);
label19.setBounds(530,21,60,24);
cbo_p5_sentido2.setEnabled(false);
borderPanel3.add(cbo_p5_sentido2);
cbo_p5_sentido2.setBounds(590,45,73,21);
try {
        tabPanel.setCurrentPanelNdx(0);
}
catch(java.beans.PropertyVetoException e) { }
btn_insertar.setLabel("Insertar");
add(btn_insertar);
btn_insertar.setBackground(new java.awt.Color(185,206,208));
btn_insertar.setFont(new Font("Dialog", Font.BOLD, 12));
btn_insertar.setBounds(804,96,86,24);
btn_modificar.setLabel("Modificar");
add(btn_modificar);
btn_modificar.setBackground(new java.awt.Color(185,206,208));
btn_modificar.setFont(new Font("Dialog", Font.BOLD, 12));
btn_modificar.setBounds(804,132,85,24);
btn_eliminar.setLabel("Eliminar");
add(btn_eliminar);
btn_eliminar.setBackground(new java.awt.Color(185,206,208));
btn_eliminar.setFont(new Font("Dialog", Font.BOLD, 12));
btn_eliminar.setBounds(804,168,85,24);
btn_cancelar.setLabel("Cancelar");
btn_cancelar.setEnabled(false);
add(btn_cancelar);
btn_cancelar.setBackground(new java.awt.Color(185,206,208));
btn_cancelar.setFont(new Font("Dialog", Font.BOLD, 12));
btn_cancelar.setBounds(804,528,86,24);
btn_aceptar.setLabel("Aceptar");
btn_aceptar.setEnabled(false);
add(btn_aceptar);
btn_aceptar.setBackground(new java.awt.Color(185,206,208));
btn_aceptar.setFont(new Font("Dialog", Font.BOLD, 12));
btn_aceptar.setBounds(804,492,86,24);
label24.setText("Sistema de Información de la Estructura Vial y Sitios de Interés");
label24.setAlignment(java.awt.Label.CENTER);
add(label24);
label24.setForeground(new java.awt.Color(0,64,128));
label24.setFont(new Font("Dialog", Font.BOLD, 16));
label24.setBounds(156,12,600,24);
setTitle("Estructura Vial y Sitios de Interés");
//}}
```

```
            inicializa();
}

private ResultSet ejecutaQuery(String sp){
            try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
            stmt = con.createStatement();
            rs = stmt.executeQuery(sp);
   }catch(Exception e){
      debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
      e.printStackTrace(System.out);
   }
   return rs;
}

private int ejecutaUpdate(String sp){
            int filasMod = -1;
            try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
            stmt = con.createStatement();
            filasMod = stmt.executeUpdate(sp);
   }catch(Exception e){
      debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
      e.printStackTrace(System.out);
   }
   return filasMod;
}

private void cierraBD(){
            try{
                        rs.close();
                        stmt.close();
                        con.close();
            }catch(Exception e){
      debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
      e.printStackTrace(System.out);
   }
}
/*
private void inicializa(){
   cargaDatosPanel0();
   cargaDatosPanel2();
   cargaDatosPanel4();
}
*/
private void inicializa(){
            llenaControles();
            llenaMultilists();
}

private void llenaControles(){
            llenaComboDelegacion(cbo_p1_delegacion);

            llenaComboCalle(cbo_p3_calle);
            llenaComboColonia(cbo_p3_colonia);
            llenaComboDelegacion(cbo_p3_delegacion);
            llenaComboCategoria(cbo_p3_categoria);

            llenaComboCalle(cbo_p5_calle);
            llenaComboColonia(cbo_p5_colonia);
            llenaComboNodo(cbo_p5_nodo1);
            llenaComboNodo(cbo_p5_nodo2);
```

```
            llenaComboSentido(cbo_p5_sentido1);
            llenaComboSentido(cbo_p5_sentido2);
            llenaComboTrafico(cbo_p5_trafico);
}

private void llenaMultilists(){
   cargaDatosPanel0();
   cargaDatosPanel1();
   cargaDatosPanel2();
   cargaDatosPanel3();
   cargaDatosPanel4();
   cargaDatosPanel5();

}


private void llenaComboDelegacion(Choice cbo){
   try{
   ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_delegacion");

      int idDel = 0;
      String deleg = "";
      cbo.removeAll();

      if(rs1!=null){
         while(rs.next()){
            idDel = rs1.getInt(1);
            deleg = rs1.getString(2).trim();

            cbo.addItem(deleg + "                                                                    >>>" +
                                                   idDel);
         }
      }
      else
           debug("rs1 fue null");

      rs1.close();
      cierraBD();
   }catch(Exception e){
      debug("Error en llenaComboDelegacion():" + e.toString() + "_Msg=" + e.getMessage());
      e.printStackTrace(System.out);
   }

}

private void llenaComboCalle(Choice cbo){
   try{
   ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_calle");

      int idCalle = 0;
      String calle = "";
      cbo.removeAll();

      if(rs1!=null){
         while(rs.next()){
            idCalle = rs1.getInt(1);
            calle = rs1.getString(2).trim();

            cbo.addItem(calle + "                                                                  >>>" +
                                                   idCalle);
         }
      }
      else
           debug("rs1 fue null");
```

```
            rs1.close();
            cierraBD();
        }catch(Exception e){
            debug("Error en llenaComboCalle():" + e.toString() + "_Msg=" + e.getMessage());
            e.printStackTrace(System.out);
        }

    }

    private void llenaComboColonia(Choice cbo){
        try{
        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_colonia");

            int idColonia = 0;
            String colonia = "";
            cbo.removeAll();

            if(rs1!=null){
                while(rs.next()){
                    idColonia = rs1.getInt(1);
                    colonia = rs1.getString(2).trim();

                    cbo.addItem(colonia + "                                                                    >>>" +
                                                        idColonia);
                }
            }
            else
                 debug("rs1 fue null");

            rs1.close();
            cierraBD();
        }catch(Exception e){
            debug("Error en llenaComboColonia():" + e.toString() + "_Msg=" + e.getMessage());
            e.printStackTrace(System.out);
        }

    }

    private void llenaComboCategoria(Choice cbo){
        try{
        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_categoria");

            int idCategoria = 0;
            String categoria = "";
            cbo.removeAll();

            if(rs1!=null){
                while(rs.next()){
                    idCategoria = rs1.getInt(1);
                    categoria = rs1.getString(2).trim();

                    cbo.addItem(categoria + "                                                                  >>>" +
                                                        idCategoria);
                }
            }
            else
                 debug("rs1 fue null");

            rs1.close();
            cierraBD();
        }catch(Exception e){
            debug("Error en llenaComboCategoria():" + e.toString() + "_Msg=" + e.getMessage());
            e.printStackTrace(System.out);
```

```
    }

}

private void llenaComboNodo(Choice cbo){
    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_nodo");

        int idNodo = 0;
        String nodo = "";
        cbo.removeAll();

        if(rs1!=null){
            while(rs.next()){
                idNodo = rs1.getInt(1);
                nodo = rs1.getString(2).trim();

                cbo.addItem(nodo + "                                                                    >>>" +
                                                    idNodo);
            }
        }
        else
             debug("rs1 fue null");

        rs1.close();
        cierraBD();
    }catch(Exception e){
        debug("Error en llenaComboNodo():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }

}

private void llenaComboSentido(Choice cbo){
    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sentido");

        int idSent = 0;
        String sent = "";
        cbo.removeAll();

        if(rs1!=null){
            while(rs.next()){
                idSent = rs1.getInt(1);
                sent = rs1.getString(2).trim();
                //debug(idSent + "=" + sent);
                cbo.addItem(sent + "                                                                    >>>" +
                                                    idSent);
            }
        }
        else
             debug("rs1 fue null");

        rs1.close();
        cierraBD();
    }catch(Exception e){
        debug("Error en llenaComboSentido():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }

}

private void llenaComboTrafico(Choice cbo){
    try{
```

```
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_trafico");

        int idTraf = 0;
        String traf = "";
        cbo.removeAll();

        if(rs1!=null){
            while(rs.next()){
                idTraf = rs1.getInt(1);
                traf = rs1.getString(2).trim();

                cbo.addItem(traf + "                                                    >>>" +
                                                    idTraf);
            }
        }
        else
             debug("rs1 fue null");

        rs1.close();
        cierraBD();
    }catch(Exception e){
        debug("Error en llenaComboTrafico():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }

}


private void debug(String str){
    System.out.println(str);
}

public LocAppMtto(String title)
{
            this();
            setTitle(title);
}

static public void main(String args[])
{
            try
            {
                        //Create a new instance of our application's frame, and make it visible.
            (new LocAppMtto()).setVisible(true);
            }
            catch (Throwable t)
            {
                        System.err.println(t);
                        t.printStackTrace();
                        //Ensure the application exits with an error condition.
                        System.exit(1);
            }
}

public void addNotify()
{
            // Record the size of the window prior to calling parents addNotify.
            Dimension d = getSize();

            super.addNotify();

            if (fComponentsAdjusted)
                        return;
```

```
		// Adjust components according to the insets
		setSize(getInsets().left + getInsets().right + d.width, getInsets().top + getInsets().bottom + d.height);
		Component components[] = getComponents();
		for (int i = 0; i < components.length; i++)
		{
			Point p = components[i].getLocation();
			p.translate(getInsets().left, getInsets().top);
			components[i].setLocation(p);
		}
		fComponentsAdjusted = true;
	}

	// Used for addNotify check.
	boolean fComponentsAdjusted = false;

	//{{DECLARE_CONTROLS
	symantec.itools.awt.TabPanel tabPanel = new symantec.itools.awt.TabPanel();
	java.awt.Panel panel0 = new java.awt.Panel();
	symantec.itools.awt.MultiList ml_calle = new symantec.itools.awt.MultiList();
	symantec.itools.awt.BorderPanel bp_calle = new symantec.itools.awt.BorderPanel();
	java.awt.TextField txt_nombre_calle = new java.awt.TextField();
	java.awt.Label label1 = new java.awt.Label();
	java.awt.Panel panel1 = new java.awt.Panel();
	symantec.itools.awt.MultiList ml_colonia = new symantec.itools.awt.MultiList();
	symantec.itools.awt.BorderPanel bp_colonia = new symantec.itools.awt.BorderPanel();
	java.awt.TextField txt_colonia = new java.awt.TextField();
	java.awt.Label label2 = new java.awt.Label();
	java.awt.Choice cbo_p1_delegacion = new java.awt.Choice();
	java.awt.Label label3 = new java.awt.Label();
	java.awt.Panel panel2 = new java.awt.Panel();
	symantec.itools.awt.MultiList ml_delegacion = new symantec.itools.awt.MultiList();
	symantec.itools.awt.BorderPanel bp_delegacion = new symantec.itools.awt.BorderPanel();
	java.awt.TextField txt_delegacion = new java.awt.TextField();
	java.awt.Label label4 = new java.awt.Label();
	java.awt.Panel panel3 = new java.awt.Panel();
	symantec.itools.awt.MultiList ml_sitio = new symantec.itools.awt.MultiList();
	symantec.itools.awt.BorderPanel borderPanel1 = new symantec.itools.awt.BorderPanel();
	java.awt.TextField txt_sitio = new java.awt.TextField();
	java.awt.Label label5 = new java.awt.Label();
	java.awt.Choice cbo_p3_calle = new java.awt.Choice();
	java.awt.Label label6 = new java.awt.Label();
	java.awt.Label label7 = new java.awt.Label();
	java.awt.Choice cbo_p3_colonia = new java.awt.Choice();
	java.awt.Label label8 = new java.awt.Label();
	java.awt.Label label9 = new java.awt.Label();
	com.symantec.itools.awt.MaskedTextField txtm_telefono = new com.symantec.itools.awt.MaskedTextField();
	com.symantec.itools.awt.MaskedTextField txtm_numero = new com.symantec.itools.awt.MaskedTextField();
	java.awt.Label label20 = new java.awt.Label();
	java.awt.Choice cbo_p3_delegacion = new java.awt.Choice();
	java.awt.Label label21 = new java.awt.Label();
	java.awt.Choice cbo_p3_categoria = new java.awt.Choice();
	java.awt.Panel panel4 = new java.awt.Panel();
	symantec.itools.awt.MultiList ml_nodo = new symantec.itools.awt.MultiList();
	symantec.itools.awt.BorderPanel borderPanel2 = new symantec.itools.awt.BorderPanel();
	java.awt.Label label10 = new java.awt.Label();
	java.awt.TextField txt_nodo = new java.awt.TextField();
	java.awt.Panel panel5 = new java.awt.Panel();
	symantec.itools.awt.MultiList ml_cuadra = new symantec.itools.awt.MultiList();
	symantec.itools.awt.BorderPanel borderPanel3 = new symantec.itools.awt.BorderPanel();
	java.awt.Label label11 = new java.awt.Label();
	java.awt.Choice cbo_p5_calle = new java.awt.Choice();
	java.awt.Label label12 = new java.awt.Label();
	java.awt.Choice cbo_p5_nodo1 = new java.awt.Choice();
	java.awt.Label label13 = new java.awt.Label();
```

```
java.awt.Choice cbo_p5_flujo1 = new java.awt.Choice();
java.awt.Label label14 = new java.awt.Label();
java.awt.Choice cbo_p5_nodo2 = new java.awt.Choice();
java.awt.Label label15 = new java.awt.Label();
java.awt.Choice cbo_p5_flujo2 = new java.awt.Choice();
java.awt.Label label16 = new java.awt.Label();
java.awt.Choice cbo_p5_colonia = new java.awt.Choice();
java.awt.Label label17 = new java.awt.Label();
com.symantec.itools.awt.MaskedTextField txtm_noinic = new com.symantec.itools.awt.MaskedTextField();
java.awt.Label label18 = new java.awt.Label();
com.symantec.itools.awt.MaskedTextField txtm_nofinal = new com.symantec.itools.awt.MaskedTextField();
java.awt.Label label22 = new java.awt.Label();
java.awt.Choice cbo_p5_trafico = new java.awt.Choice();
java.awt.Label label23 = new java.awt.Label();
java.awt.Choice cbo_p5_sentido1 = new java.awt.Choice();
java.awt.Label label19 = new java.awt.Label();
java.awt.Choice cbo_p5_sentido2 = new java.awt.Choice();
java.awt.Button btn_insertar = new java.awt.Button();
java.awt.Button btn_modificar = new java.awt.Button();
java.awt.Button btn_eliminar = new java.awt.Button();
java.awt.Button btn_cancelar = new java.awt.Button();
java.awt.Button btn_aceptar = new java.awt.Button();
java.awt.Label label24 = new java.awt.Label();
//}}

//{{DECLARE_MENUS
//}}

class SymWindow extends java.awt.event.WindowAdapter
{
        public void windowClosing(java.awt.event.WindowEvent event)
        {
                Object object = event.getSource();
                if (object == LocAppMtto.this)
                        LocAppMtto_WindowClosing(event);
        }
}

void LocAppMtto_WindowClosing(java.awt.event.WindowEvent event)
{
        // to do: code goes here.

        LocAppMtto_WindowClosing_Interaction1(event);
}


void LocAppMtto_WindowClosing_Interaction1(java.awt.event.WindowEvent event)
{
        try {
                // QuitDialog Create and show as modal
                (new QuitDialog(this, true)).setVisible(true);
        } catch (Exception e) {
        }
}


class SymAction implements java.awt.event.ActionListener
{
        public void actionPerformed(java.awt.event.ActionEvent event)
        {
                Object object = event.getSource();
                if (object == btn_insertar)
                        btnInsertar_ActionPerformed(event);
                else if (object == btn_aceptar)
```

```
                        btnAceptar_ActionPerformed(event);
                    else if (object == btn_modificar)
                        btnModificar_ActionPerformed(event);
                    else if (object == btn_eliminar)
                        btnEliminar_ActionPerformed(event);
                    else if (object == btn_cancelar)
                        btnCancelar_ActionPerformed(event);

            }
}

private void cargaDatosPanel0(){
    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_calle");

        int row=0;
        int idCalle = 0;
        String calle = "";
        ml_calle.clear();

        if(rs1!=null){
            while(rs.next()){
                idCalle = rs1.getInt(1);
                calle = rs1.getString(2).trim();
                ml_calle.addTextCell(row, 0, String.valueOf(idCalle));
                ml_calle.addTextCell(row, 1, calle);

                row++;
            }
        }
        else
             debug("rs1 fue null");

        rs1.close();
        cierraBD();
    }catch(Exception e){
        debug("Error en cargaDatosPanel0():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }
}

private void cargaDatosPanel1(){
    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_colonia");

        int row=0;

        ml_colonia.clear();

        if(rs1!=null){
            while(rs.next()){
                ml_colonia.addTextCell(row, 0, String.valueOf(rs1.getInt(1)));
                ml_colonia.addTextCell(row, 1, rs1.getString(2).trim());
                ml_colonia.addTextCell(row, 2, String.valueOf(rs1.getInt(3)));
                ml_colonia.addTextCell(row, 3, rs1.getString(4).trim());

                row++;
            }
        }
        else
             debug("rs1 fue null");

        rs1.close();
        cierraBD();
```

```
    }catch(Exception e){
        debug("Error en cargaDatosPanel1():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }
}

private void cargaDatosPanel2(){
    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_delegacion");

        int row=0;
        int idDel = 0;
        String Deleg = "";
        ml_delegacion.clear();

        if(rs1!=null){
            while(rs.next()){
                idDel = rs1.getInt(1);
                Deleg = rs1.getString(2).trim();
                ml_delegacion.addTextCell(row, 0, String.valueOf(idDel));
                ml_delegacion.addTextCell(row, 1, Deleg);

                row++;
            }
        }
        else
             debug("rs1 fue null");

        rs1.close();
        cierraBD();
    }catch(Exception e){
        debug("Error en cargaDatosPanel2():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }
}

private void cargaDatosPanel3(){
    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio");

        int row=0;
        ml_sitio.clear();

        if(rs1!=null){
            while(rs.next()){
                ml_sitio.addTextCell(row, 0, String.valueOf(rs1.getInt(1)));
                ml_sitio.addTextCell(row, 1, rs1.getString(2).trim());
                ml_sitio.addTextCell(row, 2, String.valueOf(rs1.getInt(3)));
                ml_sitio.addTextCell(row, 3, rs1.getString(4).trim());
                ml_sitio.addTextCell(row, 4, String.valueOf(rs1.getInt(5)));
                ml_sitio.addTextCell(row, 5, String.valueOf(rs1.getInt(6)));
                ml_sitio.addTextCell(row, 6, rs1.getString(7).trim());
                ml_sitio.addTextCell(row, 7, String.valueOf(rs1.getInt(8)));
                ml_sitio.addTextCell(row, 8, String.valueOf(rs1.getInt(9)));
                ml_sitio.addTextCell(row, 9, rs1.getString(10).trim());
                ml_sitio.addTextCell(row, 10, String.valueOf(rs1.getInt(11)));
                ml_sitio.addTextCell(row, 11, rs1.getString(12).trim());
                row++;
            }
        }
        else
             debug("rs1 fue null");

        rs1.close();
```

```
            cierraBD();
        }catch(Exception e){
            debug("Error en cargaDatosPanel3():" + e.toString() + "_Msg=" + e.getMessage());
            e.printStackTrace(System.out);
        }
    }

    private void cargaDatosPanel4(){
        try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_nodo");

            int row=0;
            ml_nodo.clear();

            if(rs1!=null){
                while(rs.next()){
                    ml_nodo.addTextCell(row, 0, String.valueOf(rs1.getInt(1)));
                    ml_nodo.addTextCell(row, 1, rs1.getString(2).trim());

                    row++;
                }
            }
            else
                debug("rs1 fue null");

rs1.close();
            cierraBD();
        }catch(Exception e){
            debug("Error en cargaDatosPanel4():" + e.toString() + "_Msg=" + e.getMessage());
            e.printStackTrace(System.out);
        }
    }

    private void cargaDatosPanel5(){
        try{
        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_cuadra");

            int row=0;
            ml_cuadra.clear();

            if(rs1!=null){
                while(rs.next()){
                    ml_cuadra.addTextCell(row, 0, String.valueOf(rs1.getInt(1)));
                    ml_cuadra.addTextCell(row, 1, rs1.getString(2).trim());
                    ml_cuadra.addTextCell(row, 2, String.valueOf(rs1.getInt(3)));
                    ml_cuadra.addTextCell(row, 3, rs1.getString(4).trim());
                    ml_cuadra.addTextCell(row, 4, String.valueOf(rs1.getInt(5)));
                    ml_cuadra.addTextCell(row, 5, String.valueOf(rs1.getInt(6)));
                    ml_cuadra.addTextCell(row, 6, rs1.getString(7).trim());
                    ml_cuadra.addTextCell(row, 7, String.valueOf(rs1.getInt(8)));
                    ml_cuadra.addTextCell(row, 8, String.valueOf(rs1.getInt(9)));
                    ml_cuadra.addTextCell(row, 9, rs1.getString(10).trim());
                    ml_cuadra.addTextCell(row, 10, String.valueOf(rs1.getInt(11)));
                    ml_cuadra.addTextCell(row, 11, String.valueOf(rs1.getInt(12)));
                    ml_cuadra.addTextCell(row, 12, String.valueOf(rs1.getInt(13)));
                    ml_cuadra.addTextCell(row, 13, rs1.getString(14).trim());
                    ml_cuadra.addTextCell(row, 14, String.valueOf(rs1.getInt(15)));
                    ml_cuadra.addTextCell(row, 15, rs1.getString(16).trim());
                    ml_cuadra.addTextCell(row, 16, String.valueOf(rs1.getInt(17)));
                    ml_cuadra.addTextCell(row, 17, rs1.getString(18).trim());
                    row++;
                }
            }
            else
```

```
                debug("rs1 fue null");

            rs1.close();
            cierraBD();
        }catch(Exception e){
            debug("Error en cargaDatosPanel5():" + e.toString() + "_Msg=" + e.getMessage());
            e.printStackTrace(System.out);
        }
    }

    void btnInsertar_ActionPerformed(java.awt.event.ActionEvent event)
    {
        // to do: code goes here.
        switch(tabPanel.getCurrentPanelNdx())
        {
            case 0: insertarP0(); break;
            case 1: insertarP1(); break;
            case 2: insertarP2(); break;
            case 3: insertarP3(); break;
            case 4: insertarP4(); break;
            case 5: insertarP5(); break;
        }
    }

    private void insertarP0(){
        if(operacion == NINGUNA){
            operacion = INSERTAR;
            habilitaCtrlsP0(false);
            txt_nombre_calle.requestFocus();
        }
        else if(operacion == INSERTAR){
            if(txt_nombre_calle.getText().trim().compareTo("")!=0){
                int fm = ejecutaUpdate("EXEC localizacion..loc_spi_calle '" +
txt_nombre_calle.getText().trim() + "'");
                if(fm == -1)
                    debug("Error al ejecutar localizacion..loc_spi_calle");
                else if(fm == 0)
                    debug("No se insertó registro con localizacion..loc_spi_calle");
                else if(fm == 1){
                    debug("Registro insertado con localizacion..loc_spi_calle");
                    llenaControles();
                }
            }
            else{
                debug("Mandar mensaje Tecle un nombre de calle");
            }

            operacion = NINGUNA;
            habilitaCtrlsP0(true);
            txt_nombre_calle.setText("");
            cargaDatosPanel0();
        }
    }

    private void insertarP1(){
        if(operacion == NINGUNA){
            operacion = INSERTAR;
            habilitaCtrlsP1(false);
            txt_colonia.requestFocus();
        }
        else if(operacion == INSERTAR){
            if(txt_colonia.getText().trim().compareTo("")!=0){
                String delSelected = cbo_p1_delegacion.getSelectedItem();
```

```
                                        delSelected = delSelected.substring(delSelected.indexOf(">>>") + 3,
delSelected.length());
                                        //debug("delegSele="+delSelected);

                                        int fm = ejecutaUpdate("EXEC localizacion..loc_spi_colonia '" +
txt_colonia.getText().trim() + "', " + delSelected.trim());
                                        if(fm == -1)
                                                debug("Error al ejecutar localizacion..loc_spi_colonia");
                                        else if(fm == 0)
                                                debug("No se insertó registro con localizacion..loc_spi_colonia");
                                        else if(fm == 1){
                                                debug("Registro insertado con localizacion..loc_spi_colonia");
                                                llenaControles();
                                        }

                                }
                                else{
                                        debug("Mandar mensaje Tecle un nombre de colonia");
                                }

                                operacion = NINGUNA;
                                habilitaCtrlsP1(true);
                                txt_colonia.setText("");
                                cargaDatosPanel1();
                        }
                }

                private void insertarP2(){
                        if(operacion == NINGUNA){
                                operacion = INSERTAR;
                                habilitaCtrlsP2(false);
                                txt_delegacion.requestFocus();
                        }
                        else if(operacion == INSERTAR){
                                if(txt_delegacion.getText().trim().compareTo("")!=0){
                                        int fm = ejecutaUpdate("EXEC localizacion..loc_spi_delegacion '" +
txt_delegacion.getText().trim() + "'");
                                        if(fm == -1)
                                                debug("Error al ejecutar localizacion..loc_spi_delegacion");
                                        else if(fm == 0)
                                                debug("No se insertó registro con localizacion..loc_spi_delegacion");
                                        else if(fm == 1){
                                                debug("Registro insertado con localizacion..loc_spi_delegacion");
                                                llenaControles();
                                        }
                                }
                                else{
                                        debug("Mandar mensaje Tecle un nombre de Delegacion");
                                }

                                operacion = NINGUNA;
                                habilitaCtrlsP2(true);
                                txt_delegacion.setText("");
                                cargaDatosPanel2();
                        }
                }

                private void insertarP3(){
                        if(operacion == NINGUNA){
                                operacion = INSERTAR;
                                habilitaCtrlsP3(false);
                                txt_sitio.requestFocus();
                        }
                        else if(operacion == INSERTAR){
```

```
		if((txt_sitio.getText().trim().compareTo("")!=0)||(txtm_numero.getText().trim().compareTo("")!=0)){
					String calleSelected = cbo_p3_calle.getSelectedItem();
					calleSelected = calleSelected.substring(calleSelected.indexOf(">>>") + 3,
calleSelected.length());
					String colSelected = cbo_p3_colonia.getSelectedItem();
					colSelected = colSelected.substring(colSelected.indexOf(">>>") + 3,
colSelected.length());
					String delSelected = cbo_p3_delegacion.getSelectedItem();
					delSelected = delSelected.substring(delSelected.indexOf(">>>") + 3,
delSelected.length());
					String catSelected = cbo_p3_categoria.getSelectedItem();
					catSelected = catSelected.substring(catSelected.indexOf(">>>") + 3,
catSelected.length());
					//EXEC localizacion..loc_spi_sitio 'VIPS', 1, 140, 1, 57425485
					int fm = ejecutaUpdate("EXEC localizacion..loc_spi_sitio '" +
txt_sitio.getText().trim() + "'," + calleSelected.trim() + "," + txtm_numero.getText().trim() + "," + colSelected.trim() + ","
+ txtm_telefono.getText().trim() + "," + delSelected.trim() + "," + catSelected.trim());
					if(fm == -1)
						debug("Error al ejecutar localizacion..loc_spi_sitio");
					else if(fm == 0)
						debug("No se insertó registro con localizacion..loc_spi_sitio");
					else if(fm == 1){
						debug("Registro insertado con localizacion..loc_spi_sitio");
						llenaControles();
					}

				}
				else{
					debug("Mandar mensaje Tecle un nombre del sitio y/o el numero de la calle");
				}

				operacion = NINGUNA;
				habilitaCtrlsP3(true);
				txt_sitio.setText("");
				txtm_numero.setText("");
				txtm_telefono.setText("");
				cargaDatosPanel3();
			}
		}

		private void insertarP4(){
			if(operacion == NINGUNA){
				operacion = INSERTAR;
				habilitaCtrlsP4(false);
				txt_nodo.requestFocus();
			}
			else if(operacion == INSERTAR){
				if(txt_nodo.getText().trim().compareTo("")!=0){
					int fm = ejecutaUpdate("EXEC localizacion..loc_spi_nodo '" +
txt_nodo.getText().trim() + "'");
					if(fm == -1)
						debug("Error al ejecutar localizacion..loc_spi_nodo");
					else if(fm == 0)
						debug("No se insertó registro con localizacion..loc_spi_nodo");
					else if(fm == 1){
						debug("Registro insertado con localizacion..loc_spi_nodo");
						llenaControles();
					}
				}
				else{
					debug("Mandar mensaje Tecle un nombre del Nodo");
				}
```

```
                        operacion = NINGUNA;
                        habilitaCtrlsP4(true);
                        txt_nodo.setText("");
                        cargaDatosPanel4();
                }
        }

        private void insertarP5(){
                if(operacion == NINGUNA){
                        operacion = INSERTAR;
                        habilitaCtrlsP5(false);
                        cbo_p5_calle.requestFocus();
                }
                else if(operacion == INSERTAR){

        if((txtm_noinic.getText().trim().compareTo("")!=0)||(txtm_nofinal.getText().trim().compareTo("")!=0)){
                                String calleSelected = cbo_p5_calle.getSelectedItem();
                                calleSelected = calleSelected.substring(calleSelected.indexOf(">>>") + 3,
calleSelected.length());
                                String colSelected = cbo_p5_colonia.getSelectedItem();
                                colSelected = colSelected.substring(colSelected.indexOf(">>>") + 3,
colSelected.length());
                                String nodo1Selected = cbo_p5_nodo1.getSelectedItem();
                                nodo1Selected = nodo1Selected.substring(nodo1Selected.indexOf(">>>") + 3,
nodo1Selected.length());
                                String nodo2Selected = cbo_p5_nodo2.getSelectedItem();
                                nodo2Selected = nodo2Selected.substring(nodo2Selected.indexOf(">>>") + 3,
nodo2Selected.length());

                                String sent1Selected = cbo_p5_sentido1.getSelectedItem();
                                sent1Selected = sent1Selected.substring(sent1Selected.indexOf(">>>") + 3,
sent1Selected.length());
                                String sent2Selected = cbo_p5_sentido2.getSelectedItem();
                                sent2Selected = sent2Selected.substring(sent2Selected.indexOf(">>>") + 3,
sent2Selected.length());
                                String trafSelected = cbo_p5_trafico.getSelectedItem();
                                trafSelected = trafSelected.substring(trafSelected.indexOf(">>>") + 3,
trafSelected.length());


                                /*
                                                @idCalle  INT,
                                                @idNodo1        INT,
                                                @idNodo2        INT,
                                                @flujo1         INT,
                                                @flujo2         INT,
                                                @noInic         INT,
                                                @noFinal INT,
                                                @idColonia      INT,
                                                @longitud       INT
                                */
                                //EXEC localizacion..loc_spi_cuadra 19,2,13,10,10,100,200,11,1488
                                int fm = ejecutaUpdate("EXEC localizacion..loc_spi_cuadra " + calleSelected.trim() +
","
                                                + nodo1Selected.trim() + "," + nodo2Selected.trim()
+ "," + cbo_p5_flujo1.getSelectedItem() + ","
                                                +  cbo_p5_flujo2.getSelectedItem()  +  ","  +
txtm_noinic.getText().trim() + ","
                                                +  txtm_nofinal.getText().trim()  +  ","  +
colSelected.trim() + ","
                                                + sent1Selected.trim() + "," + sent2Selected.trim() +
"," + trafSelected.trim());
                                if(fm == -1)
                                        debug("Error al ejecutar localizacion..loc_spi_cuadra");
```

```
                                        else if(fm == 0)
                                                debug("No se insertó registro con localizacion..loc_spi_cuadra");
                                        else if(fm == 1){
                                                debug("Registro insertado con localizacion..loc_spi_cuadra");
                                        }

                                }
                                else{
                                        debug("Mandar mensaje Tecle el numero inicial y final de la calle, asi como la
longitud de la misma");
                                }

                                operacion = NINGUNA;
                                habilitaCtrlsP5(true);
                                txtm_noinic.setText("");
                                txtm_nofinal.setText("");
                                //txtm_longitud.setText("");
                                cargaDatosPanel5();
                        }
                }

                private void habilitaCtrlsP0(boolean hab){
                        habilitaBtnsOper(hab);
                        txt_nombre_calle.setEnabled(!hab);
                        ml_calle.setEnabled(hab);
                        habilitaPaneles(hab);
                }
                private void habilitaCtrlsP1(boolean hab){
                        habilitaBtnsOper(hab);
                        txt_colonia.setEnabled(!hab);
                        cbo_p1_delegacion.setEnabled(!hab);
                        ml_colonia.setEnabled(hab);
                        habilitaPaneles(hab);
                }
                private void habilitaCtrlsP2(boolean hab){
                        habilitaBtnsOper(hab);
                        txt_delegacion.setEnabled(!hab);
                        ml_delegacion.setEnabled(hab);
                        habilitaPaneles(hab);
                }
                private void habilitaCtrlsP3(boolean hab){
                        habilitaBtnsOper(hab);
                        txt_sitio.setEnabled(!hab);
                        cbo_p3_calle.setEnabled(!hab);
                        cbo_p3_colonia.setEnabled(!hab);
                        cbo_p3_delegacion.setEnabled(!hab);
                        cbo_p3_categoria.setEnabled(!hab);
                        txtm_numero.setEnabled(!hab);
                        txtm_telefono.setEnabled(!hab);
                        ml_sitio.setEnabled(hab);
                        habilitaPaneles(hab);
                }
                private void habilitaCtrlsP4(boolean hab){
                        habilitaBtnsOper(hab);
                        txt_nodo.setEnabled(!hab);
                        ml_nodo.setEnabled(hab);
                        habilitaPaneles(hab);
                }

                private void habilitaCtrlsP5(boolean hab){
                        habilitaBtnsOper(hab);
                        txt_sitio.setEnabled(!hab);

                        if(operacion == INSERTAR || operacion == NINGUNA){
```

```
                cbo_p5_calle.setEnabled(!hab);
                cbo_p5_colonia.setEnabled(!hab);
                cbo_p5_nodo1.setEnabled(!hab);
                cbo_p5_nodo2.setEnabled(!hab);
        }
        cbo_p5_flujo1.setEnabled(!hab);
        cbo_p5_flujo2.setEnabled(!hab);

        txtm_noinic.setEnabled(!hab);
        txtm_nofinal.setEnabled(!hab);
        cbo_p5_sentido1.setEnabled(!hab);
        cbo_p5_sentido2.setEnabled(!hab);
        cbo_p5_trafico.setEnabled(!hab);
        ml_cuadra.setEnabled(hab);
        habilitaPaneles(hab);
}

private void habilitaPaneles(boolean opc){
        try{
                for(int i=0; i<tabPanel.countTabs(); i++)
                        tabPanel.enableTabPanel(opc, i);
        }catch(Exception e){
                debug("habilitaPaneles().-" + e.toString());
        }
}

private void habilitaBtnsOper(boolean hab){
        btn_insertar.setEnabled(hab);
        btn_modificar.setEnabled(hab);
        btn_eliminar.setEnabled(hab);

        btn_aceptar.setEnabled(!hab);
        btn_cancelar.setEnabled(!hab);
}

void btnAceptar_ActionPerformed(java.awt.event.ActionEvent event)
{
        // to do: code goes here.
        switch(tabPanel.getCurrentPanelNdx())
        {
                case 0: //calle
                        switch(operacion)
                        {
                                case INSERTAR: insertarP0(); break;
                                case MODIFICAR: modificarP0(); break;
                        }
                        break;
                case 1: //colonia
                        switch(operacion)
                        {
                                case INSERTAR: insertarP1(); break;
                                case MODIFICAR: modificarP1(); break;
                        }
                        break;
                case 2: //delegacion
                        switch(operacion)
                        {
                                case INSERTAR: insertarP2(); break;
                                case MODIFICAR: modificarP2(); break;
                        }
                        break;
                case 3: //sitio
                        switch(operacion)
                        {
```

```
                                        case INSERTAR: insertarP3(); break;
                                        case MODIFICAR: modificarP3(); break;
                                }
                                break;
                        case 4: //nodo
                                switch(operacion)
                                {
                                        case INSERTAR: insertarP4(); break;
                                        case MODIFICAR: modificarP4(); break;
                                }
                                break;
                        case 5: //cuadra
                                switch(operacion)
                                {
                                        case INSERTAR: insertarP5(); break;
                                        case MODIFICAR: modificarP5(); break;
                                }
                                break;
                }
        }


        void btnModificar_ActionPerformed(java.awt.event.ActionEvent event)
        {
                // to do: code goes here.
                switch(tabPanel.getCurrentPanelNdx())
                {
                        case 0: modificarP0(); break;
                        case 1: modificarP1(); break;
                        case 2: modificarP2(); break;
                        case 3: modificarP3(); break;
                        case 4: modificarP4(); break;
                        case 5: modificarP5(); break;
                }
        }

        private void modificarP0(){
                if(operacion == NINGUNA){
                        if(ml_calle.getSelectedRow() != -1){
                                operacion = MODIFICAR;
                                habilitaCtrlsP0(false);
                                txt_nombre_calle.setText(ml_calle.getCellText(ml_calle.getSelectedRow(), 1));
                                txt_nombre_calle.requestFocus();
                        }
                        else{
                                debug("Mandar mensaje de posicionarse en un registro");
                        }
                }
                else if(operacion == MODIFICAR){
                        if(txt_nombre_calle.getText().trim().compareTo("")!=0){
                                String idSelec = ml_calle.getCellText(ml_calle.getSelectedRow(), 0).trim();

                                int fm = ejecutaUpdate("EXEC localizacion..loc_spu_calle " + idSelec + ",'" +
txt_nombre_calle.getText().trim() + "'");
                                if(fm == -1)
                                        debug("Error al ejecutar localizacion..loc_spu_calle");
                                else if(fm == 0)
                                        debug("No se modificó registro con localizacion..loc_spu_calle");
                                else if(fm == 1){
                                        debug("Registro modificado con localizacion..loc_spu_calle");
                                        llenaControles();
                                }
                        }
                        else{
```

```
                        debug("Mandar mensaje Tecle un nombre de calle");
                    }
                    operacion = NINGUNA;
                    habilitaCtrlsP0(true);
                    txt_nombre_calle.setText("");
                    cargaDatosPanel0();
                }
            }

            private void modificarP1(){
                if(operacion == NINGUNA){
                    if(ml_colonia.getSelectedRow() != -1){
                        operacion = MODIFICAR;
                        habilitaCtrlsP1(false);
                        txt_colonia.setText(ml_colonia.getCellText(ml_colonia.getSelectedRow(), 1));

                        int idDelSelec =
Integer.parseInt(ml_colonia.getCellText(ml_colonia.getSelectedRow(), 2).trim());
                        String deleg = "";
                        int idDeleg = 0;
                        for(int i=0; i<cbo_p1_delegacion.getItemCount(); i++){
                            deleg = cbo_p1_delegacion.getItem(i);
                            idDeleg = Integer.parseInt(deleg.substring(deleg.indexOf(">>>") + 3,
deleg.length()));
                            if(idDelSelec == idDeleg){
                                cbo_p1_delegacion.select(i);
                                break;
                            }
                        }
                        txt_colonia.requestFocus();
                    }
                    else{
                        debug("Mandar mensaje de posicionarse en un registro");
                    }
                }
                else if(operacion == MODIFICAR){
                    if(txt_colonia.getText().trim().compareTo("")!=0){
                        String idSelec = ml_colonia.getCellText(ml_colonia.getSelectedRow(), 0).trim();
                        String delSelected = cbo_p1_delegacion.getSelectedItem();
                        delSelected = delSelected.substring(delSelected.indexOf(">>>") + 3,
delSelected.length());
                        //debug("idSelec=" + idSelec + "__delegSele="+delSelected);

                        int fm = ejecutaUpdate("EXEC localizacion..loc_spu_colonia " + idSelec + ",'" +
txt_colonia.getText().trim() + "'," + delSelected.trim());
                        if(fm == -1)
                            debug("Error al ejecutar localizacion..loc_spu_colonia");
                        else if(fm == 0)
                            debug("No se modificó registro con localizacion..loc_spu_colonia");
                        else if(fm == 1){
                            debug("Registro modificado con localizacion..loc_spu_colonia");
                            llenaControles();
                        }

                    }
                    else{
                        debug("Mandar mensaje Tecle un nombre de calle");
                    }
                    operacion = NINGUNA;
                    habilitaCtrlsP1(true);
                    txt_colonia.setText("");
                    cargaDatosPanel1();
                }
            }
```

```
	private void modificarP2(){
		if(operacion == NINGUNA){
			if(ml_delegacion.getSelectedRow() != -1){
				operacion = MODIFICAR;
				habilitaCtrlsP2(false);
				txt_delegacion.setText(ml_delegacion.getCellText(ml_delegacion.getSelectedRow(),
1));
				txt_delegacion.requestFocus();
			}
			else{
				debug("Mandar mensaje de posicionarse en un registro");
			}
		}
		else if(operacion == MODIFICAR){
			if(txt_delegacion.getText().trim().compareTo("")!=0){
				String idSelec = ml_delegacion.getCellText(ml_delegacion.getSelectedRow(),
0).trim();

				int fm = ejecutaUpdate("EXEC localizacion..loc_spu_delegacion " + idSelec + ",'" +
txt_delegacion.getText().trim() + "'");
				if(fm == -1)
					debug("Error al ejecutar localizacion..loc_spu_delegacion");
				else if(fm == 0)
					debug("No se modificó registro con localizacion..loc_spu_delegacion");
				else if(fm == 1){
					debug("Registro modificado con localizacion..loc_spu_delegacion");
					llenaControles();
				}
			}
			else{
				debug("Mandar mensaje Tecle un nombre de delegacion");
			}
			operacion = NINGUNA;
			habilitaCtrlsP2(true);
			txt_delegacion.setText("");
			cargaDatosPanel2();
		}
	}

	private void modificarP3(){
		if(operacion == NINGUNA){
			if(ml_sitio.getSelectedRow() != -1){
				operacion = MODIFICAR;
				habilitaCtrlsP3(false);
				txt_sitio.setText(ml_sitio.getCellText(ml_sitio.getSelectedRow(), 1));
				txtm_numero.setText(ml_sitio.getCellText(ml_sitio.getSelectedRow(), 4));
				txtm_telefono.setText(ml_sitio.getCellText(ml_sitio.getSelectedRow(), 7));

				int idCalleSelec = Integer.parseInt(ml_sitio.getCellText(ml_sitio.getSelectedRow(),
2).trim());
				String calle = "";
				int idCalle = 0;
				for(int i=0; i<cbo_p3_calle.getItemCount(); i++){
					calle = cbo_p3_calle.getItem(i);
					idCalle = Integer.parseInt(calle.substring(calle.indexOf(">>>") + 3,
calle.length()));
					if(idCalleSelec == idCalle){
						cbo_p3_calle.select(i);
						break;
					}
				}
```

```
                                int idColSelec = Integer.parseInt(ml_sitio.getCellText(ml_sitio.getSelectedRow(),
5).trim());
                                String colonia = "";
                                int idCol = 0;
                                for(int i=0; i<cbo_p3_colonia.getItemCount(); i++){
                                        colonia = cbo_p3_colonia.getItem(i);
                                        idCol = Integer.parseInt(colonia.substring(colonia.indexOf(">>>") + 3,
colonia.length()));
                                        if(idColSelec == idCol){
                                                cbo_p3_colonia.select(i);
                                                break;
                                        }
                                }

                                int idDelSelec = Integer.parseInt(ml_sitio.getCellText(ml_sitio.getSelectedRow(),
8).trim());
                                String delegacion = "";
                                int idDel = 0;
                                for(int i=0; i<cbo_p3_delegacion.getItemCount(); i++){
                                        delegacion = cbo_p3_delegacion.getItem(i);
                                        idDel = Integer.parseInt(delegacion.substring(delegacion.indexOf(">>>")
+ 3, delegacion.length()));
                                        if(idDelSelec == idDel){
                                                cbo_p3_delegacion.select(i);
                                                break;
                                        }
                                }

                                int idCatSelec = Integer.parseInt(ml_sitio.getCellText(ml_sitio.getSelectedRow(),
10).trim());
                                String categoria = "";
                                int idCat = 0;
                                for(int i=0; i<cbo_p3_categoria.getItemCount(); i++){
                                        categoria = cbo_p3_categoria.getItem(i);
                                        idCat = Integer.parseInt(categoria.substring(categoria.indexOf(">>>") +
3, categoria.length()));
                                        if(idCatSelec == idCat){
                                                cbo_p3_categoria.select(i);
                                                break;
                                        }
                                }

                                txt_sitio.requestFocus();
                        }
                        else{
                                debug("Mandar mensaje de posicionarse en un registro");
                        }
                }
                else if(operacion == MODIFICAR){

        if((txt_sitio.getText().trim().compareTo("")!=0)||(txtm_numero.getText().trim().compareTo("")!=0)){
                                String idSelec = ml_sitio.getCellText(ml_sitio.getSelectedRow(), 0).trim();
                                String calleSelected = cbo_p3_calle.getSelectedItem();
                                calleSelected = calleSelected.substring(calleSelected.indexOf(">>>") + 3,
calleSelected.length());
                                String colSelected = cbo_p3_colonia.getSelectedItem();
                                colSelected = colSelected.substring(colSelected.indexOf(">>>") + 3,
colSelected.length());
                                String delSelected = cbo_p3_delegacion.getSelectedItem();
                                delSelected = delSelected.substring(delSelected.indexOf(">>>") + 3,
delSelected.length());
                                String catSelected = cbo_p3_categoria.getSelectedItem();
                                catSelected = catSelected.substring(catSelected.indexOf(">>>") + 3,
catSelected.length());
```

```
                                        //EXEC localizacion..loc_spu_sitio 1, 'VIPS', 1, 140, 1, 57425485
                                        int fm = ejecutaUpdate("EXEC localizacion..loc_spu_sitio " + idSelec + ",'" +
txt_sitio.getText().trim() + "'," + calleSelected.trim() + "," + txtm_numero.getText().trim() + "," + colSelected.trim() + ","
+ txtm_telefono.getText().trim() + "," + delSelected.trim() + "," + catSelected.trim());
                                        if(fm == -1)
                                                debug("Error al ejecutar localizacion..loc_spu_sitio");
                                        else if(fm == 0)
                                                debug("No se modificó registro con localizacion..loc_spu_sitio");
                                        else if(fm == 1){
                                                debug("Registro modificado con localizacion..loc_spu_sitio");
                                                llenaControles();
                                        }

                                }
                                else{
                                        debug("Mandar mensaje Tecle el nombre del sitio y el numero de la calle");
                                }
                                operacion = NINGUNA;
                                habilitaCtrlsP3(true);
                                txt_sitio.setText("");
                                txtm_numero.setText("");
                                txtm_telefono.setText("");
                                cargaDatosPanel3();
                        }
                }

                private void modificarP4(){
                        if(operacion == NINGUNA){
                                if(ml_nodo.getSelectedRow() != -1){
                                        operacion = MODIFICAR;
                                        habilitaCtrlsP4(false);
                                        txt_nodo.setText(ml_nodo.getCellText(ml_nodo.getSelectedRow(), 1));
                                        txt_nodo.requestFocus();
                                }
                                else{
                                        debug("Mandar mensaje de posicionarse en un registro");
                                }
                        }
                        else if(operacion == MODIFICAR){
                                if(txt_nodo.getText().trim().compareTo("")!=0){
                                        String idSelec = ml_nodo.getCellText(ml_nodo.getSelectedRow(), 0).trim();

                                        int fm = ejecutaUpdate("EXEC localizacion..loc_spu_nodo " + idSelec + ",'" +
txt_nodo.getText().trim() + "'");
                                        if(fm == -1)
                                                debug("Error al ejecutar localizacion..loc_spu_nodo");
                                        else if(fm == 0)
                                                debug("No se modificó registro con localizacion..loc_spu_nodo");
                                        else if(fm == 1){
                                                debug("Registro modificado con localizacion..loc_spu_nodo");
                                                llenaControles();
                                        }
                                }
                                else{
                                        debug("Mandar mensaje Tecle un nombre de nodo");
                                }
                                operacion = NINGUNA;
                                habilitaCtrlsP4(true);
                                txt_nodo.setText("");
                                cargaDatosPanel4();
                        }
                }
```

```
	private void modificarP5(){
		if(operacion == NINGUNA){
			if(ml_cuadra.getSelectedRow() != -1){
				operacion = MODIFICAR;
				habilitaCtrlsP5(false);

				txtm_noinic.setText(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 10));
				txtm_nofinal.setText(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 11));

				int idCalleSelec =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 0).trim());
				String calle = "";
				int idCalle = 0;
				for(int i=0; i<cbo_p5_calle.getItemCount(); i++){
					calle = cbo_p5_calle.getItem(i);
					idCalle = Integer.parseInt(calle.substring(calle.indexOf(">>>") + 3,
calle.length()));
					if(idCalleSelec == idCalle){
						cbo_p5_calle.select(i);
						break;
					}
				}
				int idN1Selec =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 2).trim());
				String nodo1 = "";
				int idN1 = 0;
				for(int i=0; i<cbo_p5_nodo1.getItemCount(); i++){
					nodo1 = cbo_p5_nodo1.getItem(i);
					idN1 = Integer.parseInt(nodo1.substring(nodo1.indexOf(">>>") + 3,
nodo1.length()));
					if(idN1Selec == idN1){
						cbo_p5_nodo1.select(i);
						break;
					}
				}
				int idN2Selec =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 5).trim());
				String nodo2 = "";
				int idN2 = 0;
				for(int i=0; i<cbo_p5_nodo2.getItemCount(); i++){
					nodo2 = cbo_p5_nodo2.getItem(i);
					idN2 = Integer.parseInt(nodo2.substring(nodo2.indexOf(">>>") + 3,
nodo2.length()));
					if(idN2Selec == idN2){
						cbo_p5_nodo2.select(i);
						break;
					}
				}
				int idColSelec =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 8).trim());
				String colonia = "";
				int idCol = 0;
				for(int i=0; i<cbo_p5_colonia.getItemCount(); i++){
					colonia = cbo_p5_colonia.getItem(i);
					idCol = Integer.parseInt(colonia.substring(colonia.indexOf(">>>") + 3,
colonia.length()));
					if(idColSelec == idCol){
						cbo_p5_colonia.select(i);
						break;
					}
				}
				int idF1Selec =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 4).trim());
				int idF1 = 0;
```

```
                                        for(int i=0; i<cbo_p5_flujo1.getItemCount(); i++){
                                                idF1 = Integer.parseInt(cbo_p5_flujo1.getItem(i));
                                                if(idF1Selec == idF1){
                                                        cbo_p5_flujo1.select(i);
                                                        break;
                                                }
                                        }
                                        int                                     idF2Selec                                       =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 7).trim());
                                        int idF2 = 0;
                                        for(int i=0; i<cbo_p5_flujo2.getItemCount(); i++){
                                                idF2 = Integer.parseInt(cbo_p5_flujo2.getItem(i));
                                                if(idF2Selec == idF2){
                                                        cbo_p5_flujo2.select(i);
                                                        break;
                                                }
                                        }
                                        int                                     idS1Selec                                       =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 12).trim());
                                        String sent1 = "";
                                        int idS1 = 0;
                                        for(int i=0; i<cbo_p5_sentido1.getItemCount(); i++){
                                                sent1 = cbo_p5_sentido1.getItem(i);
                                                idS1    =    Integer.parseInt(sent1.substring(sent1.indexOf(">>>")    +    3,
sent1.length()));
                                                if(idS1Selec == idS1){
                                                        cbo_p5_sentido1.select(i);
                                                        break;
                                                }
                                        }
                                        int                                     idS2Selec                                       =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 14).trim());
                                        String sent2 = "";
                                        int idS2 = 0;
                                        for(int i=0; i<cbo_p5_sentido2.getItemCount(); i++){
                                                sent2 = cbo_p5_sentido2.getItem(i);
                                                idS2    =    Integer.parseInt(sent2.substring(sent2.indexOf(">>>")    +    3,
sent2.length()));
                                                if(idS2Selec == idS2){
                                                        cbo_p5_sentido2.select(i);
                                                        break;
                                                }
                                        }
                                        int                                     idTrafSelec                                     =
Integer.parseInt(ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 16).trim());
                                        String traf = "";
                                        int idtraf = 0;
                                        for(int i=0; i<cbo_p5_trafico.getItemCount(); i++){
                                                traf = cbo_p5_trafico.getItem(i);
                                                idtraf    =    Integer.parseInt(traf.substring(traf.indexOf(">>>")    +    3,
traf.length()));
                                                if(idTrafSelec == idtraf){
                                                        cbo_p5_trafico.select(i);
                                                        break;
                                                }
                                        }

                                        cbo_p5_trafico.requestFocus();
                                }
                                else{
                                        debug("Mandar mensaje de posicionarse en un registro");
                                }
                        }
                        else if(operacion == MODIFICAR){
```

```
		if((txtm_noinic.getText().trim().compareTo("")!=0)||(txtm_nofinal.getText().trim().compareTo("")!=0)){
					String calleSelected = cbo_p5_calle.getSelectedItem();
					calleSelected = calleSelected.substring(calleSelected.indexOf(">>>") + 3,
calleSelected.length());
					String n1Selected = cbo_p5_nodo1.getSelectedItem();
					n1Selected = n1Selected.substring(n1Selected.indexOf(">>>") + 3,
n1Selected.length());
					String n2Selected = cbo_p5_nodo2.getSelectedItem();
					n2Selected = n2Selected.substring(n2Selected.indexOf(">>>") + 3,
n2Selected.length());
					String colSelected = cbo_p5_colonia.getSelectedItem();
					colSelected = colSelected.substring(colSelected.indexOf(">>>") + 3,
colSelected.length());
					String f1Selected = cbo_p5_flujo1.getSelectedItem();
					String f2Selected = cbo_p5_flujo2.getSelectedItem();
					String s1Selected = cbo_p5_sentido1.getSelectedItem();
					s1Selected = s1Selected.substring(s1Selected.indexOf(">>>") + 3,
s1Selected.length());
					String s2Selected = cbo_p5_sentido2.getSelectedItem();
					s2Selected = s2Selected.substring(s2Selected.indexOf(">>>") + 3,
s2Selected.length());
					String trafSelected = cbo_p5_trafico.getSelectedItem();
					trafSelected = trafSelected.substring(trafSelected.indexOf(">>>") + 3,
trafSelected.length());
					//EXEC localizacion..loc_spu_sitio 1, 'VIPS', 1, 140, 1, 57425485
					int fm = ejecutaUpdate("EXEC localizacion..loc_spu_cuadra " + calleSelected.trim()
+ ","
+ n1Selected.trim() + "," + n2Selected.trim() + "," +
f1Selected.trim() + ","
+ f2Selected.trim() + "," +
txtm_noinic.getText().trim() + ","
+ txtm_nofinal.getText().trim() + "," +
colSelected.trim() + ","
+ s1Selected.trim() + "," + s2Selected.trim() + "," +
trafSelected.trim());
					if(fm == -1)
						debug("Error al ejecutar localizacion..loc_spu_cuadra");
					else if(fm == 0)
						debug("No se modificó registro con localizacion..loc_spu_cuadra");
					else if(fm == 1){
						debug("Registro modificado con localizacion..loc_spu_cuadra");
						llenaControles();
					}
				}
				else{
					debug("Mandar mensaje Tecle el No final e inicial");
				}
				operacion = NINGUNA;
				habilitaCtrlsP5(true);
				txtm_noinic.setText("");
				txtm_nofinal.setText("");
				cargaDatosPanel5();
			}
		}

		void btnEliminar_ActionPerformed(java.awt.event.ActionEvent event)
		{
			// to do: code goes here.
			switch(tabPanel.getCurrentPanelNdx())
			{
				case 0: eliminarP0(); break;
				case 1: eliminarP1(); break;
```

```
                    case 2: eliminarP2(); break;
                    case 3: eliminarP3(); break;
                    case 4: eliminarP4(); break;
                    case 5: eliminarP5(); break;
            }
    }

    private void eliminarP0(){
            if(operacion == NINGUNA){
                    if(ml_calle.getSelectedRow() != -1){
                            operacion = ELIMINAR;
                            //habilitaCtrlsP1(false);
                            //txt_nombre_calle.setText(ml_calle.getCellText(ml_calle.getSelectedRow(), 1));
                            //txt_nombre_calle.requestFocus();
                            eliminarP0();
                    }
                    else{
                            debug("Mandar mensaje de posicionarse en un registro");
                    }
            }
            else if(operacion == ELIMINAR){
                    String idSelec = ml_calle.getCellText(ml_calle.getSelectedRow(), 0).trim();
                    int fm = ejecutaUpdate("EXEC localizacion..loc_spd_calle " + idSelec);
                    if(fm == -1)
                            debug("Error al ejecutar localizacion..loc_spd_calle");
                    else if(fm == 0)
                            debug("No se eliminó registro con localizacion..loc_spd_calle");
                    else if(fm == 1)
                            debug("Registro eliminado con localizacion..loc_spd_calle");


                    operacion = NINGUNA;
                    //habilitaCtrlsP1(true);
                    //txt_nombre_calle.setText("");
                    cargaDatosPanel0();
            }
    }

    private void eliminarP1(){
            if(operacion == NINGUNA){
                    if(ml_colonia.getSelectedRow() != -1){
                            operacion = ELIMINAR;
                            //habilitaCtrlsP1(false);
                            //txt_nombre_calle.setText(ml_calle.getCellText(ml_calle.getSelectedRow(), 1));
                            //txt_nombre_calle.requestFocus();
                            eliminarP1();
                    }
                    else{
                            debug("Mandar mensaje de posicionarse en un registro");
                    }
            }
            else if(operacion == ELIMINAR){
                    String idSelec = ml_colonia.getCellText(ml_colonia.getSelectedRow(), 0).trim();
                    int fm = ejecutaUpdate("EXEC localizacion..loc_spd_colonia " + idSelec);
                    if(fm == -1)
                            debug("Error al ejecutar localizacion..loc_spd_colonia");
                    else if(fm == 0)
                            debug("No se eliminó registro con localizacion..loc_spd_colonia");
                    else if(fm == 1)
                            debug("Registro eliminado con localizacion..loc_spd_colonia");


                    operacion = NINGUNA;
                    //habilitaCtrlsP1(true);
```

```
                    //txt_nombre_calle.setText("");
                    cargaDatosPanel1();
                }
}

private void eliminarP2(){
        if(operacion == NINGUNA){
                if(ml_delegacion.getSelectedRow() != -1){
                        operacion = ELIMINAR;
                        eliminarP2();
                }
                else{
                        debug("Mandar mensaje de posicionarse en un registro");
                }
        }
        else if(operacion == ELIMINAR){
                String idSelec = ml_delegacion.getCellText(ml_delegacion.getSelectedRow(), 0).trim();
                int fm = ejecutaUpdate("EXEC localizacion..loc_spd_delegacion " + idSelec);
                if(fm == -1)
                        debug("Error al ejecutar localizacion..loc_spd_delegacion");
                else if(fm == 0)
                        debug("No se eliminó registro con localizacion..loc_spd_delegacion");
                else if(fm == 1)
                        debug("Registro eliminado con localizacion..loc_spd_delegacion");


                operacion = NINGUNA;
                cargaDatosPanel2();
        }
}

private void eliminarP3(){
        if(operacion == NINGUNA){
                if(ml_sitio.getSelectedRow() != -1){
                        operacion = ELIMINAR;
                        //habilitaCtrlsP1(false);
                        //txt_nombre_calle.setText(ml_calle.getCellText(ml_calle.getSelectedRow(), 1));
                        //txt_nombre_calle.requestFocus();
                        eliminarP3();
                }
                else{
                        debug("Mandar mensaje de posicionarse en un registro");
                }
        }
        else if(operacion == ELIMINAR){
                String idSelec = ml_sitio.getCellText(ml_sitio.getSelectedRow(), 0).trim();
                int fm = ejecutaUpdate("EXEC localizacion..loc_spd_sitio " + idSelec);
                if(fm == -1)
                        debug("Error al ejecutar localizacion..loc_spd_sitio");
                else if(fm == 0)
                        debug("No se eliminó registro con localizacion..loc_spd_sitio");
                else if(fm == 1)
                        debug("Registro eliminado con localizacion..loc_spd_sitio");


                operacion = NINGUNA;
                //habilitaCtrlsP1(true);
                //txt_nombre_calle.setText("");
                cargaDatosPanel3();
        }
}

private void eliminarP4(){
        if(operacion == NINGUNA){
```

```
                    if(ml_nodo.getSelectedRow() != -1){
                        operacion = ELIMINAR;
                        eliminarP4();
                    }
                    else{
                        debug("Mandar mensaje de posicionarse en un registro");
                    }
                }
                else if(operacion == ELIMINAR){
                    String idSelec = ml_nodo.getCellText(ml_nodo.getSelectedRow(), 0).trim();
                    int fm = ejecutaUpdate("EXEC localizacion..loc_spd_nodo " + idSelec);
                    if(fm == -1)
                        debug("Error al ejecutar localizacion..loc_spd_nodo");
                    else if(fm == 0)
                        debug("No se eliminó registro con localizacion..loc_spd_nodo");
                    else if(fm == 1)
                        debug("Registro eliminado con localizacion..loc_spd_nodo");


                    operacion = NINGUNA;
                    cargaDatosPanel4();
                }
            }

            private void eliminarP5(){
                if(operacion == NINGUNA){
                    if(ml_cuadra.getSelectedRow() != -1){
                        operacion = ELIMINAR;
                        //habilitaCtrlsP1(false);
                        //txt_nombre_calle.setText(ml_calle.getCellText(ml_calle.getSelectedRow(), 1));
                        //txt_nombre_calle.requestFocus();
                        eliminarP5();
                    }
                    else{
                        debug("Mandar mensaje de posicionarse en un registro");
                    }
                }
                else if(operacion == ELIMINAR){
                    /*
                                                                @idCalle  INT,
                                                                @idNodo1        INT,
                                                                @idNodo2        INT,
                                                                @idColonia      INT
                    */

                    String idCuadra = ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 0).trim();
                    idCuadra += "," + ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 2).trim();
                    idCuadra += "," + ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 5).trim();
                    idCuadra += "," + ml_cuadra.getCellText(ml_cuadra.getSelectedRow(), 8).trim();

                    int fm = ejecutaUpdate("EXEC localizacion..loc_spd_cuadra " + idCuadra.trim());
                    if(fm == -1)
                        debug("Error al ejecutar localizacion..loc_spd_cuadra");
                    else if(fm == 0)
                        debug("No se eliminó registro con localizacion..loc_spd_cuadra");
                    else if(fm == 1)
                        debug("Registro eliminado con localizacion..loc_spd_cuadra");


                    operacion = NINGUNA;
                    //habilitaCtrlsP1(true);
                    //txt_nombre_calle.setText("");
                    cargaDatosPanel5();
                }
```

```
}


void btnCancelar_ActionPerformed(java.awt.event.ActionEvent event)
{
	// to do: code goes here.
	switch(tabPanel.getCurrentPanelNdx())
	{
		case 0: //calle
			switch(operacion)
			{
				case INSERTAR: CinsertarP0(); break;
				case MODIFICAR: CmodificarP0(); break;
			}
			break;
		case 1: //colonia
			switch(operacion)
			{
				case INSERTAR: CinsertarP1(); break;
				case MODIFICAR: CmodificarP1(); break;
			}
			break;
		case 2: //delegacion
			switch(operacion)
			{
				case INSERTAR: CinsertarP2(); break;
				case MODIFICAR: CmodificarP2(); break;
			}
			break;
		case 3: //sitio
			switch(operacion)
			{
				case INSERTAR: CinsertarP3(); break;
				case MODIFICAR: CmodificarP3(); break;
			}
			break;
		case 4: //nodo
			switch(operacion)
			{
				case INSERTAR: CinsertarP4(); break;
				case MODIFICAR: CmodificarP4(); break;
			}
			break;
		case 5: //cuadra
			switch(operacion)
			{
				case INSERTAR: CinsertarP5(); break;
				case MODIFICAR: CmodificarP5(); break;
			}
			break;
	}
}

private void CinsertarP0(){
	if(operacion == INSERTAR){
		operacion = NINGUNA;
		habilitaCtrlsP0(true);
		txt_nombre_calle.setText("");
	}
}
private void CinsertarP1(){
	if(operacion == INSERTAR){
		operacion = NINGUNA;
		habilitaCtrlsP1(true);
```

```
                txt_colonia.setText("");
            }
        }

        private void CinsertarP2(){
            if(operacion == INSERTAR){
                operacion = NINGUNA;
                habilitaCtrlsP2(true);
                txt_delegacion.setText("");
            }
        }

        private void CinsertarP3(){
            if(operacion == INSERTAR){
                operacion = NINGUNA;
                habilitaCtrlsP3(true);
                txt_sitio.setText("");
                txtm_numero.setText("");
                txtm_telefono.setText("");
            }
        }

        private void CinsertarP4(){
            if(operacion == INSERTAR){
                operacion = NINGUNA;
                habilitaCtrlsP4(true);
                txt_nodo.setText("");
            }
        }

        private void CinsertarP5(){
            if(operacion == INSERTAR){
                operacion = NINGUNA;
                habilitaCtrlsP5(true);
                txtm_noinic.setText("");
                txtm_nofinal.setText("");
                //txtm_longitud.setText("");
            }
        }

        private void CmodificarP0(){
            if(operacion == MODIFICAR){
                operacion = NINGUNA;
                habilitaCtrlsP0(true);
                txt_nombre_calle.setText("");
                cargaDatosPanel0();
            }
        }

        private void CmodificarP1(){
            if(operacion == MODIFICAR){
                operacion = NINGUNA;
                habilitaCtrlsP1(true);
                txt_colonia.setText("");
            }
        }

        private void CmodificarP2(){
            if(operacion == MODIFICAR){
                operacion = NINGUNA;
                habilitaCtrlsP2(true);
                txt_delegacion.setText("");
            }
        }
```

```
	private void CmodificarP3(){
		if(operacion == MODIFICAR){
			operacion = NINGUNA;
			habilitaCtrlsP3(true);
			txt_sitio.setText("");
			txtm_numero.setText("");
			txtm_telefono.setText("");
		}
	}

	private void CmodificarP4(){
		if(operacion == MODIFICAR){
			operacion = NINGUNA;
			habilitaCtrlsP4(true);
			txt_nodo.setText("");
		}
	}

	private void CmodificarP5(){
		if(operacion == MODIFICAR){
			operacion = NINGUNA;
			habilitaCtrlsP5(true);
			txtm_noinic.setText("");
			txtm_nofinal.setText("");
		}
	}


}
```

Apéndice B

Código Fuente *(Frontend)*

LocSrvPrincipal.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;
import java.util.*;

public class LocSrvPrincipal extends HttpServlet{
	Connection con = null;
	Statement stmt = null;
	ResultSet rs = null;

	public void init(ServletConfig config) throws ServletException
	{
	super.init(config);
	}

	public void doGet(HttpServletRequest req, HttpServletResponse res) throws
	ServletException, IOException
	{
		//debug("req.getParameter('txt_usr')="+req.getParameter("txt_usr"));
		if((req.getParameter("txt_usr")!=null)&&(!(req.getParameter("txt_usr").equalsIgnoreCase("")))){
			doPost(req, res);
			return;
      }

		PrintWriter out = res.getWriter();
		res.setContentType("text/vnd.wap.wml");

		out.println("<?xml version='1.0'?>");
		out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
		out.println("<wml>");
		out.println("	<card id='cardP' title='Localización' newcontext='true' ordered='true'>");
		out.println("		<p align='center'>");
		out.println("			Favor de ingresar:");
		out.println("		</p>");
		out.println("		<p align='left'>");
		out.println("			<input name='txt_usr' title='Usuario' type='text' format='a*a'
emptyok='false' maxlength='8' />");
		out.println("			<input name='txt_pwd' title='Password' type='password'
format='a*a' emptyok='false' maxlength='8' />");
		out.println("			<a
href='LocSrvPrincipal?txt_usr=$(txt_usr)&amp;txt_pwd=$(txt_pwd)&amp;URLRand="+ Math.random() +"'>Enviar</a>");
		out.println("		</p>");
		out.println("	</card>");
		out.println("</wml>");
		out.close();
	}//doGet()

	private void debug(String str){
	   System.out.println(str);
	}

	public void doPost(HttpServletRequest req, HttpServletResponse res) throws
	ServletException, IOException
	{
		//Se leen los parametros de la pantalla de login
		String txt_usr = "";
		if(req.getParameter("txt_usr")!=null){
```

```
                txt_usr = (String)req.getParameter("txt_usr");
            }
            String txt_pwd = "";
            if(req.getParameter("txt_pwd")!=null){
                txt_pwd = (String)req.getParameter("txt_pwd");
            }
            String nvaBusq = "";
            if(req.getParameter("nvaBusq")!=null){
                nvaBusq = (String)req.getParameter("nvaBusq");
            }

            // se valida usr y pwd
            int idUsuario = validarUsuario(txt_usr, txt_pwd);
            debug("Usr:"+ txt_usr +"\nPwd:"+ txt_pwd +"\nidUsr:"+ idUsuario );

            PrintWriter out = res.getWriter();
            res.setContentType("text/vnd.wap.wml");

            if(idUsuario < 1) //Si el usuario y/o password es incorrecto
            {
                out.println("<?xml version='1.0'?>");
                out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
                out.println("<wml>");
                out.println("  <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
                out.println("    <p align='left'>");
                out.println("      Usuario y/o Pwd incorrecto(s).");
                out.println("    </p>");
                out.println("    <p align='left'>");
                out.println("      <a href='LocSrvPrincipal?URLRand="+ Math.random()
+"' title='Aceptar'>Aceptar</a>");
                out.println("    </p>");
                out.println("    <p align = 'left'>");
                //out.println("      Usr:"+ txt_usr +"-Pwd:"+ txt_pwd +"-idUsr:"+
idUsuario );
                out.println("    </p>");
                out.println("  </card>");
                out.println("</wml>");
                out.close();
            }
            else
            {
                HttpSession sesion = req.getSession(true);
                sesion.invalidate();
                sesion = req.getSession(true);

                if(nvaBusq.equalsIgnoreCase("S")){
                    String sqlString = "localizacion.dbo.loc_spd_busqueda " + idUsuario;
                    debug(sqlString);
                    ejecutaUpdate(sqlString);
                }

                sesion.putValue("idUsr", new Integer(idUsuario));
                sesion.putValue("txt_usr", new String(txt_usr));
                sesion.putValue("txt_pwd", new String(txt_pwd));

                // si son validos mostrar ultima pantalla

                // Leer loc_busqueda, si stsDir != 0 (hay busqueda pendiente) -> Realizar lo necesario
(Grabar variables de sesion, redirigir al URL almacenado, )
            int stsDir = 0;
            String URL1 = "";
            String URL2 = "";
                try
```

```
{
ResultSet rs1 = ejecutaQuery("LOCALIZACION.dbo.loc_spr_busqueda " + idUsuario);

if(rs1!=null){
if(rs1.next()){
        stsDir = Integer.parseInt(rs1.getString(2).trim());
        URL1 = rs1.getString(3).trim();
        URL2 = rs1.getString(4).trim();
}
}
else
debug("rs1 fue null");

rs1.close();
//cierraBD();
}catch(Exception e){
debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
e.printStackTrace(System.out);
}

if(stsDir == 1)
{
        res.sendRedirect(URL1);
}

if(stsDir >= 2)
{
        //cargar variables de sesion con la ubicacion de la primer direccion
        //      LocSrvDireccion?dir=Or&del=7&col=5&calle=7&num=222
        StringTokenizer stk = new StringTokenizer(URL1.substring(URL1.indexOf("?")+1,
URL1.length()), "&");
        String dir = (String)stk.nextElement();
        String parte = "";

        dir = dir.substring(dir.indexOf("=")+1, dir.length());

        while(stk.hasMoreElements())
        {
                parte = (String)stk.nextElement();
                debug("name="+parte.substring(0,parte.indexOf("=")) + dir);
                debug("value="+parte.substring(parte.indexOf("=")+1, parte.length()));
                sesion.putValue(parte.substring(0,parte.indexOf("=")) + dir, new
Integer(parte.substring(parte.indexOf("=")+1, parte.length())));
        }

        if(stsDir == 2)
        {
                if(dir.equalsIgnoreCase("Or"))
                        res.sendRedirect("LocSrvRutaDosPuntos?Rand="+
Math.random());
                if(dir.equalsIgnoreCase("Ori"))
                        res.sendRedirect("LocSrvRutaSitio?Rand="+ Math.random());
        }
}

if(stsDir == 3)
{
        res.sendRedirect(URL2);
}

if(stsDir >= 4)
{
        //cargar variables de sesion con la ubicacion de la segunda direccion
        //      LocSrvDireccion?dir=De&del=7&col=8&calle=9&num=1
```

```
                                        StringTokenizer stk = new StringTokenizer(URL2.substring(URL2.indexOf("?")+1,
URL2.length()), "&");
                                        String dir = (String)stk.nextElement();
                                        String parte = "";

                                        dir = dir.substring(dir.indexOf("=")+1, dir.length());

                                        while(stk.hasMoreElements())
                                        {
                                                parte = (String)stk.nextElement();
                                                debug("name="+parte.substring(0,parte.indexOf("=")) + dir);
                                                debug("value="+parte.substring(parte.indexOf("=")+1, parte.length()));
                                                sesion.putValue(parte.substring(0,parte.indexOf("=")) + dir, new
Integer(parte.substring(parte.indexOf("=")+1, parte.length())));
                                        }

                                        if(stsDir == 4)
                                        {
                                                if(dir.equalsIgnoreCase("De"))
                                                        res.sendRedirect("LocSrvRutaDosPuntos?Rand="+
Math.random());
                                                if(dir.equalsIgnoreCase("Sit"))
                                                        res.sendRedirect("LocSrvRutaSitio?Rand="+ Math.random());
                                        }

                                }


                                out.println("<?xml version='1.0'?>");
                                out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
                                out.println("<wml>");
                                out.println("   <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
                                out.println("           <p align='left'>");
                                out.println("                   Buscar ruta de:");
                                out.println("           </p>");
                                out.println("           <p align='left'>");
                                out.println("                   <a href='LocSrvRutaDosPuntos?Rand="+
Math.random() +"' title='R2P'>Dos Puntos</a>");
                                out.println("           </p>");
                                out.println("           <p align='left'>");
                                out.println("                   <a href='LocSrvRutaSitio?Rand="+ Math.random()
+"' title='RS'>Sitio de Interes</a>");
                                out.println("           </p>");
                                out.println("           <p align = 'left'>");
                                //out.println("                   Usr:"+ txt_usr +"-Pwd:"+ txt_pwd +"-idUsr:"+
idUsuario );
                                out.println("           </p>");
                                out.println("   </card>");
                                out.println("</wml>");
                                out.close();
                        }
        }

        private int validarUsuario(String usr, String pwd)
        {
                int idusr = 0;
                        try{
                        ResultSet rs1 = ejecutaQuery("LOCALIZACION.dbo.loc_sp_validausuario '" + usr + "', '" + pwd + "'");

                        if(rs1!=null){
                        if(rs1.next()){
                                idusr = Integer.parseInt(rs1.getString(1).trim());
                        }
```

```
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }
            return idusr;
    }

            private ResultSet ejecutaQuery(String sp){
                    try{
                            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                    stmt = con.createStatement();
                    rs = stmt.executeQuery(sp);
               }catch(Exception e){
                  debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
               return rs;
            }


            private int ejecutaUpdate(String sp){
                    int filasMod = -1;
                    try{
                            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                    stmt = con.createStatement();
                    filasMod = stmt.executeUpdate(sp);
               }catch(Exception e){
                  debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
               return filasMod;
            }
}
```

LocSrvRutaDosPuntos.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvRutaDosPuntos extends HttpServlet{
            Connection con = null;
            Statement stmt = null;
            ResultSet rs = null;

            public void init(ServletConfig config) throws ServletException
            {
            super.init(config);
            }

            public void doGet(HttpServletRequest req, HttpServletResponse res) throws
```

```
        ServletException, IOException
        {
                PrintWriter out = res.getWriter();
                res.setContentType("text/vnd.wap.wml");

                out.println("       <?xml version='1.0'?>");
                out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
                out.println("<wml>");
                out.println("       <card id='card2P' title='Dos Puntos' newcontext='true' ordered='true'>");
                out.println("               <p align='left'>");
                out.println("                       <a href='LocSrvDireccion?dir=Or' title='DO'>Dir. Origen</a>");
                out.println("               </p>");
                out.println("               <p align='left'>");
                out.println("                       <a href='LocSrvDireccion?dir=De' title='DD'>Dir.
Destino</a>");
                out.println("               </p>");
                out.println("               <p align = 'center'>");
                out.println("                       <a href='LocSrvBuscarRutaDosPuntos?Rand="+ Math.random()
+"'>Ruta Optima</a>");
                out.println("               </p>");
                out.println("               <p>");
                out.println("               <a href='LocSrvPrincipal?Rand="+ Math.random() +"'>Regresar</a>");
                out.println("               </p>");
                out.println("       </card>");
                out.println("</wml>");
        }//doGet()

        private void debug(String str){
            System.out.println(str);
        }
}
```

LocSrvDireccion.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvDireccion extends HttpServlet{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

 public void init(ServletConfig config) throws ServletException
  {
    super.init(config);
  }

  public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException
  {
        PrintWriter out = res.getWriter();
        res.setContentType("text/vnd.wap.wml");

        HttpSession sesion = req.getSession(false); // cambie true -> false

        String dir = "";
        if(req.getParameter("dir")!=null){
```

```
		dir = (String)req.getParameter("dir");
	}

	int del = -1;
	if(req.getParameter("del")!=null){
		del = Integer.parseInt((String)req.getParameter("del"));
	}

	int col = -1;
	if(req.getParameter("col")!=null){
		col = Integer.parseInt((String)req.getParameter("col"));
	}

	int calle = -1;
	if(req.getParameter("calle")!=null){
		calle = Integer.parseInt((String)req.getParameter("calle"));
	}

	int num = -1;
	if(req.getParameter("num")!=null){
		num = Integer.parseInt((String)req.getParameter("num"));
		if(dir.equalsIgnoreCase("Or")){
			sesion.putValue("delOr", new Integer(del));
			sesion.putValue("colOr", new Integer(col));
			sesion.putValue("calleOr", new Integer(calle));
			sesion.putValue("numOr", new Integer(num));
		}else if(dir.equalsIgnoreCase("De")){
			sesion.putValue("delDe", new Integer(del));
			sesion.putValue("colDe", new Integer(col));
			sesion.putValue("calleDe", new Integer(calle));
			sesion.putValue("numDe", new Integer(num));
		}
	}


	debug("-------------------------------------\nPrueba: ");
	debug("delOr=" + sesion.getValue("delOr"));
	debug("delDe=" + sesion.getValue("delDe"));
	debug("colOr=" + sesion.getValue("colOr"));
	debug("colDe=" + sesion.getValue("colDe"));
	debug("calleOr=" + sesion.getValue("calleOr"));
	debug("calleDe=" + sesion.getValue("calleDe"));
	debug("numOr=" + sesion.getValue("numOr"));
	debug("numDe=" + sesion.getValue("numDe"));

	/*
	debug("req.getClass().getName()=" + req.getClass().getName() );
	debug("req.getScheme()=" + req.getScheme() );
	debug("req.getServerPort()=" + req.getServerPort());
	debug("req.getServerName()=" + req.getServerName());
	debug("req.getRequestURI()=" + req.getRequestURI());
	debug("req.getServletPath()=" + req.getServletPath());
	debug("req.getQueryString()=" + req.getQueryString());
	*/

	//loc_spi_busqueda 2,1,1,'adlskjdkl'
	String sqlString = "localizacion.dbo.loc_spi_busqueda ";
	sqlString += (dir.equalsIgnoreCase("Or")?"1":"2") +", ";
	sqlString += sesion.getValue("idUsr") + ", ";
	sqlString += (dir.equalsIgnoreCase("Or")?((num == -1)?"1":"2"):((num == -1)?"3":"4")) +", '";
	//sqlString += req.getScheme()+ "://"+ req.getServerName() +":"+ req.getServerPort() + req.getServletPath()
+"?"+ req.getQueryString() +"'";
	sqlString += "LocSrvDireccion?"+ req.getQueryString() +"'";
	debug(sqlString);
```

```
            ejecutaUpdate(sqlString);

            if(del == -1){
                        frameDelegacion(out, dir);
            }
            else if(col == -1){
                        frameColonia(out, dir, del);
            }
            else if(calle == -1){
                        frameCalle(out, dir, del, col);
            }
            else if(num == -1){
                        frameNum(out, dir, del, col, calle);
            }
            else{
                        res.sendRedirect("LocSrvRutaDosPuntos");
            }



  }//doGet()


    private void frameNum(PrintWriter out, String dir, int deleg, int col, int calle){
                        out.println("<?xml version=\"1.0\"?>");
                        //out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML1.1//EN\"
\"http://www.wapforum.org/DTD/wml1.1.xml\">");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                        out.println("<wml>");
                        out.println("       <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
                        out.println("       <p align='left'>");
                        out.println("       Introduce:");

                        out.println("<input name='txt_numero' type='text' format='*N' emptyok='false' maxlength='5'
title='Numero'/>");
                        out.println("       </p>");
                        out.println("       <p>");
                        out.println("       <a href='LocSrvDireccion?dir="
                                                + dir + "&amp;del="
                                                + deleg + "&amp;col="
                                                + col + "'>Regresar</a>");
                        out.println("       </p>");
                        out.println("       <p>");
                        out.println("       <a href='LocSrvDireccion?dir="
                                                + dir + "&amp;del="
                                                + deleg + "&amp;col="
                                                + col + "&amp;calle="
                                                + calle + "&amp;num="
                                                + "$(txt_numero)'>Aceptar</a>");
                        out.println("       </p>");
                        out.println("</card>");
                        out.println("</wml>");
                        out.close();
  }//frameNum()


  private void frameCalle(PrintWriter out, String dir, int deleg, int col){
                        out.println("<?xml version=\"1.0\"?>");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                        out.println("<wml>");
                        out.println("       <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
```

```
                out.println("       <p align='left'>");
                out.println("       Selecciona");
                out.println("               <select title='Cal' name='cbo_calleO' multiple='false'>");

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_cuadracalle " + col);

                if(rs1!=null){
                        String str1 = "";
                        String str2 = "";
                while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str2 = rs1.getString(2).trim();
                out.println("                       <option value='" + str1
                                                + "' onpick='LocSrvDireccion?dir="
                                                + dir + "&amp;del="
                                                + deleg + "&amp;col="
                                                + col + "&amp;calle="
                                                + str1 + "'>"+str2+"</option>");
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }

                out.println("               </select>");
                out.println("       </p>");
                out.println("       <p>");
                out.println("       <a href='LocSrvDireccion?dir="
                                                + dir + "&amp;del="
                                                + deleg + "'>Regresar</a>");
                out.println("       </p>");
                out.println("</card>");
                out.println("</wml>");
                out.close();
 }//frameCalle()


 private void frameColonia(PrintWriter out, String dir, int deleg){
                out.println("<?xml version=\"1.0\"?>");
                out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                out.println("<wml>");
                out.println("       <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
                out.println("       <p align='left'>");
                out.println("       Selecciona");
                out.println("               <select title='Col' name='cbo_coloniaO' multiple='false'>");

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_colonia null," + deleg);

                if(rs1!=null){
                        String str1 = "";
                        String str2 = "";
                while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str2 = rs1.getString(2).trim();
                out.println("                       <option value='" + str1
```

```
                                                    + "' onpick='LocSrvDireccion?dir="
                                                    + dir + "&amp;del="
                                                    + deleg + "&amp;col="
                                                    + str1 + "'>"+str2+"</option>");
                    }
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("                    </select>");
                    out.println("          </p>");
                    out.println("          <p>");
                    out.println("          <a href='LocSrvDireccion?dir="
                                                    + dir + "'>Regresar</a>");
                    out.println("          </p>");
                    out.println("</card>");
                    out.println("</wml>");
                    out.close();
 } //frameColonia()

  private void frameDelegacion(PrintWriter out, String dir){
                    out.println("<?xml version=\"1.0\"?>");
                    out.println("<!DOCTYPE        wml        PUBLIC        \"-//WAPFORUM//DTD        WML        1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                    out.println("<wml>");
                    out.println("          <card id='cardP' title='Delegacion' newcontext='true' ordered='true'>");
                    out.println("          <p align='left'>");
                    out.println("          Selecciona");
                    out.println("                    <select title='Del:' name='cbo_delegacionO' multiple='false'>");

                    try{
                    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_delegacion");

                    if(rs1!=null){
                              String str1 = "";
                              String str2 = "";
                    while(rs1.next()){
                              str1 = rs1.getString(1).trim();
                              str2 = rs1.getString(2).trim();
                    out.println("                              <option value='" + str1
                                                    +"' onpick='LocSrvDireccion?dir="
                                                    + dir + "&amp;del="
                                                    + str1 + "'>"+str2+"</option>");
                    }
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("                    </select>");
                    out.println("          </p>");
```

```
                    out.println("          <p>");
                    out.println("          <a href='LocSrvRutaDosPuntos'>Regresar</a>");
                    out.println("          </p>");

                    out.println("</card>");
                    out.println("</wml>");
                    out.close();
 }//frameDelegacion()

 private ResultSet ejecutaQuery(String sp){
                    try{
                              Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                    stmt = con.createStatement();
                    rs = stmt.executeQuery(sp);
               }catch(Exception e){
                  debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
               return rs;
          }

          private int ejecutaUpdate(String sp){
                    int filasMod = -1;
                    try{
                              Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                    stmt = con.createStatement();
                    filasMod = stmt.executeUpdate(sp);
               }catch(Exception e){
                  debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
               return filasMod;
          }

          private void cierraBD(){
                    try{
                              rs.close();
                              stmt.close();
                              con.close();
                    }catch(Exception e){
                  debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
          }

          private void debug(String str){
               System.out.println(str);
          }
}
```

LocSrvBuscarRutaDosPuntos.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import Dijkstra;
```

```
public class LocSrvBuscarRutaDosPuntos extends HttpServlet
{
	private Connection con = null;
	private Statement stmt = null;
	private ResultSet rs = null;

	public void doGet(HttpServletRequest req, HttpServletResponse res) throws
	ServletException, IOException
	{
		try{
			PrintWriter out = res.getWriter();
			res.setContentType("text/vnd.wap.wml");

			HttpSession sesion = req.getSession(false); // cambie true -> false

			Integer delOr = new Integer(-1);
			Integer delDe = new Integer(-1);
			Integer colOr = new Integer(-1);
			Integer colDe = new Integer(-1);
			Integer calleOr = new Integer(-1);
			Integer calleDe = new Integer(-1);
			Integer numOr = new Integer(-1);
			Integer numDe = new Integer(-1);

			if(sesion.getValue("delOr")!=null){
				delOr = (Integer)sesion.getValue("delOr");
			}
			if(sesion.getValue("delDe")!=null){
				delDe = (Integer)sesion.getValue("delDe");
			}
			if(sesion.getValue("colOr")!=null){
				colOr = (Integer)sesion.getValue("colOr");
			}
			if(sesion.getValue("colDe")!=null){
				colDe = (Integer)sesion.getValue("colDe");
			}
			if(sesion.getValue("calleOr")!=null){
				calleOr = (Integer)sesion.getValue("calleOr");
			}
			if(sesion.getValue("calleDe")!=null){
				calleDe = (Integer)sesion.getValue("calleDe");
			}
			if(sesion.getValue("numOr")!=null){
				numOr = (Integer)sesion.getValue("numOr");
			}
			if(sesion.getValue("numDe")!=null){
				numDe = (Integer)sesion.getValue("numDe");
			}

			String txt_usr = "";
			if(sesion.getValue("txt_usr")!=null){
				txt_usr = (String)sesion.getValue("txt_usr");
			}

			String txt_pwd = "";
			if(sesion.getValue("txt_pwd")!=null){
				txt_pwd = (String)sesion.getValue("txt_pwd");
			}


			Dijkstra d = new Dijkstra(calleOr.intValue() , numOr.intValue(), calleDe.intValue(),
numDe.intValue());

			String rutaOptima[] = new String[d.getNumNodosRutaOptima()];
```

```
                                rutaOptima = d.getNombresCalles();

                                out.println("<?xml version=\"1.0\"?>");
                                out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML1.1//EN\"
\"http://www.wapforum.org/DTD/wml1.1.xml\">");
                                out.println("<wml>");
                                out.println("<card id='cardres' title='Ruta Optima'>");

                                int nume = 1;
                                for(int i=0; i<rutaOptima.length; i++){
                                        if(String.valueOf(rutaOptima[i]).indexOf("(") == -1){
                                                out.println("<p align='left' mode='nowrap'><small>" + rutaOptima[i] +
"</small></p>");
                                        }else{
                                                if(i==0){

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i+1].substring(0,rutaOptima[i+1].i
ndexOf("(")))!=0)
                                                                out.println("<p align='left' mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                                else if(i==1){
                                                        out.println("<p align='left' mode='nowrap'><small>" +
(nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                                else if(i==(rutaOptima.length - 1)){

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i-1].substring(0,rutaOptima[i-
1].indexOf("(")))!=0)
                                                                out.println("<p align='left' mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                                else{
                                                        //if(rutaOptima[i].compareTo(rutaOptima[i-1])!=0)

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i-1].substring(0,rutaOptima[i-
1].indexOf("(")))!=0)
                                                                out.println("<p align='left' mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                        }
                                }

                                out.println("                   <p align='left'>");
                                out.println("                           <br/><a                 href='LocSrvRutaDosPuntos'
title='NewD'>Cambiar Direccion</a>");
                                out.println("                   </p>");

                                out.println("                   <p align='left'>");
                                out.println("                           <a href='LocSrvPrincipal?nvaBusq=S&amp;txt_usr="+
txt_usr +"&amp;txt_pwd="+ txt_pwd +"&URLRand="+ Math.random() +"' title='NewB'>Nueva Busqueda</a>");
                                out.println("                   </p>");

                                out.println("<p align='center'><small><br/>Visita cic.ipn.mx</small></p>");
                                out.println("</card>");
                                out.println("</wml>");
                                out.close();

                        }catch(Exception e){
                        debug("Error en doGet():" + e.toString() + "_Msg=" + e.getMessage());
                        e.printStackTrace(System.out);
                        }
 }
```

```
private int encontrarNodoCercano(int idcalle, int numero){
        int nodoCercano = -1;

   int nodo1 = -1;
   int nodo2 = -1;
   int flujo1 = -1;
   int flujo2 = -1;
   int numIni = -1;
   int numFin = -1;
   int trafico = -1;

        try{
        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_buscaNodos " + idcalle + ", " + numero);

        if(rs1!=null){
        if(rs1.next()){
                nodo1 = rs1.getInt(1);
                nodo2 = rs1.getInt(2);
                flujo1 = rs1.getInt(3);
                flujo2 = rs1.getInt(4);
                numIni = rs1.getInt(5);
                numFin = rs1.getInt(6);
                trafico = rs1.getInt(7);
        }
        }
        else
        debug("rs1 fue null");

        rs1.close();
        //cierraBD();
        }catch(Exception e){
        debug("Error en encontrarNodoCercano():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
        }

        if(trafico == 0){
                // mandar mensaje de que la cuadra se encuentra bloqueada,
                // por lo que tiene que situarse en otra posicion
        }
        else if((flujo1 == 0) && (flujo2 == 0)){
                // mandar mensaje de error,
                // por lo que tiene que situarse en otra posicion
        }
        else if(flujo1 == 0){
                nodoCercano = nodo2;
        }
        else if(flujo2 == 0){
                nodoCercano = nodo1;
        }
        else{
                if((numero >= numIni) && (numero <= (numIni + (numFin-numIni)/2)))
                        nodoCercano = nodo1;
                else
                        nodoCercano = nodo2;
        }

        return nodoCercano;
}

private ResultSet ejecutaQuery(String sp){
        try{
                Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
```

```
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
            }catch(Exception e){
                debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
            }
            return rs;
        }
        private void debug(String str){
            System.out.println(str);
        }
}
```

LocSrvRutaSitio.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvRutaSitio extends HttpServlet{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

        public void init(ServletConfig config) throws ServletException
        {
                super.init(config);
        }

        public void doGet(HttpServletRequest req, HttpServletResponse res) throws
                ServletException, IOException
        {
                PrintWriter out = res.getWriter();
                res.setContentType("text/vnd.wap.wml");

                out.println("       <?xml version='1.0'?>");
                out.println("<!DOCTYPE       wml       PUBLIC       '-//WAPFORUM//DTD       WML       1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
                out.println("<wml>");
                out.println("       <card id='cardSI' title='Sitio Interes' newcontext='true' ordered='true'>");
                out.println("               <p align='left'>");
                out.println("                       <a      href='LocSrvDireccionSitio?dir=Ori'      title='DO'>Dir.
Origen</a>");
                out.println("               </p>");
                out.println("               <p align='left'>");
                out.println("                       <a href='LocSrvSitio?dir=Sit' title='DS'>Sitio</a>");
                out.println("               </p>");
                out.println("               <p align = 'center'>");
                out.println("                       <a      href='LocSrvBuscarRutaSitio?Rand="+      Math.random()
+"'>Ruta Optima</a>");
                out.println("               </p>");
                out.println("               <p>");
                out.println("               <a href='LocSrvPrincipal?Rand="+ Math.random() +"'>Regresar</a>");
                out.println("               </p>");
                out.println("       </card>");
                out.println("</wml>");
        }//doGet()

        private void debug(String str){
```

```
            System.out.println(str);
        }
}
```

LocSrvDireccionSitio.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvDireccionSitio extends HttpServlet{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

 public void init(ServletConfig config) throws ServletException
  {
    super.init(config);
  }

  public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException
  {
        PrintWriter out = res.getWriter();
        res.setContentType("text/vnd.wap.wml");

        HttpSession sesion = req.getSession(false); // cambie true -> false

        String dir = "";
        if(req.getParameter("dir")!=null){
                dir = (String)req.getParameter("dir");
        }

        int del = -1;
        if(req.getParameter("del")!=null){
                del = Integer.parseInt((String)req.getParameter("del"));
        }

        int col = -1;
        if(req.getParameter("col")!=null){
                col = Integer.parseInt((String)req.getParameter("col"));
        }

        int calle = -1;
        if(req.getParameter("calle")!=null){
                calle = Integer.parseInt((String)req.getParameter("calle"));
        }

        int num = -1;
        if(req.getParameter("num")!=null){
                num = Integer.parseInt((String)req.getParameter("num"));
                if(dir.compareTo("Ori")==0){
                        sesion.putValue("delOri", new Integer(del));
                        sesion.putValue("colOri", new Integer(col));
                        sesion.putValue("calleOri", new Integer(calle));
                        sesion.putValue("numOri", new Integer(num));
                }else if(dir.compareTo("Sit")==0){
                        sesion.putValue("delSit", new Integer(del));
                        sesion.putValue("colSit", new Integer(col));
```

```
                        sesion.putValue("calleSit", new Integer(calle));
                        sesion.putValue("numSit", new Integer(num));
                }
        }


        debug("-------*------*-------*-------*--------\nPrueba: ");
        debug("delOri=" + sesion.getValue("delOri"));
        debug("colOri=" + sesion.getValue("colOri"));
        debug("calleOri=" + sesion.getValue("calleOri"));
        debug("numOri=" + sesion.getValue("numOri"));
        debug("delSit=" + sesion.getValue("delSit"));
        debug("colSit=" + sesion.getValue("colSit"));
        debug("calleSit=" + sesion.getValue("calleSit"));
        debug("numSit=" + sesion.getValue("numSit"));

        //loc_spi_busqueda 2,1,1,'adlskjdkl'
        String sqlString = "localizacion.dbo.loc_spi_busqueda ";
        sqlString += (dir.equalsIgnoreCase("Ori")?"1":"2") +", ";
        sqlString += sesion.getValue("idUsr") + ", ";
        sqlString += (dir.equalsIgnoreCase("Ori")?((num == -1)?"1":"2"):((num == -1)?"3":"4")) +", '";
        //sqlString += req.getScheme()+ "://"+ req.getServerName() +":"+ req.getServerPort() + req.getServletPath()
+"?"+ req.getQueryString() +"'";
        sqlString += "LocSrvDireccionSitio?"+ req.getQueryString() +"'";
        debug(sqlString);
        ejecutaUpdate(sqlString);

        if(del == -1){
                frameDelegacion(out, dir);
        }
        else if(col == -1)
                frameColonia(out, dir, del);
        else if(calle == -1)
                frameCalle(out, dir, del, col);
        else if(num == -1)
                frameNum(out, dir, del, col, calle);
        else
                res.sendRedirect("LocSrvRutaSitio");



 }//doGet()


   private void frameNum(PrintWriter out, String dir, int deleg, int col, int calle){
                out.println("<?xml version=\"1.0\"?>");
                //out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML1.1//EN\"
\"http://www.wapforum.org/DTD/wml1.1.xml\">");
                out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                out.println("<wml>");
                out.println("   <card id='cardP' title='Numero' newcontext='true' ordered='true'>");
                out.println("   <p align='left'>");
                out.println("   Introduce:");

                out.println("<input name='txt_numero' type='text' format='*N' emptyok='false' maxlength='5'
title='Numero'/>");
                out.println("   </p>");
                out.println("   <p>");
                out.println("   <a href='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + deleg + "&amp;col="
                                                        + col + "'>Regresar</a>");
```

```
                out.println("       </p>");
                out.println("       <p>");
                out.println("       <a href='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + deleg + "&amp;col="
                                                        + col + "&amp;calle="
                                                        + calle + "&amp;num="
                                                        + "$(txt_numero)'>Aceptar</a>");
                out.println("       </p>");
                out.println("</card>");
                out.println("</wml>");
                out.close();
 }//frameNum()


 private void frameCalle(PrintWriter out, String dir, int deleg, int col){
                out.println("<?xml version=\"1.0\"?>");
                out.println("<!DOCTYPE       wml       PUBLIC       \"-//WAPFORUM//DTD       WML       1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                out.println("<wml>");
                out.println("       <card id='cardP' title='Calle' newcontext='true' ordered='true'>");
                out.println("       <p align='left'>");
                out.println("       Selecciona");
                out.println("               <select title=':' name='cbo_calleO' multiple='false'>");

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_cuadracalle " + col);

                if(rs1!=null){
                        String str1 = "";
                        String str2 = "";
                while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str2 = rs1.getString(2).trim();
                out.println("                       <option value='" + str1
                                                        + "' onpick='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + deleg + "&amp;col="
                                                        + col + "&amp;calle="
                                                        + str1 + "'>"+str2+"</option>");
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }

                out.println("               </select>");
                out.println("       </p>");
                out.println("       <p>");
                out.println("       <a href='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + deleg + "'>Regresar</a>");
                out.println("       </p>");
                out.println("</card>");
                out.println("</wml>");
                out.close();
 }//frameCalle()
```

```
 private void frameColonia(PrintWriter out, String dir, int deleg){
                    out.println("<?xml version=\"1.0\"?>");
                    out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                    out.println("<wml>");
                    out.println("  <card id='cardP' title='Colonia' newcontext='true' ordered='true'>");
                    out.println("  <p align='left'>");
                    out.println("  Selecciona");
                    out.println("  <select title=':' name='cbo_coloniaO' multiple='false'>");

                    try{
                    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_colonia null," + deleg);

                    if(rs1!=null){
                              String str1 = "";
                              String str2 = "";
                    while(rs1.next()){
                              str1 = rs1.getString(1).trim();
                              str2 = rs1.getString(2).trim();
                    out.println("  <option value='" + str1
                                        + "' onpick='LocSrvDireccionSitio?dir="
                                        + dir + "&amp;del="
                                        + deleg + "&amp;col="
                                        + str1 + "'>"+str2+"</option>");
                    }
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("  </select>");
                    out.println("  </p>");
                    out.println("  <p>");
                    out.println("  <a href='LocSrvDireccionSitio?dir="
                                        + dir + "'>Regresar</a>");
                    out.println("  </p>");
                    out.println("</card>");
                    out.println("</wml>");
                    out.close();
 } //frameColonia()

 private void frameDelegacion(PrintWriter out, String dir){
                    out.println("<?xml version=\"1.0\"?>");
                    out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                    out.println("<wml>");
                    out.println("  <card id='cardP' title='Delegacion' newcontext='true' ordered='true'>");
                    out.println("  <p align='left'>");
                    out.println("  Selecciona");
                    out.println("  <select title=':' name='cbo_delegacionO' multiple='false'>");

                    try{
                    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_delegacion");

                    if(rs1!=null){
                              String str1 = "";
                              String str2 = "";
```

```
                while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str2 = rs1.getString(2).trim();
                out.println("                                <option value='" + str1
                                                        +"' onpick='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + str1 + "'>"+str2+"</option>");
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }

                out.println("                </select>");
                out.println("        </p>");
                out.println("        <p>");
                out.println("        <a href='LocSrvRutaSitio'>Regresar</a>");
                out.println("        </p>");

                out.println("</card>");
                out.println("</wml>");
                out.close();
}//frameDelegacion()

private ResultSet ejecutaQuery(String sp){
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
           }catch(Exception e){
              debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return rs;
        }

        private int ejecutaUpdate(String sp){
                int filasMod = -1;
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                filasMod = stmt.executeUpdate(sp);
           }catch(Exception e){
              debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return filasMod;
        }

        private void cierraBD(){
                try{
                        rs.close();
                        stmt.close();
                        con.close();
                }catch(Exception e){
              debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
```

```
                e.printStackTrace(System.out);
            }
        }

        private void debug(String str){
            System.out.println(str);
        }
}
```

LocSrvSitio.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvSitio extends HttpServlet{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

 public void init(ServletConfig config) throws ServletException
  {
    super.init(config);
  }

  public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException
  {
        PrintWriter out = res.getWriter();
        res.setContentType("text/vnd.wap.wml");

        HttpSession sesion = req.getSession(false); // cambie true -> false

        String xcat = "";
        if(req.getParameter("xcat")!=null){
                xcat = (String)req.getParameter("xcat");
        }

        int categ = -1;
        if(req.getParameter("categ")!=null){
                categ = Integer.parseInt((String)req.getParameter("categ"));
        }

        int idSitio = -1;
        if(req.getParameter("idSitio")!=null){
                idSitio = Integer.parseInt((String)req.getParameter("idSitio"));
        }


        String dir = "";
        if(req.getParameter("dir")!=null){
                dir = (String)req.getParameter("dir");
        }

        int del = -1;
        if(req.getParameter("del")!=null){
                del = Integer.parseInt((String)req.getParameter("del"));
        }
```

```
		int col = -1;
		if(req.getParameter("col")!=null){
			col = Integer.parseInt((String)req.getParameter("col"));
		}

		int calle = -1;
		if(req.getParameter("calle")!=null){
			calle = Integer.parseInt((String)req.getParameter("calle"));
		}

		int num = -1;
		if(req.getParameter("num")!=null){
			num = Integer.parseInt((String)req.getParameter("num"));

			sesion.putValue("delSit", new Integer(del));
			sesion.putValue("colSit", new Integer(col));
			sesion.putValue("calleSit", new Integer(calle));
			sesion.putValue("numSit", new Integer(num));

			xcat = "Salir";
		}

		int masCercano = -1;
		if(req.getParameter("masCercano")!=null){
			masCercano = Integer.parseInt((String)req.getParameter("masCercano"));

			sesion.putValue("masCercanoSit", new Integer(masCercano));
			sesion.putValue("categSit", new Integer(categ));

			xcat = "Salir";
		}

		debug("-------*------*-------*-------*--------\nPrueba: ");
		debug("-->xcat=" + xcat);
		debug("delSit=" + sesion.getValue("delSit"));
		debug("colSit=" + sesion.getValue("colSit"));
		debug("calleSit=" + sesion.getValue("calleSit"));
		debug("numSit=" + sesion.getValue("numSit"));
		debug("masCercanoSit=" + sesion.getValue("masCercanoSit"));
		debug("categSit=" + sesion.getValue("categSit"));

		//loc_spi_busqueda 2,1,1,'adlskjdkl'
		String sqlString = "localizacion.dbo.loc_spi_busqueda ";
		sqlString += (dir.equalsIgnoreCase("Ori")?"1":"2") +", ";
		sqlString += sesion.getValue("idUsr") + ", ";
		sqlString += (dir.equalsIgnoreCase("Ori")?((num == -1)?"1":"2"):((num == -1)?"3":"4")) +", '";
		//sqlString += req.getScheme()+ "://"+ req.getServerName() +":"+ req.getServerPort() + req.getServletPath()
+"?"+ req.getQueryString() +"'";
		sqlString += "LocSrvSitio?"+ req.getQueryString() +"'";
		debug(sqlString);
		ejecutaUpdate(sqlString);


		if(xcat.compareTo("")==0){
			frameBuscarSitioPor(out, dir);
		}
		else if(xcat.compareTo("Si")==0){
			if(categ == -1){
				frameBuscarSitioPorCateg(out, dir);
			}
			else{
				if(idSitio == -1)
					frameBuscarSitioPorCateg(out, dir, categ);
				else if(idSitio != 0)
```

```
                                        seleccionarDireccionSitio(res, idSitio);
                                else
                                        redireccionarSinDirSitio(res, categ);
                        }
                }
                else if(xcat.compareTo("No")==0){
                        //frameBuscarSitioPorNomb(out, dir);
                }
                else if(xcat.compareTo("Salir")==0){
                        res.sendRedirect("LocSrvRutaSitio");
                }

        }//doGet()


        private void frameBuscarSitioPor(PrintWriter out, String dir){
                        out.println("<?xml version=\"1.0\"?>");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                        out.println("<wml>");
                        out.println("   <card id='cardP' title='Sitio' newcontext='true' ordered='true'>");
                        out.println("   <p align='left'>");
                        out.println("   Buscar por:<br/>");
                        out.println("   <a href='LocSrvSitio?dir=Sit&amp;xcat=Si'>Categoria</a><br/>");
                        out.println("   </p>");
                        out.println("   <p>");
                        out.println("   <a href='LocSrvRutaSitio'>Regresar</a>");
                        out.println("   </p>");

                        out.println("</card>");
                        out.println("</wml>");
                        out.close();
        }//frameBuscarSitioPor()


        private void frameBuscarSitioPorCateg(PrintWriter out, String dir){
                        out.println("<?xml version=\"1.0\"?>");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                        out.println("<wml>");
                        out.println("   <card id='cardP' title='Categoria' newcontext='true' ordered='true'>");
                        out.println("   <p align='left'>");
                        out.println("   Selecciona");
                        out.println("           <select title='Cat:' name='cbo_categoriaD' multiple='false'>");

                        try{
                        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_categoria");

                        if(rs1!=null){
                                String str1 = "";
                                String str2 = "";

                        while(rs1.next()){
                                str1 = rs1.getString(1).trim();
                                str2 = rs1.getString(2).trim();
                        out.println("                   <option value='" + str1
                                                        +"'
onpick='LocSrvSitio?dir=Sit&amp;xcat=Si&amp;categ="
                                                        + str1 + "'>"+str2+"</option>");
                        }
                        }
                        else
                        debug("rs1 fue null");
```

```
                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("                    </select>");
                    out.println("          </p>");
                    out.println("          <p>");
                    out.println("          <a href='LocSrvSitio'>Regresar</a>");
                    out.println("          </p>");

                    out.println("</card>");
                    out.println("</wml>");
                    out.close();
  }//frameBuscarSitioPorCateg()


   private void frameBuscarSitioPorCateg(PrintWriter out, String dir, int categ){
                    out.println("<?xml version=\"1.0\"?>");
                    out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                    out.println("<wml>");
                    out.println("          <card id='cardP' title='Categoria' newcontext='true' ordered='true'>");
                    out.println("          <p align='left'>");
                    out.println("          Selecciona");
                    out.println("                    <select title='Cat:' name='cbo_categoriaD' multiple='false'>");

                    try{
                    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio null, " + categ);

                    if(rs1!=null){
                              String str1 = "";
                              String str2 = "";

                   out.println("                              <option value='" + str1
                                                  +"' onpick='LocSrvSitio?dir=Sit&amp;xcat=Si&amp;categ="
                                                  + categ + "&amp;idSitio=0'>+ cercano</option>");
                              while(rs1.next()){
                              str1 = rs1.getString(1).trim();
                              str2 = rs1.getString(2).trim();
                    out.println("                              <option value='" + str1
                                                  +"'
onpick='LocSrvSitio?dir=Sit&amp;xcat=Si&amp;categ="
                                                  + categ + "&amp;idSitio="
                                                  + str1 + "'>"+str2+"</option>");
                    }
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("                    </select>");
                    out.println("          </p>");
                    out.println("          <p>");
                    out.println("          <a href='LocSrvSitio'>Regresar</a>");
                    out.println("          </p>");
```

```
                out.println("</card>");
                out.println("</wml>");
                out.close();
  }//frameBuscarSitioPorCateg()

        private void seleccionarDireccionSitio(HttpServletResponse res, int idSitio){
                String dir = "Sit";
                int deleg = -1;
                int col = -1;
                int calle = -1;
                int numero = -1;

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio " + idSitio);

                if(rs1!=null){
                if(rs1.next()){
                        calle = rs1.getInt(3);
                        numero = rs1.getInt(5);
                        col = rs1.getInt(6);
                        deleg = rs1.getInt(9);
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                debug("Redirect=LocSrvSitio?dir="
                                                        + dir + "&del="
                                                        + deleg + "&col="
                                                        + col + "&calle="
                                                        + calle + "&num="
                                                        + numero + "&masCercano=0&categ=0");
                        res.sendRedirect("LocSrvSitio?dir="
                                                        + dir + "&del="
                                                        + deleg + "&col="
                                                        + col + "&calle="
                                                        + calle + "&num="
                                                        + numero + "&masCercano=0&categ=0");

                //cierraBD();
                }catch(Exception e){
                debug("Error en seleccionarDireccionSitio():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }
        }


private void redireccionarSinDirSitio(HttpServletResponse res, int categ){
        try{
                String dir = "Sit";
                debug("Redirect=LocSrvSitio?dir="
                                + dir + "&masCercano=1&categ=" + categ
                                + "&del=0&col=0&calle=0&num=0");
                res.sendRedirect("LocSrvSitio?dir="
                                + dir + "&masCercano=1&categ=" + categ
                                + "&del=0&col=0&calle=0&num=0");
        }catch(Exception e){
                debug("Error en redireccionarSinDirSitio():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
        }
}
```

```
 private ResultSet ejecutaQuery(String sp){
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
           }catch(Exception e){
              debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return rs;
        }

        private int ejecutaUpdate(String sp){
                int filasMod = -1;
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                filasMod = stmt.executeUpdate(sp);
           }catch(Exception e){
              debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return filasMod;
        }

        private void cierraBD(){
                try{
                        rs.close();
                        stmt.close();
                        con.close();
                }catch(Exception e){
              debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
        }

        private void debug(String str){
           System.out.println(str);
        }
}
```

LocSrvBuscarRutaSitio.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import Dijkstra;

public class LocSrvBuscarRutaSitio extends HttpServlet
{
        private Connection con = null;
        private Statement stmt = null;
        private ResultSet rs = null;

        public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException
```

```
{
	try{
		PrintWriter out = res.getWriter();
		res.setContentType("text/vnd.wap.wml");

		HttpSession sesion = req.getSession(false); // cambie true -> false

		Integer delOri = new Integer(-1);
		Integer delSit = new Integer(-1);
		Integer colOri = new Integer(-1);
		Integer colSit = new Integer(-1);
		Integer calleOri = new Integer(-1);
		Integer calleSit = new Integer(-1);
		Integer numOri = new Integer(-1);
		Integer numSit = new Integer(-1);
		Integer masCercanoSit = new Integer(-1);
		Integer categSit = new Integer(-1);

		if(sesion.getValue("delOri")!=null){
			delOri = (Integer)sesion.getValue("delOri");
		}
		if(sesion.getValue("delSit")!=null){
			delSit = (Integer)sesion.getValue("delSit");
		}
		if(sesion.getValue("colOri")!=null){
			colOri = (Integer)sesion.getValue("colOri");
		}
		if(sesion.getValue("colSit")!=null){
			colSit = (Integer)sesion.getValue("colSit");
		}
		if(sesion.getValue("calleOri")!=null){
			calleOri = (Integer)sesion.getValue("calleOri");
		}
		if(sesion.getValue("calleSit")!=null){
			calleSit = (Integer)sesion.getValue("calleSit");
		}
		if(sesion.getValue("numOri")!=null){
			numOri = (Integer)sesion.getValue("numOri");
		}
		if(sesion.getValue("numSit")!=null){
			numSit = (Integer)sesion.getValue("numSit");
		}
		if(sesion.getValue("masCercanoSit")!=null){
			masCercanoSit = (Integer)sesion.getValue("masCercanoSit");
		}
		if(sesion.getValue("categSit")!=null){
			categSit = (Integer)sesion.getValue("categSit");
		}

		String txt_usr = "";
		if(sesion.getValue("txt_usr")!=null){
			txt_usr = (String)sesion.getValue("txt_usr");
		}

		String txt_pwd = "";
		if(sesion.getValue("txt_pwd")!=null){
			txt_pwd = (String)sesion.getValue("txt_pwd");
		}

		/*
		debug("-------");
		debug("delOri=" + delOri);
		debug("delSit=" + delSit);
		debug("colOri=" + colOri);
```

```
                        debug("colSit=" + colSit);
                        debug("calleOri=" + calleOri);
                        debug("calleSit=" + calleSit);
                        debug("numOri=" + numOri);
                        debug("numSit=" + numSit);
                        */
/*
                        int idNodoOrigen = encontrarNodoCercano(calleOri.intValue() , numOri.intValue());
                        int idNodoDestino = encontrarNodoCercano(calleSit.intValue(), numSit.intValue());
                        Dijkstra d = new Dijkstra();
                        d.setNodoOrig(idNodoOrigen);
                        d.setNodoDest(idNodoDestino);
                        d.ejecutarAlg();
*/
                        Dijkstra d = null;
                        if(masCercanoSit.intValue()==0)
                                d = new Dijkstra(calleOri.intValue() , numOri.intValue(), calleSit.intValue(),
numSit.intValue());
                        else
                                d = new Dijkstra(calleOri.intValue() , numOri.intValue(), categSit.intValue());

                        String rutaOptima[] = new String[d.getNumNodosRutaOptima()];
                        rutaOptima = d.getNombresCalles();
                        out.println("<?xml version=\"1.0\"?>");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML1.1//EN\"
\"http://www.wapforum.org/DTD/wml1.1.xml\">");
                        out.println("<wml>");
                        out.println("<card id='cardInfo' title='Ruta Optima'>");

                        if(masCercanoSit.intValue()!=0){
                        try{ //obtiene la categoria elegida
                        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_categoria " + categSit);

                        if(rs1!=null){
                                if(rs1.next())
                                                out.println("<p align='left' mode='nowrap'><small>El
sitio de la categoria " + rs1.getString(2).trim() + "</small></p>");
                        }
                        else
                                debug("rs1 fue null");

                        rs1.close();
                        }catch(Exception e){
                        debug("Error en doGet():" + e.toString() + "_Msg=" + e.getMessage());
                        e.printStackTrace(System.out);
                        }

                        try{
                        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio " + d.idSitioMasCercano + ",
null");

                        if(rs1!=null){
                                if(rs1.next()){
                                        String strSitio = rs1.getString(2).trim();
                                        String strCalle = rs1.getString(4).trim();
                                        String strNumer = rs1.getString(5).trim();
                                        String strCol = rs1.getString(7).trim();
                                        String strTelef = rs1.getString(8).trim();
                                        String strDeleg = rs1.getString(10).trim();

                                                out.println("<p align='left'
mode='nowrap'><small>mas cercano es " + strSitio + ", ubicado en:</small></p>");
                                                out.println("<p align='left' mode='nowrap'><small>"
+ strCalle + " " + strNumer + " Col. " + strCol + "</small></p>");
```

```
                                                        out.println("<p                                    align='left'
mode='nowrap'><small>Del. " + strDeleg + ". Tel: " + strTelef + "</small></p>");
                                                }
                        }
                        else
                                debug("rs1 fue null");

                        rs1.close();
                        }catch(Exception e){
                        debug("Error en doGet():" + e.toString() + "_Msg=" + e.getMessage());
                        e.printStackTrace(System.out);
                        }

                                out.println("                   <p align='left'>");
                                out.println("                           <a      href='#cardDirec'      title='DO'>Ver
Ruta</a>");
                                out.println("                   </p>");
                                out.println("</card>");
                                out.println("<card id='cardDirec' title='Ruta Optima'>");
                        }

                        int nume = 1;
                        for(int i=0; i<rutaOptima.length; i++){
                                if(String.valueOf(rutaOptima[i]).indexOf("(") == -1){
                                        out.println("<p  align='left'  mode='nowrap'><small>" + rutaOptima[i] +
"</small></p>");
                                }else{
                                        if(i==0){

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i+1].substring(0,rutaOptima[i+1].i
ndexOf("(")))!=0){
                                                        out.println("<p  align='left'  mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                        }
                                        else if(i==1){
                                                out.println("<p     align='left'     mode='nowrap'><small>"     +
(nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                        }
                                        else if(i==(rutaOptima.length - 1)){

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i-1].substring(0,rutaOptima[i-
1].indexOf("(")))!=0){
                                                        out.println("<p  align='left'  mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                        }
                                        else{
                                                //if(rutaOptima[i].compareTo(rutaOptima[i-1])!=0)

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i-1].substring(0,rutaOptima[i-
1].indexOf("(")))!=0){
                                                        out.println("<p  align='left'  mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                        }
                                }
                        }

                        //out.println("<p align='center'><small>Nodo Orig: " + idNodoOrigen + "</small></p>");
                        //out.println("<p align='center'><small>Nodo Dest: " + idNodoDestino + "</small></p>");

                        if(masCercanoSit.intValue()!=0){
                                out.println("                   <p align='left'>");
```

```
                                    out.println("                                   <a      href='#cardInfo'      title='DO'>Ver
Sitio</a>");
                                    out.println("                   </p>");
                            }


                            out.println("                   <p align='left'>");
                            out.println("                           <br/><a href='LocSrvRutaSitio' title='NewD'>Cambiar
Dir o Sit</a>");
                            out.println("                   </p>");

                            out.println("                   <p align='left'>");
                            out.println("                           <a href='LocSrvPrincipal?nvaBusq=S&amp;txt_usr="+
txt_usr +"&amp;txt_pwd="+ txt_pwd +"&URLRand="+ Math.random() +"' title='NewB'>Nueva Busqueda</a>");
                            out.println("                   </p>");

                            out.println("<p align='center'><small><br/>Visita cic.ipn.mx</small></p>");
                            out.println("</card>");
                            out.println("</wml>");
                            out.close();

                    }catch(Exception e){
                    debug("Error en doGet():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }
 }


        private int encontrarNodoCercano(int idcalle, int numero){
                int nodoCercano = -1;

           int nodo1 = -1;
           int nodo2 = -1;
           int flujo1 = -1;
           int flujo2 = -1;
           int numIni = -1;
           int numFin = -1;
           int trafico = -1;

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_buscaNodos " + idcalle + ", " + numero);

                if(rs1!=null){
                if(rs1.next()){
                        nodo1 = rs1.getInt(1);
                        nodo2 = rs1.getInt(2);
                        flujo1 = rs1.getInt(3);
                        flujo2 = rs1.getInt(4);
                        numIni = rs1.getInt(5);
                        numFin = rs1.getInt(6);
                        trafico = rs1.getInt(7);
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en encontrarNodoCercano():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }
```

```
                if(trafico == 0){
                        // mandar mensaje de que la cuadra se encuentra bloqueada,
                        // por lo que tiene que situarse en otra posicion
                }
                else if((flujo1 == 0) && (flujo2 == 0)){
                        // mandar mensaje de error,
                        // por lo que tiene que situarse en otra posicion
                }
                else if(flujo1 == 0){
                        nodoCercano = nodo2;
                }
                else if(flujo2 == 0){
                        nodoCercano = nodo1;
                }
                else{
                        if((numero >= numIni) && (numero <= (numIni + (numFin-numIni)/2)))
                                nodoCercano = nodo1;
                        else
                                nodoCercano = nodo2;
                }

                return nodoCercano;
        }

        private ResultSet ejecutaQuery(String sp){
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
           }catch(Exception e){
              debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return rs;
        }
        private void debug(String str){
           System.out.println(str);
        }
}
```

Dijkstra.java

```
import java.sql.*;
import java.util.Vector;
import java.util.Enumeration;

public class Dijkstra
{
        private Connection con = null;
        private Statement stmt = null;
        private ResultSet rs = null;

   private int MAXNODOS;
   private int MAX;
   int DIJKSTRA = 1;

   private int nodoInicial=0;              // nodo inicial del grafo
   private int nodoOrig = 0;
   private int nodoDest = 0;
```

```
private int idNodos[];                                  //almacena los identificadores de cada nodo
private int idNodosRutaOptima[];    //almacena los identificadores de cada nodo
private int numNodosRutaOptima;
public String callesRutaOptima[];    //almacena los nombres de las calles de la ruta optima
private int peso[][];                                   //almacena los pesos de un nodo a otro

// informacion del grafo al ejecutar el algoritmo
boolean rutaMasCorta[][];
int dist[];
int distFinal[];
boolean cambioDist[];                                   // indica cambio de distancia durante el algoritmo

private String mensaje = new String("");

private int numcambioDist =0;
private int colindantes=0;
int paso=0;                                                     // numero de iteracion

// informacion usada por el algoritmo para encontrar el siguiente nodo con minima distancia
int mindist;
int minnodo;
int minstart;
int minend;
int numnodos=0;                                         // numero de nodos
int nodo1, nodo2;                                       // numero de nodos envueltos en la accion

boolean ejecutando = false;
int algoritmo;

int o_idcalle = 0;
int o_numero = 0;
int d_idcalle = 0;
int d_numero = 0;
int categ=0;
Vector sitios = null;
public int idSitioMasCercano = 0;

public Dijkstra() {
        debug("Dijkstra()");
        this.MAXNODOS = getNumeroNodos();
                this.MAX = this.MAXNODOS + 1;
                //debug("MAXNODOS=" + MAXNODOS);
                idNodos = new int[MAX];
                idNodosRutaOptima = new int[MAX];
                peso = new int[MAX][MAX];
                rutaMasCorta = new boolean[MAX][MAX];
                dist = new int[MAX];
                distFinal = new int[MAX];
                cambioDist = new boolean[MAX];
                numNodosRutaOptima = 0;

        cargaGrafo();
        }

public Dijkstra(int o_idcalle, int o_numero, int d_idcalle, int d_numero) {
        debug("Dijkstra(args), v1.103");

        this.MAXNODOS = getNumeroNodos();
                this.MAX = this.MAXNODOS + 1;
                //debug("MAXNODOS=" + MAXNODOS);
                idNodos = new int[MAX];
                idNodosRutaOptima = new int[MAX];
                peso = new int[MAX][MAX];
```

```
            rutaMasCorta = new boolean[MAX][MAX];
            dist = new int[MAX];
            distFinal = new int[MAX];
            cambioDist = new boolean[MAX];
            numNodosRutaOptima = 0;

cargaGrafo();

this.o_idcalle = o_idcalle;
this.o_numero = o_numero;
this.d_idcalle = d_idcalle;
this.d_numero = d_numero;

setNodoOrig(encontrarNodoCercano(o_idcalle, o_numero));
setNodoDest(encontrarNodoCercano(d_idcalle, d_numero));

debug("NO="+this.nodoOrig);
debug("ND="+this.nodoDest);


if((this.nodoOrig == -1) || (this.nodoDest == -1)){
            callesRutaOptima = new String[2];
            callesRutaOptima[0] = "";
            callesRutaOptima[1] = "";
}

if(this.nodoOrig == -1){
            callesRutaOptima[numNodosRutaOptima++] = "La direccion origen se encuentra bloqueada.";
}

if(this.nodoDest == -1){
            callesRutaOptima[numNodosRutaOptima++] = "La direccion destino se encuentra bloqueada.";
}

if((this.nodoOrig != -1) && (this.nodoDest != -1)){
            ejecutarAlg();
}
}// construc con 4 args

public Dijkstra(int o_idcalle, int o_numero, int categ) {
debug("Dijkstra(args), v1.104");

this.MAXNODOS = getNumeroNodos();
            this.MAX = this.MAXNODOS + 1;
            //debug("MAXNODOS=" + MAXNODOS);
            idNodos = new int[MAX];
            idNodosRutaOptima = new int[MAX];
            peso = new int[MAX][MAX];
            rutaMasCorta = new boolean[MAX][MAX];
            dist = new int[MAX];
            distFinal = new int[MAX];
            cambioDist = new boolean[MAX];
            numNodosRutaOptima = 0;

cargaGrafo();

this.o_idcalle = o_idcalle;
this.o_numero = o_numero;
this.categ = categ;

setNodoOrig(encontrarNodoCercano(o_idcalle, o_numero));
debug("NO="+this.nodoOrig);

sitios = new Vector();
```

```
            try{
            ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio null, " + categ);

            if(rs1!=null){
                        String str1 = "";
                        String str3 = "";
                        String str5 = "";

                        while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str3 = rs1.getString(3).trim();
                        str5 = rs1.getString(5).trim();

                        sitios.addElement(new String(str1 + "!" + str3 + "!" + str5));
            }
            }
            else
            debug("rs1 fue null");

            rs1.close();
            //cierraBD();
            }catch(Exception e){
            debug("Error en Dijkstra(3 args):" + e.toString() + "_Msg=" + e.getMessage());
            e.printStackTrace(System.out);
            }


if(this.nodoOrig == -1){
            callesRutaOptima = new String[1];
            callesRutaOptima[0] = "";
}

if(this.nodoOrig == -1){
            callesRutaOptima[numNodosRutaOptima++] = "La direccion origen se encuentra bloqueada.";
}

if(this.nodoOrig != -1){
            ejecutarAlg();
}
}// construc con 3 args



public int getNumNodosRutaOptima(){
            return numNodosRutaOptima;
}

public int[] getIdNodosRutaOptima(){
            return idNodosRutaOptima;
}

private int getNumeroNodos(){
   try{
   ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_numeronodos");

      if(rs1!=null){
            if(rs1.next())
            this.numnodos = rs1.getInt(1);
   }
      else
            debug("rs1 fue null");

      rs1.close();
      //cierraBD();
```

```
    }catch(Exception e){
        debug("Error en getNumeroNodos():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }
    return this.numnodos;
}

private int setIdNumNodos(){
    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_nodo");
    int id=0;

        if(rs1!=null){
            while(rs1.next())
            idNodos[id++] = rs1.getInt(1);
    }
        else
            debug("rs1 fue null");

        rs1.close();
        //cierraBD();
    }catch(Exception e){
        debug("Error en setIdNumNodos():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }
    return this.numnodos;
}


private void cargaGrafo(){
            inicializar();

            for(int i=0; i<MAXNODOS; i++) {
                        for (int j=0; j<MAXNODOS;j++)
                        peso[i][j]=0;
            }

    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_flujonodos");

                        int n1;
                        int n2;
                        int f1;
                        int f2;
                        int t;

    if(rs1!=null){
          while(rs1.next()){
            n1 = rs1.getInt(1);
            n2 = rs1.getInt(2);
            f1 = rs1.getInt(3);
            f2 = rs1.getInt(4);
            t = rs1.getInt(5);

            if((f2 == 0) || (t == 0))
                        peso[n1][n2] = 0;
            else
                        peso[n1][n2] = (int)((9*(11-f2)) + (25*t));

            if((f1 == 0) || (t == 0))
                        peso[n2][n1] = 0;
            else
                        peso[n2][n1] = (int)((9*(11-f1)) + (25*t));
```

```
                    //debug("peso[" + n1 + "][" + n2 + "] = " + peso[n1][n2]);
                    //debug("peso[" + n2 + "][" + n1 + "] = " + peso[n2][n1]);
                  }
               }
               else
                    debug("rs1 fue null");

        rs1.close();
               //cierraBD();
               ejecutarPaso();
            }catch(Exception e){
               debug("Error en cargaGrafo():" + e.toString() + "_Msg=" + e.getMessage());
               e.printStackTrace(System.out);
            }
         }


   public void ejecutarAlg() {
                    inicializarAlg();
                    ejecutando = true;
                    for(int i=0; i<numnodos; i++){
                              ejecutarSiguientePasoAlg();
                    }
   }
   /*
   public void ejecutarPrimerPasoAlg() {
          // le permite pasar por el algoritmo
                    inicializarAlg();
                    ejecutando = true;
                    ejecutarSiguientePasoAlg();
   }
          */
   public void inicializarAlg() {
                    inicializar();

                    for(int i=0; i<MAXNODOS; i++) {
                              dist[i]=-1;
                              distFinal[i]=-1;
                    }
      dist[nodoInicial]=0;
                    distFinal[nodoInicial]=0;
                    paso=0;
   }

          public void inicializar(){
                    for (int i=0;i<MAXNODOS;i++) {
                              for (int j=0; j<MAXNODOS;j++)
                              rutaMasCorta[i][j]=false;
                    }

                    ejecutando = false;
                    algoritmo=DIJKSTRA;
          }

          private int encontrarNodoCercano(int idcalle, int numero){
                    int nodoCercano = -1;

              int nodo1 = -1;
              int nodo2 = -1;
              int flujo1 = -1;
              int flujo2 = -1;
              int numIni = -1;
              int numFin = -1;
              int trafico = -1;
```

```
                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_buscaNodos " + idcalle + ", " + numero);

                if(rs1!=null){
                if(rs1.next()){
                        nodo1 = rs1.getInt(1);
                        nodo2 = rs1.getInt(2);
                        flujo1 = rs1.getInt(3);
                        flujo2 = rs1.getInt(4);
                        numIni = rs1.getInt(5);
                        numFin = rs1.getInt(6);
                        trafico = rs1.getInt(7);
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en encontrarNodoCercano():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }

                if(trafico == 0){
                        // mandar mensaje de que la cuadra se encuentra bloqueada,
                        // por lo que tiene que situarse en otra posicion
                        nodoCercano = -1;
                }
                else if((flujo1 == 0) && (flujo2 == 0)){
                        // mandar mensaje de error,
                        // por lo que tiene que situarse en otra posicion
                        nodoCercano = -1;
                }
                else if(flujo1 == 0){
                        nodoCercano = nodo1;
                }
                else if(flujo2 == 0){
                        nodoCercano = nodo2;
                }
                else{
                        if((numero >= numIni) && (numero <= (numIni + (numFin-numIni)/2)))
                                nodoCercano = nodo1;
                        else
                                nodoCercano = nodo2;
                }

                return nodoCercano;
        }

public String[] getNombresCalles(){
        return callesRutaOptima;
}

public void setNodoInicial(int nodoInicial){
        this.nodoInicial = nodoInicial;
}

public void setNodoOrig(int nodoOrig){
        this.nodoOrig = nodoOrig;
        setNodoInicial(nodoOrig);
}
```

```
public void setNodoDest(int nodoDest){
	this.nodoDest = nodoDest;
}

private void imprimirRuta(){
	debug("Ruta:");
	int nodoAnt = nodoDest;

	/*
	for(int i=0; i<MAXNODOS; i++) {
			for (int j=0; j<MAXNODOS;j++){
				debug(i + "," + j + "=" + rutaMasCorta[i][j]);
			}
		}
		*/

		for(int i=0; i<MAXNODOS; i++) {
			for (int j=0; j<MAXNODOS;j++){
		if(rutaMasCorta[j][nodoAnt] == true){
			//debug("nodo " + nodoAnt);
			nodoAnt = j;
			if(nodoAnt == nodoOrig){
				i = MAXNODOS;
			}
			break;
		}
	}

		}
}

public void setRutaOptima(){
	debug("setRutaOptima:");

	int nodoAnt = nodoDest;
	idNodosRutaOptima[++numNodosRutaOptima] = nodoAnt;
	/*
	for(int i=0; i<MAXNODOS; i++) {
			for (int j=0; j<MAXNODOS;j++){
				debug(i + "," + j + "=" + rutaMasCorta[i][j]);
			}
		}
		*/

		for(int i=0; i<MAXNODOS; i++) {
			for (int j=0; j<MAXNODOS;j++){
		if(rutaMasCorta[j][nodoAnt] == true){
			//debug("nodo " + nodoAnt);
			nodoAnt = j;
			++numNodosRutaOptima;
			idNodosRutaOptima[numNodosRutaOptima] = nodoAnt;
			if(nodoAnt == nodoOrig){
				i = MAXNODOS;
			}
			break;
		}
	}
		}

		++numNodosRutaOptima;

		debug("numNodosRutaOptima-"+numNodosRutaOptima);

		for(int i=numNodosRutaOptima-1; i>0; i--)  //imprimiendo los idnodos de la ruta optima
```

```
                debug("XXXXXXXXXXXXX.-"+idNodosRutaOptima[i]);


            callesRutaOptima = new String[numNodosRutaOptima];
            int nOrig = 0;
            int nDest = 0;

            int idCalle = 0;
            String nomCalle = "";
            String nomColonia = "";
            int noInic = 0;
            int noFinal = 0;
            String sentido = "";
            String trafico = "";

            int cont = 0;

            for(int i=numNodosRutaOptima; i>0; i--){
                try{
                    ResultSet rs1;

                    if(i == numNodosRutaOptima){
                        nDest = idNodosRutaOptima[i-1];
                        rs1    =    ejecutaQuery("LOCALIZACION..loc_sp_nombreCalleOrig    "    +
o_idcalle + ", " + o_numero + ", " + nDest);
                    }
                    else if(i == 1){
                        nOrig = idNodosRutaOptima[i];
                        rs1    =    ejecutaQuery("LOCALIZACION..loc_sp_nombreCalleDest    "    +
d_idcalle + ", " + d_numero + ", " + nOrig);
                    }
                    else{
                        nOrig = idNodosRutaOptima[i];
                        nDest = idNodosRutaOptima[i-1];
                        rs1 = ejecutaQuery("LOCALIZACION..loc_sp_nombreCalle " + nOrig + ", "
+ nDest);
                    }//else

                    if(rs1!=null){
                if(rs1.next()){
                                idCalle = rs1.getInt(1);
                                nomCalle = rs1.getString(2).trim();
                                nomColonia = rs1.getString(3).trim();
                                noInic = rs1.getInt(4);
                                noFinal = rs1.getInt(5);
                                sentido = rs1.getString(6).trim();
                                trafico = rs1.getString(7).trim();

                                callesRutaOptima[cont++] = nomCalle + " (" + sentido + ")";
                }
                }
                else{
                debug("rs1 fue null");
                callesRutaOptima[cont++] = "Error";
            }

            debug("X--X--X.-"+callesRutaOptima[cont - 1]);

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en setRutaOptima():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }
```

```
                }//for
    }//setRutaOptima()

    public void ejecutarSiguientePasoAlg() {
            // Calcula un paso mas del algoritmo (encuentra un camino mas corto al siguiente nodo).
            //debug("ejecutarSiguientePasoAlg()");
            distFinal[minend]=mindist;
                    rutaMasCorta[minstart][minend]=true;

                    if(categ==0){
                    if ((minend == nodoDest) && (ejecutando == true)) {
                                    imprimirRuta();
                                    setRutaOptima();
                                    ejecutando = false;
                                    return;
                            }
                    }
                    else{
                            Enumeration e = sitios.elements();
                            String strSit = "";
                            int nodod = -1;
                            while(e.hasMoreElements()){
                                    strSit = (String)e.nextElement();
                                    nodod = Integer.parseInt(     String.valueOf(       encontrarNodoCercano(
                                                                                        Integer.parseInt(
strSit.substring(strSit.indexOf("!")+1, strSit.lastIndexOf("!"))) ,

            Integer.parseInt(strSit.substring(strSit.lastIndexOf("!")+1, strSit.length()))        )            )            );
                                    if ((minend == nodod) && (ejecutando == true) ) {
                                    setNodoDest(nodod);
                                    idSitioMasCercano = Integer.parseInt(strSit.substring(0, strSit.indexOf("!")));
                                    this.d_idcalle       =       Integer.parseInt(       strSit.substring(strSit.indexOf("!")+1,
strSit.lastIndexOf("!")) );
                                    this.d_numero     =     Integer.parseInt(     strSit.substring(strSit.lastIndexOf("!")+1,
strSit.length()) );
                                    debug("NODODEST--"+this.nodoDest + " idSitio+Cercano--"+idSitioMasCercano);
                                            imprimirRuta();
                                            setRutaOptima();
                                            ejecutando = false;
                                            return;
                                    }
                            }
                    }

                    paso++;
                    ejecutarPaso();

    }

    //myp
    public void ejecutarPaso() {
       mindist         = 0;
                    minstart = 0;
                    minend = MAXNODOS;

       for(int i=0; i<MAXNODOS; i++)
            cambioDist[i]=false;

       numcambioDist=0;
       colindantes=0;


                    for (int i=0; i<numnodos; i++)
```

```
                        for (int j=0; j<numnodos; j++)
                        if (peso [i][j]>0) {
                        // si el algoritmo se esta ejecutando entonces hacer el siguiente paso para esta arista
                        if (ejecutando)
                                iniciarPaso(i, j);
                        }


                if (ejecutando)
                        finalizarPaso();

    }

        public void iniciarPaso(int i, int j) {
        // mas algoritmos pueden ser aniadidos
                if (algoritmo==DIJKSTRA)
                iniciarPasoDijkstra(i, j);
    }
    public void finalizarPaso() {
        // mas algoritmos pueden ser aniadidos
                if (algoritmo==DIJKSTRA)
                finalizarPasoDijkstra();
    }

    public void iniciarPasoDijkstra(int i, int j) {
        // checar que arista entre nodo i y nodo j esta cerca de la arista s para
        //escoger durante este paso del algoritmo
        // checar si el nodo j tiene la siguiente minima distancia al nodo_inicial
                if ( (distFinal[i] != -1) && (distFinal[j] == -1) ) {
                        if ( (dist[j]==-1) || (dist[j]>=(dist[i]+peso[i][j])) ) {
        if ( (dist[i]+peso[i][j])<dist[j] ) {
                cambioDist[j]=true;
                numcambioDist++;
        }
                        dist[j] = dist[i]+peso[i][j];
                        if ( (mindist==0) || (dist[j]<mindist) ) {
                        mindist=dist[j];
                        minstart=i;
                        minend=j;
                        }
                        }
                }
    }

    public void finalizarPasoDijkstra() {
                if ((ejecutando) && (mindist==0))
                {
                        int nalcanzable = 0;

                        for (int i=0; i<numnodos; i++)
                        if (distFinal[i] > 0)
                                        nalcanzable++;

                        if (nalcanzable == 0){
                                debug("El algoritmo ha terminado, no hay nodos alcanzables desde el nodo
inicial.");
        }
        else if (nalcanzable< (numnodos-1)){
                        debug("El algoritmo ha terminado, No hay caminos del nodo_inicial a ningun otro nodo.");
                        callesRutaOptima = new String[1];
                        callesRutaOptima[0] = "No se encontraron rutas disponibles.";
                        numNodosRutaOptima = 1;
                        }else{
                        debug("El algoritmo ha terminado.");
```

```
                    }

                    }
    }

        private ResultSet ejecutaQuery(String sp){
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
           }catch(Exception e){
              debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return rs;
        }

        private void cierraBD(){
                try{
                        rs.close();
                        stmt.close();
                        con.close();
                }catch(Exception e){
              debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
        }

    private void debug(String texto){
        System.out.println(texto);
    }

    public void cargarEjemplo() {
        inicializar();
                numnodos=10;

                for(int i=0; i<MAXNODOS; i++) {
                        for (int j=0; j<MAXNODOS;j++)
                        peso[i][j]=0;
                }

        peso[0][2]=13;      peso[2][0]=13;
                peso[0][4]=16;      peso[4][0]=16;
                peso[0][5]=8;       peso[5][0]=8;
                peso[1][3]=6;       peso[3][1]=6;
                peso[1][5]=10;      peso[5][1]=10;
                peso[2][3]=14;      peso[3][2]=14;
                peso[2][5]=11;      peso[5][2]=11;
                peso[3][4]=5;       peso[4][3]=5;
                peso[3][5]=17;      peso[5][3]=17;
                peso[4][5]=7;       peso[5][4]=7;


                ejecutarPaso();
    }

}
```

LocSrvPrincipal.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;
import java.util.*;

public class LocSrvPrincipal extends HttpServlet{
	Connection con = null;
	Statement stmt = null;
	ResultSet rs = null;

	public void init(ServletConfig config) throws ServletException
	{
	super.init(config);
	}

	public void doGet(HttpServletRequest req, HttpServletResponse res) throws
	ServletException, IOException
	{
		//debug("req.getParameter('txt_usr')="+req.getParameter("txt_usr"));
		if((req.getParameter("txt_usr")!=null)&&(!(req.getParameter("txt_usr").equalsIgnoreCase("")))){
			doPost(req, res);
			return;
      }

		PrintWriter out = res.getWriter();
		res.setContentType("text/vnd.wap.wml");

		out.println("<?xml version='1.0'?>");
		out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
		out.println("<wml>");
		out.println("	<card id='cardP' title='Localización' newcontext='true' ordered='true'>");
		out.println("		<p align='center'>");
		out.println("			Favor de ingresar:");
		out.println("		</p>");
		out.println("		<p align='left'>");
		out.println("			<input name='txt_usr' title='Usuario' type='text' format='a*a'
emptyok='false' maxlength='8' />");
		out.println("			<input name='txt_pwd' title='Password' type='password'
format='a*a' emptyok='false' maxlength='8' />");
		out.println("			<a
href='LocSrvPrincipal?txt_usr=$(txt_usr)&amp;txt_pwd=$(txt_pwd)&amp;URLRand="+ Math.random() +"'>Enviar</a>");
		out.println("		</p>");
		out.println("	</card>");
		out.println("</wml>");
		out.close();
	}//doGet()

	private void debug(String str){
	   System.out.println(str);
	}

	public void doPost(HttpServletRequest req, HttpServletResponse res) throws
	ServletException, IOException
	{
		//Se leen los parametros de la pantalla de login
		String txt_usr = "";
		if(req.getParameter("txt_usr")!=null){
```

```
                txt_usr = (String)req.getParameter("txt_usr");
            }
            String txt_pwd = "";
            if(req.getParameter("txt_pwd")!=null){
                txt_pwd = (String)req.getParameter("txt_pwd");
            }
            String nvaBusq = "";
            if(req.getParameter("nvaBusq")!=null){
                nvaBusq = (String)req.getParameter("nvaBusq");
            }

            // se valida usr y pwd
            int idUsuario = validarUsuario(txt_usr, txt_pwd);
            debug("Usr:"+ txt_usr +"\nPwd:"+ txt_pwd +"\nidUsr:"+ idUsuario );

            PrintWriter out = res.getWriter();
            res.setContentType("text/vnd.wap.wml");

            if(idUsuario < 1) //Si el usuario y/o password es incorrecto
            {
                out.println("<?xml version='1.0'?>");
                out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
                out.println("<wml>");
                out.println("  <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
                out.println("    <p align='left'>");
                out.println("      Usuario y/o Pwd incorrecto(s).");
                out.println("    </p>");
                out.println("    <p align='left'>");
                out.println("      <a href='LocSrvPrincipal?URLRand="+ Math.random()
+"' title='Aceptar'>Aceptar</a>");
                out.println("    </p>");
                out.println("    <p align = 'left'>");
                //out.println("      Usr:"+ txt_usr +"-Pwd:"+ txt_pwd +"-idUsr:"+
idUsuario );
                out.println("    </p>");
                out.println("  </card>");
                out.println("</wml>");
                out.close();
            }
            else
            {
                HttpSession sesion = req.getSession(true);
                sesion.invalidate();
                sesion = req.getSession(true);

                if(nvaBusq.equalsIgnoreCase("S")){
                    String sqlString = "localizacion.dbo.loc_spd_busqueda " + idUsuario;
                    debug(sqlString);
                    ejecutaUpdate(sqlString);
                }

                sesion.putValue("idUsr", new Integer(idUsuario));
                sesion.putValue("txt_usr", new String(txt_usr));
                sesion.putValue("txt_pwd", new String(txt_pwd));

                // si son validos mostrar ultima pantalla

                // Leer loc_busqueda, si stsDir != 0 (hay busqueda pendiente) -> Realizar lo necesario
(Grabar variables de sesion, redirigir al URL almacenado, )
            int stsDir = 0;
            String URL1 = "";
            String URL2 = "";
                try
```

```
{
ResultSet rs1 = ejecutaQuery("LOCALIZACION.dbo.loc_spr_busqueda " + idUsuario);

if(rs1!=null){
if(rs1.next()){
        stsDir = Integer.parseInt(rs1.getString(2).trim());
        URL1 = rs1.getString(3).trim();
        URL2 = rs1.getString(4).trim();
}
}
else
debug("rs1 fue null");

rs1.close();
//cierraBD();
}catch(Exception e){
debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
e.printStackTrace(System.out);
}

if(stsDir == 1)
{
        res.sendRedirect(URL1);
}

if(stsDir >= 2)
{
        //cargar variables de sesion con la ubicacion de la primer direccion
        //      LocSrvDireccion?dir=Or&del=7&col=5&calle=7&num=222
        StringTokenizer stk = new StringTokenizer(URL1.substring(URL1.indexOf("?")+1,
URL1.length()), "&");
        String dir = (String)stk.nextElement();
        String parte = "";

        dir = dir.substring(dir.indexOf("=")+1, dir.length());

        while(stk.hasMoreElements())
        {
                parte = (String)stk.nextElement();
                debug("name="+parte.substring(0,parte.indexOf("=")) + dir);
                debug("value="+parte.substring(parte.indexOf("=")+1, parte.length()));
                sesion.putValue(parte.substring(0,parte.indexOf("=")) + dir, new
Integer(parte.substring(parte.indexOf("=")+1, parte.length())));
        }

        if(stsDir == 2)
        {
                if(dir.equalsIgnoreCase("Or"))
                        res.sendRedirect("LocSrvRutaDosPuntos?Rand="+
Math.random());
                if(dir.equalsIgnoreCase("Ori"))
                        res.sendRedirect("LocSrvRutaSitio?Rand="+ Math.random());
        }
}

if(stsDir == 3)
{
        res.sendRedirect(URL2);
}

if(stsDir >= 4)
{
        //cargar variables de sesion con la ubicacion de la segunda direccion
        //      LocSrvDireccion?dir=De&del=7&col=8&calle=9&num=1
```

```
                                        StringTokenizer stk = new StringTokenizer(URL2.substring(URL2.indexOf("?")+1,
URL2.length()), "&");
                                        String dir = (String)stk.nextElement();
                                        String parte = "";

                                        dir = dir.substring(dir.indexOf("=")+1, dir.length());

                                        while(stk.hasMoreElements())
                                        {
                                                parte = (String)stk.nextElement();
                                                debug("name="+parte.substring(0,parte.indexOf("=")) + dir);
                                                debug("value="+parte.substring(parte.indexOf("=")+1, parte.length()));
                                                sesion.putValue(parte.substring(0,parte.indexOf("=")) + dir, new
Integer(parte.substring(parte.indexOf("=")+1, parte.length())));
                                        }

                                        if(stsDir == 4)
                                        {
                                                if(dir.equalsIgnoreCase("De"))
                                                        res.sendRedirect("LocSrvRutaDosPuntos?Rand="+
Math.random());
                                                if(dir.equalsIgnoreCase("Sit"))
                                                        res.sendRedirect("LocSrvRutaSitio?Rand="+ Math.random());
                                        }

                                }


                                out.println("<?xml version='1.0'?>");
                                out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
                                out.println("<wml>");
                                out.println("   <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
                                out.println("           <p align='left'>");
                                out.println("                   Buscar ruta de:");
                                out.println("           </p>");
                                out.println("           <p align='left'>");
                                out.println("                   <a href='LocSrvRutaDosPuntos?Rand="+
Math.random() +"' title='R2P'>Dos Puntos</a>");
                                out.println("           </p>");
                                out.println("           <p align='left'>");
                                out.println("                   <a href='LocSrvRutaSitio?Rand="+ Math.random()
+"' title='RS'>Sitio de Interes</a>");
                                out.println("           </p>");
                                out.println("           <p align = 'left'>");
                                //out.println("                   Usr:"+ txt_usr +"-Pwd:"+ txt_pwd +"-idUsr:"+
idUsuario );
                                out.println("           </p>");
                                out.println("   </card>");
                                out.println("</wml>");
                                out.close();
                        }
        }

        private int validarUsuario(String usr, String pwd)
        {
                int idusr = 0;
                        try{
                        ResultSet rs1 = ejecutaQuery("LOCALIZACION.dbo.loc_sp_validausuario '" + usr + "', '" + pwd + "'");

                        if(rs1!=null){
                        if(rs1.next()){
                                idusr = Integer.parseInt(rs1.getString(1).trim());
                        }
```

```
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }
            return idusr;
    }

            private ResultSet ejecutaQuery(String sp){
                    try{
                            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                    stmt = con.createStatement();
                    rs = stmt.executeQuery(sp);
               }catch(Exception e){
                  debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
               return rs;
            }


            private int ejecutaUpdate(String sp){
                    int filasMod = -1;
                    try{
                            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                    stmt = con.createStatement();
                    filasMod = stmt.executeUpdate(sp);
               }catch(Exception e){
                  debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
               return filasMod;
            }
}
```

LocSrvRutaDosPuntos.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvRutaDosPuntos extends HttpServlet{
            Connection con = null;
            Statement stmt = null;
            ResultSet rs = null;

            public void init(ServletConfig config) throws ServletException
            {
            super.init(config);
            }

            public void doGet(HttpServletRequest req, HttpServletResponse res) throws
```

```
        ServletException, IOException
        {
                PrintWriter out = res.getWriter();
                res.setContentType("text/vnd.wap.wml");

                out.println("       <?xml version='1.0'?>");
                out.println("<!DOCTYPE wml PUBLIC '-//WAPFORUM//DTD WML 1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
                out.println("<wml>");
                out.println("       <card id='card2P' title='Dos Puntos' newcontext='true' ordered='true'>");
                out.println("               <p align='left'>");
                out.println("                       <a href='LocSrvDireccion?dir=Or' title='DO'>Dir. Origen</a>");
                out.println("               </p>");
                out.println("               <p align='left'>");
                out.println("                       <a href='LocSrvDireccion?dir=De' title='DD'>Dir.
Destino</a>");
                out.println("               </p>");
                out.println("               <p align = 'center'>");
                out.println("                       <a href='LocSrvBuscarRutaDosPuntos?Rand="+ Math.random()
+"'>Ruta Optima</a>");
                out.println("               </p>");
                out.println("               <p>");
                out.println("               <a href='LocSrvPrincipal?Rand="+ Math.random() +"'>Regresar</a>");
                out.println("               </p>");
                out.println("       </card>");
                out.println("</wml>");
        }//doGet()

        private void debug(String str){
            System.out.println(str);
        }
}
```

LocSrvDireccion.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvDireccion extends HttpServlet{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

 public void init(ServletConfig config) throws ServletException
  {
    super.init(config);
  }

  public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException
  {
        PrintWriter out = res.getWriter();
        res.setContentType("text/vnd.wap.wml");

        HttpSession sesion = req.getSession(false); // cambie true -> false

        String dir = "";
        if(req.getParameter("dir")!=null){
```

```
		dir = (String)req.getParameter("dir");
	}

	int del = -1;
	if(req.getParameter("del")!=null){
		del = Integer.parseInt((String)req.getParameter("del"));
	}

	int col = -1;
	if(req.getParameter("col")!=null){
		col = Integer.parseInt((String)req.getParameter("col"));
	}

	int calle = -1;
	if(req.getParameter("calle")!=null){
		calle = Integer.parseInt((String)req.getParameter("calle"));
	}

	int num = -1;
	if(req.getParameter("num")!=null){
		num = Integer.parseInt((String)req.getParameter("num"));
		if(dir.equalsIgnoreCase("Or")){
			sesion.putValue("delOr", new Integer(del));
			sesion.putValue("colOr", new Integer(col));
			sesion.putValue("calleOr", new Integer(calle));
			sesion.putValue("numOr", new Integer(num));
		}else if(dir.equalsIgnoreCase("De")){
			sesion.putValue("delDe", new Integer(del));
			sesion.putValue("colDe", new Integer(col));
			sesion.putValue("calleDe", new Integer(calle));
			sesion.putValue("numDe", new Integer(num));
		}
	}


	debug("-------------------------------------\nPrueba: ");
	debug("delOr=" + sesion.getValue("delOr"));
	debug("delDe=" + sesion.getValue("delDe"));
	debug("colOr=" + sesion.getValue("colOr"));
	debug("colDe=" + sesion.getValue("colDe"));
	debug("calleOr=" + sesion.getValue("calleOr"));
	debug("calleDe=" + sesion.getValue("calleDe"));
	debug("numOr=" + sesion.getValue("numOr"));
	debug("numDe=" + sesion.getValue("numDe"));

	/*
	debug("req.getClass().getName()=" + req.getClass().getName() );
	debug("req.getScheme()=" + req.getScheme() );
	debug("req.getServerPort()=" + req.getServerPort());
	debug("req.getServerName()=" + req.getServerName());
	debug("req.getRequestURI()=" + req.getRequestURI());
	debug("req.getServletPath()=" + req.getServletPath());
	debug("req.getQueryString()=" + req.getQueryString());
	*/

	//loc_spi_busqueda 2,1,1,'adlskjdkl'
	String sqlString = "localizacion.dbo.loc_spi_busqueda ";
	sqlString += (dir.equalsIgnoreCase("Or")?"1":"2") +", ";
	sqlString += sesion.getValue("idUsr") + ", ";
	sqlString += (dir.equalsIgnoreCase("Or")?((num == -1)?"1":"2"):((num == -1)?"3":"4")) +", '";
	//sqlString += req.getScheme()+ "://"+ req.getServerName() +":"+ req.getServerPort() + req.getServletPath()
+"?"+ req.getQueryString() +"'";
	sqlString += "LocSrvDireccion?"+ req.getQueryString() +"'";
	debug(sqlString);
```

```
            ejecutaUpdate(sqlString);

            if(del == -1){
                        frameDelegacion(out, dir);
            }
            else if(col == -1){
                        frameColonia(out, dir, del);
            }
            else if(calle == -1){
                        frameCalle(out, dir, del, col);
            }
            else if(num == -1){
                        frameNum(out, dir, del, col, calle);
            }
            else{
                        res.sendRedirect("LocSrvRutaDosPuntos");
            }



  }//doGet()


    private void frameNum(PrintWriter out, String dir, int deleg, int col, int calle){
                        out.println("<?xml version=\"1.0\"?>");
                        //out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML1.1//EN\"
\"http://www.wapforum.org/DTD/wml1.1.xml\">");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                        out.println("<wml>");
                        out.println("       <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
                        out.println("       <p align='left'>");
                        out.println("       Introduce:");

                        out.println("<input name='txt_numero' type='text' format='*N' emptyok='false' maxlength='5'
title='Numero'/>");
                        out.println("       </p>");
                        out.println("       <p>");
                        out.println("       <a href='LocSrvDireccion?dir="
                                                + dir + "&amp;del="
                                                + deleg + "&amp;col="
                                                + col + "'>Regresar</a>");
                        out.println("       </p>");
                        out.println("       <p>");
                        out.println("       <a href='LocSrvDireccion?dir="
                                                + dir + "&amp;del="
                                                + deleg + "&amp;col="
                                                + col + "&amp;calle="
                                                + calle + "&amp;num="
                                                + "$(txt_numero)'>Aceptar</a>");
                        out.println("       </p>");
                        out.println("</card>");
                        out.println("</wml>");
                        out.close();
  }//frameNum()


  private void frameCalle(PrintWriter out, String dir, int deleg, int col){
                        out.println("<?xml version=\"1.0\"?>");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                        out.println("<wml>");
                        out.println("       <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
```

```
                out.println("       <p align='left'>");
                out.println("       Selecciona");
                out.println("               <select title='Cal' name='cbo_calleO' multiple='false'>");

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_cuadracalle " + col);

                if(rs1!=null){
                        String str1 = "";
                        String str2 = "";
                while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str2 = rs1.getString(2).trim();
                out.println("                       <option value='" + str1
                                                + "' onpick='LocSrvDireccion?dir="
                                                + dir + "&amp;del="
                                                + deleg + "&amp;col="
                                                + col + "&amp;calle="
                                                + str1 + "'>"+str2+"</option>");
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }

                out.println("               </select>");
                out.println("       </p>");
                out.println("       <p>");
                out.println("       <a href='LocSrvDireccion?dir="
                                                + dir + "&amp;del="
                                                + deleg + "'>Regresar</a>");
                out.println("       </p>");
                out.println("</card>");
                out.println("</wml>");
                out.close();
 }//frameCalle()


 private void frameColonia(PrintWriter out, String dir, int deleg){
                out.println("<?xml version=\"1.0\"?>");
                out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                out.println("<wml>");
                out.println("       <card id='cardP' title='Localización' newcontext='true' ordered='true'>");
                out.println("       <p align='left'>");
                out.println("       Selecciona");
                out.println("               <select title='Col' name='cbo_coloniaO' multiple='false'>");

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_colonia null," + deleg);

                if(rs1!=null){
                        String str1 = "";
                        String str2 = "";
                while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str2 = rs1.getString(2).trim();
                out.println("                       <option value='" + str1
```

```
                                                    + "' onpick='LocSrvDireccion?dir="
                                                    + dir + "&amp;del="
                                                    + deleg + "&amp;col="
                                                    + str1 + "'>"+str2+"</option>");
                    }
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("                    </select>");
                    out.println("          </p>");
                    out.println("          <p>");
                    out.println("          <a href='LocSrvDireccion?dir="
                                                    + dir + "'>Regresar</a>");
                    out.println("          </p>");
                    out.println("</card>");
                    out.println("</wml>");
                    out.close();
 } //frameColonia()

  private void frameDelegacion(PrintWriter out, String dir){
                    out.println("<?xml version=\"1.0\"?>");
                    out.println("<!DOCTYPE        wml        PUBLIC        \"-//WAPFORUM//DTD        WML        1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                    out.println("<wml>");
                    out.println("          <card id='cardP' title='Delegacion' newcontext='true' ordered='true'>");
                    out.println("          <p align='left'>");
                    out.println("          Selecciona");
                    out.println("                    <select title='Del:' name='cbo_delegacionO' multiple='false'>");

                    try{
                    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_delegacion");

                    if(rs1!=null){
                              String str1 = "";
                              String str2 = "";
                    while(rs1.next()){
                              str1 = rs1.getString(1).trim();
                              str2 = rs1.getString(2).trim();
                    out.println("                              <option value='" + str1
                                                    +"' onpick='LocSrvDireccion?dir="
                                                    + dir + "&amp;del="
                                                    + str1 + "'>"+str2+"</option>");
                    }
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("                    </select>");
                    out.println("          </p>");
```

```
                    out.println("          <p>");
                    out.println("          <a href='LocSrvRutaDosPuntos'>Regresar</a>");
                    out.println("          </p>");

                    out.println("</card>");
                    out.println("</wml>");
                    out.close();
 }//frameDelegacion()

 private ResultSet ejecutaQuery(String sp){
                    try{
                              Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                    stmt = con.createStatement();
                    rs = stmt.executeQuery(sp);
               }catch(Exception e){
                  debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
               return rs;
          }

          private int ejecutaUpdate(String sp){
                    int filasMod = -1;
                    try{
                              Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                    stmt = con.createStatement();
                    filasMod = stmt.executeUpdate(sp);
               }catch(Exception e){
                  debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
               return filasMod;
          }

          private void cierraBD(){
                    try{
                              rs.close();
                              stmt.close();
                              con.close();
                    }catch(Exception e){
                  debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
                  e.printStackTrace(System.out);
               }
          }

          private void debug(String str){
               System.out.println(str);
          }
}
```

LocSrvBuscarRutaDosPuntos.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import Dijkstra;
```

```
public class LocSrvBuscarRutaDosPuntos extends HttpServlet
{
	private Connection con = null;
	private Statement stmt = null;
	private ResultSet rs = null;

	public void doGet(HttpServletRequest req, HttpServletResponse res) throws
	ServletException, IOException
	{
		try{
			PrintWriter out = res.getWriter();
			res.setContentType("text/vnd.wap.wml");

			HttpSession sesion = req.getSession(false); // cambie true -> false

			Integer delOr = new Integer(-1);
			Integer delDe = new Integer(-1);
			Integer colOr = new Integer(-1);
			Integer colDe = new Integer(-1);
			Integer calleOr = new Integer(-1);
			Integer calleDe = new Integer(-1);
			Integer numOr = new Integer(-1);
			Integer numDe = new Integer(-1);

			if(sesion.getValue("delOr")!=null){
				delOr = (Integer)sesion.getValue("delOr");
			}
			if(sesion.getValue("delDe")!=null){
				delDe = (Integer)sesion.getValue("delDe");
			}
			if(sesion.getValue("colOr")!=null){
				colOr = (Integer)sesion.getValue("colOr");
			}
			if(sesion.getValue("colDe")!=null){
				colDe = (Integer)sesion.getValue("colDe");
			}
			if(sesion.getValue("calleOr")!=null){
				calleOr = (Integer)sesion.getValue("calleOr");
			}
			if(sesion.getValue("calleDe")!=null){
				calleDe = (Integer)sesion.getValue("calleDe");
			}
			if(sesion.getValue("numOr")!=null){
				numOr = (Integer)sesion.getValue("numOr");
			}
			if(sesion.getValue("numDe")!=null){
				numDe = (Integer)sesion.getValue("numDe");
			}

			String txt_usr = "";
			if(sesion.getValue("txt_usr")!=null){
				txt_usr = (String)sesion.getValue("txt_usr");
			}

			String txt_pwd = "";
			if(sesion.getValue("txt_pwd")!=null){
				txt_pwd = (String)sesion.getValue("txt_pwd");
			}


			Dijkstra d = new Dijkstra(calleOr.intValue() , numOr.intValue(), calleDe.intValue(),
numDe.intValue());

			String rutaOptima[] = new String[d.getNumNodosRutaOptima()];
```

```
                                rutaOptima = d.getNombresCalles();

                                out.println("<?xml version=\"1.0\"?>");
                                out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML1.1//EN\"
\"http://www.wapforum.org/DTD/wml1.1.xml\">");
                                out.println("<wml>");
                                out.println("<card id='cardres' title='Ruta Optima'>");

                                int nume = 1;
                                for(int i=0; i<rutaOptima.length; i++){
                                        if(String.valueOf(rutaOptima[i]).indexOf("(") == -1){
                                                out.println("<p align='left' mode='nowrap'><small>" + rutaOptima[i] +
"</small></p>");
                                        }else{
                                                if(i==0){

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i+1].substring(0,rutaOptima[i+1].i
ndexOf("(")))!=0)
                                                                out.println("<p align='left' mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                                else if(i==1){
                                                        out.println("<p align='left' mode='nowrap'><small>" +
(nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                                else if(i==(rutaOptima.length - 1)){

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i-1].substring(0,rutaOptima[i-
1].indexOf("(")))!=0)
                                                                out.println("<p align='left' mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                                else{
                                                        //if(rutaOptima[i].compareTo(rutaOptima[i-1])!=0)

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i-1].substring(0,rutaOptima[i-
1].indexOf("(")))!=0)
                                                                out.println("<p align='left' mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                        }
                                }

                                out.println("                   <p align='left'>");
                                out.println("                           <br/><a                 href='LocSrvRutaDosPuntos'
title='NewD'>Cambiar Direccion</a>");
                                out.println("                   </p>");

                                out.println("                   <p align='left'>");
                                out.println("                           <a href='LocSrvPrincipal?nvaBusq=S&amp;txt_usr="+
txt_usr +"&amp;txt_pwd="+ txt_pwd +"&URLRand="+ Math.random() +"' title='NewB'>Nueva Busqueda</a>");
                                out.println("                   </p>");

                                out.println("<p align='center'><small><br/>Visita cic.ipn.mx</small></p>");
                                out.println("</card>");
                                out.println("</wml>");
                                out.close();

                        }catch(Exception e){
                        debug("Error en doGet():" + e.toString() + "_Msg=" + e.getMessage());
                        e.printStackTrace(System.out);
                        }
 }
```

```
private int encontrarNodoCercano(int idcalle, int numero){
        int nodoCercano = -1;

   int nodo1 = -1;
   int nodo2 = -1;
   int flujo1 = -1;
   int flujo2 = -1;
   int numIni = -1;
   int numFin = -1;
   int trafico = -1;

        try{
        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_buscaNodos " + idcalle + ", " + numero);

        if(rs1!=null){
        if(rs1.next()){
                nodo1 = rs1.getInt(1);
                nodo2 = rs1.getInt(2);
                flujo1 = rs1.getInt(3);
                flujo2 = rs1.getInt(4);
                numIni = rs1.getInt(5);
                numFin = rs1.getInt(6);
                trafico = rs1.getInt(7);
        }
        }
        else
        debug("rs1 fue null");

        rs1.close();
        //cierraBD();
        }catch(Exception e){
        debug("Error en encontrarNodoCercano():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
        }

        if(trafico == 0){
                // mandar mensaje de que la cuadra se encuentra bloqueada,
                // por lo que tiene que situarse en otra posicion
        }
        else if((flujo1 == 0) && (flujo2 == 0)){
                // mandar mensaje de error,
                // por lo que tiene que situarse en otra posicion
        }
        else if(flujo1 == 0){
                nodoCercano = nodo2;
        }
        else if(flujo2 == 0){
                nodoCercano = nodo1;
        }
        else{
                if((numero >= numIni) && (numero <= (numIni + (numFin-numIni)/2)))
                        nodoCercano = nodo1;
                else
                        nodoCercano = nodo2;
        }

        return nodoCercano;
}

private ResultSet ejecutaQuery(String sp){
        try{
                Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
```

```
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
            }catch(Exception e){
                debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
            }
            return rs;
        }
        private void debug(String str){
            System.out.println(str);
        }
}
```

LocSrvRutaSitio.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvRutaSitio extends HttpServlet{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

        public void init(ServletConfig config) throws ServletException
        {
                super.init(config);
        }

        public void doGet(HttpServletRequest req, HttpServletResponse res) throws
                ServletException, IOException
        {
                PrintWriter out = res.getWriter();
                res.setContentType("text/vnd.wap.wml");

                out.println("       <?xml version='1.0'?>");
                out.println("<!DOCTYPE       wml       PUBLIC       '-//WAPFORUM//DTD       WML       1.2//EN'
'http://www.wapforum.org/DTD/wml12.dtd'>");
                out.println("<wml>");
                out.println("       <card id='cardSI' title='Sitio Interes' newcontext='true' ordered='true'>");
                out.println("               <p align='left'>");
                out.println("                       <a      href='LocSrvDireccionSitio?dir=Ori'      title='DO'>Dir.
Origen</a>");
                out.println("               </p>");
                out.println("               <p align='left'>");
                out.println("                       <a href='LocSrvSitio?dir=Sit' title='DS'>Sitio</a>");
                out.println("               </p>");
                out.println("               <p align = 'center'>");
                out.println("                       <a      href='LocSrvBuscarRutaSitio?Rand="+      Math.random()
+"'>Ruta Optima</a>");
                out.println("               </p>");
                out.println("               <p>");
                out.println("               <a href='LocSrvPrincipal?Rand="+ Math.random() +"'>Regresar</a>");
                out.println("               </p>");
                out.println("       </card>");
                out.println("</wml>");
        }//doGet()

        private void debug(String str){
```

```
            System.out.println(str);
        }
}
```

LocSrvDireccionSitio.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvDireccionSitio extends HttpServlet{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

 public void init(ServletConfig config) throws ServletException
  {
    super.init(config);
  }

  public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException
  {
        PrintWriter out = res.getWriter();
        res.setContentType("text/vnd.wap.wml");

        HttpSession sesion = req.getSession(false); // cambie true -> false

        String dir = "";
        if(req.getParameter("dir")!=null){
                dir = (String)req.getParameter("dir");
        }

        int del = -1;
        if(req.getParameter("del")!=null){
                del = Integer.parseInt((String)req.getParameter("del"));
        }

        int col = -1;
        if(req.getParameter("col")!=null){
                col = Integer.parseInt((String)req.getParameter("col"));
        }

        int calle = -1;
        if(req.getParameter("calle")!=null){
                calle = Integer.parseInt((String)req.getParameter("calle"));
        }

        int num = -1;
        if(req.getParameter("num")!=null){
                num = Integer.parseInt((String)req.getParameter("num"));
                if(dir.compareTo("Ori")==0){
                        sesion.putValue("delOri", new Integer(del));
                        sesion.putValue("colOri", new Integer(col));
                        sesion.putValue("calleOri", new Integer(calle));
                        sesion.putValue("numOri", new Integer(num));
                }else if(dir.compareTo("Sit")==0){
                        sesion.putValue("delSit", new Integer(del));
                        sesion.putValue("colSit", new Integer(col));
```

```
                        sesion.putValue("calleSit", new Integer(calle));
                        sesion.putValue("numSit", new Integer(num));
                }
        }


        debug("-------*------*-------*-------*--------\nPrueba: ");
        debug("delOri=" + sesion.getValue("delOri"));
        debug("colOri=" + sesion.getValue("colOri"));
        debug("calleOri=" + sesion.getValue("calleOri"));
        debug("numOri=" + sesion.getValue("numOri"));
        debug("delSit=" + sesion.getValue("delSit"));
        debug("colSit=" + sesion.getValue("colSit"));
        debug("calleSit=" + sesion.getValue("calleSit"));
        debug("numSit=" + sesion.getValue("numSit"));

        //loc_spi_busqueda 2,1,1,'adlskjdkl'
        String sqlString = "localizacion.dbo.loc_spi_busqueda ";
        sqlString += (dir.equalsIgnoreCase("Ori")?"1":"2") +", ";
        sqlString += sesion.getValue("idUsr") + ", ";
        sqlString += (dir.equalsIgnoreCase("Ori")?((num == -1)?"1":"2"):((num == -1)?"3":"4")) +", '";
        //sqlString += req.getScheme()+ "://"+ req.getServerName() +":"+ req.getServerPort() + req.getServletPath()
+"?"+ req.getQueryString() +"'";
        sqlString += "LocSrvDireccionSitio?"+ req.getQueryString() +"'";
        debug(sqlString);
        ejecutaUpdate(sqlString);

        if(del == -1){
                frameDelegacion(out, dir);
        }
        else if(col == -1)
                frameColonia(out, dir, del);
        else if(calle == -1)
                frameCalle(out, dir, del, col);
        else if(num == -1)
                frameNum(out, dir, del, col, calle);
        else
                res.sendRedirect("LocSrvRutaSitio");



 }//doGet()


   private void frameNum(PrintWriter out, String dir, int deleg, int col, int calle){
                out.println("<?xml version=\"1.0\"?>");
                //out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML1.1//EN\"
\"http://www.wapforum.org/DTD/wml1.1.xml\">");
                out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                out.println("<wml>");
                out.println("   <card id='cardP' title='Numero' newcontext='true' ordered='true'>");
                out.println("   <p align='left'>");
                out.println("   Introduce:");

                out.println("<input name='txt_numero' type='text' format='*N' emptyok='false' maxlength='5'
title='Numero'/>");
                out.println("   </p>");
                out.println("   <p>");
                out.println("   <a href='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + deleg + "&amp;col="
                                                        + col + "'>Regresar</a>");
```

```
                out.println("       </p>");
                out.println("       <p>");
                out.println("       <a href='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + deleg + "&amp;col="
                                                        + col + "&amp;calle="
                                                        + calle + "&amp;num="
                                                        + "$(txt_numero)'>Aceptar</a>");
                out.println("       </p>");
                out.println("</card>");
                out.println("</wml>");
                out.close();
 }//frameNum()


 private void frameCalle(PrintWriter out, String dir, int deleg, int col){
                out.println("<?xml version=\"1.0\"?>");
                out.println("<!DOCTYPE       wml       PUBLIC       \"-//WAPFORUM//DTD       WML       1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                out.println("<wml>");
                out.println("       <card id='cardP' title='Calle' newcontext='true' ordered='true'>");
                out.println("       <p align='left'>");
                out.println("       Selecciona");
                out.println("               <select title=':' name='cbo_calleO' multiple='false'>");

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_cuadracalle " + col);

                if(rs1!=null){
                        String str1 = "";
                        String str2 = "";
                while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str2 = rs1.getString(2).trim();
                out.println("                       <option value='" + str1
                                                        + "' onpick='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + deleg + "&amp;col="
                                                        + col + "&amp;calle="
                                                        + str1 + "'>"+str2+"</option>");
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }

                out.println("               </select>");
                out.println("       </p>");
                out.println("       <p>");
                out.println("       <a href='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + deleg + "'>Regresar</a>");
                out.println("       </p>");
                out.println("</card>");
                out.println("</wml>");
                out.close();
 }//frameCalle()
```

```
 private void frameColonia(PrintWriter out, String dir, int deleg){
                    out.println("<?xml version=\"1.0\"?>");
                    out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                    out.println("<wml>");
                    out.println("  <card id='cardP' title='Colonia' newcontext='true' ordered='true'>");
                    out.println("  <p align='left'>");
                    out.println("  Selecciona");
                    out.println("  <select title=':' name='cbo_coloniaO' multiple='false'>");

                    try{
                    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_colonia null," + deleg);

                    if(rs1!=null){
                              String str1 = "";
                              String str2 = "";
                    while(rs1.next()){
                              str1 = rs1.getString(1).trim();
                              str2 = rs1.getString(2).trim();
                    out.println("  <option value='" + str1
                                        + "' onpick='LocSrvDireccionSitio?dir="
                                        + dir + "&amp;del="
                                        + deleg + "&amp;col="
                                        + str1 + "'>"+str2+"</option>");
                    }
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("  </select>");
                    out.println("  </p>");
                    out.println("  <p>");
                    out.println("  <a href='LocSrvDireccionSitio?dir="
                                        + dir + "'>Regresar</a>");
                    out.println("  </p>");
                    out.println("</card>");
                    out.println("</wml>");
                    out.close();
 } //frameColonia()

 private void frameDelegacion(PrintWriter out, String dir){
                    out.println("<?xml version=\"1.0\"?>");
                    out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                    out.println("<wml>");
                    out.println("  <card id='cardP' title='Delegacion' newcontext='true' ordered='true'>");
                    out.println("  <p align='left'>");
                    out.println("  Selecciona");
                    out.println("  <select title=':' name='cbo_delegacionO' multiple='false'>");

                    try{
                    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_delegacion");

                    if(rs1!=null){
                              String str1 = "";
                              String str2 = "";
```

```
                while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str2 = rs1.getString(2).trim();
                out.println("                                <option value='" + str1
                                                        +"' onpick='LocSrvDireccionSitio?dir="
                                                        + dir + "&amp;del="
                                                        + str1 + "'>"+str2+"</option>");
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }

                out.println("                </select>");
                out.println("        </p>");
                out.println("        <p>");
                out.println("        <a href='LocSrvRutaSitio'>Regresar</a>");
                out.println("        </p>");

                out.println("</card>");
                out.println("</wml>");
                out.close();
}//frameDelegacion()

private ResultSet ejecutaQuery(String sp){
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
           }catch(Exception e){
              debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return rs;
        }

        private int ejecutaUpdate(String sp){
                int filasMod = -1;
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                filasMod = stmt.executeUpdate(sp);
           }catch(Exception e){
              debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return filasMod;
        }

        private void cierraBD(){
                try{
                        rs.close();
                        stmt.close();
                        con.close();
                }catch(Exception e){
              debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
```

```
                e.printStackTrace(System.out);
            }
        }

        private void debug(String str){
            System.out.println(str);
        }
}
```

LocSrvSitio.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LocSrvSitio extends HttpServlet{
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;

 public void init(ServletConfig config) throws ServletException
  {
    super.init(config);
  }

  public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException
  {
        PrintWriter out = res.getWriter();
        res.setContentType("text/vnd.wap.wml");

        HttpSession sesion = req.getSession(false); // cambie true -> false

        String xcat = "";
        if(req.getParameter("xcat")!=null){
                xcat = (String)req.getParameter("xcat");
        }

        int categ = -1;
        if(req.getParameter("categ")!=null){
                categ = Integer.parseInt((String)req.getParameter("categ"));
        }

        int idSitio = -1;
        if(req.getParameter("idSitio")!=null){
                idSitio = Integer.parseInt((String)req.getParameter("idSitio"));
        }


        String dir = "";
        if(req.getParameter("dir")!=null){
                dir = (String)req.getParameter("dir");
        }

        int del = -1;
        if(req.getParameter("del")!=null){
                del = Integer.parseInt((String)req.getParameter("del"));
        }
```

```
		int col = -1;
		if(req.getParameter("col")!=null){
			col = Integer.parseInt((String)req.getParameter("col"));
		}

		int calle = -1;
		if(req.getParameter("calle")!=null){
			calle = Integer.parseInt((String)req.getParameter("calle"));
		}

		int num = -1;
		if(req.getParameter("num")!=null){
			num = Integer.parseInt((String)req.getParameter("num"));

			sesion.putValue("delSit", new Integer(del));
			sesion.putValue("colSit", new Integer(col));
			sesion.putValue("calleSit", new Integer(calle));
			sesion.putValue("numSit", new Integer(num));

			xcat = "Salir";
		}

		int masCercano = -1;
		if(req.getParameter("masCercano")!=null){
			masCercano = Integer.parseInt((String)req.getParameter("masCercano"));

			sesion.putValue("masCercanoSit", new Integer(masCercano));
			sesion.putValue("categSit", new Integer(categ));

			xcat = "Salir";
		}

		debug("-------*------*-------*-------*--------\nPrueba: ");
		debug("-->xcat=" + xcat);
		debug("delSit=" + sesion.getValue("delSit"));
		debug("colSit=" + sesion.getValue("colSit"));
		debug("calleSit=" + sesion.getValue("calleSit"));
		debug("numSit=" + sesion.getValue("numSit"));
		debug("masCercanoSit=" + sesion.getValue("masCercanoSit"));
		debug("categSit=" + sesion.getValue("categSit"));

		//loc_spi_busqueda 2,1,1,'adlskjdkl'
		String sqlString = "localizacion.dbo.loc_spi_busqueda ";
		sqlString += (dir.equalsIgnoreCase("Ori")?"1":"2") +", ";
		sqlString += sesion.getValue("idUsr") + ", ";
		sqlString += (dir.equalsIgnoreCase("Ori")?((num == -1)?"1":"2"):((num == -1)?"3":"4")) +", '";
		//sqlString += req.getScheme()+ "://"+ req.getServerName() +":"+ req.getServerPort() + req.getServletPath()
+"?"+ req.getQueryString() +"'";
		sqlString += "LocSrvSitio?"+ req.getQueryString() +"'";
		debug(sqlString);
		ejecutaUpdate(sqlString);


		if(xcat.compareTo("")==0){
			frameBuscarSitioPor(out, dir);
		}
		else if(xcat.compareTo("Si")==0){
			if(categ == -1){
				frameBuscarSitioPorCateg(out, dir);
			}
			else{
				if(idSitio == -1)
					frameBuscarSitioPorCateg(out, dir, categ);
				else if(idSitio != 0)
```

```
                                        seleccionarDireccionSitio(res, idSitio);
                                else
                                        redireccionarSinDirSitio(res, categ);
                        }
                }
                else if(xcat.compareTo("No")==0){
                        //frameBuscarSitioPorNomb(out, dir);
                }
                else if(xcat.compareTo("Salir")==0){
                        res.sendRedirect("LocSrvRutaSitio");
                }

        }//doGet()


        private void frameBuscarSitioPor(PrintWriter out, String dir){
                        out.println("<?xml version=\"1.0\"?>");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                        out.println("<wml>");
                        out.println("   <card id='cardP' title='Sitio' newcontext='true' ordered='true'>");
                        out.println("   <p align='left'>");
                        out.println("   Buscar por:<br/>");
                        out.println("   <a href='LocSrvSitio?dir=Sit&amp;xcat=Si'>Categoria</a><br/>");
                        out.println("   </p>");
                        out.println("   <p>");
                        out.println("   <a href='LocSrvRutaSitio'>Regresar</a>");
                        out.println("   </p>");

                        out.println("</card>");
                        out.println("</wml>");
                        out.close();
        }//frameBuscarSitioPor()


        private void frameBuscarSitioPorCateg(PrintWriter out, String dir){
                        out.println("<?xml version=\"1.0\"?>");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                        out.println("<wml>");
                        out.println("   <card id='cardP' title='Categoria' newcontext='true' ordered='true'>");
                        out.println("   <p align='left'>");
                        out.println("   Selecciona");
                        out.println("           <select title='Cat:' name='cbo_categoriaD' multiple='false'>");

                        try{
                        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_categoria");

                        if(rs1!=null){
                                String str1 = "";
                                String str2 = "";

                        while(rs1.next()){
                                str1 = rs1.getString(1).trim();
                                str2 = rs1.getString(2).trim();
                        out.println("                   <option value='" + str1
                                                        +"'
onpick='LocSrvSitio?dir=Sit&amp;xcat=Si&amp;categ="
                                                        + str1 + "'>"+str2+"</option>");
                        }
                        }
                        else
                        debug("rs1 fue null");
```

```
                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("                    </select>");
                    out.println("          </p>");
                    out.println("          <p>");
                    out.println("          <a href='LocSrvSitio'>Regresar</a>");
                    out.println("          </p>");

                    out.println("</card>");
                    out.println("</wml>");
                    out.close();
  }//frameBuscarSitioPorCateg()


   private void frameBuscarSitioPorCateg(PrintWriter out, String dir, int categ){
                    out.println("<?xml version=\"1.0\"?>");
                    out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML 1.2//EN\"
\"http://www.wapforum.org/DTD/wml12.dtd\">");
                    out.println("<wml>");
                    out.println("          <card id='cardP' title='Categoria' newcontext='true' ordered='true'>");
                    out.println("          <p align='left'>");
                    out.println("          Selecciona");
                    out.println("                    <select title='Cat:' name='cbo_categoriaD' multiple='false'>");

                    try{
                    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio null, " + categ);

                    if(rs1!=null){
                              String str1 = "";
                              String str2 = "";

                   out.println("                              <option value='" + str1
                                                  +"' onpick='LocSrvSitio?dir=Sit&amp;xcat=Si&amp;categ="
                                                  + categ + "&amp;idSitio=0'>+ cercano</option>");
                              while(rs1.next()){
                              str1 = rs1.getString(1).trim();
                              str2 = rs1.getString(2).trim();
                    out.println("                              <option value='" + str1
                                                  +"'
onpick='LocSrvSitio?dir=Sit&amp;xcat=Si&amp;categ="
                                                  + categ + "&amp;idSitio="
                                                  + str1 + "'>"+str2+"</option>");
                    }
                    }
                    else
                    debug("rs1 fue null");

                    rs1.close();
                    //cierraBD();
                    }catch(Exception e){
                    debug("Error en():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }

                    out.println("                    </select>");
                    out.println("          </p>");
                    out.println("          <p>");
                    out.println("          <a href='LocSrvSitio'>Regresar</a>");
                    out.println("          </p>");
```

```
                out.println("</card>");
                out.println("</wml>");
                out.close();
  }//frameBuscarSitioPorCateg()

        private void seleccionarDireccionSitio(HttpServletResponse res, int idSitio){
                String dir = "Sit";
                int deleg = -1;
                int col = -1;
                int calle = -1;
                int numero = -1;

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio " + idSitio);

                if(rs1!=null){
                if(rs1.next()){
                        calle = rs1.getInt(3);
                        numero = rs1.getInt(5);
                        col = rs1.getInt(6);
                        deleg = rs1.getInt(9);
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                debug("Redirect=LocSrvSitio?dir="
                                                        + dir + "&del="
                                                        + deleg + "&col="
                                                        + col + "&calle="
                                                        + calle + "&num="
                                                        + numero + "&masCercano=0&categ=0");
                        res.sendRedirect("LocSrvSitio?dir="
                                                        + dir + "&del="
                                                        + deleg + "&col="
                                                        + col + "&calle="
                                                        + calle + "&num="
                                                        + numero + "&masCercano=0&categ=0");

                //cierraBD();
                }catch(Exception e){
                debug("Error en seleccionarDireccionSitio():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }
        }


private void redireccionarSinDirSitio(HttpServletResponse res, int categ){
        try{
                String dir = "Sit";
                debug("Redirect=LocSrvSitio?dir="
                                + dir + "&masCercano=1&categ=" + categ
                                + "&del=0&col=0&calle=0&num=0");
                res.sendRedirect("LocSrvSitio?dir="
                                + dir + "&masCercano=1&categ=" + categ
                                + "&del=0&col=0&calle=0&num=0");
        }catch(Exception e){
                debug("Error en redireccionarSinDirSitio():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
        }
}
```

```
 private ResultSet ejecutaQuery(String sp){
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
           }catch(Exception e){
              debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return rs;
        }

        private int ejecutaUpdate(String sp){
                int filasMod = -1;
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                filasMod = stmt.executeUpdate(sp);
           }catch(Exception e){
              debug("Error en ejecutaUpdate("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return filasMod;
        }

        private void cierraBD(){
                try{
                        rs.close();
                        stmt.close();
                        con.close();
                }catch(Exception e){
              debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
        }

        private void debug(String str){
           System.out.println(str);
        }
}
```

LocSrvBuscarRutaSitio.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import Dijkstra;

public class LocSrvBuscarRutaSitio extends HttpServlet
{
        private Connection con = null;
        private Statement stmt = null;
        private ResultSet rs = null;

        public void doGet(HttpServletRequest req, HttpServletResponse res) throws
        ServletException, IOException
```

```
{
	try{
		PrintWriter out = res.getWriter();
		res.setContentType("text/vnd.wap.wml");

		HttpSession sesion = req.getSession(false); // cambie true -> false

		Integer delOri = new Integer(-1);
		Integer delSit = new Integer(-1);
		Integer colOri = new Integer(-1);
		Integer colSit = new Integer(-1);
		Integer calleOri = new Integer(-1);
		Integer calleSit = new Integer(-1);
		Integer numOri = new Integer(-1);
		Integer numSit = new Integer(-1);
		Integer masCercanoSit = new Integer(-1);
		Integer categSit = new Integer(-1);

		if(sesion.getValue("delOri")!=null){
			delOri = (Integer)sesion.getValue("delOri");
		}
		if(sesion.getValue("delSit")!=null){
			delSit = (Integer)sesion.getValue("delSit");
		}
		if(sesion.getValue("colOri")!=null){
			colOri = (Integer)sesion.getValue("colOri");
		}
		if(sesion.getValue("colSit")!=null){
			colSit = (Integer)sesion.getValue("colSit");
		}
		if(sesion.getValue("calleOri")!=null){
			calleOri = (Integer)sesion.getValue("calleOri");
		}
		if(sesion.getValue("calleSit")!=null){
			calleSit = (Integer)sesion.getValue("calleSit");
		}
		if(sesion.getValue("numOri")!=null){
			numOri = (Integer)sesion.getValue("numOri");
		}
		if(sesion.getValue("numSit")!=null){
			numSit = (Integer)sesion.getValue("numSit");
		}
		if(sesion.getValue("masCercanoSit")!=null){
			masCercanoSit = (Integer)sesion.getValue("masCercanoSit");
		}
		if(sesion.getValue("categSit")!=null){
			categSit = (Integer)sesion.getValue("categSit");
		}

		String txt_usr = "";
		if(sesion.getValue("txt_usr")!=null){
			txt_usr = (String)sesion.getValue("txt_usr");
		}

		String txt_pwd = "";
		if(sesion.getValue("txt_pwd")!=null){
			txt_pwd = (String)sesion.getValue("txt_pwd");
		}

		/*
		debug("-------");
		debug("delOri=" + delOri);
		debug("delSit=" + delSit);
		debug("colOri=" + colOri);
```

```
                        debug("colSit=" + colSit);
                        debug("calleOri=" + calleOri);
                        debug("calleSit=" + calleSit);
                        debug("numOri=" + numOri);
                        debug("numSit=" + numSit);
                        */
/*
                        int idNodoOrigen = encontrarNodoCercano(calleOri.intValue() , numOri.intValue());
                        int idNodoDestino = encontrarNodoCercano(calleSit.intValue(), numSit.intValue());
                        Dijkstra d = new Dijkstra();
                        d.setNodoOrig(idNodoOrigen);
                        d.setNodoDest(idNodoDestino);
                        d.ejecutarAlg();
*/
                        Dijkstra d = null;
                        if(masCercanoSit.intValue()==0)
                                d = new Dijkstra(calleOri.intValue() , numOri.intValue(), calleSit.intValue(),
numSit.intValue());
                        else
                                d = new Dijkstra(calleOri.intValue() , numOri.intValue(), categSit.intValue());

                        String rutaOptima[] = new String[d.getNumNodosRutaOptima()];
                        rutaOptima = d.getNombresCalles();
                        out.println("<?xml version=\"1.0\"?>");
                        out.println("<!DOCTYPE wml PUBLIC \"-//WAPFORUM//DTD WML1.1//EN\"
\"http://www.wapforum.org/DTD/wml1.1.xml\">");
                        out.println("<wml>");
                        out.println("<card id='cardInfo' title='Ruta Optima'>");

                        if(masCercanoSit.intValue()!=0){
                        try{ //obtiene la categoria elegida
                        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_categoria " + categSit);

                        if(rs1!=null){
                                if(rs1.next())
                                                out.println("<p align='left' mode='nowrap'><small>El
sitio de la categoria " + rs1.getString(2).trim() + "</small></p>");
                        }
                        else
                                debug("rs1 fue null");

                        rs1.close();
                        }catch(Exception e){
                        debug("Error en doGet():" + e.toString() + "_Msg=" + e.getMessage());
                        e.printStackTrace(System.out);
                        }

                        try{
                        ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio " + d.idSitioMasCercano + ",
null");

                        if(rs1!=null){
                                if(rs1.next()){
                                        String strSitio = rs1.getString(2).trim();
                                        String strCalle = rs1.getString(4).trim();
                                        String strNumer = rs1.getString(5).trim();
                                        String strCol = rs1.getString(7).trim();
                                        String strTelef = rs1.getString(8).trim();
                                        String strDeleg = rs1.getString(10).trim();

                                                out.println("<p align='left'
mode='nowrap'><small>mas cercano es " + strSitio + ", ubicado en:</small></p>");
                                                out.println("<p align='left' mode='nowrap'><small>"
+ strCalle + " " + strNumer + " Col. " + strCol + "</small></p>");
```

```
                                                        out.println("<p                                    align='left'
mode='nowrap'><small>Del. " + strDeleg + ". Tel: " + strTelef + "</small></p>");
                                                }
                        }
                        else
                                debug("rs1 fue null");

                        rs1.close();
                        }catch(Exception e){
                        debug("Error en doGet():" + e.toString() + "_Msg=" + e.getMessage());
                        e.printStackTrace(System.out);
                        }

                                out.println("                   <p align='left'>");
                                out.println("                           <a      href='#cardDirec'      title='DO'>Ver
Ruta</a>");
                                out.println("                   </p>");
                                out.println("</card>");
                                out.println("<card id='cardDirec' title='Ruta Optima'>");
                        }

                        int nume = 1;
                        for(int i=0; i<rutaOptima.length; i++){
                                if(String.valueOf(rutaOptima[i]).indexOf("(") == -1){
                                        out.println("<p  align='left'  mode='nowrap'><small>" + rutaOptima[i] +
"</small></p>");
                                }else{
                                        if(i==0){

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i+1].substring(0,rutaOptima[i+1].i
ndexOf("(")))!=0){
                                                        out.println("<p  align='left'  mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                        }
                                        else if(i==1){
                                                out.println("<p     align='left'     mode='nowrap'><small>"     +
(nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                        }
                                        else if(i==(rutaOptima.length - 1)){

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i-1].substring(0,rutaOptima[i-
1].indexOf("(")))!=0){
                                                        out.println("<p  align='left'  mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                        }
                                        else{
                                                //if(rutaOptima[i].compareTo(rutaOptima[i-1])!=0)

        if(rutaOptima[i].substring(0,rutaOptima[i].indexOf("(")).compareTo(rutaOptima[i-1].substring(0,rutaOptima[i-
1].indexOf("(")))!=0){
                                                        out.println("<p  align='left'  mode='nowrap'><small>"
+ (nume++) + ".-" + rutaOptima[i] + "</small></p>");
                                                }
                                        }
                                }
                        }

                        //out.println("<p align='center'><small>Nodo Orig: " + idNodoOrigen + "</small></p>");
                        //out.println("<p align='center'><small>Nodo Dest: " + idNodoDestino + "</small></p>");

                        if(masCercanoSit.intValue()!=0){
                                out.println("                   <p align='left'>");
```

```
                                    out.println("                                   <a      href='#cardInfo'      title='DO'>Ver
Sitio</a>");
                                    out.println("                   </p>");
                            }


                            out.println("                   <p align='left'>");
                            out.println("                           <br/><a href='LocSrvRutaSitio' title='NewD'>Cambiar
Dir o Sit</a>");
                            out.println("                   </p>");

                            out.println("                   <p align='left'>");
                            out.println("                           <a href='LocSrvPrincipal?nvaBusq=S&amp;txt_usr="+
txt_usr +"&amp;txt_pwd="+ txt_pwd +"&URLRand="+ Math.random() +"' title='NewB'>Nueva Busqueda</a>");
                            out.println("                   </p>");

                            out.println("<p align='center'><small><br/>Visita cic.ipn.mx</small></p>");
                            out.println("</card>");
                            out.println("</wml>");
                            out.close();

                    }catch(Exception e){
                    debug("Error en doGet():" + e.toString() + "_Msg=" + e.getMessage());
                    e.printStackTrace(System.out);
                    }
 }


        private int encontrarNodoCercano(int idcalle, int numero){
                int nodoCercano = -1;

           int nodo1 = -1;
           int nodo2 = -1;
           int flujo1 = -1;
           int flujo2 = -1;
           int numIni = -1;
           int numFin = -1;
           int trafico = -1;

                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_buscaNodos " + idcalle + ", " + numero);

                if(rs1!=null){
                if(rs1.next()){
                        nodo1 = rs1.getInt(1);
                        nodo2 = rs1.getInt(2);
                        flujo1 = rs1.getInt(3);
                        flujo2 = rs1.getInt(4);
                        numIni = rs1.getInt(5);
                        numFin = rs1.getInt(6);
                        trafico = rs1.getInt(7);
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en encontrarNodoCercano():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }
```

```
                if(trafico == 0){
                        // mandar mensaje de que la cuadra se encuentra bloqueada,
                        // por lo que tiene que situarse en otra posicion
                }
                else if((flujo1 == 0) && (flujo2 == 0)){
                        // mandar mensaje de error,
                        // por lo que tiene que situarse en otra posicion
                }
                else if(flujo1 == 0){
                        nodoCercano = nodo2;
                }
                else if(flujo2 == 0){
                        nodoCercano = nodo1;
                }
                else{
                        if((numero >= numIni) && (numero <= (numIni + (numFin-numIni)/2)))
                                nodoCercano = nodo1;
                        else
                                nodoCercano = nodo2;
                }

                return nodoCercano;
        }

        private ResultSet ejecutaQuery(String sp){
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
           }catch(Exception e){
              debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return rs;
        }
        private void debug(String str){
           System.out.println(str);
        }
}
```

Dijkstra.java

```
import java.sql.*;
import java.util.Vector;
import java.util.Enumeration;

public class Dijkstra
{
        private Connection con = null;
        private Statement stmt = null;
        private ResultSet rs = null;

   private int MAXNODOS;
   private int MAX;
   int DIJKSTRA = 1;

   private int nodoInicial=0;              // nodo inicial del grafo
   private int nodoOrig = 0;
   private int nodoDest = 0;
```

```
private int idNodos[];                                  //almacena los identificadores de cada nodo
private int idNodosRutaOptima[];    //almacena los identificadores de cada nodo
private int numNodosRutaOptima;
public String callesRutaOptima[];    //almacena los nombres de las calles de la ruta optima
private int peso[][];                                   //almacena los pesos de un nodo a otro

// informacion del grafo al ejecutar el algoritmo
boolean rutaMasCorta[][];
int dist[];
int distFinal[];
boolean cambioDist[];                                   // indica cambio de distancia durante el algoritmo

private String mensaje = new String("");

private int numcambioDist =0;
private int colindantes=0;
int paso=0;                                                     // numero de iteracion

// informacion usada por el algoritmo para encontrar el siguiente nodo con minima distancia
int mindist;
int minnodo;
int minstart;
int minend;
int numnodos=0;                                         // numero de nodos
int nodo1, nodo2;                                       // numero de nodos envueltos en la accion

boolean ejecutando = false;
int algoritmo;

int o_idcalle = 0;
int o_numero = 0;
int d_idcalle = 0;
int d_numero = 0;
int categ=0;
Vector sitios = null;
public int idSitioMasCercano = 0;

public Dijkstra() {
        debug("Dijkstra()");
        this.MAXNODOS = getNumeroNodos();
                this.MAX = this.MAXNODOS + 1;
                //debug("MAXNODOS=" + MAXNODOS);
                idNodos = new int[MAX];
                idNodosRutaOptima = new int[MAX];
                peso = new int[MAX][MAX];
                rutaMasCorta = new boolean[MAX][MAX];
                dist = new int[MAX];
                distFinal = new int[MAX];
                cambioDist = new boolean[MAX];
                numNodosRutaOptima = 0;

        cargaGrafo();
        }

public Dijkstra(int o_idcalle, int o_numero, int d_idcalle, int d_numero) {
        debug("Dijkstra(args), v1.103");

        this.MAXNODOS = getNumeroNodos();
                this.MAX = this.MAXNODOS + 1;
                //debug("MAXNODOS=" + MAXNODOS);
                idNodos = new int[MAX];
                idNodosRutaOptima = new int[MAX];
                peso = new int[MAX][MAX];
```

```
            rutaMasCorta = new boolean[MAX][MAX];
            dist = new int[MAX];
            distFinal = new int[MAX];
            cambioDist = new boolean[MAX];
            numNodosRutaOptima = 0;

cargaGrafo();

this.o_idcalle = o_idcalle;
this.o_numero = o_numero;
this.d_idcalle = d_idcalle;
this.d_numero = d_numero;

setNodoOrig(encontrarNodoCercano(o_idcalle, o_numero));
setNodoDest(encontrarNodoCercano(d_idcalle, d_numero));

debug("NO="+this.nodoOrig);
debug("ND="+this.nodoDest);


if((this.nodoOrig == -1) || (this.nodoDest == -1)){
            callesRutaOptima = new String[2];
            callesRutaOptima[0] = "";
            callesRutaOptima[1] = "";
}

if(this.nodoOrig == -1){
            callesRutaOptima[numNodosRutaOptima++] = "La direccion origen se encuentra bloqueada.";
}

if(this.nodoDest == -1){
            callesRutaOptima[numNodosRutaOptima++] = "La direccion destino se encuentra bloqueada.";
}

if((this.nodoOrig != -1) && (this.nodoDest != -1)){
            ejecutarAlg();
}
}// construc con 4 args

public Dijkstra(int o_idcalle, int o_numero, int categ) {
debug("Dijkstra(args), v1.104");

this.MAXNODOS = getNumeroNodos();
            this.MAX = this.MAXNODOS + 1;
            //debug("MAXNODOS=" + MAXNODOS);
            idNodos = new int[MAX];
            idNodosRutaOptima = new int[MAX];
            peso = new int[MAX][MAX];
            rutaMasCorta = new boolean[MAX][MAX];
            dist = new int[MAX];
            distFinal = new int[MAX];
            cambioDist = new boolean[MAX];
            numNodosRutaOptima = 0;

cargaGrafo();

this.o_idcalle = o_idcalle;
this.o_numero = o_numero;
this.categ = categ;

setNodoOrig(encontrarNodoCercano(o_idcalle, o_numero));
debug("NO="+this.nodoOrig);

sitios = new Vector();
```

```
            try{
            ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_sitio null, " + categ);

            if(rs1!=null){
                        String str1 = "";
                        String str3 = "";
                        String str5 = "";

                        while(rs1.next()){
                        str1 = rs1.getString(1).trim();
                        str3 = rs1.getString(3).trim();
                        str5 = rs1.getString(5).trim();

                        sitios.addElement(new String(str1 + "!" + str3 + "!" + str5));
            }
            }
            else
            debug("rs1 fue null");

            rs1.close();
            //cierraBD();
            }catch(Exception e){
            debug("Error en Dijkstra(3 args):" + e.toString() + "_Msg=" + e.getMessage());
            e.printStackTrace(System.out);
            }


if(this.nodoOrig == -1){
            callesRutaOptima = new String[1];
            callesRutaOptima[0] = "";
}

if(this.nodoOrig == -1){
            callesRutaOptima[numNodosRutaOptima++] = "La direccion origen se encuentra bloqueada.";
}

if(this.nodoOrig != -1){
            ejecutarAlg();
}
}// construc con 3 args



public int getNumNodosRutaOptima(){
            return numNodosRutaOptima;
}

public int[] getIdNodosRutaOptima(){
            return idNodosRutaOptima;
}

private int getNumeroNodos(){
   try{
   ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_numeronodos");

      if(rs1!=null){
            if(rs1.next())
            this.numnodos = rs1.getInt(1);
   }
      else
            debug("rs1 fue null");

      rs1.close();
      //cierraBD();
```

```
    }catch(Exception e){
        debug("Error en getNumeroNodos():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }
    return this.numnodos;
}

private int setIdNumNodos(){
    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_nodo");
    int id=0;

        if(rs1!=null){
            while(rs1.next())
            idNodos[id++] = rs1.getInt(1);
    }
        else
            debug("rs1 fue null");

        rs1.close();
        //cierraBD();
    }catch(Exception e){
        debug("Error en setIdNumNodos():" + e.toString() + "_Msg=" + e.getMessage());
        e.printStackTrace(System.out);
    }
    return this.numnodos;
}


private void cargaGrafo(){
            inicializar();

            for(int i=0; i<MAXNODOS; i++) {
                        for (int j=0; j<MAXNODOS;j++)
                        peso[i][j]=0;
            }

    try{
    ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_sp_flujonodos");

                        int n1;
                        int n2;
                        int f1;
                        int f2;
                        int t;

    if(rs1!=null){
          while(rs1.next()){
            n1 = rs1.getInt(1);
            n2 = rs1.getInt(2);
            f1 = rs1.getInt(3);
            f2 = rs1.getInt(4);
            t = rs1.getInt(5);

            if((f2 == 0) || (t == 0))
                        peso[n1][n2] = 0;
            else
                        peso[n1][n2] = (int)((9*(11-f2)) + (25*t));

            if((f1 == 0) || (t == 0))
                        peso[n2][n1] = 0;
            else
                        peso[n2][n1] = (int)((9*(11-f1)) + (25*t));
```

```
                    //debug("peso[" + n1 + "][" + n2 + "] = " + peso[n1][n2]);
                    //debug("peso[" + n2 + "][" + n1 + "] = " + peso[n2][n1]);
                  }
               }
               else
                    debug("rs1 fue null");

        rs1.close();
               //cierraBD();
               ejecutarPaso();
            }catch(Exception e){
               debug("Error en cargaGrafo():" + e.toString() + "_Msg=" + e.getMessage());
               e.printStackTrace(System.out);
            }
         }


   public void ejecutarAlg() {
                    inicializarAlg();
                    ejecutando = true;
                    for(int i=0; i<numnodos; i++){
                              ejecutarSiguientePasoAlg();
                    }
   }
   /*
   public void ejecutarPrimerPasoAlg() {
          // le permite pasar por el algoritmo
                    inicializarAlg();
                    ejecutando = true;
                    ejecutarSiguientePasoAlg();
   }
          */
   public void inicializarAlg() {
                    inicializar();

                    for(int i=0; i<MAXNODOS; i++) {
                              dist[i]=-1;
                              distFinal[i]=-1;
                    }
      dist[nodoInicial]=0;
                    distFinal[nodoInicial]=0;
                    paso=0;
   }

          public void inicializar(){
                    for (int i=0;i<MAXNODOS;i++) {
                              for (int j=0; j<MAXNODOS;j++)
                              rutaMasCorta[i][j]=false;
                    }

                    ejecutando = false;
                    algoritmo=DIJKSTRA;
          }

          private int encontrarNodoCercano(int idcalle, int numero){
                    int nodoCercano = -1;

              int nodo1 = -1;
              int nodo2 = -1;
              int flujo1 = -1;
              int flujo2 = -1;
              int numIni = -1;
              int numFin = -1;
              int trafico = -1;
```

```
                try{
                ResultSet rs1 = ejecutaQuery("LOCALIZACION..loc_spr_buscaNodos " + idcalle + ", " + numero);

                if(rs1!=null){
                if(rs1.next()){
                        nodo1 = rs1.getInt(1);
                        nodo2 = rs1.getInt(2);
                        flujo1 = rs1.getInt(3);
                        flujo2 = rs1.getInt(4);
                        numIni = rs1.getInt(5);
                        numFin = rs1.getInt(6);
                        trafico = rs1.getInt(7);
                }
                }
                else
                debug("rs1 fue null");

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en encontrarNodoCercano():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }

                if(trafico == 0){
                        // mandar mensaje de que la cuadra se encuentra bloqueada,
                        // por lo que tiene que situarse en otra posicion
                        nodoCercano = -1;
                }
                else if((flujo1 == 0) && (flujo2 == 0)){
                        // mandar mensaje de error,
                        // por lo que tiene que situarse en otra posicion
                        nodoCercano = -1;
                }
                else if(flujo1 == 0){
                        nodoCercano = nodo1;
                }
                else if(flujo2 == 0){
                        nodoCercano = nodo2;
                }
                else{
                        if((numero >= numIni) && (numero <= (numIni + (numFin-numIni)/2)))
                                nodoCercano = nodo1;
                        else
                                nodoCercano = nodo2;
                }

                return nodoCercano;
        }

public String[] getNombresCalles(){
        return callesRutaOptima;
}

public void setNodoInicial(int nodoInicial){
        this.nodoInicial = nodoInicial;
}

public void setNodoOrig(int nodoOrig){
        this.nodoOrig = nodoOrig;
        setNodoInicial(nodoOrig);
}
```

```
public void setNodoDest(int nodoDest){
	this.nodoDest = nodoDest;
}

private void imprimirRuta(){
	debug("Ruta:");
	int nodoAnt = nodoDest;

	/*
	for(int i=0; i<MAXNODOS; i++) {
			for (int j=0; j<MAXNODOS;j++){
				debug(i + "," + j + "=" + rutaMasCorta[i][j]);
			}
		}
		*/

		for(int i=0; i<MAXNODOS; i++) {
			for (int j=0; j<MAXNODOS;j++){
		if(rutaMasCorta[j][nodoAnt] == true){
			//debug("nodo " + nodoAnt);
			nodoAnt = j;
			if(nodoAnt == nodoOrig){
				i = MAXNODOS;
			}
			break;
		}
	}

		}
}

public void setRutaOptima(){
	debug("setRutaOptima:");

	int nodoAnt = nodoDest;
	idNodosRutaOptima[++numNodosRutaOptima] = nodoAnt;
	/*
	for(int i=0; i<MAXNODOS; i++) {
			for (int j=0; j<MAXNODOS;j++){
				debug(i + "," + j + "=" + rutaMasCorta[i][j]);
			}
		}
		*/

		for(int i=0; i<MAXNODOS; i++) {
			for (int j=0; j<MAXNODOS;j++){
		if(rutaMasCorta[j][nodoAnt] == true){
			//debug("nodo " + nodoAnt);
			nodoAnt = j;
			++numNodosRutaOptima;
			idNodosRutaOptima[numNodosRutaOptima] = nodoAnt;
			if(nodoAnt == nodoOrig){
				i = MAXNODOS;
			}
			break;
		}
	}
		}

		++numNodosRutaOptima;

		debug("numNodosRutaOptima-"+numNodosRutaOptima);

		for(int i=numNodosRutaOptima-1; i>0; i--)  //imprimiendo los idnodos de la ruta optima
```

```
                debug("XXXXXXXXXXXXX.-"+idNodosRutaOptima[i]);


            callesRutaOptima = new String[numNodosRutaOptima];
            int nOrig = 0;
            int nDest = 0;

            int idCalle = 0;
            String nomCalle = "";
            String nomColonia = "";
            int noInic = 0;
            int noFinal = 0;
            String sentido = "";
            String trafico = "";

            int cont = 0;

            for(int i=numNodosRutaOptima; i>0; i--){
                try{
                    ResultSet rs1;

                    if(i == numNodosRutaOptima){
                        nDest = idNodosRutaOptima[i-1];
                        rs1    =    ejecutaQuery("LOCALIZACION..loc_sp_nombreCalleOrig    "    +
o_idcalle + ", " + o_numero + ", " + nDest);
                    }
                    else if(i == 1){
                        nOrig = idNodosRutaOptima[i];
                        rs1    =    ejecutaQuery("LOCALIZACION..loc_sp_nombreCalleDest    "    +
d_idcalle + ", " + d_numero + ", " + nOrig);
                    }
                    else{
                        nOrig = idNodosRutaOptima[i];
                        nDest = idNodosRutaOptima[i-1];
                        rs1 = ejecutaQuery("LOCALIZACION..loc_sp_nombreCalle " + nOrig + ", "
+ nDest);
                    }//else

                    if(rs1!=null){
                if(rs1.next()){
                                idCalle = rs1.getInt(1);
                                nomCalle = rs1.getString(2).trim();
                                nomColonia = rs1.getString(3).trim();
                                noInic = rs1.getInt(4);
                                noFinal = rs1.getInt(5);
                                sentido = rs1.getString(6).trim();
                                trafico = rs1.getString(7).trim();

                                callesRutaOptima[cont++] = nomCalle + " (" + sentido + ")";
                }
                }
                else{
                debug("rs1 fue null");
                callesRutaOptima[cont++] = "Error";
            }

            debug("X--X--X.-"+callesRutaOptima[cont - 1]);

                rs1.close();
                //cierraBD();
                }catch(Exception e){
                debug("Error en setRutaOptima():" + e.toString() + "_Msg=" + e.getMessage());
                e.printStackTrace(System.out);
                }
```

```
                }//for
    }//setRutaOptima()

    public void ejecutarSiguientePasoAlg() {
            // Calcula un paso mas del algoritmo (encuentra un camino mas corto al siguiente nodo).
            //debug("ejecutarSiguientePasoAlg()");
            distFinal[minend]=mindist;
                    rutaMasCorta[minstart][minend]=true;

                    if(categ==0){
                    if ((minend == nodoDest) && (ejecutando == true)) {
                                    imprimirRuta();
                                    setRutaOptima();
                                    ejecutando = false;
                                    return;
                            }
                    }
                    else{
                            Enumeration e = sitios.elements();
                            String strSit = "";
                            int nodod = -1;
                            while(e.hasMoreElements()){
                                    strSit = (String)e.nextElement();
                                    nodod = Integer.parseInt(     String.valueOf(       encontrarNodoCercano(
                                                                                        Integer.parseInt(
strSit.substring(strSit.indexOf("!")+1, strSit.lastIndexOf("!"))) ,

            Integer.parseInt(strSit.substring(strSit.lastIndexOf("!")+1, strSit.length()))        )            )            );
                                    if ((minend == nodod) && (ejecutando == true) ) {
                                    setNodoDest(nodod);
                                    idSitioMasCercano = Integer.parseInt(strSit.substring(0, strSit.indexOf("!")));
                                    this.d_idcalle       =       Integer.parseInt(       strSit.substring(strSit.indexOf("!")+1,
strSit.lastIndexOf("!")) );
                                    this.d_numero     =     Integer.parseInt(     strSit.substring(strSit.lastIndexOf("!")+1,
strSit.length()) );
                                    debug("NODODEST--"+this.nodoDest + " idSitio+Cercano--"+idSitioMasCercano);
                                            imprimirRuta();
                                            setRutaOptima();
                                            ejecutando = false;
                                            return;
                                    }
                            }
                    }

                    paso++;
                    ejecutarPaso();

    }

    //myp
    public void ejecutarPaso() {
       mindist         = 0;
                    minstart = 0;
                    minend = MAXNODOS;

       for(int i=0; i<MAXNODOS; i++)
            cambioDist[i]=false;

       numcambioDist=0;
       colindantes=0;


                    for (int i=0; i<numnodos; i++)
```

```
                        for (int j=0; j<numnodos; j++)
                        if (peso [i][j]>0) {
                        // si el algoritmo se esta ejecutando entonces hacer el siguiente paso para esta arista
                        if (ejecutando)
                                iniciarPaso(i, j);
                        }


                if (ejecutando)
                        finalizarPaso();

    }

        public void iniciarPaso(int i, int j) {
        // mas algoritmos pueden ser aniadidos
                if (algoritmo==DIJKSTRA)
                iniciarPasoDijkstra(i, j);
    }
    public void finalizarPaso() {
        // mas algoritmos pueden ser aniadidos
                if (algoritmo==DIJKSTRA)
                finalizarPasoDijkstra();
    }

    public void iniciarPasoDijkstra(int i, int j) {
        // checar que arista entre nodo i y nodo j esta cerca de la arista s para
        //escoger durante este paso del algoritmo
        // checar si el nodo j tiene la siguiente minima distancia al nodo_inicial
                if ( (distFinal[i] != -1) && (distFinal[j] == -1) ) {
                        if ( (dist[j]==-1) || (dist[j]>=(dist[i]+peso[i][j])) ) {
        if ( (dist[i]+peso[i][j])<dist[j] ) {
                cambioDist[j]=true;
                numcambioDist++;
        }
                        dist[j] = dist[i]+peso[i][j];
                        if ( (mindist==0) || (dist[j]<mindist) ) {
                        mindist=dist[j];
                        minstart=i;
                        minend=j;
                        }
                        }
                }
    }

    public void finalizarPasoDijkstra() {
                if ((ejecutando) && (mindist==0))
                {
                        int nalcanzable = 0;

                        for (int i=0; i<numnodos; i++)
                        if (distFinal[i] > 0)
                                        nalcanzable++;

                        if (nalcanzable == 0){
                                debug("El algoritmo ha terminado, no hay nodos alcanzables desde el nodo
inicial.");
        }
        else if (nalcanzable< (numnodos-1)){
                        debug("El algoritmo ha terminado, No hay caminos del nodo_inicial a ningun otro nodo.");
                        callesRutaOptima = new String[1];
                        callesRutaOptima[0] = "No se encontraron rutas disponibles.";
                        numNodosRutaOptima = 1;
                        }else{
                        debug("El algoritmo ha terminado.");
```

```
                    }

                    }
    }

        private ResultSet ejecutaQuery(String sp){
                try{
                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection("jdbc:odbc:odbc_monitor", "", "");
                stmt = con.createStatement();
                rs = stmt.executeQuery(sp);
           }catch(Exception e){
              debug("Error en ejecutaQuery("+sp+"):" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
           return rs;
        }

        private void cierraBD(){
                try{
                        rs.close();
                        stmt.close();
                        con.close();
                }catch(Exception e){
              debug("Error en cierraBD():" + e.toString() + "_Msg=" + e.getMessage());
              e.printStackTrace(System.out);
           }
        }

    private void debug(String texto){
        System.out.println(texto);
    }

    public void cargarEjemplo() {
        inicializar();
                numnodos=10;

                for(int i=0; i<MAXNODOS; i++) {
                        for (int j=0; j<MAXNODOS;j++)
                        peso[i][j]=0;
                }

        peso[0][2]=13;      peso[2][0]=13;
                peso[0][4]=16;      peso[4][0]=16;
                peso[0][5]=8;       peso[5][0]=8;
                peso[1][3]=6;       peso[3][1]=6;
                peso[1][5]=10;      peso[5][1]=10;
                peso[2][3]=14;      peso[3][2]=14;
                peso[2][5]=11;      peso[5][2]=11;
                peso[3][4]=5;       peso[4][3]=5;
                peso[3][5]=17;      peso[5][3]=17;
                peso[4][5]=7;       peso[5][4]=7;


                ejecutarPaso();
    }

}
```

Apéndice C

Script de Tablas y Servicios de Base de Datos (SQL)

Tablas

```
CREATE TABLE [dbo].[loc_busqueda] (
	[idUsr] [int] NOT NULL ,
	[stsDir] [int] NULL ,
	[URL1] [varchar] (256) COLLATE Traditional_Spanish_CI_AS NULL ,
	[URL2] [varchar] (256) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_catCalle] (
	[idCalle] [int] NOT NULL ,
	[nombre] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_catCategoria] (
	[idCategoria] [int] IDENTITY (1, 1) NOT FOR REPLICATION  NOT NULL ,
	[Nombre] [char] (70) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_catColonia] (
	[idColonia] [int] IDENTITY (1, 1) NOT FOR REPLICATION  NOT NULL ,
	[nombre] [char] (30) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[idDeleg] [int] NOT NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_catDelegacion] (
	[idDeleg] [int] IDENTITY (1, 1) NOT NULL ,
	[nombre] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_catsentido] (
	[idsentido] [int] IDENTITY (1, 1) NOT FOR REPLICATION  NOT NULL ,
	[descr] [char] (3) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_cattrafico] (
	[idTrafico] [int] IDENTITY (1, 1) NOT FOR REPLICATION  NOT NULL ,
	[descr] [char] (20) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_cuadra] (
	[idCalle] [int] NOT NULL ,
	[idNodo1] [int] NOT NULL ,
	[idNodo2] [int] NOT NULL ,
	[flujo1] [int] NOT NULL ,
	[flujo2] [int] NOT NULL ,
	[noInic] [int] NOT NULL ,
	[noFinal] [int] NOT NULL ,
	[idColonia] [int] NOT NULL ,
	[sent1] [int] NOT NULL ,
	[sent2] [int] NOT NULL ,
	[trafico] [int] NOT NULL
) ON [PRIMARY]
GO
```

```
CREATE TABLE [dbo].[loc_nodo] (
	[idNodo] [int] IDENTITY (1, 1) NOT FOR REPLICATION  NOT NULL ,
	[nombre] [char] (50) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_sitio] (
	[idSitio] [int] IDENTITY (1, 1) NOT FOR REPLICATION  NOT NULL ,
	[nombre] [char] (60) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[idCalle] [int] NOT NULL ,
	[numero] [int] NULL ,
	[idColonia] [int] NOT NULL ,
	[telefono] [int] NULL ,
	[idDeleg] [int] NOT NULL ,
	[idCategoria] [int] NOT NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[loc_usuario] (
	[idUsuario] [int] IDENTITY (1, 1) NOT FOR REPLICATION  NOT NULL ,
	[login] [varchar] (10) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[pwd] [varchar] (10) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_busqueda] ADD
	CONSTRAINT [DF_loc_busqueda_dir1] DEFAULT (0) FOR [stsDir],
	CONSTRAINT [PK_loc_busqueda] PRIMARY KEY  CLUSTERED
	(
		[idUsr]
	)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_catCalle] ADD
	CONSTRAINT [PK_loc_calle] PRIMARY KEY  CLUSTERED
	(
		[idCalle]
	)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_catCategoria] ADD
	CONSTRAINT [PK_loc_catcategoria] PRIMARY KEY  CLUSTERED
	(
		[idCategoria]
	)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_catColonia] ADD
	CONSTRAINT [PK_loc_colonia] PRIMARY KEY  CLUSTERED
	(
		[idColonia]
	)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_catDelegacion] ADD
	CONSTRAINT [PK_loc_delegacion] PRIMARY KEY  CLUSTERED
	(
		[idDeleg]
	)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_catsentido] ADD
	CONSTRAINT [PK_loc_catsentido] PRIMARY KEY  CLUSTERED
```

```
		(
			[idsentido]
		)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_cattrafico] ADD
		CONSTRAINT [PK_loc_cattrafico] PRIMARY KEY  CLUSTERED
		(
			[idTrafico]
		)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_cuadra] ADD
		CONSTRAINT [PK_loc_cuadra] PRIMARY KEY  CLUSTERED
		(
			[idCalle],
			[idNodo1],
			[idNodo2],
			[idColonia]
		)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_nodo] ADD
		CONSTRAINT [PK_loc_catnodo] PRIMARY KEY  CLUSTERED
		(
			[idNodo]
		)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_sitio] ADD
		CONSTRAINT [PK_loc_sitio] PRIMARY KEY  CLUSTERED
		(
			[idSitio]
		)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_usuario] ADD
		CONSTRAINT [PK_loc_usuario] PRIMARY KEY  CLUSTERED
		(
			[idUsuario]
		)  ON [PRIMARY]
GO

ALTER TABLE [dbo].[loc_catColonia] ADD
		CONSTRAINT [FK_loc_catColonia_loc_catDelegacion] FOREIGN KEY
		(
			[idDeleg]
		) REFERENCES [dbo].[loc_catDelegacion] (
			[idDeleg]
		)
GO

ALTER TABLE [dbo].[loc_cuadra] ADD
		CONSTRAINT [FK_loc_cuadra_loc_catColonia] FOREIGN KEY
		(
			[idColonia]
		) REFERENCES [dbo].[loc_catColonia] (
			[idColonia]
		),
		CONSTRAINT [FK_loc_cuadra_loc_catsentido] FOREIGN KEY
		(
			[sent1]
		) REFERENCES [dbo].[loc_catsentido] (
			[idsentido]
```

```
	),
	CONSTRAINT [FK_loc_cuadra_loc_catsentido1] FOREIGN KEY
	(
		[sent2]
	) REFERENCES [dbo].[loc_catsentido] (
		[idsentido]
	),
	CONSTRAINT [FK_loc_cuadra_loc_cattrafico] FOREIGN KEY
	(
		[trafico]
	) REFERENCES [dbo].[loc_cattrafico] (
		[idTrafico]
	),
	CONSTRAINT [FK_loc_grafocalle_loc_calle] FOREIGN KEY
	(
		[idCalle]
	) REFERENCES [dbo].[loc_catCalle] (
		[idCalle]
	),
	CONSTRAINT [FK_loc_grafocalle_loc_nodo] FOREIGN KEY
	(
		[idNodo1]
	) REFERENCES [dbo].[loc_nodo] (
		[idNodo]
	),
	CONSTRAINT [FK_loc_grafocalle_loc_nodo1] FOREIGN KEY
	(
		[idNodo2]
	) REFERENCES [dbo].[loc_nodo] (
		[idNodo]
	)
GO

ALTER TABLE [dbo].[loc_sitio] ADD
	CONSTRAINT [FK_loc_sitio_loc_catCalle] FOREIGN KEY
	(
		[idCalle]
	) REFERENCES [dbo].[loc_catCalle] (
		[idCalle]
	),
	CONSTRAINT [FK_loc_sitio_loc_catCategoria] FOREIGN KEY
	(
		[idCategoria]
	) REFERENCES [dbo].[loc_catCategoria] (
		[idCategoria]
	),
	CONSTRAINT [FK_loc_sitio_loc_catColonia] FOREIGN KEY
	(
		[idColonia]
	) REFERENCES [dbo].[loc_catColonia] (
		[idColonia]
	),
	CONSTRAINT [FK_loc_sitio_loc_catDelegacion] FOREIGN KEY
	(
		[idDeleg]
	) REFERENCES [dbo].[loc_catDelegacion] (
		[idDeleg]
	)
GO
```

Procedimientos Almacenados (Servicios)

```
CREATE PROCEDURE dbo.loc_sp_cuadracalle
/**************************************
Consulta de cuadras
**************************************/
        @idColonia          INT = NULL
AS
BEGIN
        if @idColonia is null
                SELECT  DISTINCT
                        A.idCalle,
                        B.nombre 'nomCalle'
                FROM    localizacion..loc_cuadra A (NOLOCK),
                        localizacion..loc_catCalle B (NOLOCK),
                        localizacion..loc_catColonia D(NOLOCK)
                WHERE   A.idCalle = B.idCalle
                AND     A.idColonia = D.idColonia
                ORDER BY 2
        else
                SELECT  DISTINCT
                        A.idCalle,
                        B.nombre 'nomCalle'
                FROM    localizacion..loc_cuadra A (NOLOCK),
                        localizacion..loc_catCalle B (NOLOCK),
                        localizacion..loc_catColonia D(NOLOCK)
                WHERE   A.idCalle = B.idCalle
                AND     A.idColonia = D.idColonia
                AND     @idColonia = A.idColonia
                ORDER BY 2
END


CREATE PROCEDURE dbo.loc_sp_flujonodos
/**************************************
Consulta de flujos de nodo a nodo
**************************************/
AS
BEGIN
        SELECT  idNodo1,
                idNodo2,
                flujo1,
                flujo2,
                trafico
        FROM    localizacion..loc_cuadra (NOLOCK)
END


CREATE PROCEDURE dbo.loc_sp_nombreCalle
/**************************************
Consulta de nombre de calle situada entre dos nodos dados
**************************************/
        @idNodoOri          INT,
        @idNodoDes          INT
AS
BEGIN
        SELECT  A.idCalle,
                B.nombre 'nomCalle',
                D.nombre 'nomColonia',
                A.noInic,
                A.noFinal,
```

```
		E.descr 'sentido',
		F.descr 'trafico'
	FROM	localizacion..loc_cuadra A (NOLOCK),
		localizacion..loc_catCalle B (NOLOCK),
		localizacion..loc_catColonia D(NOLOCK),
		localizacion..loc_catsentido E(NOLOCK),
		localizacion..loc_cattrafico F(NOLOCK)
	WHERE	A.idCalle = B.idCalle
	AND	A.idColonia = D.idColonia
	AND	E.idsentido =	CASE
					WHEN (A.idNodo1 = @idNodoOri AND A.idNodo2 = @idNodoDes) THEN
A.sent2
					WHEN (A.idNodo2 = @idNodoOri AND A.idNodo1 = @idNodoDes) THEN
A.sent1
				END
	AND	A.trafico = F.idtrafico
	AND	(	(A.idNodo1 = @idNodoOri AND A.idNodo2 = @idNodoDes)
		OR	(A.idNodo1 = @idNodoDes AND A.idNodo2 = @idNodoOri)
		)
END


CREATE PROCEDURE dbo.loc_sp_nombreCalle2
/**************************************
Consulta de cuadras
**************************************/
	@idCalleOri		INT,
	@numeroOri		INT,
	@idNodoDes		INT
AS
BEGIN
	SELECT	A.idCalle,
		B.nombre 'nomCalle',
		D.nombre 'nomColonia',
		A.noInic,
		A.noFinal,
		E.descr 'sentido',
		F.descr 'trafico'
	FROM	localizacion..loc_cuadra A (NOLOCK),
		localizacion..loc_catCalle B (NOLOCK),
		localizacion..loc_catColonia D(NOLOCK),
		localizacion..loc_catsentido E(NOLOCK),
		localizacion..loc_cattrafico F(NOLOCK)
	WHERE	A.idCalle = B.idCalle
	AND	A.idColonia = D.idColonia
	AND	E.idsentido =	CASE
					WHEN A.idNodo2 = @idNodoDes THEN A.sent2
					WHEN A.idNodo1 = @idNodoDes THEN A.sent1
				END
	AND	A.trafico = F.idtrafico
	AND	A.idCalle = @idCalleOri
	AND	@numeroOri between A.noInic AND A.noFinal
	AND	(A.idNodo1 = @idNodoDes OR A.idNodo2 = @idNodoDes)

END


CREATE PROCEDURE dbo.loc_sp_nombreCalleDest
/**************************************
Consulta de cuadras
**************************************/
	@idCalleDes		INT,
	@numeroDes		INT,
	@idNodoOri		INT
```

```
AS
BEGIN
        SELECT  A.idCalle,
                B.nombre 'nomCalle',
                D.nombre 'nomColonia',
                A.noInic,
                A.noFinal,
                E.descr 'sentido',
                F.descr 'trafico'
        FROM    localizacion..loc_cuadra A (NOLOCK),
                localizacion..loc_catCalle B (NOLOCK),
                localizacion..loc_catColonia D(NOLOCK),
                localizacion..loc_catsentido E(NOLOCK),
                localizacion..loc_cattrafico F(NOLOCK)
        WHERE   A.idCalle = B.idCalle
        AND     A.idColonia = D.idColonia
        AND     E.idsentido =       CASE
                                        WHEN A.idNodo2 = @idNodoOri THEN A.sent1
                                        WHEN A.idNodo1 = @idNodoOri THEN A.sent2
                                    END
        AND     A.trafico = F.idtrafico
        AND     A.idCalle = @idCalleDes
        AND     @numeroDes between A.noInic AND A.noFinal
        AND     (A.idNodo1 = @idNodoOri OR A.idNodo2 = @idNodoOri)

END


CREATE PROCEDURE dbo.loc_sp_nombreCalleOrig
/*************************************
Consulta de cuadras
*************************************/
        @idCalleOri         INT,
        @numeroOri          INT,
        @idNodoDes          INT
AS
BEGIN
        SELECT  A.idCalle,
                B.nombre 'nomCalle',
                D.nombre 'nomColonia',
                A.noInic,
                A.noFinal,
                E.descr 'sentido',
                F.descr 'trafico'
        FROM    localizacion..loc_cuadra A (NOLOCK),
                localizacion..loc_catCalle B (NOLOCK),
                localizacion..loc_catColonia D(NOLOCK),
                localizacion..loc_catsentido E(NOLOCK),
                localizacion..loc_cattrafico F(NOLOCK)
        WHERE   A.idCalle = B.idCalle
        AND     A.idColonia = D.idColonia
        AND     E.idsentido =       CASE
                                        WHEN A.idNodo2 = @idNodoDes THEN A.sent2
                                        WHEN A.idNodo1 = @idNodoDes THEN A.sent1
                                    END
        AND     A.trafico = F.idtrafico
        AND     A.idCalle = @idCalleOri
        AND     @numeroOri between A.noInic AND A.noFinal
        AND     (A.idNodo1 = @idNodoDes OR A.idNodo2 = @idNodoDes)

END


CREATE PROCEDURE dbo.loc_sp_numerocuadras
```

```
/*************************************
Consulta de numero de cuadras
*************************************/
AS
DECLARE
        @flujos Integer
BEGIN
        SELECT  @flujos = count(*)
        FROM    localizacion..loc_cuadra (NOLOCK)
        WHERE   flujo1 > 0

        SELECT  (@flujos + count(*)) 'cuadras'
        FROM    localizacion..loc_cuadra (NOLOCK)
        WHERE   flujo2 > 0

END


CREATE PROCEDURE dbo.loc_sp_numeronodos
/*************************************
Consulta de numero de nodos
*************************************/
AS
BEGIN
        SELECT  count(*) 'nodos'
        FROM    localizacion..loc_nodo (NOLOCK)
END


CREATE PROCEDURE dbo.loc_sp_validausuario
/*************************************
Verifica un usuario y password dados
*************************************/
        @login          VARCHAR(10) = null,
        @password       VARCHAR(10) = null
AS
BEGIN
        if EXISTS (     SELECT  idUsuario
                        FROM    localizacion.dbo.loc_usuario (NOLOCK)
                        WHERE   UPPER(LTRIM(RTRIM(login))) like UPPER(LTRIM(RTRIM(@login)))
                        )
        Begin
                if EXISTS (     SELECT  idUsuario
                                FROM    localizacion.dbo.loc_usuario (NOLOCK)
                                WHERE   UPPER(LTRIM(RTRIM(login))) like UPPER(LTRIM(RTRIM(@login)))
                                AND     pwd like @password
                                )
                        SELECT  idUsuario
                        FROM    localizacion.dbo.loc_usuario (NOLOCK)
                        WHERE   UPPER(LTRIM(RTRIM(login))) like UPPER(LTRIM(RTRIM(@login)))
                        AND     pwd like @password
                else
                        SELECT  -1
        End
        else
                SELECT  -2
END


CREATE PROCEDURE dbo.loc_spd_busqueda
/*************************************
Elimina una busqueda de usuario
*************************************/
        @idUsr          INT
```

```
AS
BEGIN
	DELETE  localizacion.dbo.loc_busqueda
	WHERE   idUsr = @idUsr

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar de localizacion.dbo.loc_busqueda',18,2) WITH SETERROR
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spd_calle
/*************************************
Elimina una calle
*************************************/
	@idcalle  INT
AS
BEGIN
	DELETE localizacion..loc_cuadra
	WHERE   idcalle = @idcalle

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_cuadra',18,2) WITH SETERROR
		Return 0
	End

	DELETE localizacion..loc_sitio
	WHERE   idcalle = @idcalle

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_sitio',18,2) WITH SETERROR
		Return 0
	End

	DELETE localizacion..loc_catcalle
	WHERE idcalle = @idcalle

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_catcalle',18,2) WITH SETERROR
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spd_categoria
/*************************************
Elimina una calle
*************************************/
	@idcategoria	INT
AS
BEGIN
	DELETE localizacion..loc_sitio
	WHERE   idcategoria = @idcategoria

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_sitio',18,2) WITH SETERROR
		Return 0
	End
```

```
	DELETE localizacion..loc_catcategoria
	WHERE idcategoria = @idcategoria

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_catcategoria',18,2) WITH SETERROR
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spd_colonia
/**************************************
Elimina una colonia
**************************************/
	@idcolonia		INT
AS
BEGIN
	DELETE localizacion..loc_cuadra
	WHERE	idcolonia = @idcolonia

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_cuadra',18,2) WITH SETERROR
		Return 0
	End

	DELETE localizacion..loc_sitio
	WHERE	idcolonia = @idcolonia

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_sitio',18,2) WITH SETERROR
		Return 0
	End

	DELETE localizacion..loc_catcolonia
	WHERE idcolonia = @idcolonia

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_catcolonia',18,2) WITH SETERROR
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spd_cuadra
/**************************************
Elimina una cuadra
**************************************/
	@idCalle  INT,
	@idNodo1		INT,
	@idNodo2		INT,
	@idColonia		INT
AS
BEGIN
	DELETE	localizacion..loc_cuadra
	WHERE	idCalle = @idCalle
	AND	idNodo1 = @idNodo1
	AND	idNodo2 = @idNodo2
	AND	idColonia = @idColonia
```

```
	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_cuadra',18,2) WITH SETERROR
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spd_delegacion
/**************************************
Elimina una delegacion
**************************************/
	@idDeleg		INT
AS
BEGIN
	DELETE	localizacion..loc_catcolonia
	WHERE	idDeleg = @idDeleg

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_catcolonia',18,2) WITH SETERROR
		Return 0
	End

	DELETE	localizacion..loc_catDelegacion
	WHERE	idDeleg = @idDeleg

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_catDelegacion',18,2) WITH SETERROR
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spd_nodo
/**************************************
Elimina un nodo
**************************************/
	@idnodo  INT
AS
BEGIN
	DELETE localizacion..loc_cuadra
	WHERE	idNodo1 = @idnodo
	OR	idNodo2 = @idnodo

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_cuadra',18,2) WITH SETERROR
		Return 0
	End

	DELETE localizacion..loc_nodo
	WHERE idnodo = @idnodo

	If @@Error <> 0
	Begin
		Raiserror('Error al Eliminar en localizacion..loc_nodo',18,2) WITH SETERROR
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spd_sitio
```

```
/*************************************
Elimina un sitio de interes
*************************************/
        @idSitio   INT
AS
BEGIN
        DELETE   localizacion..loc_sitio
        WHERE    idSitio = @idSitio

        If @@Error <> 0
        Begin
                Raiserror('Error al Eliminar en localizacion..loc_sitio',18,2) WITH SETERROR
                Return 0
        End
END


CREATE PROCEDURE dbo.loc_spi_busqueda
/*************************************
Inserta una busqueda de usuario
*************************************/
        @d1od2              INT,
        @idUsr              INT,
--      @tipoBusq
        @sts                INT,
        @URL                VARCHAR(256)
AS
BEGIN
        if (NOT EXISTS (    SELECT   idUsr
                                FROM     localizacion.dbo.loc_busqueda (NOLOCK)
                                WHERE    idUsr = @idUsr
                            )
           )
        BEGIN
                INSERT INTO localizacion.dbo.loc_busqueda
                VALUES   (          @idUsr,
                                    0,
                                    '',
                                    ''
                            )

                If @@Error <> 0
                Begin
                        Raiserror('Error al Insertar en localizacion.dbo.loc_busqueda',18,2) WITH SETERROR
                        Return 0
                End
        END

        IF @d1od2 = 1
        BEGIN
                UPDATE  localizacion.dbo.loc_busqueda
                SET     stsDir = @sts,
                        URL1 = @URL
                WHERE   idUsr = @idUsr

                If @@Error <> 0
                Begin
                        Raiserror('Error al Actualizar en localizacion.dbo.loc_busqueda',18,2) WITH SETERROR
                        Return 0
                End
        END
        ELSE
        BEGIN
                UPDATE  localizacion.dbo.loc_busqueda
```

```
                SET     stsDir = @sts,
                        URL2 = @URL
                WHERE   idUsr = @idUsr

                If @@Error <> 0
                Begin
                        Raiserror('Error al Actualizar en localizacion.dbo.loc_busqueda',18,2) WITH SETERROR
                        Return 0
                End
        END
END


CREATE PROCEDURE dbo.loc_spi_calle
/*************************************
Inserta una nueva calle
*************************************/
        @nombre VARCHAR(30)
AS
DECLARE
        @maxIndex       INT
BEGIN
        --BEGIN TRAN inserta_calle
        SELECT @maxIndex = indice
        FROM localizacion..loc_indices (NOLOCK)
        WHERE idIndex = 1

        SELECT @maxIndex = @maxIndex + 1

        UPDATE localizacion..loc_indices
        SET indice = @maxIndex
        WHERE idIndex = 1

        If @@Error <> 0
        Begin
                Raiserror('Error al Actualizar en localizacion..loc_indices',18,2) WITH SETERROR
                --ROLLBACK inserta_calle
                Return 0
        End

        INSERT INTO localizacion..loc_catcalle
        VALUES (@maxIndex,@nombre)

        If @@Error <> 0
        Begin
                Raiserror('Error al Insertar en localizacion..loc_catcalle',18,2) WITH SETERROR
                --ROLLBACK inserta_calle
                Return 0
        End

        --COMMIT TRAN inserta_calle
END

CREATE PROCEDURE dbo.loc_spi_colonia
/*************************************
Inserta una nueva colonia
*************************************/
        @nombre         VARCHAR(30),
        @idDeleg        INT
AS
BEGIN
        INSERT INTO localizacion..loc_catcolonia
        VALUES (@nombre, @idDeleg)
```

```
	If @@Error <> 0
	Begin
		Raiserror('Error al Insertar en localizacion..loc_catcolonia',18,2) WITH SETERROR
		Return 0
	End

	--COMMIT TRAN inserta_colonia
END


CREATE PROCEDURE dbo.loc_spi_cuadra
/*************************************
Inserta una nueva cuadra
*************************************/
	@idCalle  INT,
	@idNodo1		INT,
	@idNodo2		INT,
	@flujo1		INT,
	@flujo2		INT,
	@noInic		INT,
	@noFinal INT,
	@idColonia		INT,
	@sent1		INT,
	@sent2		INT,
	@trafico  INT
AS
BEGIN
	if ((NOT EXISTS (	SELECT	idCalle
				FROM	localizacion..loc_cuadra (NOLOCK)
				WHERE	(	(idNodo1 = @idnodo1 AND idNodo2 = @idNodo2)
					OR	(idNodo1 = @idnodo2 AND idNodo2 = @idNodo1)
					)
			)
	   ) AND (@idNodo1 <> @idNodo2))
	BEGIN
		INSERT INTO localizacion..loc_cuadra
		VALUES (@idCalle,
			@idNodo1,
			@idNodo2,
			@flujo1,
			@flujo2,
			@noInic,
			@noFinal,
			@idColonia,
			@sent1,
			@sent2,
			@trafico
			 )

		If @@Error <> 0
		Begin
			Raiserror('Error al Insertar en localizacion..loc_cuadra',18,2) WITH SETERROR
			Return 0
		End
	END
	ELSE
		SELECT 'Ya existe, ok'
END


CREATE PROCEDURE dbo.loc_spi_delegacion
/*************************************
Inserta una nueva delegacion
*************************************/
```

```
	@nombre VARCHAR(30)
AS
BEGIN
	INSERT INTO localizacion..loc_catDelegacion
	VALUES (@nombre)

	If @@Error <> 0
	Begin
		Raiserror('Error al Insertar en localizacion..loc_catDelegacion',18,2) WITH SETERROR
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spi_nodo
/*************************************
Inserta una nueva calle
*************************************/
	@nombre VARCHAR(50)
AS
BEGIN
	INSERT INTO localizacion..loc_nodo
	VALUES (@nombre)

	If @@Error <> 0
	Begin
		Raiserror('Error al Insertar en localizacion..loc_nodo',18,2) WITH SETERROR
		--ROLLBACK inserta_nodo
		Return 0
	End

	--COMMIT TRAN inserta_nodo
END


CREATE PROCEDURE dbo.loc_spi_sitio
/*************************************
Inserta un nuevo sitio de interes
*************************************/
	@nombre		VARCHAR(60),
	@idcalle	INT,
	@numero		INT,
	@idColonia	INT,
	@telefono	INT,
	@iddelegacion	INT,
	@idcategoria	INT
AS
BEGIN
	INSERT INTO localizacion..loc_sitio
	VALUES (@nombre,
		@idcalle,
		@numero,
		@idcolonia,
		@telefono,
		@iddelegacion,
		@idcategoria
		)

	If @@Error <> 0
	Begin
		Raiserror('Error al Insertar en localizacion..loc_sitio',18,2) WITH SETERROR
		Return 0
	End
END
```

```
CREATE PROCEDURE dbo.loc_spr_buscaNodoCercano
/**************************************
Encuentra el nodo mas cercano a una calle y numero dado
**************************************/
        @idcalle int,
        @numero int
AS
BEGIN
        SELECT  CA.nombre,
                CU.idNodo1,
                CU.idNodo2,
                CU.noInic,
                CU.noFinal
        FROM    localizacion..loc_cuadra CU,
                localizacion..loc_catcalle CA
        WHERE   @idcalle = CU.idcalle
        AND     @idcalle = CA.idcalle
        AND     @numero between CU.noInic AND CU.noFinal
END


CREATE PROCEDURE dbo.loc_spr_buscaNodos
/**************************************
Encuentra los dos nodos que cubren una calle y numero dados
**************************************/
        @idcalle int,
        @numero int
AS
BEGIN
        SELECT  CU.idNodo1,
                CU.idNodo2,
                CU.flujo1,
                CU.flujo2,
                CU.noInic,
                CU.noFinal,
                CU.trafico
        FROM    localizacion..loc_cuadra CU (nolock)
        WHERE   @idcalle = CU.idcalle
        AND     @numero between CU.noInic AND CU.noFinal


END


CREATE PROCEDURE dbo.loc_spr_buscaSitio
/**************************************
Encuentra un Sitio de Interes
**************************************/
        @idcalle int,
        @numero int
AS
BEGIN
        SELECT  CA.nombre,
                CU.idNodo1,
                CU.idNodo2,
                CU.noInic,
                CU.noFinal
        FROM    localizacion..loc_cuadra CU (NOLOCK),
                localizacion..loc_catcalle CA (NOLOCK)
        WHERE   @idcalle = CU.idcalle
        AND     @idcalle = CA.idcalle
        AND     @numero between CU.noInic AND CU.noFinal
```

```
END


CREATE PROCEDURE dbo.loc_spr_busqueda
/*************************************
Recupera los datos de una busqueda
*************************************/
        @idUsr          INT
AS
BEGIN
        if (NOT EXISTS (    SELECT   idUsr
                                FROM     localizacion.dbo.loc_busqueda (NOLOCK)
                                WHERE    idUsr = @idUsr
                            )
          )
        BEGIN
                INSERT INTO localizacion.dbo.loc_busqueda
                VALUES   (@idUsr, 0, '', '')

                If @@Error <> 0
                Begin
                        Raiserror('Error al Insertar en localizacion.dbo.loc_busqueda',18,2) WITH SETERROR
                        Return 0
                End
        END

        SELECT   idUsr,
                stsDir,
                URL1,
                URL2
        FROM     localizacion.dbo.loc_busqueda (NOLOCK)
        WHERE    idUsr = @idUsr
END


CREATE PROCEDURE dbo.loc_spr_calle
/*************************************
Consulta de calle
*************************************/
        @idcalle  INT = null
AS
BEGIN
        if @idcalle is null
                SELECT   A.idCalle, A.nombre--, B.idcolonia, B.nombre
                FROM     localizacion..loc_catcalle A (NOLOCK)--,
                         --localizacion..loc_catcolonia B
--              WHERE    A.idcalle = B.idcalle
--              AND      B.idcolonia = C.idcolonia
                ORDER BY 2
        else
                SELECT   A.idCalle, A.nombre--, C.idcolonia, C.nombre
                FROM     localizacion..loc_catcalle A (NOLOCK)--,
--                       localizacion..loc_catcolonia B
                WHERE    A.idcalle = @idcalle
--              AND      A.idcalle = B.@idcalle
                ORDER BY 2
END


CREATE PROCEDURE dbo.loc_spr_callecol
/*********************************************************

Catalogo de Calles y colonias a las que pertenecen
```

```
************************************************************/
AS
BEGIN
        SELECT DISTINCT A.idcalle, A.nombre, B.idcolonia, B.nombre
        FROM    LOCALIZACION..loc_catcalle A (NOLOCK),
        LOCALIZACION..loc_catcolonia B(NOLOCK),
        LOCALIZACION..loc_cuadra C (NOLOCK)
        WHERE A.idcalle = C.idcalle
        AND   B.idcolonia = C.idcolonia
        ORDER BY 2,1
END


CREATE PROCEDURE dbo.loc_spr_categoria
/*************************************
Consulta de calle
*************************************/
        @idcategoria        INT = null
AS
BEGIN
        if @idcategoria is null
                SELECT  A.idcategoria, A.nombre--, B.idcolonia, B.nombre
                FROM    localizacion..loc_catcategoria A (NOLOCK)--,
                        --localizacion..loc_catcolonia B
--              WHERE   A.idcategoria = B.idcategoria
--              AND     B.idcolonia = C.idcolonia
                ORDER BY 2
        else
                SELECT  A.idcategoria, A.nombre--, C.idcolonia, C.nombre
                FROM    localizacion..loc_catcategoria A (NOLOCK)--,
--                      localizacion..loc_catcolonia B
                WHERE   A.idcategoria = @idcategoria
--              AND     A.idcategoria = B.@idcategoria
                ORDER BY 2
END


CREATE PROCEDURE dbo.loc_spr_colonia
/*************************************
Consulta de colonias
*************************************/
        @idcolonia          INT = null,
        @idDel              INT = null
AS
BEGIN
   if @idDel is null
   begin
        if @idcolonia is null
                SELECT  A.idcolonia, A.nombre, A.idDeleg, B.nombre
                FROM    localizacion..loc_catcolonia A (NOLOCK),
                        localizacion..loc_catdelegacion B (NOLOCK)
                WHERE   A.idDeleg = B.idDeleg
                ORDER BY 2
        else
                SELECT  A.idcolonia, A.nombre, A.idDeleg, B.nombre
                FROM    localizacion..loc_catcolonia A (NOLOCK),
                        localizacion..loc_catdelegacion B (NOLOCK)
                WHERE   A.idcolonia = @idcolonia
                AND     A.idDeleg = B.idDeleg
                ORDER BY 2
   end
   else
   begin
        if @idcolonia is null
```

```
                SELECT  A.idcolonia, A.nombre, A.idDeleg, B.nombre
                FROM    localizacion..loc_catcolonia A (NOLOCK),
                        localizacion..loc_catdelegacion B (NOLOCK)
                WHERE   A.idDeleg = B.idDeleg
                AND     A.idDeleg = @idDel
                ORDER BY 2
        else
                SELECT  A.idcolonia, A.nombre, A.idDeleg, B.nombre
                FROM    localizacion..loc_catcolonia A (NOLOCK),
                        localizacion..loc_catdelegacion B (NOLOCK)
                WHERE   A.idcolonia = @idcolonia
                AND     A.idDeleg = B.idDeleg
                AND     A.idDeleg = @idDel
                ORDER BY 2
    end
END


CREATE PROCEDURE dbo.loc_spr_cuadra
/*************************************
Consulta de cuadras
*************************************/
        @idCalle  INT = NULL,
        @idNodo1        INT = NULL,
        @idNodo2        INT = NULL,
        @idColonia      INT = NULL
AS
BEGIN
        if @idCalle is null
                SELECT  A.idCalle,
                        B.nombre 'nomCalle',
                        A.idNodo1,
                        C1.nombre 'nomNodo1',
                        A.flujo1,
                        A.idNodo2,
                        C2.nombre 'nomNodo2',
                        A.flujo2,
                        A.idcolonia,
                        D.nombre 'nomColonia',
                        A.noInic,
                        A.noFinal,
                        A.sent1,
                        E1.descr 'descrSent1',
                        A.sent2,
                        E2.descr 'descrSent2',
                        A.trafico,
                        F.descr 'descrTraf'
                FROM    localizacion..loc_cuadra A (NOLOCK),
                        localizacion..loc_catCalle B (NOLOCK),
                        localizacion..loc_nodo      C1 (NOLOCK),
                        localizacion..loc_nodo      C2 (NOLOCK),
                        localizacion..loc_catColonia D(NOLOCK),
                        localizacion..loc_catsentido E1(NOLOCK),
                        localizacion..loc_catsentido E2(NOLOCK),
                        localizacion..loc_cattrafico F(NOLOCK)
                WHERE   A.idCalle = B.idCalle
                AND     A.idNodo1 = C1.idNodo
                AND     A.idNodo2 = C2.idNodo
                AND     A.idColonia = D.idColonia
                AND     A.sent1 = E1.idsentido
                AND     A.sent2 = E2.idsentido
                AND     A.trafico = F.idtrafico
        else
                SELECT  A.idCalle,
```

```
                B.nombre 'nomCalle',
                A.idNodo1,
                C1.nombre 'nomNodo1',
                A.flujo1,
                A.idNodo2,
                C2.nombre 'nomNodo2',
                A.flujo2,
                A.idcolonia,
                D.nombre 'nomColonia',
                A.noInic,
                A.noFinal,
                A.sent1,
                E1.descr 'descrSent1',
                A.sent2,
                E2.descr 'descrSent2',
                A.trafico,
                F.descr 'descrTraf'
        FROM    localizacion..loc_cuadra A (NOLOCK),
                localizacion..loc_catCalle B (NOLOCK),
                localizacion..loc_nodo      C1 (NOLOCK),
                localizacion..loc_nodo      C2 (NOLOCK),
                localizacion..loc_catColonia D(NOLOCK),
                localizacion..loc_catsentido E1(NOLOCK),
                localizacion..loc_catsentido E2(NOLOCK),
                localizacion..loc_cattrafico F(NOLOCK)
        WHERE   A.idCalle = B.idCalle
        AND     A.idNodo1 = C1.idNodo
        AND     A.idNodo2 = C2.idNodo
        AND     A.idColonia = D.idColonia
        AND     A.sent1 = E1.idsentido
        AND     A.sent2 = E2.idsentido
        AND     A.trafico = F.idtrafico
        AND     A.idCalle = @idCalle
END


CREATE PROCEDURE dbo.loc_spr_delegacion
/**************************************
Consulta de delegaciones
**************************************/
        @idDeleg        INT = null
AS
BEGIN
        if @idDeleg is null
                SELECT  A.idDeleg, A.nombre
                FROM    localizacion..loc_catDelegacion A (NOLOCK)
        else
                SELECT  A.idDeleg, A.nombre
                FROM    localizacion..loc_catDelegacion A (NOLOCK)
                WHERE   A.idDeleg = @idDeleg
END


CREATE PROCEDURE dbo.loc_spr_nodo
/**************************************
Consulta de nodos
**************************************/
        @idnodo  INT = null
AS
BEGIN
        if @idnodo is null
                SELECT  A.idnodo, A.nombre
                FROM    localizacion..loc_nodo A (NOLOCK)
        else
```

```
                SELECT  A.idnodo, A.nombre
                FROM    localizacion..loc_nodo A (NOLOCK)
                WHERE   A.idnodo = @idnodo
END


CREATE PROCEDURE dbo.loc_spr_sentido
/*************************************
Consulta de sentidos
*************************************/
AS
BEGIN
                SELECT  idsentido, descr
                FROM    localizacion..loc_catsentido  (NOLOCK)
END


CREATE PROCEDURE dbo.loc_spr_sitio
/*************************************
Consulta de sitios de interes
*************************************/
        @idSitio   INT = null,
        @idcateg INT = null
AS
BEGIN

                SELECT  A.idSitio,
                        A.nombre,
                        A.idCalle,
                        B.nombre 'nomCalle',
                        A.numero,
                        A.idcolonia,
                        C.nombre 'nomColonia',
                        A.telefono,
                        A.iddeleg,
                        D.nombre 'nomDelegacion',
                        A.idcategoria,
                        E.nombre 'nomcategoria'
                FROM    localizacion..loc_sitio A (NOLOCK),
                        localizacion..loc_catcalle B (NOLOCK),
                        localizacion..loc_catcolonia C (NOLOCK),
                        localizacion..loc_catdelegacion D (NOLOCK),
                        localizacion..loc_catcategoria E (NOLOCK)
                WHERE   A.idcalle = B.idcalle
                AND     A.idcolonia = C.idcolonia
                AND     A.iddeleg = D.iddeleg
                AND     A.idcategoria = E.idcategoria
                AND     (A.idSitio = @idSitio OR @idSitio IS NULL)
                AND     (A.idcategoria = @idcateg OR @idcateg IS NULL)

END


CREATE PROCEDURE dbo.loc_spr_trafico
/*************************************
Consulta de carga del trafico
*************************************/
AS
BEGIN
                SELECT  idtrafico, descr
                FROM    localizacion..loc_cattrafico  (NOLOCK)
END
```

```
CREATE PROCEDURE dbo.loc_spu_calle
/*************************************
Actualiza una calle
*************************************/
	@idcalle  INT,
	@NuevoNombre	VARCHAR(30)
AS
BEGIN
	UPDATE localizacion..loc_catcalle
	SET Nombre = @NuevoNombre
	WHERE idcalle = @idcalle

	If @@Error <> 0
	Begin
		Raiserror('Error al Actualizar en localizacion..loc_catcalle',18,2) WITH SETERROR
		--ROLLBACK inserta_calle
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spu_categoria
/*************************************
Actualiza una calle
*************************************/
	@idcategoria		INT,
	@NuevoNombre	VARCHAR(30)
AS
BEGIN
	UPDATE localizacion..loc_catcategoria
	SET Nombre = @NuevoNombre
	WHERE idcategoria = @idcategoria

	If @@Error <> 0
	Begin
		Raiserror('Error al Actualizar en localizacion..loc_catcategoria',18,2) WITH SETERROR
		--ROLLBACK inserta_calle
		Return 0
	End
END


CREATE PROCEDURE dbo.loc_spu_colonia
/*************************************
Actualiza una colonia
*************************************/
	@idcolonia		INT,
	@NuevoNombre	VARCHAR(30),
	@idDeleg INT
AS
BEGIN
	UPDATE localizacion..loc_catcolonia
	SET	Nombre = @NuevoNombre,
		idDeleg = @idDeleg
	WHERE idcolonia = @idcolonia

	If @@Error <> 0
	Begin
		Raiserror('Error al Actualizar en localizacion..loc_catcolonia',18,2) WITH SETERROR
		Return 0
	End
END
```

```
CREATE PROCEDURE dbo.loc_spu_cuadra
/*************************************
Actualiza una cuadra
*************************************/
        @idCalle  INT,
        @idNodo1        INT,
        @idNodo2        INT,
        @flujo1         INT,
        @flujo2         INT,
        @noInic         INT,
        @noFinal INT,
        @idColonia      INT,
        @sent1          INT,
        @sent2          INT,
        @trafico  INT
AS
BEGIN

        UPDATE localizacion..loc_cuadra
        SET     flujo1 = @flujo1,
                flujo2 = @flujo2,
                noInic = @noInic,
                noFinal = @noFinal,
                sent1 = @sent1,
                sent2 = @sent2,
                trafico = @trafico
        WHERE   idCalle = @idCalle
        AND     idNodo1 = @idNodo1
        AND     idNodo2 = @idNodo2
        AND     idColonia = @idColonia

        If @@Error <> 0
        Begin
                Raiserror('Error al Actualizar en localizacion..loc_cuadra',18,2) WITH SETERROR
                Return 0
        End

END


CREATE PROCEDURE dbo.loc_spu_delegacion
/*************************************
Actualiza una delegacion
*************************************/
        @idDeleg        INT,
        @NuevoNombre    VARCHAR(30)
AS
BEGIN
        UPDATE localizacion..loc_catDelegacion
        SET Nombre = @NuevoNombre
        WHERE idDeleg = @idDeleg

        If @@Error <> 0
        Begin
                Raiserror('Error al Actualizar en localizacion..loc_catDelegacion',18,2) WITH SETERROR
                Return 0
        End
END


CREATE PROCEDURE dbo.loc_spu_nodo
/*************************************
Actualiza un nodo
*************************************/
```

```
        @idnodo  INT,
        @NuevoNombre     VARCHAR(30)
AS
BEGIN
        UPDATE localizacion..loc_nodo
        SET Nombre = @NuevoNombre
        WHERE idnodo = @idnodo

        If @@Error <> 0
        Begin
                Raiserror('Error al Actualizar en localizacion..loc_nodo',18,2) WITH SETERROR
                Return 0
        End
END


CREATE PROCEDURE dbo.loc_spu_sitio
/*************************************
Actualiza un sitio de interes
*************************************/
        @idSitio   INT,
        @NvoNombre       VARCHAR(60),
        @Nvoidcalle      INT,
        @Nvonumero       INT,
        @NvoidColonia    INT,
        @Nvotelefono     INT,
        @Nvoiddelegacion INT,
        @Nvoidcategoria  INT
AS
BEGIN
        UPDATE localizacion..loc_sitio
        SET     nombre = @NvoNombre,
                idcalle = @Nvoidcalle,
                numero = @Nvonumero,
                idcolonia = @NvoidColonia,
                telefono = @Nvotelefono,
                iddeleg = @Nvoiddelegacion,
                idcategoria = @Nvoidcategoria
        WHERE idSitio = @idSitio

        If @@Error <> 0
        Begin
                Raiserror('Error al Actualizar en localizacion..loc_sitio',18,2) WITH SETERROR
                Return 0
        End
END
```

Apéndice D

Diagramas UML

Mantenimiento de Delegación (Backend)

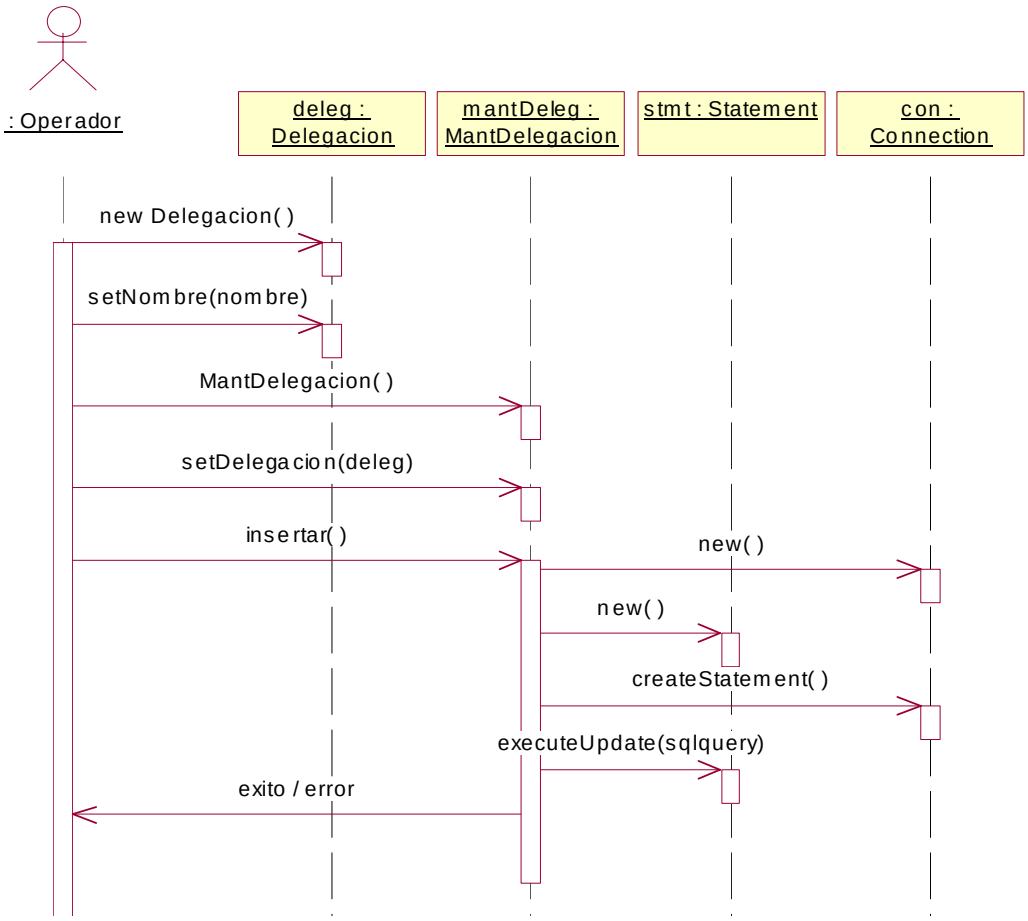


Figura D.1 Diagrama de Secuencia para la inserción de una Delegación

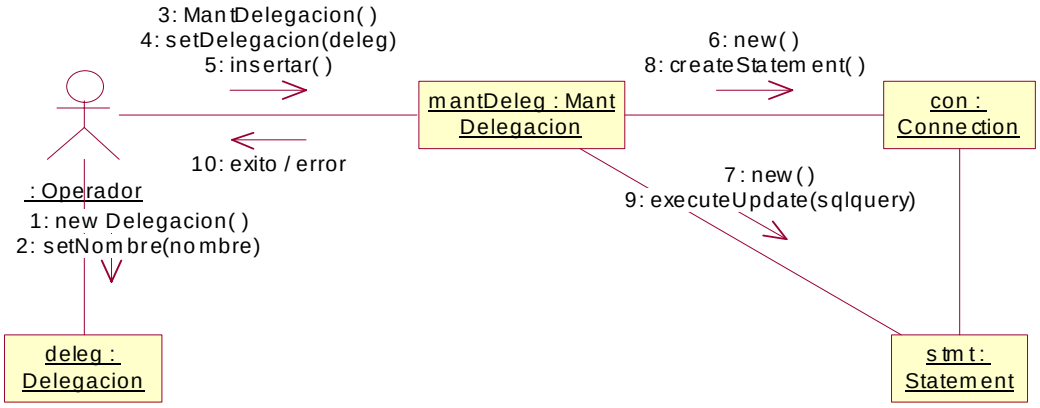


Figura D.2 Diagrama de Colaboración para la inserción de una Delegación

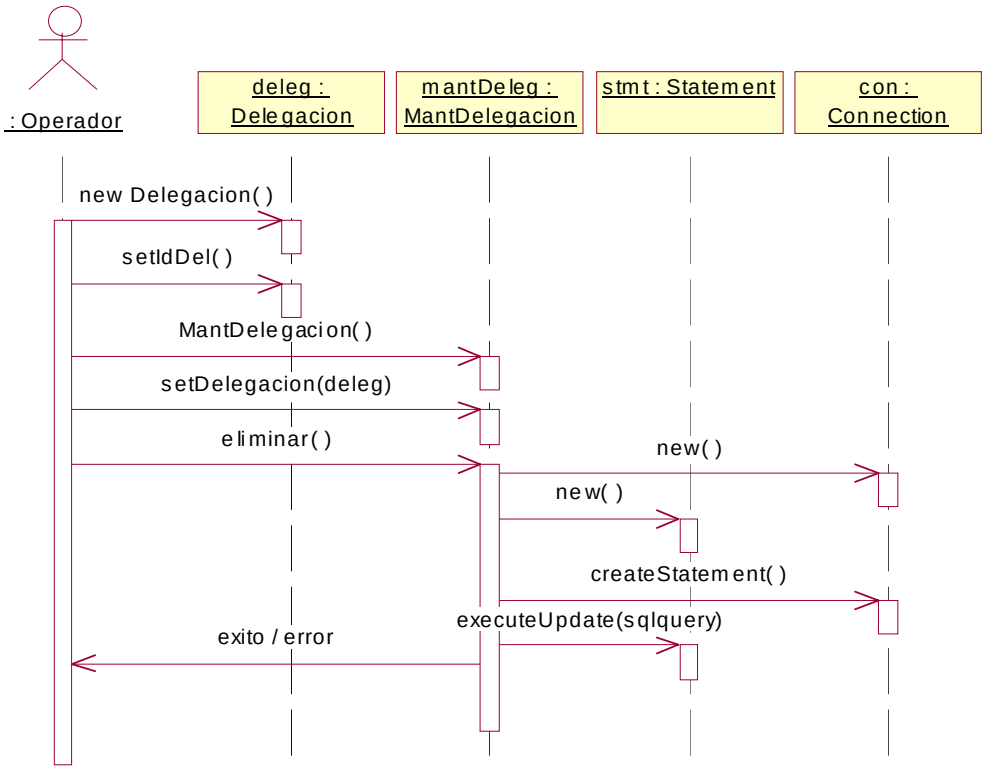


Figura D.3 Diagrama de Secuencia para la eliminación de una Delegación

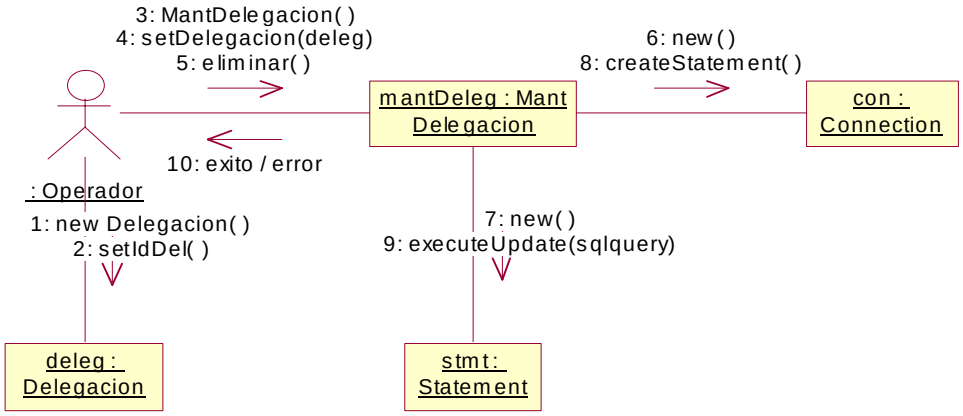


Figura D.4 Diagrama de Colaboración para la eliminación de una Delegación

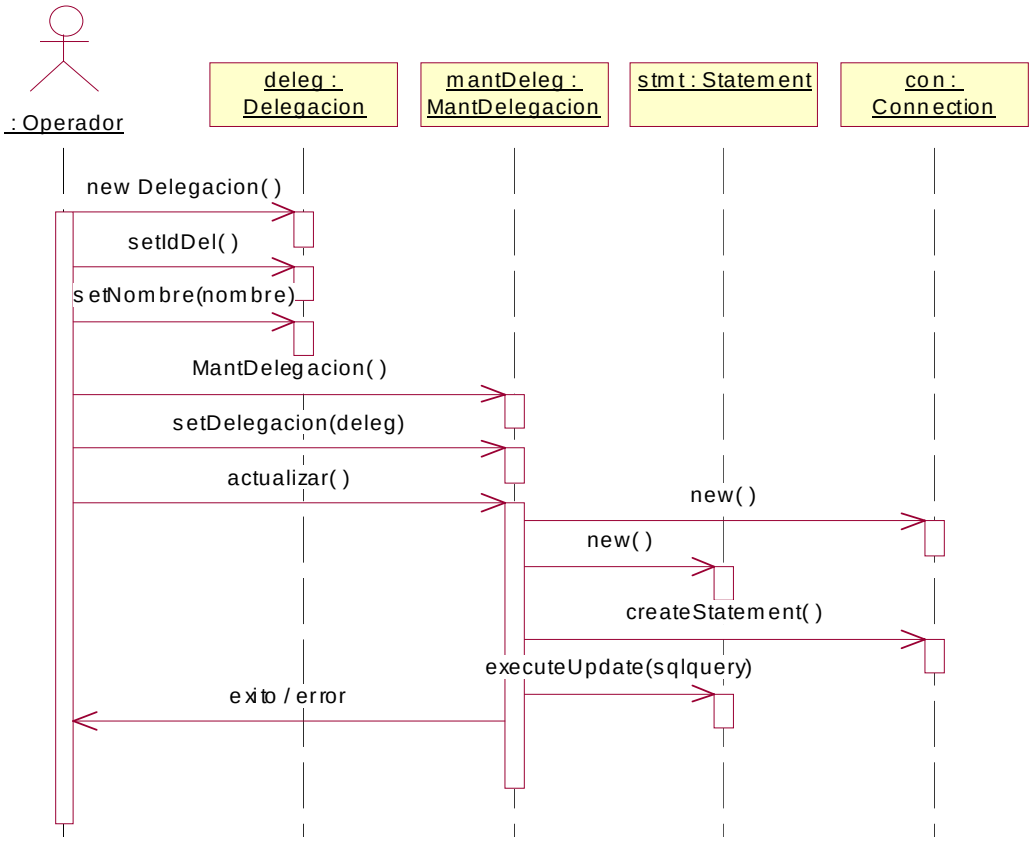


Figura D.5 Diagrama de Secuencia para la modificación de una Delegación

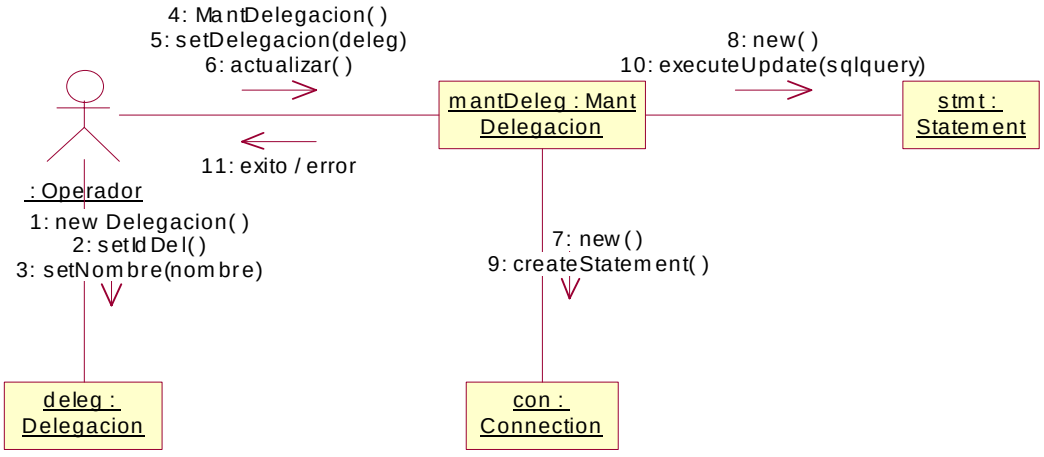


Figura D.6 Diagrama de Colaboración para la modificación de una Delegación

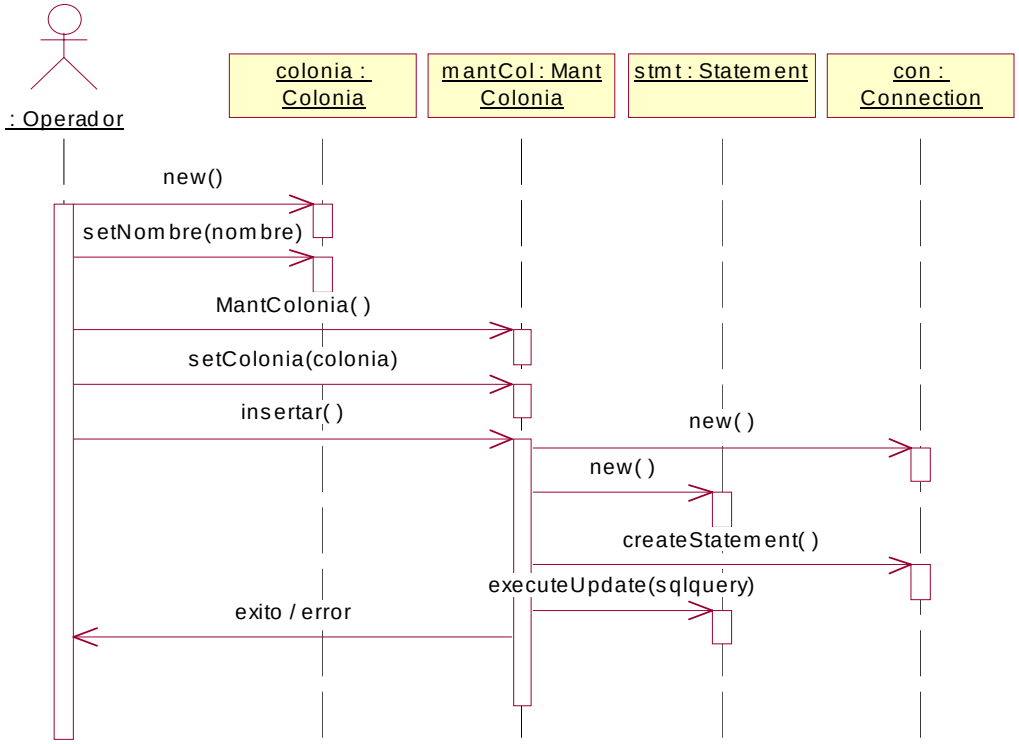


Figura D.7 Diagrama de Secuencia para la inserción de una Colonia

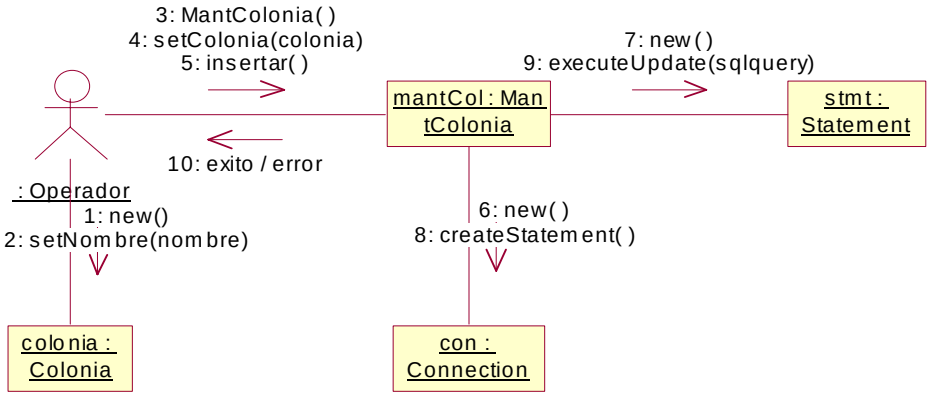


Figura D.8 Diagrama de Colaboración para la inserción de una Colonia

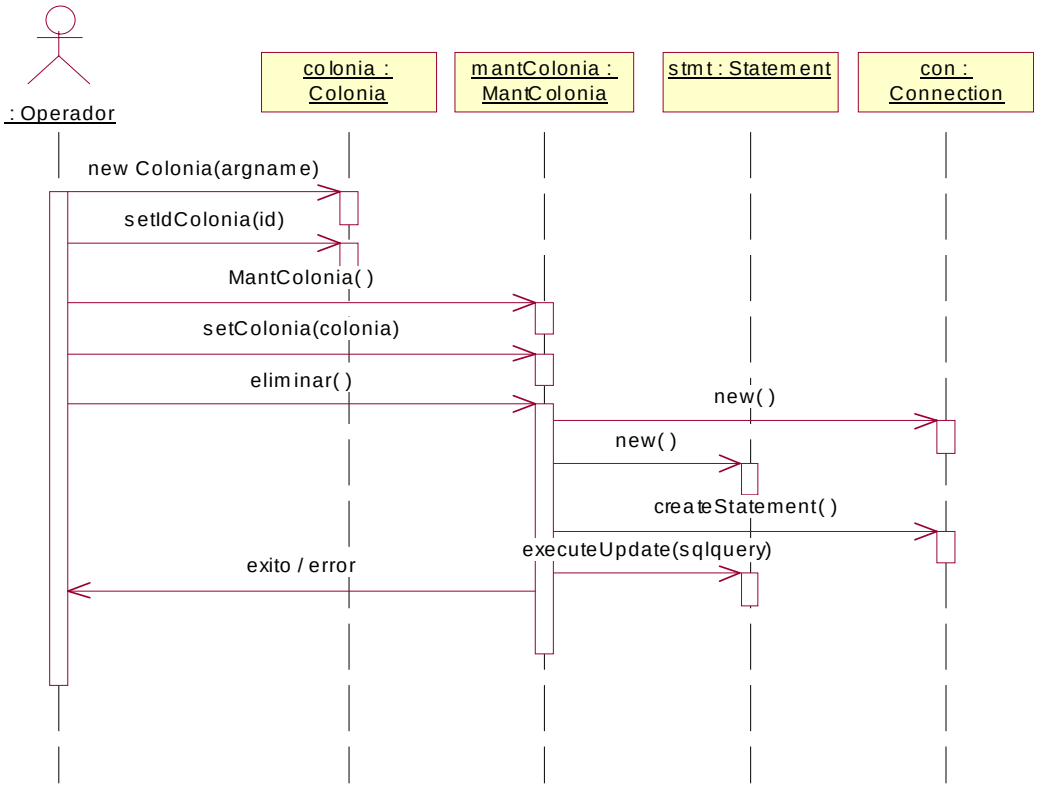


Figura D.9 Diagrama de Secuencia para la eliminación de una Colonia

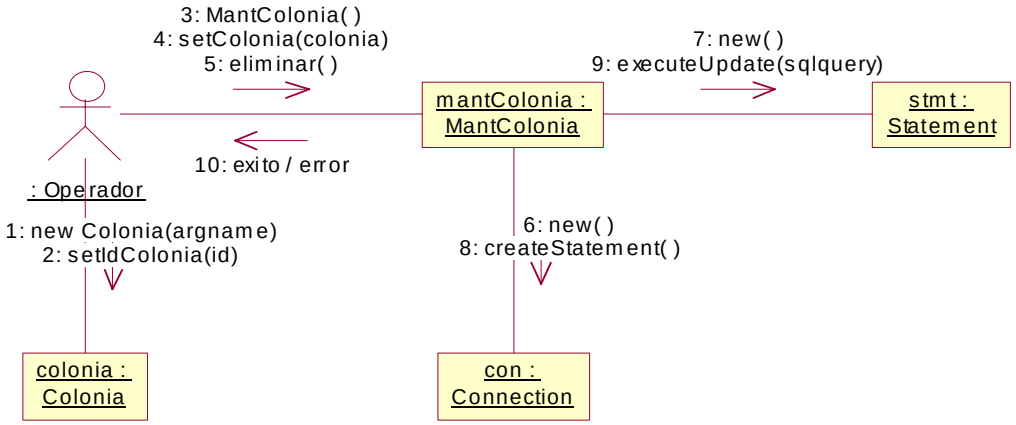


Figura D.10 Diagrama de Colaboración para la eliminación de una Colonia

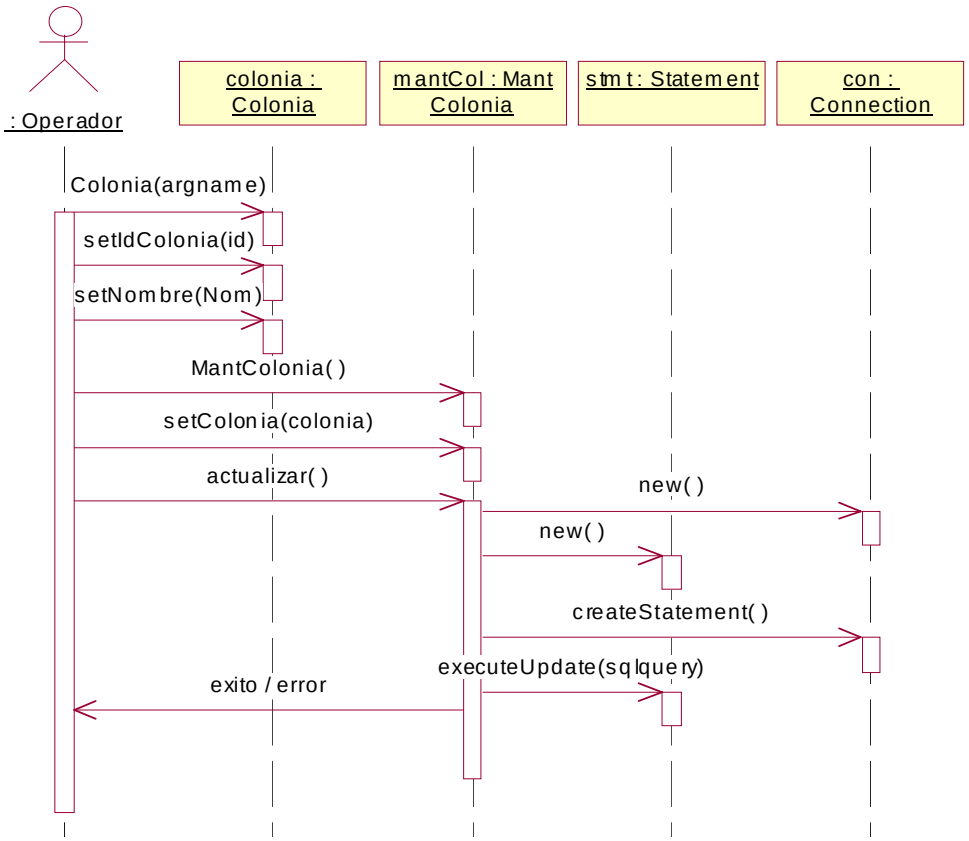


Figura D.11 Diagrama de Secuencia para la modificación de una Colonia

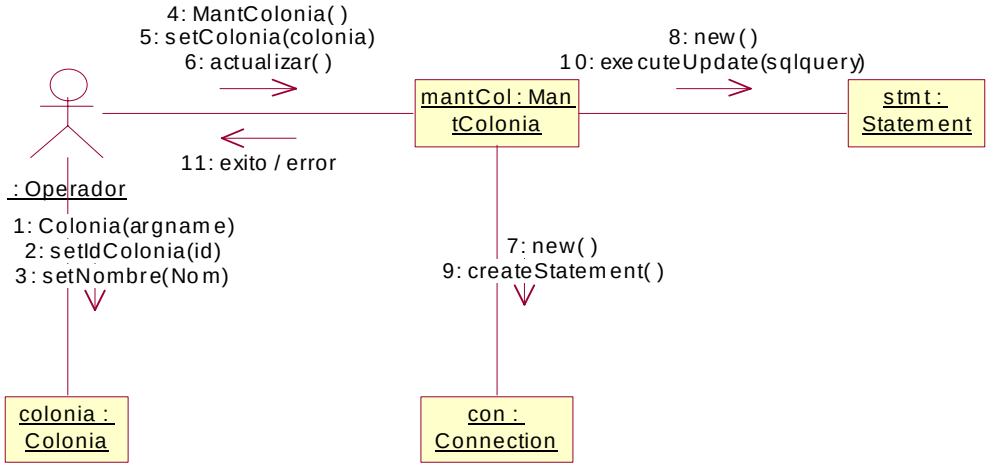


Figura D.12 Diagrama de Colaboración para la modificación de una Colonia

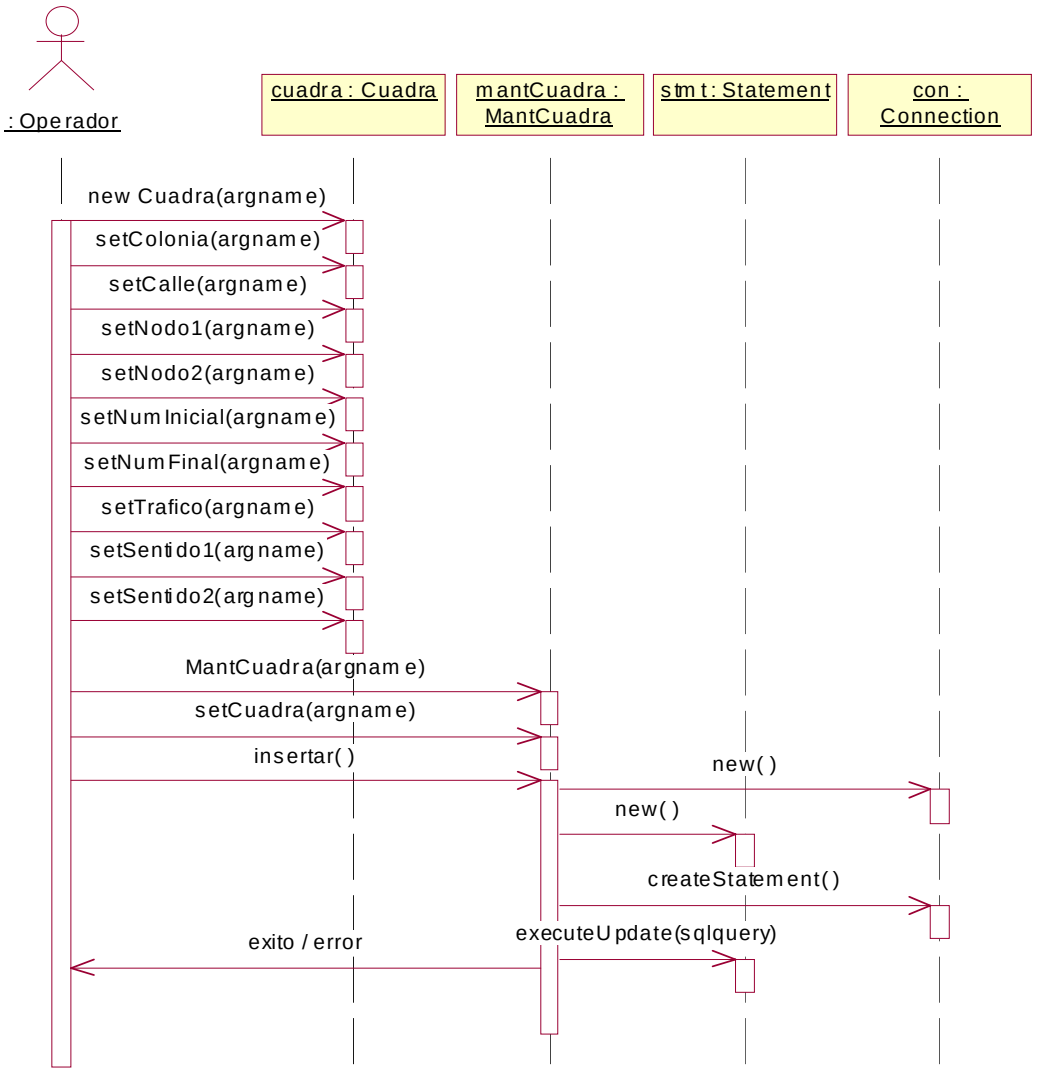


Figura D.13 Diagrama de Secuencia para la inserción de una Cuadra

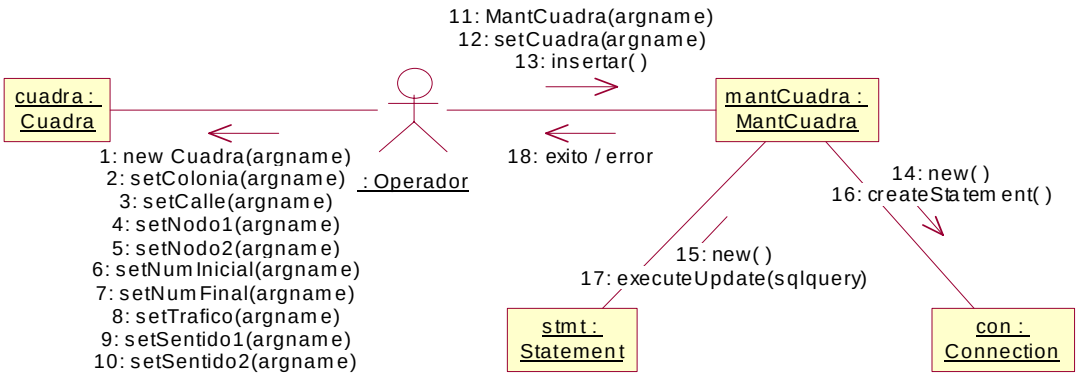


Figura D.14 Diagrama de Colaboración para la inserción de una Cuadra

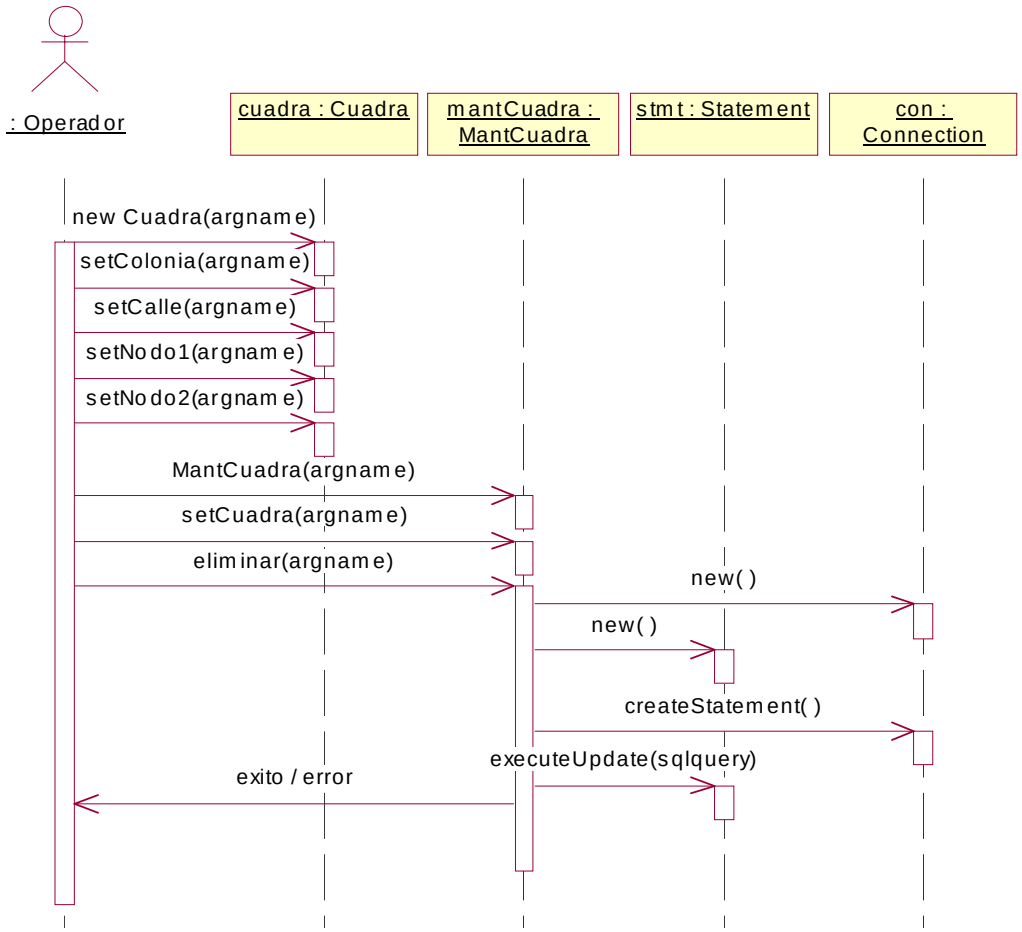


Figura D.15 Diagrama de Secuencia para la eliminación de una Cuadra

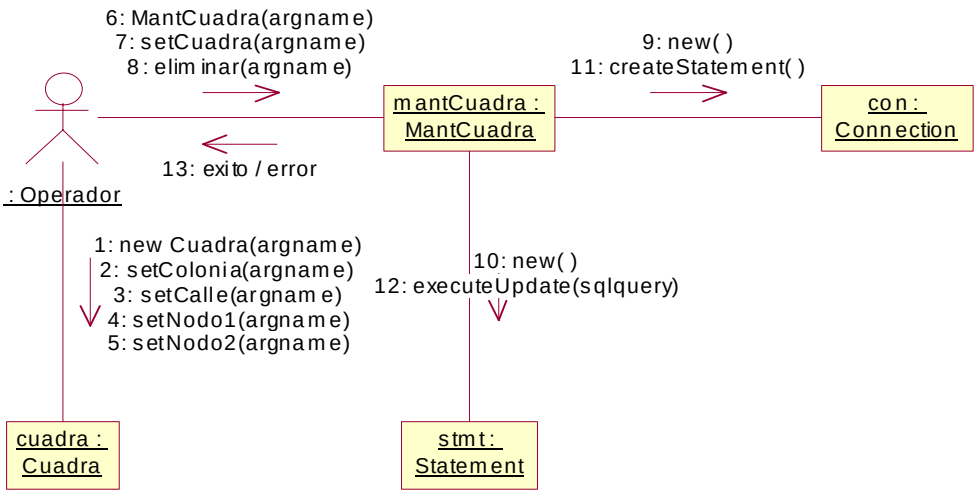


Figura D.16 Diagrama de Colaboración para la eliminación de una Cuadra

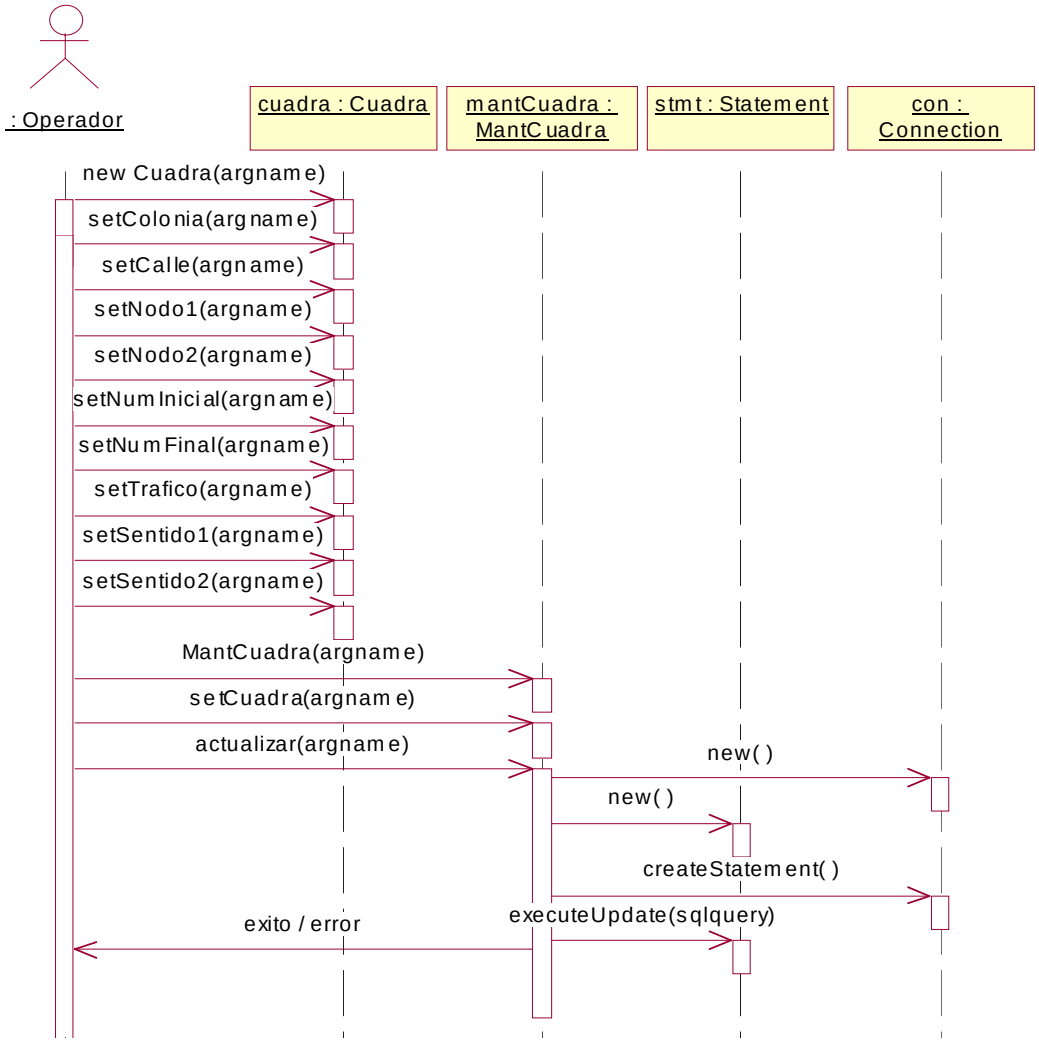


Figura D.17 Diagrama de Secuencia para la modificación de una Cuadra

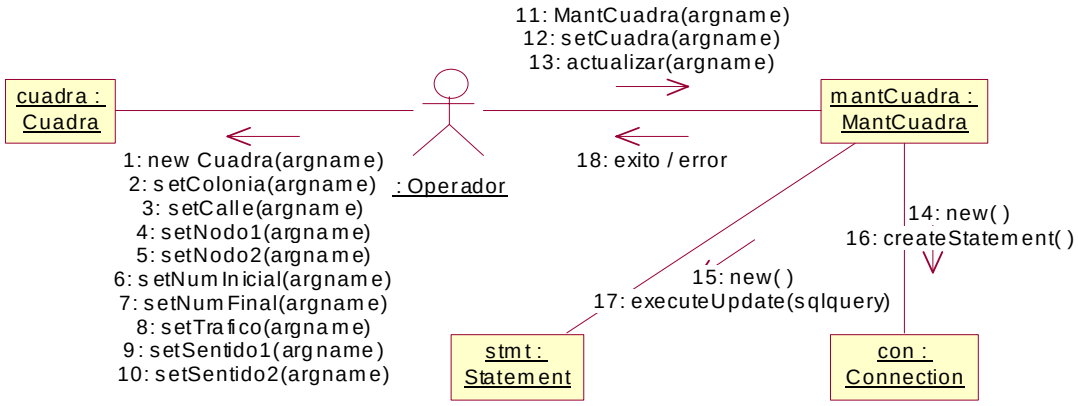


Figura D.18 Diagrama de Colaboración para la modificación de una Cuadra

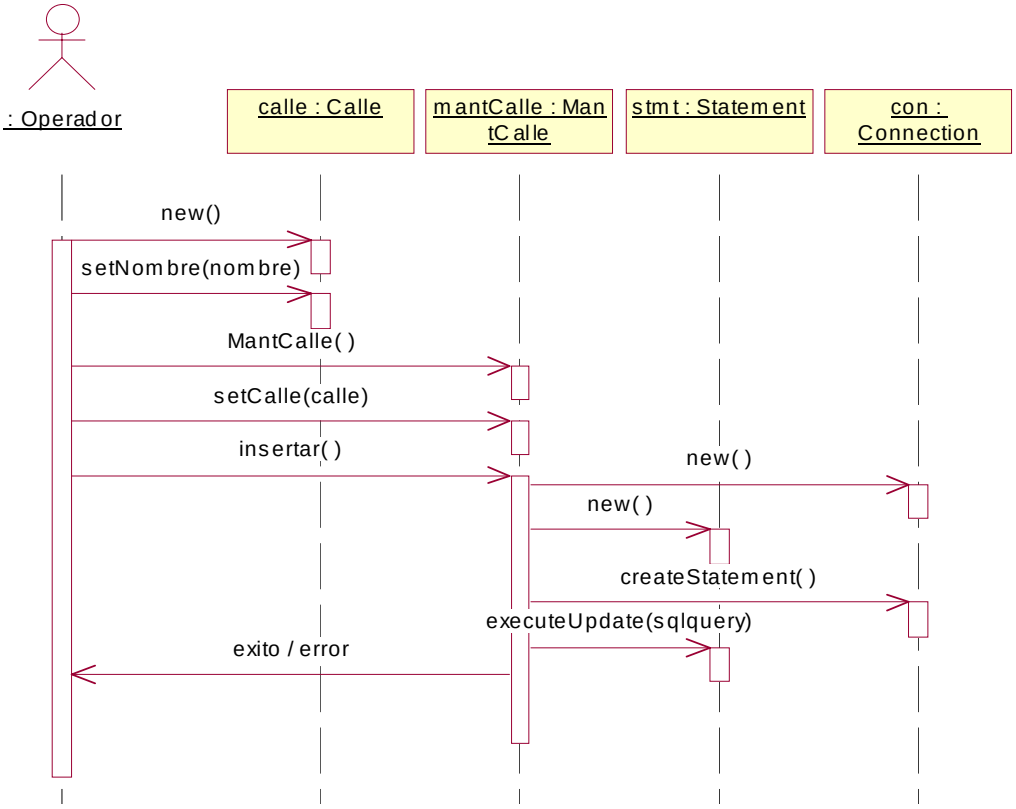


Figura D.19 Diagrama de Secuencia para la inserción de una Calle

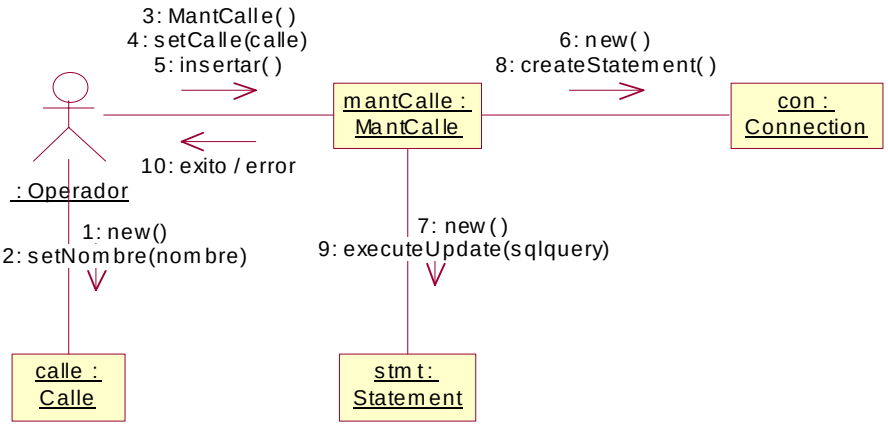


Figura D.20 Diagrama de Colaboración para la inserción de una Calle

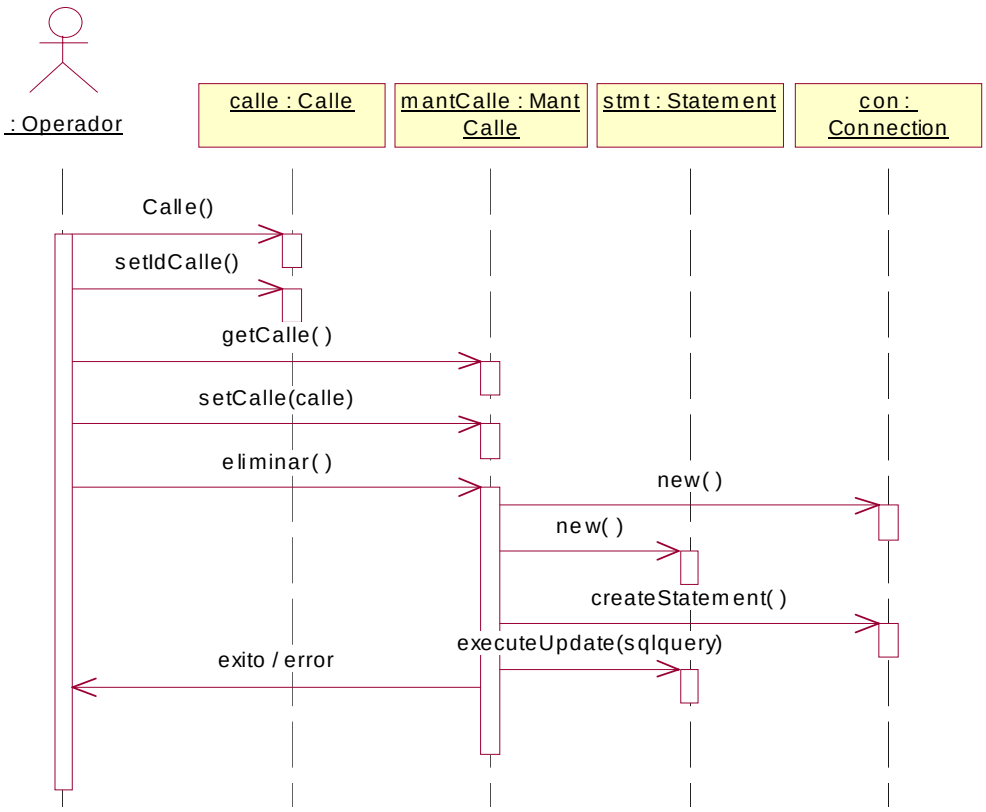


Figura D.21 Diagrama de Secuencia para la eliminación de una Calle

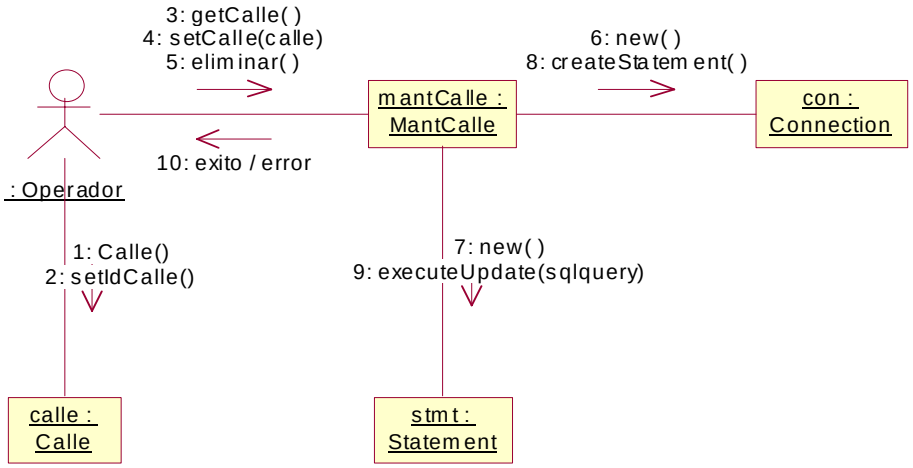


Figura D.22 Diagrama de Colaboración para la eliminación de una Calle

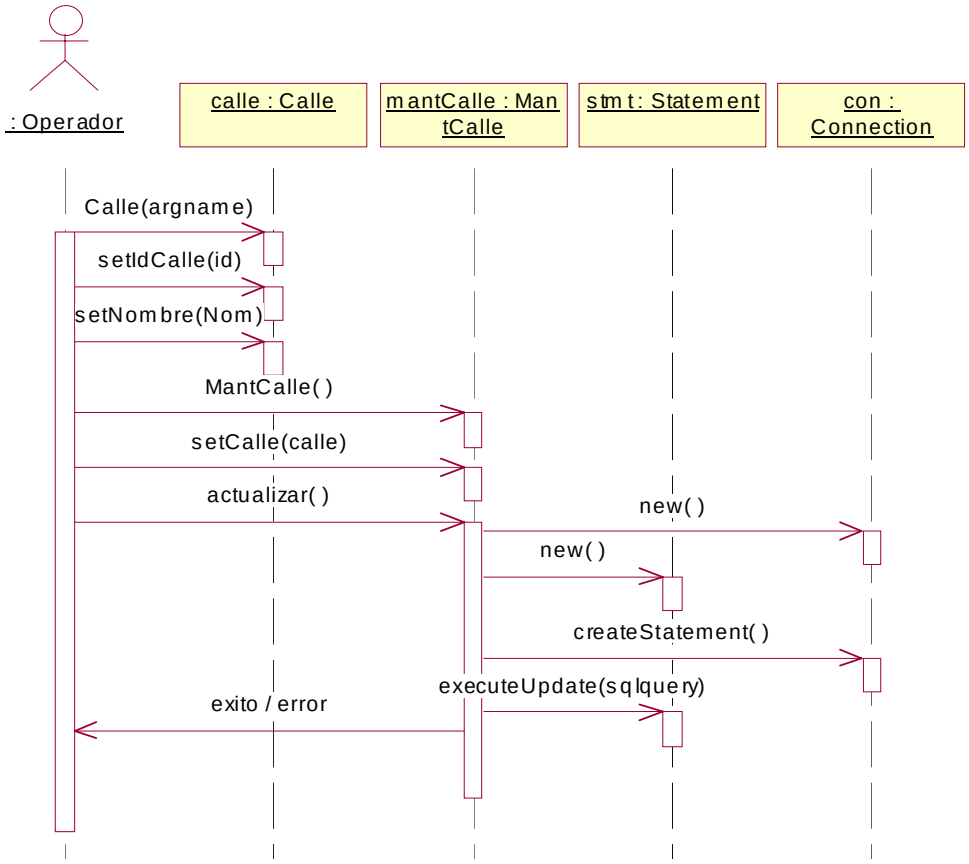


Figura D.23 Diagrama de Secuencia para la modificación de una Calle

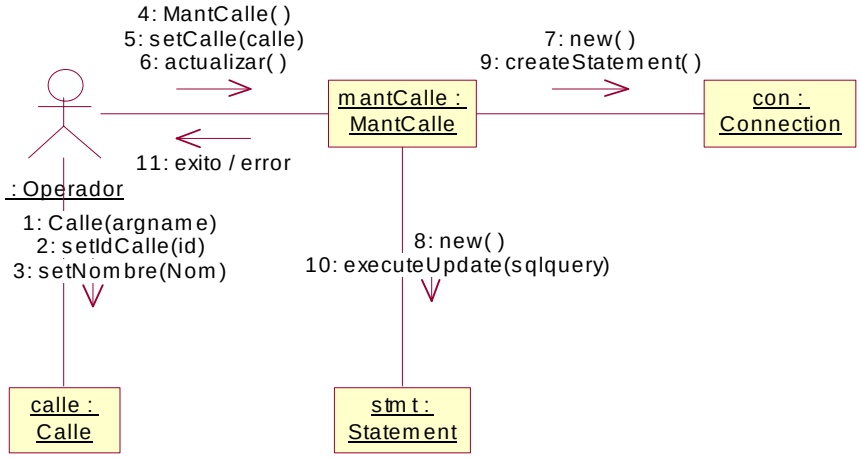


Figura D.24 Diagrama de Colaboración para la modificación de una Calle

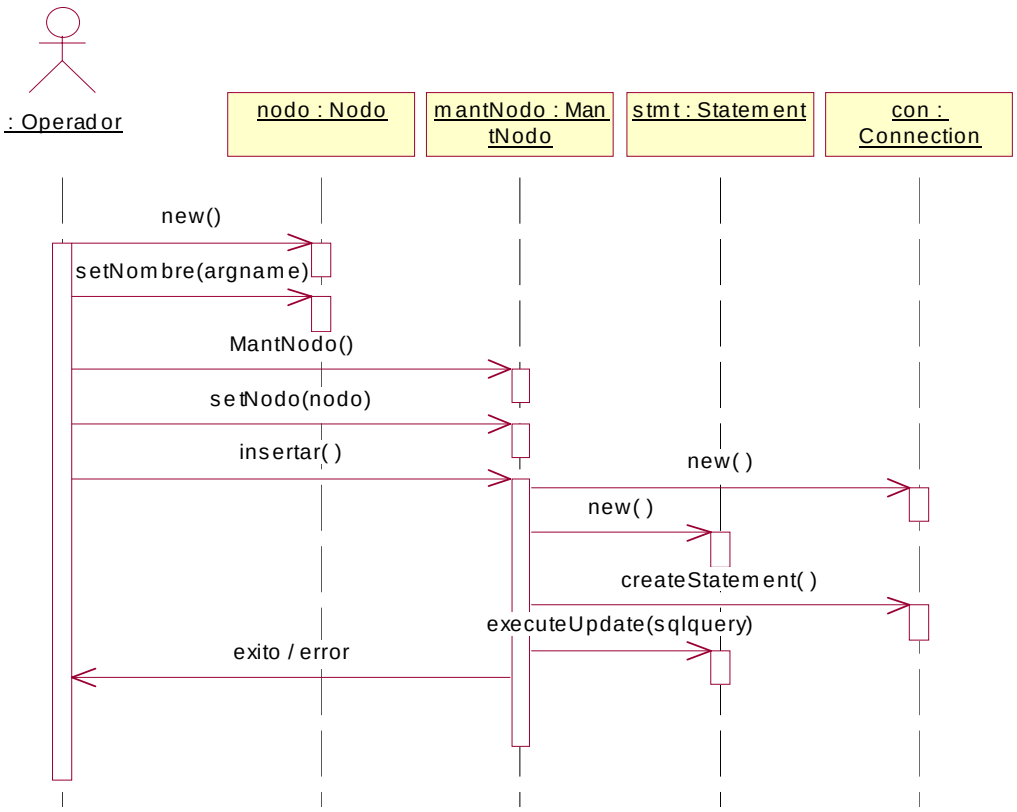


Figura D.25 Diagrama de Secuencia para la inserción de un Nodo

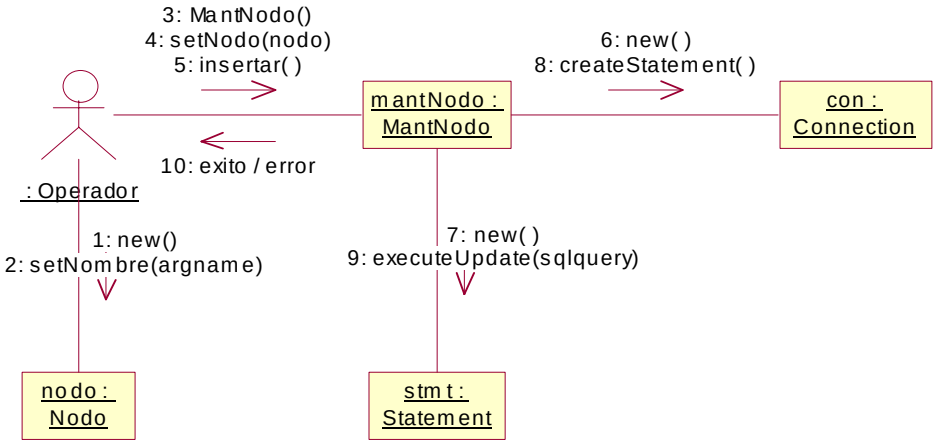


Figura D.26 Diagrama de Colaboración para la inserción de un Nodo

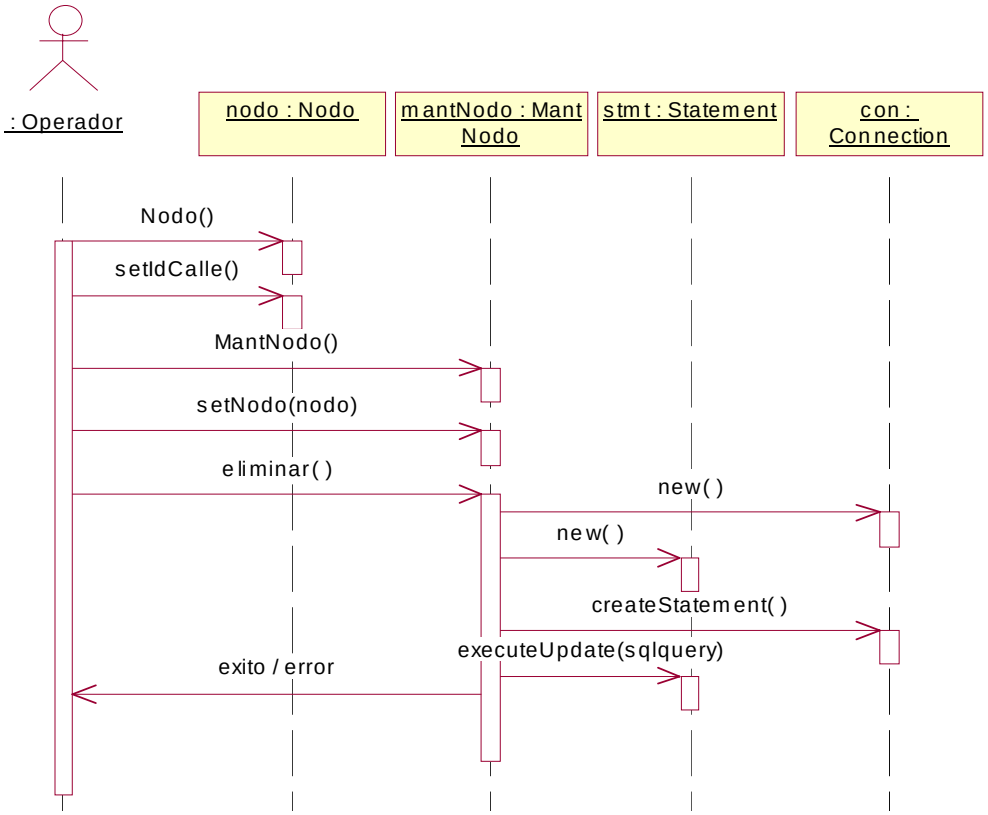


Figura D.27 Diagrama de Secuencia para la eliminación de un Nodo

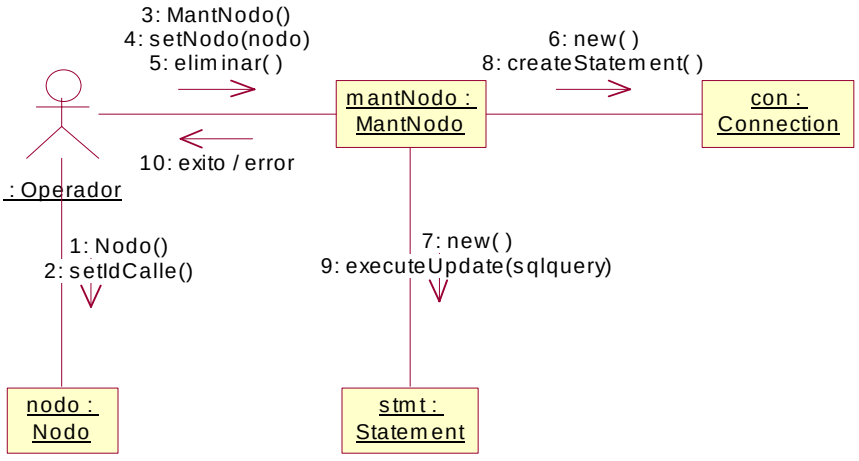


Figura D.28 Diagrama de Colaboración para la eliminación de un Nodo

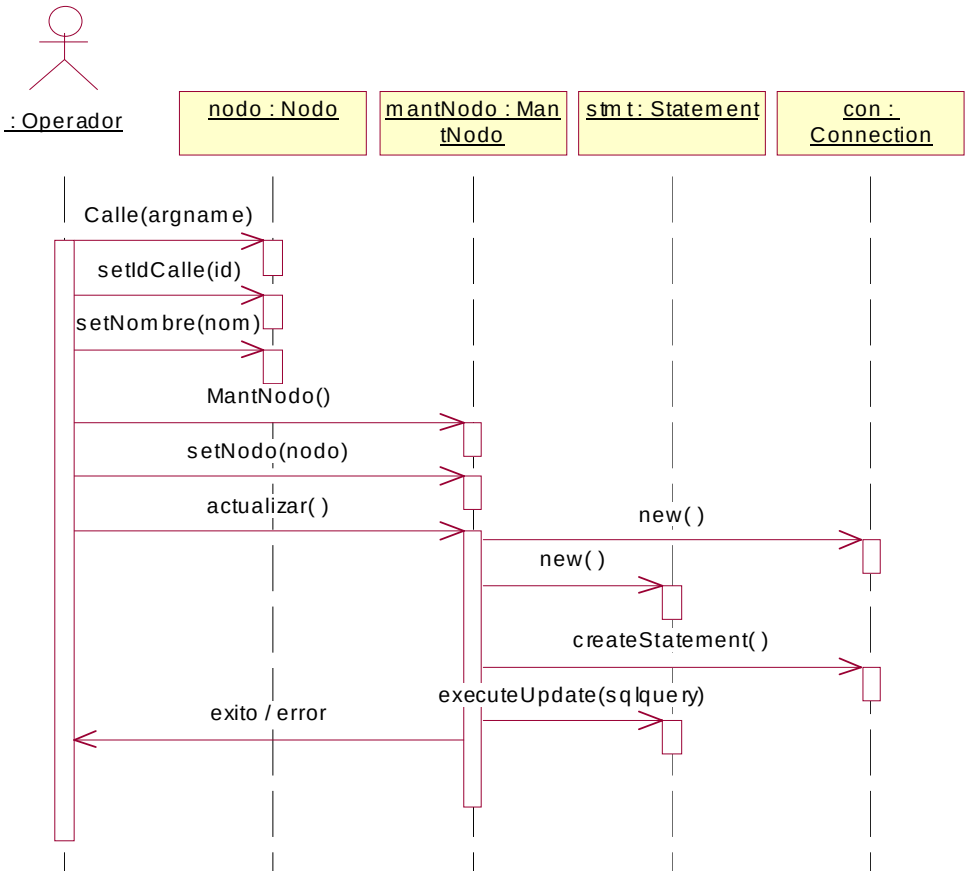


Figura D.29 Diagrama de Secuencia para la modificación de un Nodo

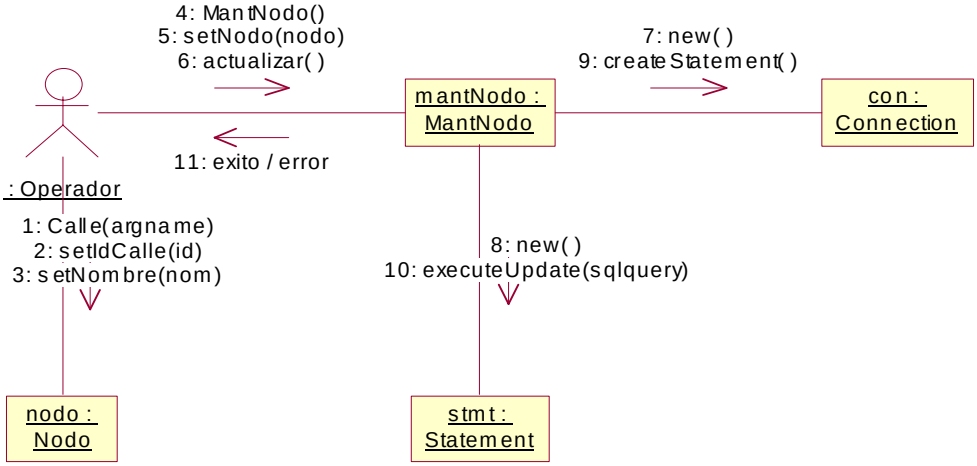


Figura D.30 Diagrama de Colaboración para la modificación de un Nodo

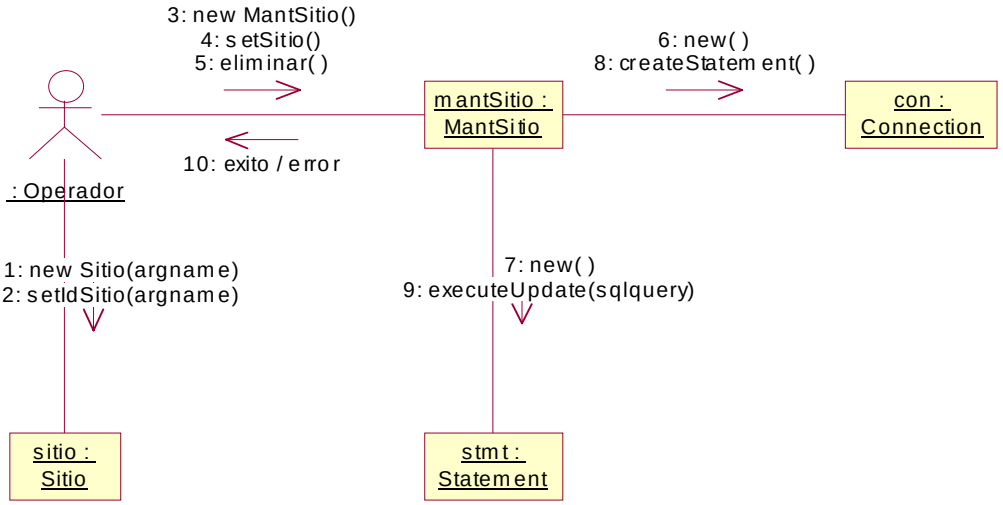


Figura D.31 Diagrama de Colaboración para la eliminación de un Sitio de Interés

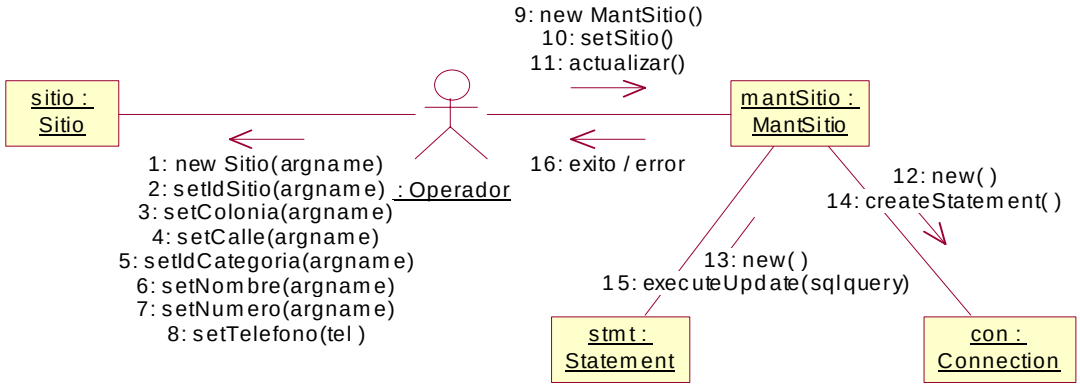


Figura D.32 Diagrama de Colaboración para la modificación de un Sitio de Interés

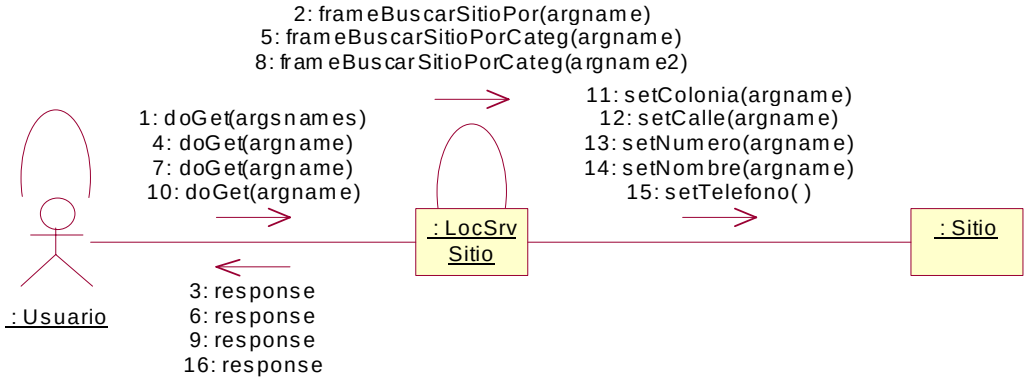


Figura D.33 Diagrama de Colaboración para la Selección de un Sitio de Interés

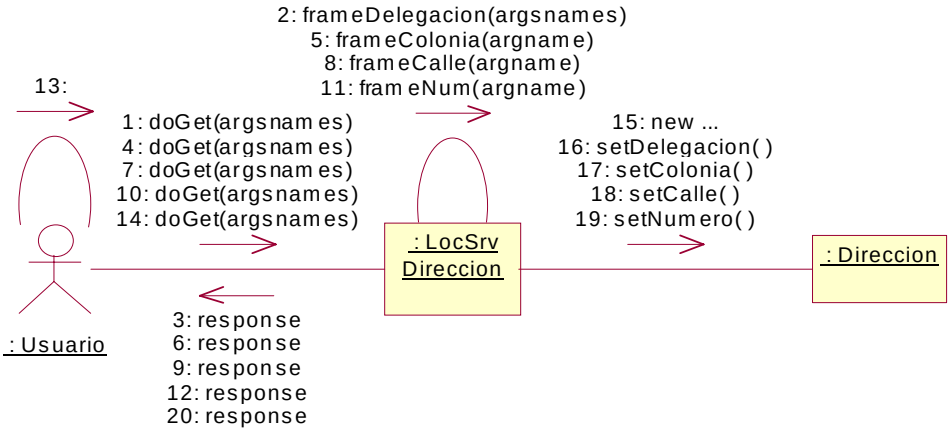


Figura D.34 Diagrama de Colaboración para la Selección de una Dirección

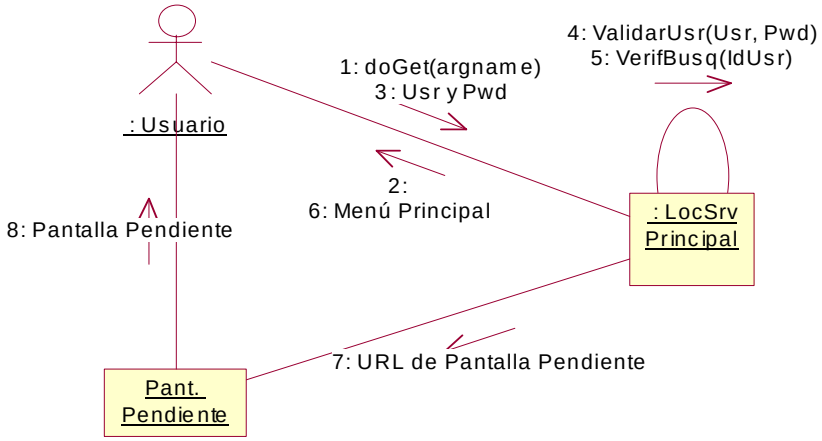


Figura D.35 Diagrama de Colaboración para el módulo de Registro y Búsqueda Pendiente

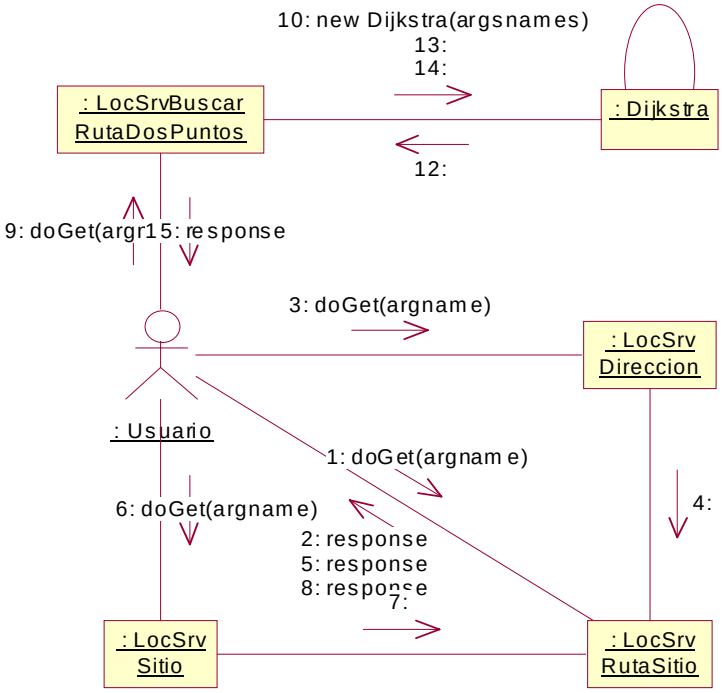


Figura D.36 Diagrama de Colaboración para Buscar la Ruta Óptima entre Dos Puntos

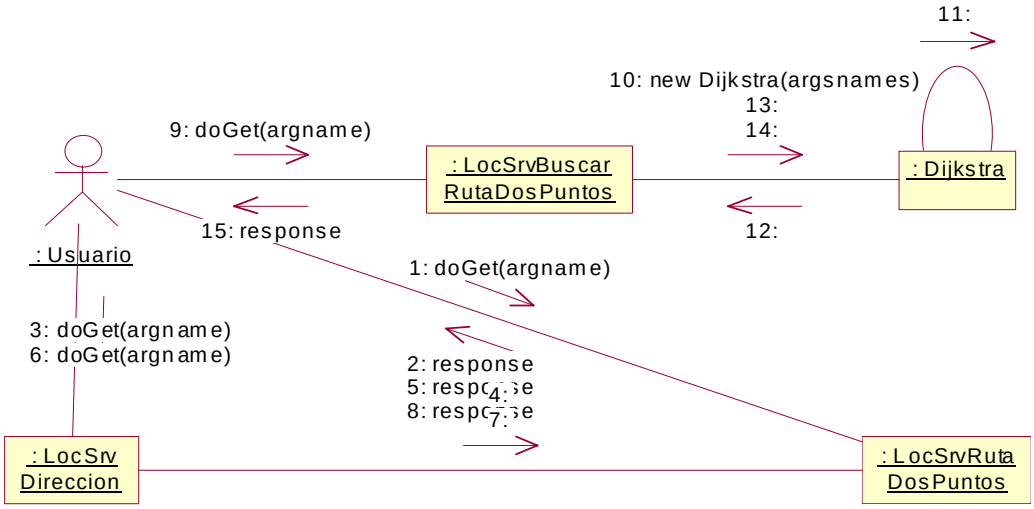


Figura D. 37 Diagrama de Colaboración para Buscar la Ruta Óptima hacia un Sitio de Interés

Bibliografía

Referencia	Descripción
[BAL00]	Balladares, Leandro, Ambiente CASE y Lenguaje Visual para la Asignación de Comportamiento Complejo a Componentes de Mundos Virtuales, Centro de Investigación en Computación – IPN. 2000.
[BEH81]	Mehdi Behzad,Gary Chartrand y Linda Lesniak-Foster; Graphs & Digrafos; Wadsworth International Group. 1981.
[BON76]	J. A. Bondy and U.S.R. Murty, Graph Theory with Applications, North-Holland New York.Amsterdam.Oxford. 1976.
[BZK98]	Dimitri Bertzekas, Network Optimization. Continuous and discrete models, Athena Scientific, Belmont, Massachusetts.
[DJK59]	Dijkstra, Edsger W.; A Note on Two Problems in Connection with Graphs. Numer. Math. Vol 1. 1959. Pp 269-271.
[DJK80]	Dijkstra, Edsger W.; Scholten, C.S; Termination Detection for Diffusing Computations Information Processing Letters. Vol 11 - nro 1. Agosto 1980. Pp 1-4.
[GAM01]	Gama, Luis, Sistema Punto de Venta para Farmacias InterFarma, Centro de Investigación en Computación – IPN. 2001
[KEN90]	Kenneth A. Ross y Charles R.B. Wright, Matemáticas Discretas, pHH Prentice Hall. 1990.
[LOO85]	Maty E.S. Loomis, Estructura de Datos y Organización de Archivos pHH Prentice Hall. 1985.
[RAM00]	James Rumbaugh, Ivar Jacobson y Grady Booch. El Lenguaje Unificado de Modelado. Manual de Referencia Rational Software Corporation. Addison Wesley Ed., Año 2000
[WEI91]	Weiser, Mark The Computer for de 21st Century Scientific American, Año 1991
[WWW1]	Ingenet http://ingenet.ulpgc.es/~ablesa/telecom/optimizaredes/routing.htm
[WWW2]	Departamento de Ciencias de la Computación, Universidad de Chile http://www.dcc.uchile.cl/~cc30a/apuntes/Grafos/
[WWW3]	Resumen de la Metodología IDEF0 http://www.aqa.es/doc/Metodologia%20%20IDEF0%20Resumen.pdf

[WWW4]	Desarrollo de Software Orientado a Objetos usando UML, Patricio Letelier Torres. Departamento Sistemas Informáticos y Computación (DSIC) Universidad Politécnica de Valencia (UPV) - España http://www.dsic.upv.es/~uml
[WWW5]	Ingeniería del Software. Diseño Lenguajes y Ciencias de la Computación Universidad de Málaga - España http://www.lcc.uma.es/docencia/ETSIInf/isd/Isd_3.pdf
[WWW6]	Weitzenfeld, Alfredo Ingeniería de Software Orientada a Objetos con UML, Java e Internet Departamento Académico de Computación Instituto Tecnológico Autónomo de México http://cannes.rhon.itam.mx/Alfredo/Espaniol/Publicaciones/MINT/Analisis.pdf
[WWW7]	Peter M. Chen, Associate Professor of Department of Electrical Engineering and Computer Science University of Michigan. http://www.eecs.umich.edu/~pmchen/
[WWW8]	Portal de programación avanzada en el lenguaje Java. http://www.programacion.com/java/tutorial/servlets_jsp/1/
[WWW9]	Sitio oficial de la tecnología Java Servlet. Sun Microsystems. http://java.sun.com/products/servlet/
[WWW10]	Weitzenfeld, Alfredo Ingeniería de Software Orientada a Objetos con UML, Java e Internet Departamento Académico de Computación Instituto Tecnológico Autónomo de México http://cannes.rhon.itam.mx/Alfredo/Espaniol/Publicaciones/MINT/Pruebas.pdf
[WWW11]	Hernando Fernández, Miguel Angel Introducción al Cómputo Ubicuo http://gsyc.escet.urjc.es/~mhernand/docu_pfc/memoria/node6.html
[WWW12]	MoviStar e-moción Cerca de Tí http://www.movistar.com/emocion/lomasutil/cerca/cerca.htm
[WWW13]	Radiomóvil Dipsa S.A. de C.V. (TELCEL) *RUTA Marcar *7882 desde un celular con servicio de Telcel
[WWW14]	Centro de Computación Latinoamericano El Protocolo Inalámbrico de Aplicaciones http://www.cclca.com/mantenimiento/wap.pdf

Glosario

Abreviatura	Descripción
CSD	Datos Conmutados por Circuito (Circuit Switched Data)
DBMS	Sistema Manejador de Base de Datos (Database Management System)
GPRS	Servicio General de Paquetes de Radio (General Packet Radio Service)
GSM	Sistema Global para Comunicaciones Móviles (Global System for Mobile Communications)
GUI	Interfaz Gráfica de Usuario (Graphic User Interface)
HDML	Lenguaje de Marcado para Dispositivos PDAs (Handheld Device Markup Language)
HDTP	Protocolo de Transporte para Dispositivos PDAs (Handheld Device Transport Protocol)
HTML	Lenguaje de Marcado de Hipertexto (Hyper Text Markup Language)
HTTP	Protocolo de Transferencia de Hipertexto (Hyper Text Transfer Protocol)
HTTPS	Protocolo Seguro para Transferencia de Hipertexto (Hyper Text Transfer Protocol Secure)
IDE	Ambiente para Desarrollo Integridad (Integrated Development Environment)
IP	Protocolo de Internet (Internet Protocol)
ITTP	Protocolo para Transferencia Inteligente entre Terminales (Intelligent Terminal Transfer Protocol)
JDBC	Conectividad para Base de Datos Java (Java Database Connectivity)
JSP	Página de Servidor Java (Java Server Page)
MTU	Unidad de Transmisión Máxima (Maximum Transmission Unit)
PDA	Asistente Personal Digital (Personal Digital Assistant)
PDU	Unidad de Datos del Protocolo (Protocol Data Unit)

RDBMS	Sistema Manejador de Bases de Datos Relacionales (Relational Database Management System)
SDK	Kit de Desarrollo de Software (Software Development Kit)
SDU	Unidad de Datos del Servicio (Service Data Unit)
SMS	Servicio de Mensajes Cortos (Short Message Service)
SSL	Capa de *Sockets* Seguros (Secure Sockets Layer)
TCP	Protocolo de Control de Transmisión (Transmission Control Protocol)
TTML	Lenguaje de Marcado de Texto Etiquetado (Tagged Text Markup Language)
UDP	Protocolo de Datagramas de Usuario (User Datagram Protocol)
UML	Lenguaje de Modelado Unificado (Unified Modeling Language)
UMTS	Sistema Universal de Telecomunicaciones Móviles (Universal Mobile Telecommunications System)
URI	Identificador Uniforme de Recursos (Uniform Resource Identifier)
URL	Localizador Uniforme de Recursos (Uniform Resource Locator)
WAE	Entorno Inalámbrico de Aplicación (Wireless Application Environment)
WAP	Protocolo de Aplicaciones Inalámbricas (Wireless Application Protocol)
WDP	Protocolo Inalámbrico de Datagramas (Wireless Datagram Protocol)
WML	Lenguaje de Marcado Inalámbrico (Wireless Markup Language)
WSP	Protocolo Inalámbrico de Sesión (Wireless Session Protocol)
WTA	Aplicaciones de Telefonía Inalámbrica (Wireless Telephony Applications)
WTAI	Interfaz para Aplicaciones de Telefonía Inalámbrica (Wireless Telephony Applications Interface)
WTLS	Capa de Transporte Seguro Inalámbrico (Wireless Transport Layer Security)

WTP	Protocolo Inalámbrico de Transacción (Wireless Transaction Protocol)
XML	Lenguaje de Marcado Extensible (eXtensible Markup Language)