



**UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS**  
**FACULTAD 6**



**DEPARTAMENTO DE GEOINFORMÁTICA / PROYECTO SIG-DESKTOP**

**SISTEMA DE INFORMACIÓN GEOGRÁFICA PARA LA UBICACIÓN Y GESTIÓN  
DE LA FLOTA DE ÓMNIBUS DE LA EMPRESA DE ÓMNIBUS NACIONALES  
(EON)**

**TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS  
INFORMÁTICAS**

**Autores:**

Yensaire Sencial Frometa  
Carlos Yoesly Oviedo Becerra

**Tutor:** Ing. Nilberto Caridad Chavez Marquez

**Co-tutor:** Ing. Alberto Menendez Romero

La Habana, junio de 2013  
"Año 55 de la Revolución"

## DECLARACIÓN DE AUTORÍA

Declaramos ser los únicos autores del presente trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste se firma la presente:

A los \_\_\_\_\_ días del mes de \_\_\_\_\_ del año \_\_\_\_\_.

---

Yensaire Sencial Frometa  
Autor

---

Carlos Yoesly Oviedo Becerra  
Autor

---

Ing. Nilberto Caridad Chavez Marquez  
Tutor

## DATOS DEL TUTOR

**Nombre y apellidos:** Ing. Nilberto Caridad Chavez Marquez.

**Correo electrónico:** [nchavez@uci.cu](mailto:nchavez@uci.cu)

**Categoría docente:** Asistente.

**Año de graduación:** 2007

**Profesión:** Ingeniero en Ciencias Informáticas.

## AGRADECIMIENTOS

Yensaire

A mi tutor Nilberto, por todo su apoyo, confianza y ayuda.

Al tribunal por la seriedad puesta durante la investigación y por la ayuda brindada.

A mis padres por toda su entrega, por todo el sacrificio hecho hasta el día de hoy.

A mi abuela y mi tío por estar siempre a mi lado.

A mis hermanos por ser parte de mi vida.

A mi novio Alberto por ser lo mejor que me ha pasado en la universidad y por la confianza, el cariño y apoyo que me brinda día a día.

A mis amigos Guille y Maritza, Marbelis por ser como unos padres para mí.

Carlos Yoesly

A mis padres Maritza y Carlos por ayudarme, guiarme, aconsejarme y respetarme las decisiones que he tomado en mi vida.

A mis abuelos maternos Laura (yeya), Ramón (papi), por su preocupación, dedicación, desvelo y el amor que me han brindado en cada paso de mi vida.

A mi tío Ramón Eddy por ser otro padre para mí, y ayudarme en cada instante que lo he necesitado.

A mi hermana Yaritza, por su gran amor y hermandad eterna.

A mi familia en general por ser mi mayor felicidad y la inspiración de todas mis metas alcanzadas hasta el momento, de veras muchas gracias por confiar en mí.

## *Agradecimientos*

---

A mis amigos, que han estado presentes en los momentos más cruciales de esta etapa,  
la RedBull, mi gente muchas gracias.

A mi hermanazo Eduardo por compartir todos estos años de duro trabajo.

A mi tutor Nilberto por su apoyo y guía certera.

A Alberto, por su gran ayuda incondicional, y por su paciencia hacia nosotros.

A los miembros del tribunal por su preocupación, dedicación y exigencia durante el  
periodo de desarrollo del trabajo de diploma.

A todas mis amistades de la Universidad que durante 5 años se convirtieron en mi  
familia y principal apoyo, y lo seguirán siendo; resulta realmente imposible listarlos a  
todos y agradecerles con la magnitud que se merecen.

A la Universidad y a la Revolución por darme la oportunidad de formarme como una  
profesional, de poder hacer realidad mis sueños y haberme permitido el privilegio de  
conocer tantas personas maravillosas.

## DEDICATORIA

Yensaire

A mis padre: Oneida y Rafael.

A mis hermanos: Yeimi, Yenima, Carlos Rafael y Yordanis.

A mi abuela: Mirian.

A mi tío: Orleydis.

A mi novio incondicional: Alberto.

A mis amigos, esos que siempre están ahí cuando estas en situaciones buenas y malas de la vida.

Carlos Yoesly

Dedico el presente trabajo de diploma a mi familia por su apoyo, cariño, comprensión y ejemplo a lo largo de mi vida, en especial a una persona, a mi abuela Laura.

*“Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber.”*

*Albert Einstein*

## RESUMEN

La ciencia y la tecnología en las últimas décadas se han desarrollado tomando auge la utilización de sistemas informáticos para el control y la visualización de flotas de ómnibus en las empresas. El crecimiento sostenido de estas flotas debido a la creación de nuevas rutas y el aumento del personal, han requerido el uso de estos sistemas informáticos en la automatización de los principales procesos de gestión de la flota. Los Sistemas de Información Geográfica juegan un papel importante en este sentido, dadas las bondades que brindan en el manejo y la representación de la información espacial de dichas flotas, facilitando el almacenamiento y la manipulación de la información asociadas a ellas. La presente investigación esboza el desarrollo de un Sistema de Información Geográfica para la ubicación y la gestión de la flota de ómnibus de la Empresa de Ómnibus Nacionales (EON), el cual proporciona una herramienta útil para el control de las flotas, los recursos humanos asociados a ella, así como la visualización de los recorridos en un intervalo de tiempo. Para el desarrollo de la solución se tuvo en cuenta las tendencias, tecnologías y herramientas que existen a nivel mundial, así como las políticas de software libre establecidas.

**Palabras claves:** Control de Flota, Ruta, Sistema de Información Geográfica.



## ÍNDICE

Introducción .....	1
CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA .....	5
1.1    Conceptos asociados al dominio del problema .....	5
1.2    Objeto de estudio .....	8
1.2.1    Descripción General.....	8
1.2.2    Descripción actual del dominio del problema .....	9
1.2.3    Situación problemática .....	9
1.3    Existencia de soluciones semejantes a la presente investigación .....	11
1.4    Conclusiones parciales .....	15
CAPÍTULO 2 TENDENCIAS Y TECNOLOGÍAS ACTUALES A UTILIZAR .....	17
2.1    Arquitectura de Software.....	17
2.1.1    Arquitectura Orientada a Objetos .....	17
2.2    Lenguaje de Programación .....	17
2.2.1    C++.....	18
2.3    Sistemas Gestores de Base de Datos.....	18
2.3.1    PostgresSQL.....	19
2.3.2    PostGis .....	19
2.4    Metodología de desarrollo de software.....	20
2.4.1    Proceso Unificado Racional (RUP) .....	21
2.5    Lenguaje de Modelado.....	23
2.5.1    Lenguaje Unificado de Modelado (UML) .....	23
2.6    Herramienta CASE.....	24
2.6.1    Visual Paradigm.....	24
2.7    Entorno de Desarrollo Integrado (IDE) .....	25
2.7.1    QT Creator .....	25
2.8    GeoQ: Como plataforma para desarrollo SIG.....	26
2.9    Conclusiones parciales .....	26
CAPÍTULO 3 PRESENTACIÓN DE LA SOLUCIÓN PROPUESTA.....	27
3.1    Modelo del negocio .....	27
3.1.1    Actores y trabajadores del negocio .....	27
3.1.2    Diagrama de casos de uso del negocio.....	29
3.1.3    Diagrama de actividades.....	29
3.1.4    Descripción de los casos de uso del negocio .....	30
3.2    Especificación de los requisitos.....	31
3.2.1    Requisitos funcionales .....	31
3.2.2    Requisitos no funcionales .....	33
3.3    Descripción del sistema propuesto.....	36
3.3.1    Descripción de los actores del sistema .....	37
3.3.2    Descripción de casos de uso del sistema.....	38
3.3.3    Conclusiones del capítulo .....	42
CAPÍTULO 4 CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA.....	43
4.1    Diseño.....	43
4.1.1    Patrones .....	43
4.1.2    Diagrama de clases del diseño .....	47
4.2    Diseño de la base de datos.....	48
4.2.1    Diagrama de clases persistentes .....	48

4.2.2	Modelo Entidad-Relación .....	49
4.3	Implementación.....	50
4.3.1	Modelo de Despliegue .....	50
4.3.2	Diagrama de componentes .....	50
4.4	Prueba .....	52
4.4.1	Pruebas de caja negra .....	53
4.5	Conclusiones parciales .....	58
	CONCLUSIONES .....	59
	RECOMENDACIONES .....	60
	REFERENCIAS BIBLIOGRÁFICAS .....	61
	BIBLIOGRAFÍA .....	68

## ÍNDICE DE FIGURAS

Figura 1 Arquitectura de OODISMAL. ....	13
Figura 2 Arquitectura de MovilWeb. ....	14
Figura 3 Modelo de objetos del negocio .....	28
Figura 4 Diagrama de casos de uso del negocio.....	29
Figura 5 Diagrama de actividades del caso de uso del negocio "Elaborar estado de la flota".....	29
Figura 6 Diagrama de actividades del caso de uso del negocio "Representar flota" .....	30
Figura 7 Diagrama de casos de uso del sistema .....	38
Figura 8 Vista global de clases y paquetes más significativos del diseño.....	47
Figura 9 Diagrama de clases del diseño del caso de uso "Gestionar planificación" .....	48
Figura 10 Diagrama de clases persistentes.....	49
Figura 11 Modelo Entidad - Relación (modelo físico) .....	49
Figura 12 Modelo de despliegue .....	50
Figura 13 Diagrama de componentes del caso de uso "Gestionar planificación".....	51
Figura 14 Cantidad de no conformidades.....	57

## ÍNDICE DE TABLAS

Tabla 1 Actores del negocio .....	28
Tabla 2 Trabajadores del negocio .....	28
Tabla 3 Descripción textual del caso de uso del negocio "Elaborar estado de la flota" ..	31
Tabla 4 Descripción de los actores del sistema.....	37
Tabla 5 Descripción textual del caso de uso del sistema "Gestionar planificación" .....	42
Tabla 6 Trazabilidad y encapsulamiento de los elementos del diseño en componentes	52
Tabla 7 Secciones a probar en el caso de uso "Gestionar planificación" .....	55
Tabla 8 Descripción de las variables para el caso de uso "Gestionar planificación" .....	56
Tabla 9 Matriz de datos. SC1: "Registrar nueva planificación". CU "Gestionar planificación" .....	56

## Introducción

El mundo actual impone a las empresas la necesidad de conocer en todo momento las tareas que se están realizando en la misma, y a la par administrar con eficiencia los recursos y el personal con el cual labora. De esta forma se pueden establecer un conjunto de medidas correctivas y evitar desviaciones en la ejecución de planes.

El control es de mucha importancia para cualquier institución, ya que significa la regla número uno para alcanzar la realización exitosa de sus planes. Además ayuda a realizar análisis para determinar rápidamente las causas de un problema surgido, localizar los responsables y proporciona la información necesaria sobre la situación actual de la ejecución de las tareas.

El término control, adquiere mayor relevancia, en aquellas instituciones que cuentan con un elevado número de personal. Esto implica un numeroso y amplio parque tecnológico de transporte; donde es necesario administrar correctamente los itinerarios y rutas para así disminuir costos. De este modo se introduce, el uso de control de flotas. Cuyo objetivo es brindar los mecanismos necesarios que permitan conocer constantemente el uso diario de cada vehículo de la institución.

Con la rápida evolución de las Tecnologías de la Información y la Comunicación (TIC) surgen nuevas herramientas que ayudan a realizar un mayor control sobre los vehículos de las empresas, ejemplo de ello son los Sistemas de Información Geográfica (SIG), el cual automatiza el manejo y la representación de la información de forma espacial, territorial y geográfica. Con la ayuda de un SIG las empresas pueden representar sobre un mapa la ubicación exacta de cada uno de sus vehículos y a partir de ello tomar decisiones más acertadas.

En Cuba, la Empresa de Ómnibus Nacionales (EON) es la principal encargada de la transportación de pasajeros mediante su flota de vehículos y rutas interprovinciales; prestando sus servicios a cualquier persona jurídica y natural, nacional o extranjero. Con el incremento de su parque de transporte, el cual ascendió a más de 1100 ómnibus en el año 2005, se hizo necesario la búsqueda de nuevas estrategias que permitieran una correcta administración y ejecución de las rutas.

En la actualidad la estructura de la EON está dada por un Director General al cual se subordinan varios departamentos y subdirecciones. En el Puesto de Mando de la Subdirección de Operaciones es donde se controla el estado de los vehículos, se realiza un seguimiento y se trazan nuevas rutas. Este proceso es de vital importancia para la empresa puesto que a partir del correcto control se pueden minimizar los gastos de recursos de sus vehículos y efectuar mejores recorridos por vehículos.

Los operadores son los encargados de dar seguimiento a cada vehículo de la empresa y para ello cuentan con modelos en hojas electrónicas de cálculo realizadas en Microsoft Excel donde llenan un conjunto de información por cada vehículo y actualizan los tiempos que demora un vehículo en recorrer determinada ruta. Además se mantiene un seguimiento del histórico para conocer cuándo un vehículo necesita entrar en reparación o mantenimiento.

Producto del volumen de información y la amplia flota con la que cuenta la EON, los operadores no pueden realizar las tareas de seguimiento con eficiencia y precisión. Además, al contar con herramientas informáticas no diseñadas para esta tarea se hace molesto y abre paso a la ocurrencia de errores, lo cual producen resultados de análisis incorrectos.

Es por tal motivo que se ha identificado como **problema a resolver** de la presente investigación: ¿Cómo disminuir los errores asociados a la asignación de rutas en la Empresa de Ómnibus Nacionales para lograr un mayor control de su flota de ómnibus?

Planteándose como **objeto de estudio**: El proceso de control de la flota de ómnibus, quedando definido como el **campo de acción**: Automatización del proceso de control de la flota de ómnibus en la Empresa de Ómnibus Nacionales.

Se define como **objetivo general**: Implementar un Sistema de Información Geográfica que permita analizar y representar espacialmente la información referente a la flota de ómnibus de la Empresa de Ómnibus Nacionales.

Como parte de la investigación se propone la siguiente **idea a defender**: Si se implementa un Sistema de Información Geográfica que analice y represente espacialmente la información referente a la flota de ómnibus de la Empresa de Ómnibus

Nacionales, entonces se disminuirán los errores asociados a la asignación de rutas y se logrará un mayor control de la flota de ómnibus de la empresa.

Para dar cumplimiento a los objetivos trazados se desarrollarán las siguientes **tareas** durante la investigación:

1. Caracterizar el proceso de gestión y ubicación de la flota de la Empresa de Ómnibus Nacionales.
2. Realizar una revisión de las tendencias y tecnologías actuales aplicables al proceso de negocio.
3. Modelar los procesos de Negocios, Diseño e Implementación del sistema.
4. Implementar la solución propuesta.
5. Probar y validar la solución propuesta.

Finalizada la investigación se espera obtener como **resultados**:

1. La documentación técnica del proceso de desarrollo.
2. El Sistema de Información Geográfica para la ubicación y gestión de la flota de ómnibus de la Empresa de Ómnibus Nacionales.

Para el desarrollo de la investigación se tendrán en cuenta los siguientes métodos científicos que serán de suma importancia para la realización satisfactoria de la misma.

Para el desarrollo de esta investigación científica, se utilizarán como métodos:

### **Métodos Teóricos**

1. Histórico - Lógico: En la primera fase de la investigación se desarrolla un estudio del estado del arte de la problemática analizada, revisando de forma crítica cada uno de los documentos para lograr un mejor entendimiento de lo que se debe desarrollar.
2. Analítico - Sintético: Se consultan distintas fuentes bibliográficas con el objetivo de seleccionar las más adecuadas a este trabajo. Se estudia con profundidad la información acerca de las tecnologías, metodologías y herramientas posibles a ser utilizadas, pudiendo definir con mayor certeza las mismas, para un mejor entendimiento de la situación y poder elaborar la solución propuesta.

3. Modelación: Se utiliza para la modelación de diagramas, representar el proceso de desarrollo y propiciar un mejor entendimiento de la solución a implementar.

### **Métodos Empíricos**

1. Observación: Se realizan visitas a la Empresa de Ómnibus Nacionales con el objetivo de observar los procesos que allí se desarrollan, las técnicas y mecanismos de gestión de las rutas y vehículos.
2. Entrevistas: Se utiliza este método para establecer el alcance del presente trabajo, el objetivo, el flujo actual de la información así como los requisitos que contendrá la aplicación a desarrollar. Se espera poder realizar entrevistas a los especialistas del Departamento de Operaciones de la EON.

La investigación está dividida en 4 capítulos fundamentales:

En el **capítulo 1** se expone el objeto de estudio, la descripción del entorno donde se desarrolla la investigación y se describen los conceptos fundamentales relacionados con el dominio del problema.

En el **capítulo 2** se argumenta la selección de las tecnologías y herramientas que se utilizarán en el proceso de desarrollo del software, lenguajes de programación y modelado así como la arquitectura, la metodología a utilizar y el entorno de desarrollo.

En el **capítulo 3** se expone la solución propuesta con todos sus argumentos, el modelo de negocio, los requisitos funcionales y no funcionales, así como la descripción de los casos de usos y los actores del sistema.

Finalmente en el **capítulo 4** se plantea la construcción propuesta en el capítulo 3. Se exponen los diagramas de clases del diseño para cada caso de uso del sistema, los diagramas de interacción, el diagrama de despliegue, el modelo de implementación y se realiza el diseño de los casos de prueba.



## **CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA**

En este capítulo se plantean los conceptos relacionados con la fundamentación teórica de la presente investigación, el objeto de estudio y la existencia de soluciones informáticas que respondan a las exigencias actuales de la EON. Finalmente se detallan los principales procesos que se llevan a cabo hoy como parte de las actividades de los especialistas en el Departamento de Operaciones.

### **1.1 Conceptos asociados al dominio del problema**

Cada vez que se realiza una investigación surgen conceptos esenciales asociados al problema tratado, los cuales constituyen la base fundamental y resulta de suma importancia su comprensión para el desarrollo del trabajo. A continuación se relacionan los principales conceptos asociados al dominio del problema, los cuales facilitarán la comprensión de la presente investigación.

#### **Sistema de Información Geográfica**

Un Sistema de Información Geográfica (SIG) es un conjunto de herramientas computacionales y algoritmos matemáticos para trabajar con los datos referentes a una determinada información geográfica, el trabajo consta en capturar, almacenar, analizar, transformar y presentar toda la información geográfica y de sus atributos con el fin de satisfacer las necesidades de determinados usuarios. La función principal de este tipo de software es la de contar con una cartografía ubicada en una determinada base de datos y permitir a los clientes interactuar con ella y la información asociada a esta **(Martín, 2011)** .

Un Sistema de Información Geográfica también puede definirse como la integración de hardware, software y un conjunto de procedimientos diseñados con el objetivo de facilitar la obtención, manipulación, gestión, modelado, análisis, representación y salida de información referenciada geográficamente, para resolver problemas complejos de planificación y gestión, y contribuir a la toma de decisiones, dando además una perspectiva totalmente nueva y dinámica de la información **(Linde, 2012)**.

En esencia, pueden concretarse como sistemas con la capacidad de manipular datos espaciales, brindando una herramienta que permite visualizar y analizar la información de forma versátil e intuitiva (**Farré, y otros, 2010**). De este modo se hace fácil la gestión de la información espacial, que es almacenada en una base de datos geográfica, combinándola con la información temática; donde cada tema representa una capa con la información geográfica de los objetos espaciales junto con la tabla de información asociada a la misma.

Los SIG contribuyen al estudio de la distribución y monitoreo de recursos de todo tipo, tanto naturales como humanos y muchas actividades en las que es necesario el manejo de grandes masas de información geográficamente referenciada.

## **Dato espacial**

Un dato espacial es una variable asociada a una localización del espacio, las formas de un objeto geográfico y las relaciones entre ellos, comúnmente con coordenadas y topología (**Guillaumet, 2000**).

## **Control**

De acuerdo al Diccionario de la Real Academia de Lengua Española la palabra control en una de sus acepciones significa: comprobación, inspección, fiscalización, intervención. También puede definirse control como un mecanismo que permite corregir desviaciones a través de indicadores cualitativos y cuantitativos dentro de un contexto social amplio, a fin de lograr el cumplimiento de los objetivos claves para el éxito organizacional, entendiéndose no como un proceso netamente técnico de seguimiento, sino también como un proceso informal donde se evalúan factores culturales, organizativos, humanos y grupales (**RAE, 2012**).

El control debe verse como una función administrativa que permite constatar si determinada actividad, proceso o sistema alcanza los resultados que se esperan; aunque una empresa cuente con magníficos planes, una estructura organizacional adecuada y una dirección eficiente, sin un mecanismo que se cerciore si los hechos van de acuerdo con los objetivos, no se podrá verificar cual es la situación real de la organización.

## **Flota**

Una flota se puede entender como un conjunto de vehículos que pueden ser tanto marítimos, terrestres o aéreos. Cuando los vehículos comparten características semejantes se dice que la flota es homogénea, y si son diferentes son clasificados como flota heterogénea (**Fernández, 2011**).

## **Control de flota**

El control de flotas está centrado en la localización, mensajería, telemetría e información de históricos, relativo a la planificación de rutas, cálculo de costos y optimización de cargas. Permite la planificación de las rutas al detalle y la importación y exportación de datos (**CyMSat, 2005**).

A partir de los conceptos Control y Flota se puede llegar al consenso de que control de flota se refiere al mecanismo que permite verificar y constatar el comportamiento de una flota de vehículos de acuerdo a indicadores como la ubicación, la velocidad, rutas y trayectorias recorridas; con el propósito de lograr los objetivos propuestos en la planificación de estos recursos, detectar desvío de los patrones normales establecidos y corregir estas desviaciones en el proceder de las flotas.

## **Tiempo de motor**

A los efectos del problema analizado, se puede definir tiempo de motor como la unidad medible que refleja el espacio de tiempo en que estuvo funcionando el motor del ómnibus. Se expresa en horas (h), minutos.

El valor final de tiempo se expresa como la suma de todos los espacios de tiempo en que se mantuvo encendido el motor mientras duró el viaje.

## **Ruta**

Camino o dirección que permite transitar desde un lugar a otro con un propósito. Es la unión de varios puntos que posibilita la unión de un punto origen y un punto destino (**RAE, 2012**).

## 1.2 Objeto de estudio

### 1.2.1 Descripción General

La Empresa de Ómnibus Nacionales (EON) surge a partir de la necesidad de un servicio de transportación de pasajeros mediante rutas interprovinciales y fletes; prestando de esta forma servicios a cualquier persona jurídica y natural, nacional o extranjero. Los principales servicios que brinda se puede enumerar en:

- Venta de pasajes a la población en pesos cubanos y en pesos convertible a las entidades nacionales y extranjeras.
- Alquiler de ómnibus, minibuses y microbuses.
- Servicio de mensajería, paquetería y custodia de equipajes a la población.
- Servicios de asistencia en la vía, remolques y mantenimiento de medios de transporte.
- Arrendamiento y alojamiento no turístico a los trabajadores del sistema del Ministerio de Transporte en funciones de trabajo y plan vacacional.

En el año 1995 la EON transitó por varias etapas en busca de una mejora en su estructura interna, pasando de este modo por la Asociación de Transporte por Ómnibus y luego al Grupo Empresarial de Transporte por Ómnibus. En aquel entonces su parque automotor apenas superaba los 100 carros en la transportación nacional, el cual fue incrementado el 13 de junio del 2005 con la llegada de 1100 ómnibus tipo Yutong producidos en China fruto de las relaciones diplomáticas de ambas naciones.

La revitalización del transporte devino en un total de 130 rutas (66 desde La Habana) y 199 salidas (89 desde La Habana), de ellas 149 diarias y 50 alternas lo que suponía una cifra histórica para la empresa y el país.

En el 2011 la empresa cambia su nombre a Empresa de Ómnibus Nacionales y se plantea una reestructuración interna en cuanto a organización, concibiendo 19 unidades empresariales distribuidas por todo el territorio nacional y una oficina central que controla y coordina las unidades. Esta oficina central radicada en la provincia La Habana está compuesta por un director general, varias subdirecciones y departamentos.

## 1.2.2 Descripción actual del dominio del problema

Producto del crecimiento del parque automotor de la EON, se hace necesario un estricto control de las rutas y viajes de sus vehículos, así como mantener un histórico de las operaciones realizadas con el objetivo de poder ganar en eficiencia y ahorrar recursos.

Es el Departamento de Operaciones el cual tiene estas responsabilidades. En este departamento los especialistas mantienen un registro de los viajes que realizan los ómnibus de la empresa, crean y dan seguimiento a las rutas de viajes y asignan el personal necesario para cumplir con las tareas requeridas. Cada día los especialistas mantienen un registro de los tiempos empleados para recorrer determinada ruta, y a su vez controlan aspectos como: el kilometraje que lleva cada vehículo para saber cuándo debe entrar en mantenimiento, los puntos más cercanos para abastecer el combustible, el tiempo de motor, entre otros.

En cada viaje, es responsabilidad del chofer el llenar una planilla de viaje conocida como hoja de ruta, donde se reflejan varios aspectos de interés para el Departamento como: la hora de salida y la hora de llega, el tiempo de recorrido empleado, cantidad de pasajeros transportados, puntos donde se abasteció el combustible, etc. Además, con cierta regularidad durante el viaje, el chofer informa su posición en la ruta para así tener conocimiento por parte del Departamento de la ubicación exacta de sus vehículos.

Los trabajadores del Departamento de Operaciones recogen por cada ómnibus toda esta información en tablas realizadas en Microsoft Excel con el objetivo de mantener un histórico de operaciones y a partir de estas poder realizar análisis y actuar en consecuencia.

## 1.2.3 Situación problemática

El Departamento de Operaciones mantiene un registro de los viajes realizados por sus vehículos, guardando esta información en modelos de Microsoft Excel. Producto de las numerosas rutas y viajes que se realizan en el día, con el transcurso del tiempo la cantidad de documentos crece considerablemente, lo que trae problemas en el momento de consultar el historial para determinar cualquier indicador requerido.

El no contar con herramientas adecuadas para el manejo de esta información acarrea una serie de problemas que no permiten la correcta ejecución y control de las principales operaciones de la empresa. Entre estos problemas se pueden mencionar:

- La recopilación de la información de las rutas no se obtiene de manera eficiente, lo que provoca un empleo de tiempo mayor y por ende disminuye la eficiencia de esta operación.
- No se garantiza el cumplimiento de los planes de ruta o plan de consumo de recursos asignados (como combustible) debido a que las hojas de rutas no siempre tienen la precisión adecuada.
- La información con la que se trabaja se gestiona escasamente debido al cúmulo de datos y los análisis que se obtienen no son los más precisos por lo que disminuye considerablemente el control que se establecen sobre las rutas de viajes y los gastos de las mismas.
- A pesar que durante el recorrido de los ómnibus estos informan su posición usando distintas vías, el Departamento de Operaciones no posee una visión general ni una representación de sus vehículos en la vía.
- Las herramientas actuales no proveen los mecanismos necesarios para realizar simulaciones basados en pronósticos en el tiempo, de nuevas rutas o con las ya existentes, así como lograr una visión global de las rutas, cruces, puntos de abastecimiento, etc.
- En ocasiones se dificulta determinar cuándo un ómnibus está listo para entrar al taller por mantenimiento o la vida útil del mismo, ya que consultar el histórico de los datos de dicho ómnibus es una tarea tediosa y requiere de tiempo.
- Se dificulta el cálculo de combustible consumido el cual se obtiene a partir de la información de combustible servido al vehículo, agregando el que tienen en el depósito al inicio y descontando el restante al final del período.
- También se dificulta en menor medida determinar la aptitud de la flota para su utilización; que no es más que el coeficiente de disponibilidad técnica obtenido a partir de la suma de los vehículos en buen estado técnico entre la suma de los vehículos existentes en la entidad.

Es por lo anteriormente mencionado que se puede afirmar que los datos de la EON no se gestionan de manera eficiente y el proceso de ubicación y gestión de la flota de

ómnibus en el Departamento de Operaciones se realiza con el apoyo de varias herramientas ofimáticas o de manera manual lo que trae incongruencias en la geo-referenciación de los datos y errores a la hora de realizar los análisis. Lo cual constituye el problema a resolver de la investigación actual.

### 1.3 Existencia de soluciones semejantes a la presente investigación

Analizado los principales procesos que se realizan en el Departamento de Operaciones y luego de haber identificado los principales problemas que enfrentan los especialistas; se hace un estudio en busca de soluciones de software existentes que brinden alguna solución. El objetivo de este análisis está dirigido a determinar cuáles de estos sistemas pudieran contribuir y servir de apoyo para el planteamiento de la solución a la cual se desea arribar al culminar esta investigación.

Con el incremento de la competencia en el sector del transporte y la necesidad de disminuir los costes de servicios y aprovechar al máximo los recursos; ha llevado a varias empresas a interesarse por los sistemas de seguimiento y control de flotas. Un ejemplo lo constituye la empresa española Micronav, que a partir de su creación en el 2004, atiende solicitudes de pequeñas y medianas empresas con el fin de brindar soluciones tecnológicas estándares para la gestión y control eficiente de los recursos disponibles en una flota.

Dentro de las soluciones que ofrece la empresa se encuentra **Micronav Localización GPS Vehículos**, un sistema de software y hardware para el control y gestión de flotas mediante las tecnologías Sistema de Posicionamiento Global (GPS) e Internet. Esta solución cuenta con varios paquetes de servicios adaptables de forma flexible a los requisitos del cliente, el cual debe pagar por cada paquete de funcionalidades elegido.

La solución está diseñada especialmente para empresas instaladoras, de reparto y de alquiler de vehículos o maquinarias. Entre las principales funcionalidades que brinda se pueden mencionar: **(Micronav, 2009)**

- Control de la flota en tiempo real.
- Generación de informes completos de la actividad de toda la flota o por vehículo.
- Gestionar puntos de paso y supervisar el cumplimiento de rutas.

- Visualizar la información cartográfica mediante Google Earth.

Otras de las empresas en el plano internacional que brindan soluciones para el control de flotas es la compañía mexicana GlobalSat. Con más de 10 años de experiencia en la integración de tecnologías satelitales y líder en servicios avanzados de telecomunicación satelital, esta empresa comercializa la solución **Localización de flotas GlobalSat**.

El sistema Localización de flotas GlobalSat y el hardware que lo acompaña, permite ver sobre un mapa el lugar exacto en el cual se encuentra en todo momento la flota de vehículos que se controla; además dónde han parado y cuánto tiempo han dedicado a cada parada. El programa permite comprobar de un vistazo el cumplimiento de la ruta prevista, para cualquier día y cualquier vehículo, ayudando al control de flotas.

Este sistema de localización es compatible con los navegadores GPS actuales e implementa el módulo de localización automática de vehículos con dispositivo de telefonía móvil donde la información es almacenada para futuros controles que se deseen realizar. Algunas características con las que cuenta la aplicación son **(GlobalSat, 2011)** :

- Seguimiento de vehículos automático y en tiempo real. La precisión de la localización de los vehículos es de 10 metros, y las posiciones se registran cada 8 segundos.
- La cartografía de la aplicación de control de flotas admite diversos niveles de zoom, y puede ser incorporada a medida.
- Brinda la posibilidad de programar alarmas que envían avisos automáticos mediante SMS a móviles, o vía e-mails. Estas alarmas avisan si el vehículo a estado o no en cierto sitio, y admiten todo tipo de condiciones.

Otra solución válida de mencionar es la creada por un grupo de Investigación y Desarrollado (I+D) integrado por ingenieros y profesores adscritos al departamento de Informática e Ingeniería con sede en el Centro Politécnico de la Universidad de Zaragoza. De los esfuerzos de sus integrantes, surge el Sistema de Información Distribuido Orientado a Objeto para el Seguimiento y Control de Flotas (**OODISMAL**, del inglés Object Oriented Distributed Information System for Automatic Movil Location); el



mismo proporciona los componentes básicos para integrar las comunicaciones de radio, los datos GPS adquiridos en tiempo real y su análisis con SIG.

Como principal premisa de esta solución se tiene el empleo de una arquitectura distribuida cliente–servidor que explota las nuevas tecnologías del desarrollo de software orientado a objeto, lo que posibilita el desarrollo de aplicaciones ligeras con facilidad de escalamiento para ajustarlo a las peculiaridades de los clientes (Ver Figura 1).

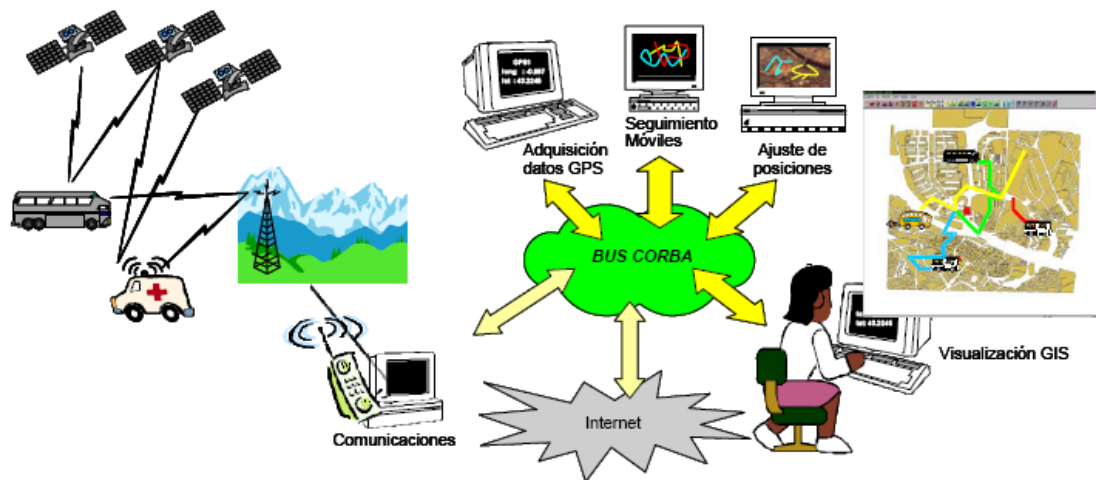


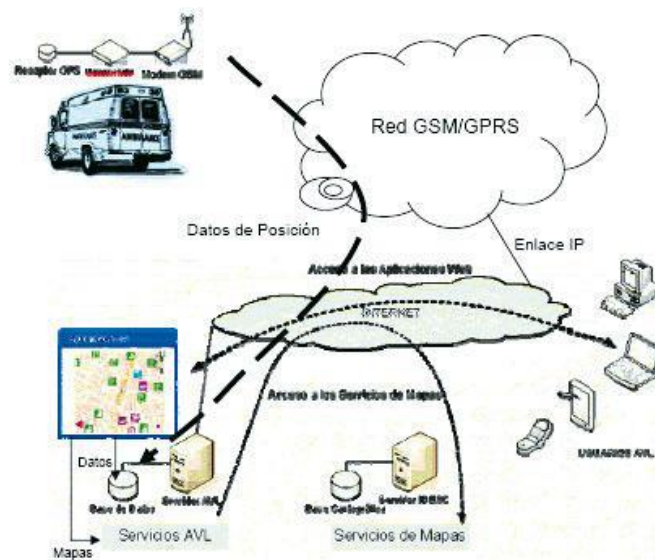
Figura 1 Arquitectura de OODISMAL.

Entre las funcionalidades más destacadas de esta solución se pueden mencionar:

- Adquisición de datos de los móviles a través de enlaces de comunicación.
- Utilidades para mejoras de precisión y de ajuste a rutas, análisis de rutas relacionados con distancias, tiempos, velocidades y posiciones, además de la correlación entre rutas reales y previstas.
- Interfaz gráfica de usuario para el acceso a los datos y visualización de datos en mapas digitales que pueden realizarse por Internet.

Tal es la popularidad de los sistemas de control de flotas y las necesidades reales en las empresas, que en Cuba se han iniciado varios proyectos con vista a obtener productos nacionales con este fin. **MovilWeb** es una aplicación web para el control de flotas basada en la Infraestructura de Datos Espaciales de la República de Cuba y diseñada para controlar los diferentes vehículos sobre una cartografía.

Con una arquitectura cliente – servidor y a su vez orientada a servicios, MovilWeb integra varios servicios de procesamiento de datos geospaciales distribuidos a través de la web; permitiendo obtener datos de posicionamiento de los móviles, en tiempo real y diferido y enviarlos a un servidor central de Localización Automática de Vehículos (AVL, del inglés Automatic Vehicle Location) (Ver Figura 2).



**Figura 2 Arquitectura de MovilWeb.**

Inicialmente, en el año 2006, la solución se desarrolló con la tecnología ASP<sup>1</sup> y el IIS<sup>2</sup>, utilizando el sistema gestor de base de datos Microsoft SQL Server 2000 y los componentes principales del sistema estaban desarrollados en Borland Delphi 7. A partir de los avances obtenidos en el campo del software libre se desarrolló una nueva versión de MovilWeb con similares prestaciones; pero esta vez en el lenguaje de programación Java con el sistema gestor de base de datos PostgreSQL. **(González, y otros, 2011)**

Entre las prestaciones principales de este sistema se encuentran: **(González, y otros, 2008)**

- Permite hacer un seguimiento online de los móviles.

<sup>1</sup> Del inglés Active Server Pages. Es una tecnología de Microsoft que se ejecuta en el lado del servidor para páginas web generadas dinámicamente.

<sup>2</sup> Del inglés Internet Information Server. Es un servidor para páginas web de la empresa Microsoft.

- Permite el registro del comportamiento de la flota mediante tarjetas de memoria, lo cual posibilita el trabajo en modo desconexión (offline).
- Posibilita auditar el comportamiento de los móviles y explorar la información del historial.
- Facilita la creación de reportes esenciales como distancias recorridas, consumo de combustible, registro de velocidades, etc.
- Realiza operaciones de análisis temático en cuanto a velocidad y detenciones de los vehículos.

Como resultado del análisis de las soluciones existentes presentadas se puede determinar el alto grado de acabado, prestación y el eficiente manejo de la información espacial y geográfica para el control de flotas. No obstante, en su mayoría son soluciones propietarias que requieren la adquisición de licencias para su uso y/o la autorización de sus propietarios. Además, la adquisición de una de estas soluciones no implica la obtención de su código fuente, por lo que se hace casi imposible incorporar nuevas funcionalidades al sistema sin contar con la empresa que la confecciona. Tanto Localización de Flotas GlobalSat, como Micronav Localización GPS Vehículos y OODISMAL son aplicaciones que demandan la autorización de sus propietarios para su explotación a través de licencia, estas soluciones existentes para el control de flotas no cumplen con la política de migración de software libre aunque podrían contribuir al aporte de elementos a tener en cuenta a la hora de impulsar el desarrollo del sistema.

## 1.4 Conclusiones parciales

En el presente capítulo se identificaron los principales conceptos y definiciones asociados a los SIG y a los sistemas para el control de flotas; lo cual provee una mejor comprensión de la investigación y el problema que se desea solucionar.

Con la búsqueda y análisis de soluciones que existen en el contexto nacional e internacional relacionado con la temática abordada, se arribó a la conclusión que a pesar de ser soluciones completas en su mayoría, utilizan tecnologías propietarias que son de difícil acceso por su tipo de patente; además requieren de una inversión de capital para poner a punto la infraestructura tecnológica necesaria; por ello es necesario buscar otro tipo de soluciones.

El análisis de las herramientas que se utilizan en el Departamento de Operaciones de la EON, arrojó que son herramientas no acordes con los procesos que allí se realizan, lo cual interrumpen o atrasan la forma en que se controlan los vehículos de la empresa, invirtiendo de este modo en tiempo y no pudiendo controlar de forma correcta los recursos de la empresa.

## **CAPÍTULO 2 TENDENCIAS Y TECNOLOGÍAS ACTUALES A UTILIZAR**

En este capítulo se argumenta la selección de las tecnologías y herramientas que se utilizarán en el proceso de desarrollo del software, lenguajes de programación y de modelado así como la arquitectura, la metodología a utilizar y el entorno de desarrollo.

### **2.1 Arquitectura de Software**

La arquitectura de una aplicación es la vista conceptual de la estructura de ella misma. Toda aplicación contiene código de presentación, código de procesamiento de datos y código de almacenamiento de datos. La arquitectura de las aplicaciones difiere según como está distribuido este código. **(CORNEJO, 2001)**

“La Arquitectura de Software constituye un puente entre el requerimiento y el código, ocupando el lugar que en los gráficos antiguos se reservaba para el diseño”. **(Reynoso, 2004)**

Existen numerosas definiciones, que los expertos en el tema precisan para la Arquitectura de software, sin embargo esta investigación se regirá por la que aparece redactada en el documento de IEEE Standard 1471-2000 donde define la Arquitectura de Software como la organización de un sistema en términos de sus componentes de software, incluyendo los subsistemas y las relaciones e interacciones entre ellos, los principios que guían el diseño de ese sistema de software.

#### **2.1.1 Arquitectura Orientada a Objetos**

Los componentes de este estilo son los objetos, o más bien instancias de los tipos de datos abstractos; se basan en principios Orientado a Objetos: encapsulamiento, herencia y polimorfismo. Los objetos son asimismo las unidades de modelado, diseño e implementación, y los objetos y sus interacciones son el centro de las incumbencias en el diseño de la arquitectura y en la estructura de la aplicación. Las interfaces están separadas de las implementaciones. En general la distribución de objetos es transparente **(Reynoso, 2004)**.

### **2.2 Lenguaje de Programación**

Un lenguaje de programación es una técnica de comunicación estandarizada para expresar las instrucciones a una computadora. Se trata de un conjunto de reglas sintácticas y semánticas utilizadas para definir los programas de ordenador **(Cueva, 1998)**.

Un lenguaje de programación de alto nivel se caracteriza por expresar los algoritmos necesarios para la creación de los programas informáticos de una manera adecuada a la capacidad cognitiva humana en lugar de a la capacidad ejecutora de las máquinas, es decir, facilita la comunicación con un computador mediante signos convencionales cercanos a los de un lenguaje natural **(Cueva, 1998)**.

### 2.2.1 C++

El C++ es un lenguaje versátil, potente y general. Su éxito entre los programadores profesionales le ha llevado a ocupar uno de los primeros puestos como herramienta de desarrollo de aplicaciones de escritorio. El C++ mantiene las ventajas del lenguaje C en cuanto a riqueza de operadores y expresiones, flexibilidad, concisión y eficiencia. Además, ha eliminado algunas de las dificultades y limitaciones del C original **(Brazales, y otros, 1998)**.

Se puede decir que C++ es un lenguaje multiparadigma que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos. Actualmente es muy popular en la programación de aplicaciones.

Este lenguaje se eligió como el lenguaje principal para la implementación del negocio del sistema de información geográfica por sus características.

### 2.3 Sistemas Gestores de Base de Datos

En la actualidad un porcentaje elevado de las tareas que se realizan en un ordenador, computadora o sistema informático requiere de la preservación de cierta cantidad de datos con el objetivo de que perduren en el tiempo, lo cual se ha convertido en una necesidad de vital importancia para la mayoría de las organizaciones e instituciones. Es por esto que han surgido técnicas y tecnologías que aportan los mecanismos necesarios a las aplicaciones informáticas para que cumplan con este objetivo.

Una base de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. Este término “...no incluye la aplicación informática que consiste en los formulario y reportes con que el usuario interactúa” (RIORDAN, 1999).

Un Sistema Gestor de Base de Datos (en lo adelante SGBD) es un conjunto de programas informáticos que permiten crear y mantener una base de datos, asegurando la integridad, confidencialidad y seguridad de los datos que posteriormente se convertirán en información relevante para una organización. Entre sus principales funciones se encuentra el de centralizar los accesos a los datos y actuar como intermediario entre los datos físicos y el usuario.

### 2.3.1 PostgreSQL

PostgreSQL es un SGBD objeto-relacional, orientado a objetos y multiplataforma, dirigido por una comunidad de desarrolladores y organizaciones comerciales denominada PGDG (PostgreSQL Global Development Group). Se puede considerar el gestor de base de datos de código abierto más avanzado que se puede utilizar hoy en día (PostgreSQL, 2013).

Dentro de las características principales que se pueden atribuir a este sistema se encuentra que ofrece soporte para el lenguaje SQL92\SQL3, integridad de transacciones y extensibilidad de tipos de datos. Además, sirve de soporte a los lenguajes de programación más populares como PHP, C, C++, Java, entre otros.

Entre los módulos con lo que cuenta se encuentra PostGis, de gran utilidad a la hora de trabajar con bases de datos espaciales. Esta tipología de base de datos se diferencia por contener además información referente a la localización espacial de objetos geográficos, elementos que tienen forma y ubicación en el espacio.

### 2.3.2 PostGis

PostGIS es el módulo espacial de PostgreSQL permite la manipulación y almacenamiento de datos espaciales, es capaz de tratar grandes volúmenes de datos

con escalabilidad y puede usarse, con adaptaciones, en cualquier plataforma. Es de código abierto, distribuido bajo la licencia GNU (General Public License).

Soporta funciones básicas para el análisis espacial de objetos. Entre esas funciones se pueden encontrar: **(Espinosa, 2010)**:

- Analizar si dos geometrías cuentan con un punto en común (Disjoint ()).
- Analizar si dos geometrías poseen alguna intersección (Intersects ()).
- Analizar si dos geometrías se cruzan (Crosses ()).
- Analizar si una geometría contiene a la otra (Contains ()).

### 2.4 Metodología de desarrollo de software

En la actualidad la construcción de una aplicación informática eficiente que cumpla con los requerimientos planteados por el cliente es una tarea intensa y difícil de cumplir. Las metodologías para el desarrollo del software imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente.

Una metodología de desarrollo del software tiene como principal objetivo aumentar la calidad del software que se produce en todas y cada una de sus fases de desarrollo; definiendo Quién debe hacer Qué, Cuándo y Cómo debe hacerlo. Estas metodologías guían a los desarrolladores de un software en el diseño e implementación del mismo.

Hoy en día existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo y seleccionar la adecuada que posibilite obtener los resultados óptimos es uno de los principales problemas a que se enfrentan los equipos de desarrollo.

- **Metodologías tradicionales o pesadas:** aquellas metodologías orientadas al control de los procesos, estableciendo rigurosamente las actividades a desarrollar, herramientas a utilizar y notaciones que se usarán.
- **Metodologías ágiles o ligeras:** aquellas metodologías orientadas a la interacción con el cliente y el desarrollo incremental del software, mostrando versiones parcialmente funcionales del software al cliente en intervalos cortos de



tiempo, para que pueda evaluar y sugerir cambios en el producto según se va desarrollando.

### 2.4.1 Proceso Unificado Racional (RUP)

Entre las metodologías tradicionales o pesadas se encuentra el Proceso Unificado Racional (Rational Unified Process en inglés, abreviado como RUP), enfocada en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar, y requiere una extensa documentación.

RUP está constituido por 9 flujos de trabajo (ver Figura 3), los 6 primeros son conocidos como flujos de ingeniería: modelación del negocio, requisitos, análisis y diseño, implementación, prueba, instalación; y los 3 últimos de apoyo: gestión de configuración y cambios, gestión de proyectos y entorno.

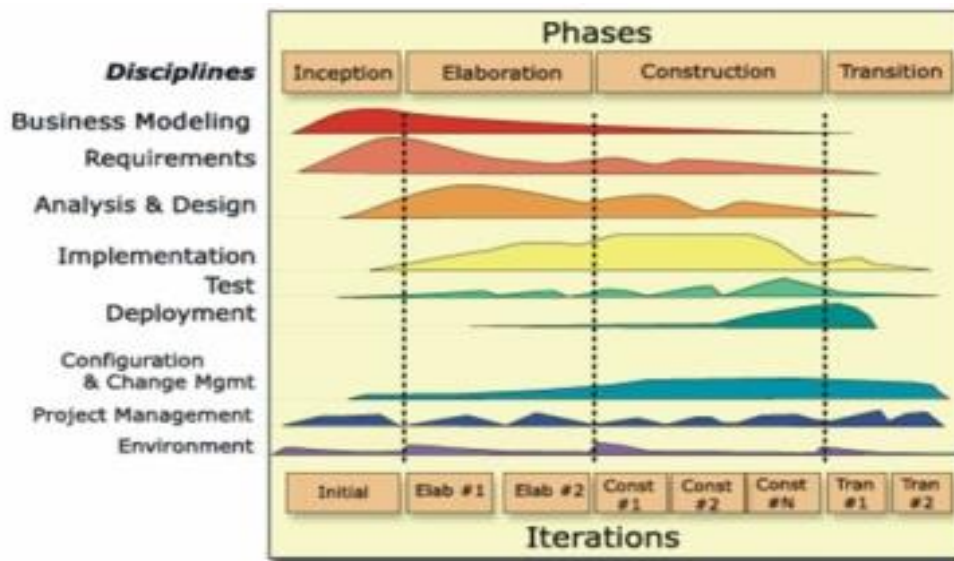


Figura 3 Estructura del ciclo de vida del RUP.

La metodología RUP posee tres características esenciales: (Jacobson, y otros, 2000)

- Dirigido por casos de uso: Los casos de usos representan el hilo conductor que orienta las actividades de desarrollo. Se centra en la funcionalidad que el sistema debe poseer para satisfacer las necesidades del usuario.

- Centrado en la arquitectura: Establecer una arquitectura candidata al inicio es un paso muy importante, pues será ella la que guíe el desarrollo del sistema. Esto permitirá el desarrollo en paralelo, minimizar la repetición de trabajos e incrementar la probabilidad de reutilización de componentes.
- Iterativo e incremental: El desarrollo iterativo brinda la posibilidad de que los elementos sean integrados progresivamente, facilita el rehúso y resulta un producto más robusto pues los errores se van corrigiendo en cada iteración.

Las 4 fases o etapas de la metodología RUP son: **(Tobarra, 2003)**

- Concepción: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos.
- Elaboración: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos.
- Construcción: se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario.
- Transición: se implementa el producto en el cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados.

Cada una de estas fases es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones anteriores.

Debido a la complejidad y alcance de la propuesta del software a realizar, la investigación se desarrollará utilizando la metodología de desarrollo RUP ya que es la metodología más adecuada, dada la magnitud del proyecto. En cuanto a las características de la misma permite garantizar la corrección de los errores en cada iteración, para una mayor calidad de la solución. De igual manera las metodologías ágiles no son factible para el proyecto SIG-DESKTOP puesto que estas solo se comprometen con pequeñas soluciones y requieren una constante interacción con el cliente, lo que no es posible, además, no define una arquitectura temprana y en cambio la va reconstruyendo en consecuencia de las versiones y requisitos que va presentando el usuario final, lo que para proyectos de gran escala sería un costoso gasto de tiempo y recurso.

### 2.5 Lenguaje de Modelado

El lenguaje de modelado pretende dar apoyo a la mayoría de los procesos de desarrollo de software, son desarrollados con el esfuerzo de simplificar y consolidar el gran número de métodos de desarrollo que han surgido.

#### 2.5.1 Lenguaje Unificado de Modelado (UML)

La falta de estandarización en la forma de representar gráficamente un modelo impedía que los diseños gráficos realizados se pudieran compartir fácilmente entre distintos diseñadores. Se necesitaba por tanto un lenguaje no sólo para comunicar las ideas a otros desarrolladores sino también para servir de apoyo en los procesos de análisis de un problema.

Lenguaje Unificado de Modelado prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. **(Pockin, 2008)**

UML (del inglés Unified Modeling Language) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema de software. Sus creadores pretendieron, con este lenguaje, unificar las experiencias acumuladas sobre técnicas de modelado e incorporar las mejores prácticas en un acercamiento estándar. **(Pockin, 2008)**

UML ayuda a los usuarios a entender la realidad desde un punto de vista de la tecnología y la posibilidad de que reflexione antes de invertir y gastar grandes cantidades de dinero en proyectos que no estén seguros en su desarrollo, reduciendo el costo y el tiempo empleado en la construcción de los módulos que construirán el software. **(Rumbaugh, y otros, 2000)**

Entre los rasgos principales que han contribuido a hacer de UML el estándar de la industria en la actualidad, se pueden mencionar: **(Rumbaugh, y otros, 2000)**

- Permite modelar sistemas utilizando técnicas orientadas a objetos.
- Adecuado a las necesidades de conectividades actuales y futuras.

- Es un lenguaje muy expresivo que cubre las vistas necesarias para desarrollar y luego desplegar los sistemas.
- Ampliamente utilizado por la industria del software.
- Reemplaza a decenas de notaciones empleadas por otros lenguajes.
- Modela estructuras complejas.
- Comportamiento del sistema: casos de usos, diagramas de secuencia, de colaboración, que sirven para evaluar el estado de las máquinas.

### 2.6 Herramienta CASE

Las Herramientas CASE (del inglés Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) son aplicaciones informáticas con el fin de aumentar la productividad de las áreas de desarrollo y mantenimiento de los sistemas informáticos, mejorar la calidad del software, y reducir el costo en términos de dinero y tiempo (**Menéndez, 2010**).

#### 2.6.1 Visual Paradigm

Visual Paradigm es una herramienta CASE que utiliza UML como lenguaje de modelado. Está diseñada para una amplia gama de usuarios interesados en construir sistemas fiables con el uso del paradigma orientado a objetos, incluyendo actividades como ingeniería de software, análisis de sistemas y análisis de negocios (**Rumbaugh, y otros, 2000**).

Sus características principales:

- Entorno de creación de diagramas para UML.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad de integrarse en los principales IDEs.
- Disponibilidad en múltiples plataformas.

Visual Paradigm tiene como ventajas la disponibilidad en múltiples plataformas entre ellas: Microsoft Windows (98, 2000, XP, o Vista), Linux, Mac OS X, Solaris; así como también brinda el apoyo todo el básico en cuanto a artefactos generados en las etapas de definición de requisitos y de especificación de componentes.

### 2.7 Entorno de Desarrollo Integrado (IDE)

Un Entorno de Desarrollo Integrado, llamado también IDE (sigla en inglés de Integrated Development Environment), es un programa informático compuesto por un conjunto de herramientas de programación, generalmente un editor de código, un compilador, un depurador, un constructor de interfaz gráfica y, eventualmente, un sistema de control de versiones.

#### 2.7.1 QT Creator

Qt Creator constituye una multiplataforma que se ajusta a las necesidades de los desarrolladores. Se centra en proporcionar características que ayudan a los nuevos usuarios de Qt a aprender y comenzar a desarrollar rápidamente, también aumenta la productividad de los desarrolladores con experiencia en Qt (**Chavez, 2010**).

Es un excelente IDE multiplataforma para el desarrollo de aplicaciones en C++ de manera sencilla y rápida. Entre sus principales características se encuentra: (**Ronny, 2010**).

- Posee un avanzado editor de código C++.
- Soporta los lenguajes: C#/.NET Languages (Mono), Python: PyQt y PySide, Ada, Pascal, Perl, PHP y Ruby.
- Posee una GUI integrada con diseñador de formularios.
- Herramienta para proyectos y administración.
- Ayuda sensible integrada al contexto.
- Depurador visual. Resaltado y auto-completado de código.
- Soporte para refactorización de código.

Qt Creator permite el desarrollo de aplicaciones en entornos MS Windows, Mac OSX y Linux. Se caracteriza por ser de código abierto, gratuito y muy eficiente.

### 2.8 GeoQ: Como plataforma para desarrollo SIG

GeoQ es una plataforma del centro de desarrollo de software GEsED de la Universidad de las Ciencias Informáticas; está basado en Quantum GIS (QGIS), es de código abierto y trabaja bajo la licencia General Publica (GNU/GPL). Está desarrollado en el lenguaje C++ y se ejecuta en múltiples plataformas. Proporciona un número cada vez mayor de las capacidades proporcionadas por las funciones básicas y plugins<sup>3</sup>. Permite visualizar, gestionar, editar y analizar datos geoespaciales (**Escobar, 2011**).

Las características de la arquitectura de GeoQ, permite la creación de personalizaciones para dar solución a los disímiles problemas de negocios que tengan los clientes; donde una personalización no es más que la adaptación de las funcionalidades de GeoQ, modificar el entorno en cuanto a colores, iconografía, distribución de funcionalidades y habilitar o agregar características propias del negocio.

### 2.9 Conclusiones parciales

Después de una comparación y evaluación de las herramientas y tecnologías; se decide utilizar para el desarrollo del sistema propuesto: como metodología de desarrollo RUP la cual genera la documentación necesaria para garantizar un adecuado entendimiento del negocio; como herramienta CASE Visual Paradigm 8.0, utilizando UML en su versión 2.1 como lenguaje de modelado, y como gestor de base de datos PostgreSQL 9.1 que será el encargado de gestionar los datos de la aplicación con su módulo espacial PostGIS 1.5. Además para el desarrollo de la aplicación se hará uso del framework de desarrollo GeoQ en su versión 1.0, utilizando como lenguaje de programación C++ y como IDE de desarrollo QtCreator en su versión 2.5.0. Una vez definidas las herramientas y tecnologías se puede decir que están creadas las bases para el posterior desarrollo de la solución propuesta.

---

<sup>3</sup> Del inglés plug-in, hace referencia a pequeñas aplicaciones funcionales creadas para ser añadidas a programas con el propósito de ampliar su capacidad en alguna actividad.

## CAPÍTULO 3 PRESENTACIÓN DE LA SOLUCIÓN PROPUESTA

En este capítulo se presenta una descripción de la solución propuesta y de los procesos del negocio, así como la especificación de los requisitos funcionales y no funcionales que debe presentar el sistema a construir. Además, se determinan los casos de uso del sistema y los actores que interactuarán con este, elaborándose una descripción de cada uno.

### 3.1 Modelo del negocio

El modelo de negocio describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización. Tiene como objetivo comprender la estructura y la dinámica de la organización en la cual se va a implantar un sistema; comprende los problemas actuales de la organización e identifica las mejoras potenciales.

El modelado del negocio está soportado por dos tipos de modelos de UML: modelo de casos de uso y modelo de objetos. El modelo de casos de uso del negocio describe los procesos de negocio de una empresa en términos de casos de uso del negocio y actores del negocio que se corresponden con los procesos del negocio y los clientes (Jacobson, y otros, 2000).

#### 3.1.1 Actores y trabajadores del negocio

Un actor del negocio representa un individuo, grupo, organización, máquina o sistema que interactúa con el negocio y se beneficia de su existencia.

Luego de analizar los procesos del negocio en el cual está enmarcada la investigación actual, se identificó un actor del negocio, dos procesos de negocio y un trabajador del negocio; también se identificaron las posibles tareas a ser automatizadas.

Actor	Descripción
-------	-------------

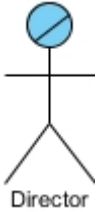
 <p>Director</p>	<p>Es el encargado de coordinar los principales procesos del negocio. Responsable de elaborar un informe periódico que refleja el estado del parque tecnológico de su base de ómnibus y de mantener un control de la flota.</p>
---	---

Tabla 1 Actores del negocio

Los trabajadores del negocio representan personas o sistemas que están involucrados en uno o más procesos del negocio, que participan en ellos, pero no obtienen ningún resultado de valor.


Trabajadores	Descripción
 <p>Operativo</p>	<p>Participa en el proceso de elaboración del estado técnico de la flota y de su representación en el mapa.</p>

Tabla 2 Trabajadores del negocio

La siguiente figura representa el Modelo de Objetos del Negocio.

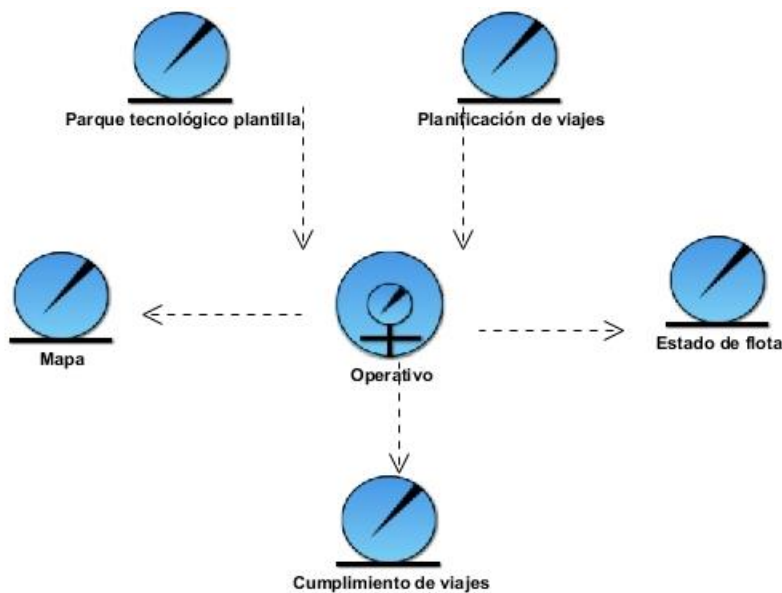


Figura 3 Modelo de objetos del negocio



### 3.1.2 Diagrama de casos de uso del negocio

El diagrama de casos de uso del negocio describe las relaciones que existen entre un caso de uso del negocio y un actor del negocio.

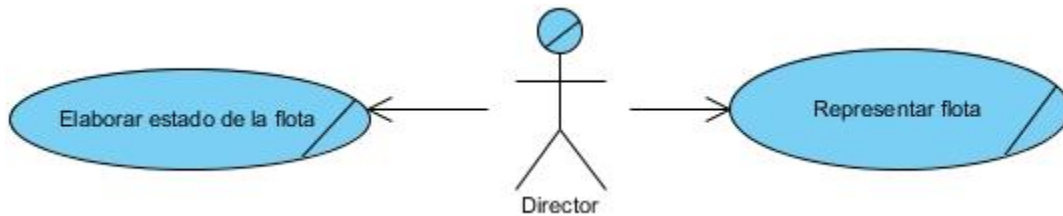


Figura 4 Diagrama de casos de uso del negocio

### 3.1.3 Diagrama de actividades

Los diagrama de actividades del negocio ayudan a describir en detalle lo que pasa dentro del negocio y permite examinar los roles específicos que juegan las personas (trabajadores del negocio) y las actividades que realizan, también identifica qué funciones deberá asumir el producto del software y quiénes serán los actores del futuro sistema.

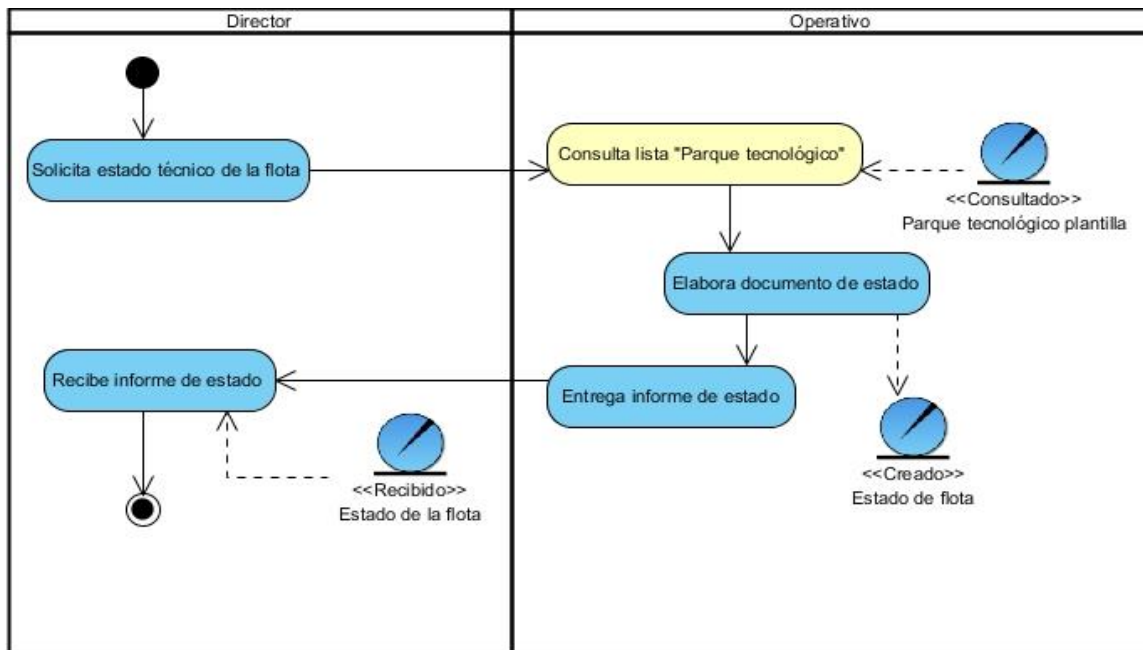


Figura 5 Diagrama de actividades del caso de uso del negocio "Elaborar estado de la flota"

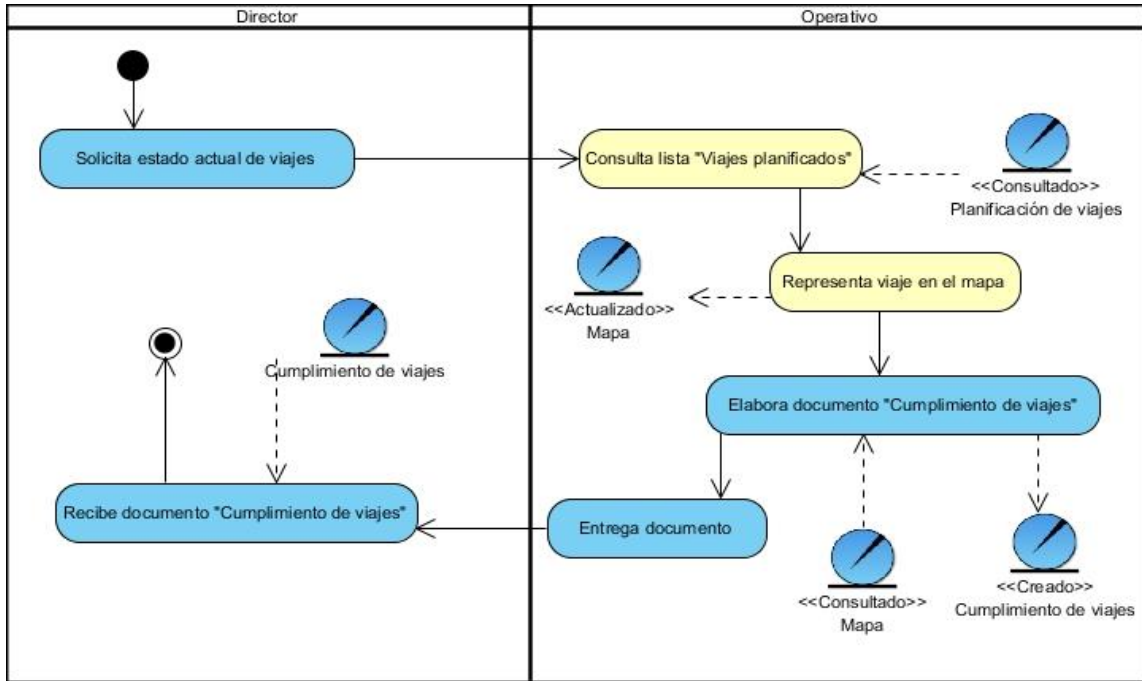


Figura 6 Diagrama de actividades del caso de uso del negocio "Representar flota"

### 3.1.4 Descripción de los casos de uso del negocio

La siguiente tabla recoge una descripción textual para el caso de uso del negocio "Elaborar estado de la flota".

<b>Caso de uso del Negocio</b>	Elaborar estado de la flota
<b>Actor</b>	Director
<b>Trabajador</b>	Operativo
<b>Resumen</b>	El caso de uso comienza cuando el Director solicita la información relacionada con el estado técnico de la flota que incluye: indicadores y disponibilidad de la flota. El caso de uso finaliza cuando el Director recibe el informe.
<b>Casos de usos asociados</b>	Ninguno
<b>Flujo Normal de Eventos</b>	
<b>Acción del Actor</b>	<b>Respuesta del Proceso del Negocio</b>
1. Solicita la confección de un informe técnico que refleja el estado técnico de	2. El Operativo consulta la plantilla técnica de la base de ómnibus

la flota.	donde se recoge la lista de ómnibus real con que cuenta la empresa.
	3. El Operativo elabora un documento donde describe la disponibilidad de cada ómnibus. Esta disponibilidad está dada en: “ómnibus disponible” y “ómnibus no disponible”. En caso de no estar disponible se especifica el motivo.
	4. Calcula el coeficiente de disponibilidad técnica dado por la cantidad de ómnibus disponibles entre el total de ómnibus. Este coeficiente se incluye en el documento que se elabora y varía en el intervalo de 0 a 1.
	5. Se entrega el documento de estado técnico de la flota.
<b>Poscondición</b>	Se confecciona y se entrega el informe técnico de la flota.

Tabla 3 Descripción textual del caso de uso del negocio “Elaborar estado de la flota”

### 3.2 Especificación de los requisitos

#### 3.2.1 Requisitos funcionales

Los requisitos funcionales constituyen capacidades o condiciones que el sistema deberá cumplir, o sea, las utilidades expuestas por la solución para los usuarios (**Jacobson, 2000**). A continuación se listan las funcionalidades que brindará el sistema: (ver el documento *GeoQ-EON Especificación de requisitos de software.docx* para una descripción detallada, variables de entrada y variables de salidas de cada requisito)

- RF1: Autenticar usuario.
- RF2: Registrar un viaje.

- RF3: Mostrar la lista de viajes.
- RF4: Eliminar una viaje previamente seleccionado.
- RF5: Registrar una nueva planificación.
- RF6: Editar los datos asociados a una planificación.
- RF7: Eliminar una planificación determinada.
- RF8: Registrar un nuevo ómnibus.
- RF9: Editar los datos asociados a un ómnibus determinado.
- RF10: Eliminar un ómnibus previamente seleccionado.
- RF11: Registrar un nuevo chofer.
- RF12: Editar los datos asociados a un chofer.
- RF13: Eliminar un chofer previamente seleccionado.
- RF 14: Identificar un elemento.
- RF15: Calcular coeficiente de disponibilidad de la flota.
- RF16: Detener y reanudar un viaje que se encuentre en ejecución.
- RF17: Mostrar la lista de choferes.
- RF 18: Calcular distancia entre paradas.
- RF19: Calcular tiempo entre paradas.
- RF20: Mostrar ómnibus disponibles y en recorrido.
- RF21: Mostrar choferes que se encuentran de viaje.
- RF22: Mostrar los puntos de parada por planificación de viaje.
- RF23: Listar los puntos donde cambia la velocidad en el recorrido.
- RF24: Registrar una nueva velocidad.
- RF25: Editar los datos asociados a una velocidad.
- RF26: Eliminar una velocidad.
- RF27: Registrar un nueva parada.
- RF28: Editar los datos asociados a una parada.
- RF29: Eliminar una parada previamente seleccionada.
- RF30: Pausar la simulación.
- RF31: Reanudar la simulación.
- RF32: Cambiar la frecuencia con que varía el tiempo.
- RF33: Mostrar y ocultar barra de tiempo.
- RF34: Listar toda la planificación de viaje.
- RF35: Visualizar el estado de los viajes en el transcurso del tiempo.

### 3.2.2 Requisitos no funcionales

Los requisitos no funcionales representan propiedades o cualidades que el producto debe tener lo que hace del mismo que sea más atractivo, usable, rápido y confiable.

#### Usabilidad

- El sistema podrá ser usado por personas con conocimientos básicos en el manejo de computadoras. Se emplearán componentes que indiquen al usuario el estado de los procesos que por su complejidad requieran de un tiempo de procesamiento apreciable.
- El Sistema tendrá una correcta Arquitectura de la Información, con iconografía relevante a la función realizada y con colores acordes al problema planteado.
- Las funcionalidades del sistema estarán agrupadas por zona de interés o grupos relacionados.

#### Fiabilidad

- La herramienta de implementación a utilizar debe tener soporte para recuperación ante fallos y errores.
- La información manejada por el sistema estará protegida de acceso no autorizado y divulgación.
- Debido a la arquitectura que presenta el sistema, siendo más robusto al no tratarse de un sistema de gestión que requiera mantenimiento y optimización en el almacenamiento, se estima un tiempo promedio de 6 meses entre posibles fallas.
- El tiempo medio de reparación, en caso de un fallo es de 7 días.

#### Eficiencia

- El tiempo de respuesta estará dado por la cantidad de información a procesar, entre mayor cantidad de información, mayor será el tiempo de procesamiento. Para un mapa con extensión municipal, el tiempo de respuesta promedio es menor que 3 segundos. Para una cartografía de extensión de datos igual a

Cuba, por cada capa podría demorarse hasta 4 y 7 segundos dependiendo necesariamente de la cantidad de objetos de la misma.

- Al igual que el tiempo de respuesta, la velocidad de procesamiento de la información, la actualización y la recuperación dependerán de la cantidad de información que tenga que procesar la aplicación.

### **Soporte**

- La aplicación recibirá mantenimiento en el período de tiempo determinado por el equipo de desarrollo y los clientes.

### **Restricciones de diseño e implementación**

- Diseño sencillo, con pocas entradas, donde no sea necesario un alto nivel de entrenamiento para utilizar el sistema.
- El producto de software final debe diseñarse sobre una arquitectura en capas.
- Se deben emplear los estándares establecidos (diseño de interfaces, base de datos y codificación). Se debe lograr un producto altamente configurable y extensible, teniendo en cuenta que se desarrollará sobre la plataforma GeoQ y que constituye una plataforma de desarrollo para ser personalizada como aplicaciones a la medida, pudiéndose incorporar a estas nuevas funcionalidades.

### **Componentes comprados**

- No se han comprado componentes para el desarrollo del software.

### **Interfaz**

#### **Interfaz de Usuario**

El sistema debe:

- Tener una apariencia profesional y un diseño gráfico sencillo.
- Posibilitarle al usuario la configuración del entorno de trabajo.
- Ser intuitivo.

### Interfaces de Hardware

Para las PCs clientes:

- Se requiere al menos 512 MB de memoria RAM.
- Se requiere al menos 2 GB libres de disco duro.
- Procesador 512 MHz como mínimo.

### Interfaces de Software

La construcción de la aplicación funcionará bajo los conceptos de arquitectura en capas centrada en los datos. Las PCs clientes:

- El sistema GeoQ-EON es un sistema multiplataforma que puede ejecutarse en los siguientes sistemas operativos:
  - ✓ GNU/Linux.
  - ✓ Mac OS X.
  - ✓ Windows XP o superior
  - ✓ Windows Server 2000 o superior.
- PostgreSQL (Versión 9.1) como Sistema Gestor de Base de Datos.
- PostGis (Versión 1.5) como extensión de PostgreSQL para el soporte de datos espaciales.

### Interfaz de Comunicación

- El producto GeoQ-EON garantizará mediante su interfaz la configuración del entorno de trabajo mediante funcionalidades propias como ocultar y mostrar paneles, así como elementos para cambiar las vistas, las escalas y las capas que serán visibles en la interacción.

### Requisitos de Licencia

- GeoQ-EON se publica bajo la licencia pública general de GNU (GNU GPL). Desarrollar este producto bajo esta licencia quiere decir que se puede inspeccionar y modificar el código fuente.

### **Requisitos Legales, Derecho del Autor y otros**

- El sistema debe ajustarse y regirse por la ley, decretos leyes, decretos, resoluciones y manuales (órdenes) establecidos, que norman los procesos que serán automatizados.
- La mayoría de las herramientas de desarrollo son libres y del resto, las licencias están avaladas.

### **Estándares Aplicables**

- Estándar OGC (Open Geospatial Consortium): Se utiliza para desarrollar e implementar estándares de los contenidos y servicios geoespaciales.

### **Confidencialidad**

- La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

### **Disponibilidad**

- A los usuarios autorizados se les garantizará el acceso a la información, los dispositivos o mecanismos utilizados para lograr la seguridad, no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.
- La información y las funcionalidades del sistema estarán disponibles y el usuario podrá acceder a ellas las 24 horas de los 7 días de la semana.

### **3.3 Descripción del sistema propuesto**

Los actores se definen como los roles que puede tener un usuario; pueden ser humanos, otros sistemas, máquinas, hardware, etc. que interactúan con un sistema para de esta forma intercambiar datos; generalmente estos actores suelen corresponderse con los trabajadores (o actores del negocio) en un negocio.



3.3.1 Descripción de los actores del sistema


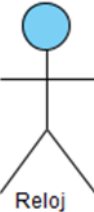
Actor	Descripción
 <p>Usuario</p>	<p>Es el encargado de realizar todas las funcionalidades del sistema que requieran la intervención humana para la configuración del mismo, la introducción de datos y la confirmación de acciones.</p>
 <p>Reloj</p>	<p>Es el encargado generar intervalos de tiempos regulares y actualizar la representación visual de los viajes realizados por los ómnibus.</p>

Tabla 4 Descripción de los actores del sistema

Un caso de uso constituye una técnica utilizada para describir el comportamiento del sistema, a través de un documento narrativo que define la secuencia de acciones que obtienen resultados de valor para un actor que utiliza un sistema para completar un proceso, sin importar los detalles de la implementación.

Cada forma en que los actores usan el sistema se representa con un caso de uso. Los casos de uso son “fragmentos” de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. De manera más precisa, un caso de uso especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores, incluyendo alternativas dentro de la secuencia.

A continuación se muestra el diagrama de casos de uso del sistema donde están representados los distintos actores que interactúan con el sistema mediante los casos de uso del sistema y luego la descripción textual para el caso de uso “Gestionar planificación”.

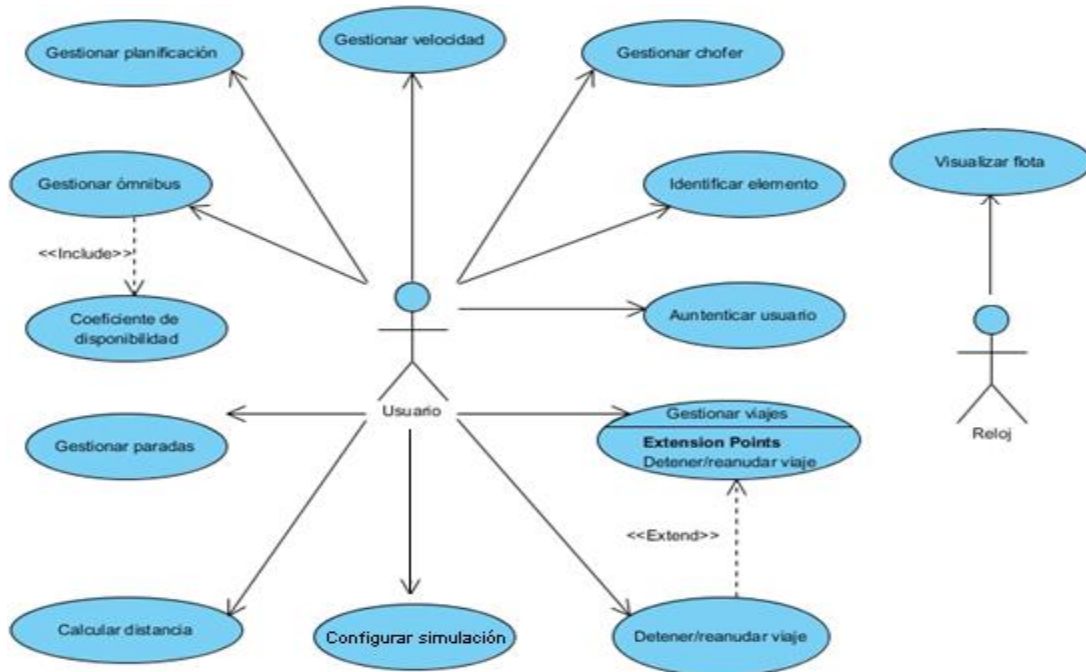


Figura 7 Diagrama de casos de uso del sistema

### 3.3.2 Descripción de casos de uso del sistema

<b>Caso de Uso</b>	<b>Gestionar planificación</b>
<b>Actor</b>	Usuario
<b>Propósito</b>	Capacidad del usuario para gestionar una planificación.
<b>Resumen:</b>	El caso de uso inicia cuando el usuario selecciona la opción Planificación. El sistema muestra una interfaz con la lista de las planificaciones y permite registrar, editar y eliminar una planificación. El usuario selecciona los datos de la planificación y selecciona la opción deseada. El sistema guarda los cambios y concluye el caso de uso.
<b>Precondiciones</b>	El usuario debe estar autenticado en el sistema.
<b>Referencias</b>	RF 5, RF 6, RF 7, RF34.
<b>Prioridad</b>	Secundario.
<b>Flujo Normal de Eventos</b>	
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>
1. El caso de uso se inicia cuando el usuario selecciona de la barra de herramienta la	2. El sistema muestra una interfaz con la lista de planificaciones y brinda las

opción Planificación.

opciones de:

- Registrar nueva planificación. Ver sección “Registrar nueva planificación.”
- Editar una planificación. Ver sección “Editar una planificación.”
- Eliminar una planificación seleccionada. Ver sección “Eliminar una planificación seleccionada.” **Ver interfaz 1**

**Sección “Registrar nueva planificación”**

**Acción del Actor**

**Respuesta del Sistema**

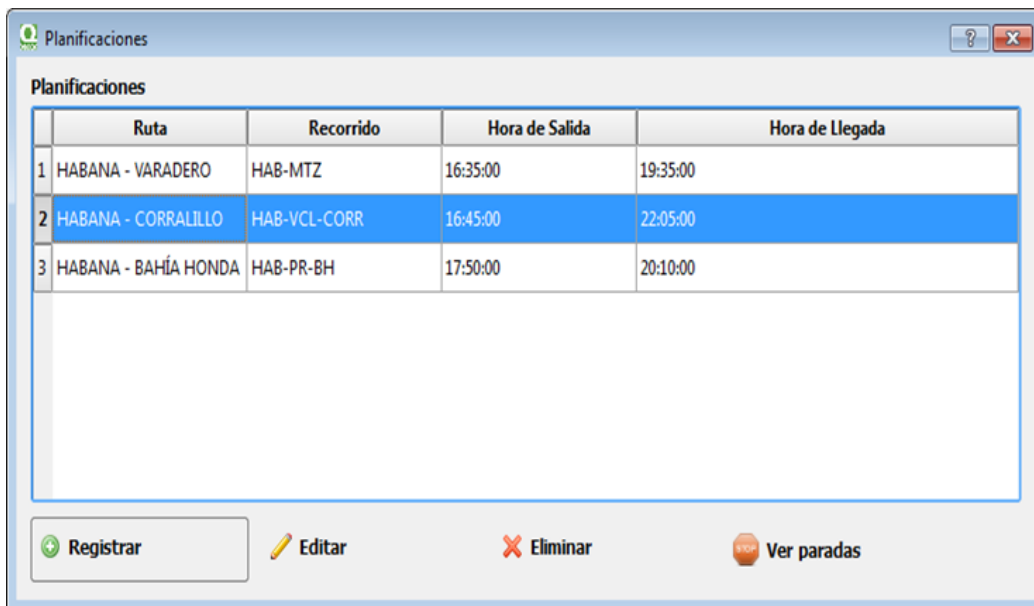
3. El usuario hace clic izquierdo y selecciona la opción Registrar.

4. El sistema muestra una interfaz con los datos a registrar.

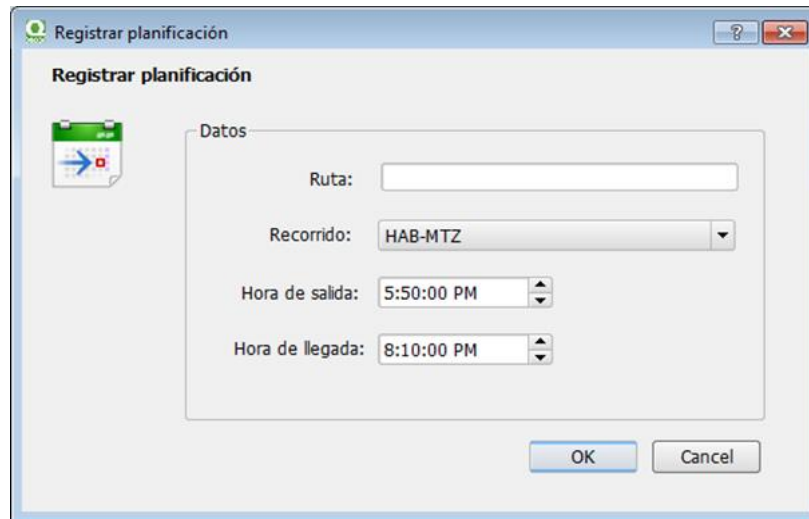
5. El usuario introduce los datos y hace clic en el botón OK.

6. El sistema valida los datos y permite registrar una nueva planificación, se guardan los cambios y termina el caso de uso. **Ver interfaz 2**

**Prototipo de Interfaz 1**



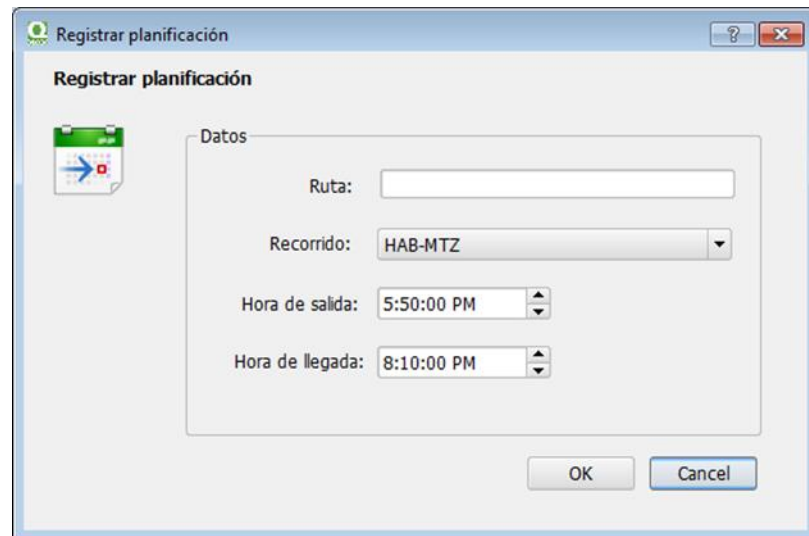
**Prototipo de Interfaz 2**



**Flujos Alternos**

Acción del Actor	Respuesta del Sistema
5.1 El usuario introduce los datos y hace clic izquierdo en el botón "Cancel".	4.1 El sistema no realiza ningún cambio y termina el caso de uso. <b>Ver interfaz 3</b>

**Prototipo de Interfaz 3**

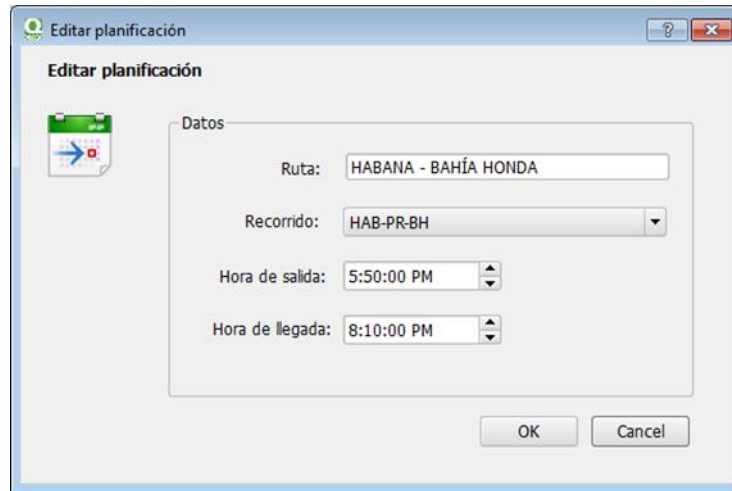


**Sección "Editar una planificación"**

Acción del Actor	Respuesta del Sistema
7. El usuario hace clic izquierdo y selecciona la opción Editar.	8. El sistema muestra una interfaz con los datos que desea editar.

- |   |  |
|---|--|
| <p>9. El usuario realiza cambios en los datos que desee y hace clic en el botón OK.</p> | <p>10. El sistema permite editar los datos asociados a una planificación, se guardan los cambios y termina el caso de uso. <b>Ver interfaz 4</b></p> |
|---|--|

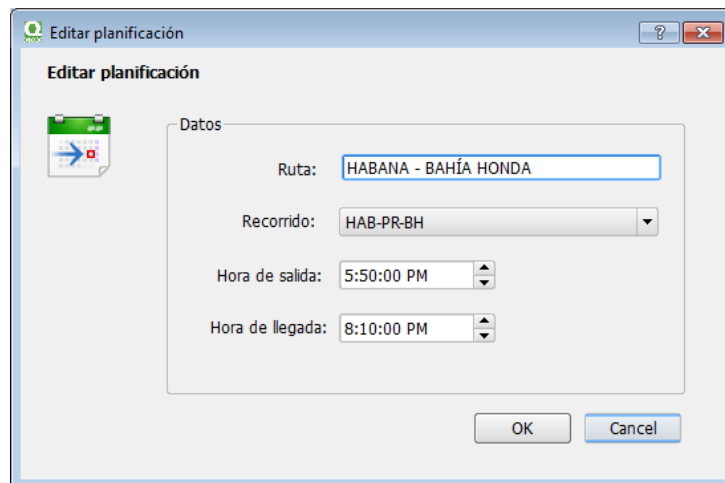
**Prototipo de Interfaz 4**



**Flujos Alternos**

Acción del Actor	Respuesta del Sistema
<p>9.1 El usuario realiza cambios en los datos deseados y hace clic izquierdo en el botón "Cancel".</p>	<p>10.1 El sistema no realiza ningún cambio y termina el caso de uso. <b>Ver interfaz 5</b></p>

**Prototipo de Interfaz 5**



**Sección "Eliminar una planificación seleccionada"**

Acción del Actor	Respuesta del Sistema
<p>11. El usuario hace clic izquierdo y selecciona una</p>	<p>13. El sistema muestra una interfaz para</p>

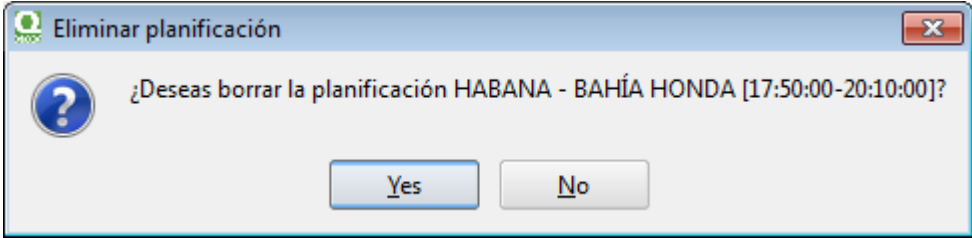
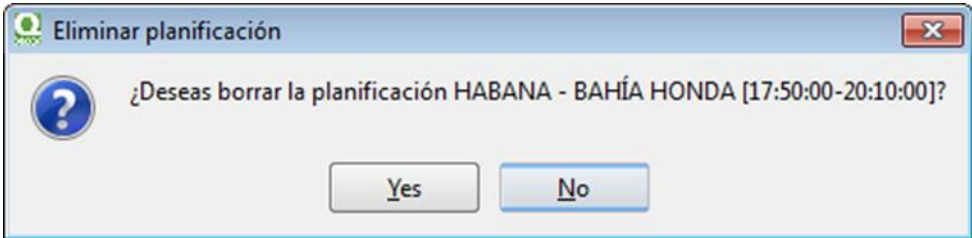
planificación determinada.	eliminar la planificación.
12. El usuario hace clic izquierdo y selecciona la opción Eliminar.	15. El sistema permite eliminar una planificación seleccionada, se guardan los cambios y termina el caso de uso. <b>Ver interfaz 6</b>
14. El usuario hace clic izquierdo en el botón Yes.	
<b>Prototipo de Interfaz 6</b>	
	
<b>Flujos Alternos</b>	
<b>Acción del Actor</b>	<b>Respuesta del Sistema</b>
7.1 El usuario hace clic izquierdo en el botón "No".	8.1 El sistema no realiza ningún cambio y termina el caso de uso. <b>Ver interfaz 7</b>
<b>Prototipo de Interfaz 7</b>	
	
<b>Poscondición</b>	Se realizó correctamente la gestión de la planificación.

Tabla 5 Descripción textual del caso de uso del sistema "Gestionar planificación"

### 3.3.3 Conclusiones del capítulo

En el presente capítulo se describió detalladamente el negocio compuesto principalmente por dos procesos de negocio. Estos procesos se modelaron a través del Diagrama de Casos de Uso del Negocio y luego se precisaron las características del sistema en términos de requisitos funcionales y no funcionales. Se identificaron 13 casos de usos que recogen las funcionalidades identificadas del sistema; las características trazadas facilitan la creación de un software de fácil utilización y elevada operatividad.

## CAPÍTULO 4 CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA

El presente capítulo describe la solución en términos de diagrama de clases del diseño dividido por secciones de casos de uso para facilitar una mejor comprensión; se presenta el diseño de la base de datos a través del diagrama Entidad-Relación. Se define el modelo de despliegue originado por la selección de los artefactos más importantes para el sistema donde se precisan los componentes que conforman la estructura física de la aplicación. Finalmente se realizan las pruebas pertinentes al sistema para demostrar el cumplimiento de los estándares de calidad necesarios para satisfacer los requisitos propuestos.

### 4.1 Diseño

En el diseño se modela y se da forma al sistema para que soporte los requisitos funcionales y no funcionales definidos. Es en la etapa (o flujo de trabajo) de diseño donde se crea una entrada apropiada y un punto de partida para actividades de implementación capturando los requisitos o subsistemas individuales, interfaces y clases. **(Rumbaugh, 2004)**.

#### 4.1.1 Patrones

Un patrón involucra una descripción general de una solución concurrente a un problema recurrente con soluciones recurrentes con diversos objetivos y restricciones. Los patrones cubren diferentes rangos de escala de abstracción. Algunos ayudan a estructurar un sistema en subsistemas, otros a refinar otros subsistemas y sus relaciones, otros solucionan una implementación en particular. **(Rosánigo, 2000)**

Los patrones de diseño proponen una forma de reutilizar la experiencia de los desarrolladores, para ello clasifica y describe formas de solucionar problemas que ocurren de forma frecuente en el desarrollo; por tanto está basado en la recopilación del conocimiento de los expertos en desarrollo de software.

Los patrones dados las características que posee se dividen en dos grandes grupos: los patrones arquitectónicos y los patrones de diseño.

### Patrones arquitectónicos

Los patrones arquitectónicos describen los principios fundamentales de la arquitectura de un sistema de software. Estos a su vez identifican los subsistemas, define sus responsabilidades y establece las reglas y guías para organizar la relación entre ellos **(Rosánigo, 2000)**.

El patrón arquitectónico utilizado en el proceso de desarrollo del sistema es el Modelo–Vista–Controlador. Este patrón separa o define 3 capas para organizar las funcionalidades del sistema y su estructura lógica: el modelo, la vista y el controlador.

La Vista se encarga de todo lo referente a la presentación de la aplicación, como son las imágenes a mostrar, cajas de texto, distribución de los elementos en la pantalla, colores, formatos, entre otros. El Modelo se encarga de la lógica o reglas del negocio y del acceso a datos. El Controlador actúa como el interlocutor entre la Vista y el Modelo, llevando las solicitudes del usuario al Modelo y notificando a la Vista según sea necesario **(Paredes, 2009)**.

Los beneficios de aplicar este patrón radican principalmente en que la capa Vista puede cambiarse fácilmente; aporta además reusabilidad de componentes y los datos del sistema están encapsulados en entidades lógicas que pueden manipularse y representarse desde vistas distintas.

### Patrones de diseño

Los patrones de diseño proveen un esquema para refinar los subsistemas y las relaciones entre los componentes de un sistema. Describen una estructura recurrente de comunicación entre componentes para resolver un problema general de diseño dentro de un contexto particular. Se encuentran en un nivel intermedio de granularidad entre los patrones arquitectónicos y las expresiones idiomáticas. Tienden a operar independientemente de un paradigma particular de programación o lenguaje de programación **(Rosánigo, 2000)**.



En la actualidad existen tres grandes grupos de patrones de diseño: de creación, de estructura y de comportamiento.

### **Patrones de diseño: de creación**

Los patrones de creación abstraen la forma en que se crean los objetos, de forma que permite tratar las clases a crear de forma genérica apartando la decisión de qué clases crear o como crearlas.

- Patrón Singleton (Única instancia): restringe la cantidad de instancias que se crean a partir de una clase. Este patrón brinda los mecanismos necesarios para garantizar que se cree una y solo una instancia de la clase; la cual es compartida de forma única para todas las llamadas a la clase en cuestión; de este modo se evita el uso de variables globales. El uso de este patrón permitió mantener una conexión común y global para todo el sistema, de este modo los parámetros de conexión están centralizados en una única clase la cual puede modificarse con facilidad. La clase que implementa este patrón forma parte del subsistema GeoQ, llama GeoQConexión.

### **Patrones de diseño: de estructura**

Los patrones estructurales describen cómo formar estructuras complejas a partir de elementos más simples. Existen dos tipos de patrones de este tipo, de clase y de objeto.

- Patrón Facade (Fachada): este patrón simplifica el problema del acceso a un conjunto de clases o interfaces reduciendo las dependencias con estas. Cuando se tienen subsistemas complejos o la ejecución de una operación conlleva el uso y dependencia de múltiples clases, se introduce el término fachada el cual brinda una interfaz de comunicación con el subsistema en cuestión reduciendo así el número de dependencias. De esto modo, la fachada es la encargada de todas las llamadas internas al subsistema y la orquestación del flujo de llamadas; mientras que las clases externas al subsistema se comunican a través de una interfaz sólida y unificada. Este patrón se evidencia en la clase EonApp el cual

es el punto de entrada para toda la lógica del sistema que se construye. De este modo es sencillo conectar el subsistema desarrollado dentro de GeoQ.

### **Patrones de diseño: de comportamiento**

Los patrones de comportamiento hablan de cómo interaccionan entre sí los objetos para conseguir ciertos resultados.

- Patrón Observer (Observador): este patrón brinda un mecanismo más para prevenir el acoplamiento entre objetos. Su implementación garantiza que los objetos al cambiar algunas de sus propiedades o su estado puedan notificar dicho cambio a los objetos que están interesados. Este patrón se utilizó principalmente para manejar desde las clases controladores el flujo de datos y las peticiones del usuario realizadas desde las clases interfaces. De este modo todas las peticiones e interacciones son manipuladas desde los controladores, asignando así la responsabilidad de controlar el flujo del sistema y decidir las posibles respuestas.

### **Patrones de diseño: Mapeo de objetos y relaciones**

Los patrones englobados en esta categoría proporcionan las herramientas necesarias para mantener una capa de dominio y de acceso a datos desacoplada de las restantes capas, abstrayendo de este modo la lógica subyacente en cómo acceder a los datos del sistema.

- Patrón Foreign Key Mapping (Mapeo de llave foránea): en sistemas complejos o de mediana envergadura los objetos persistentes mantienen relaciones con otros objetos que igualmente son persistente, la forma más común de implementar estas relaciones se hace conservando identificadores que representan llaves foráneas. Este patrón soluciona el problema de referirse a los objetos por su propia referencia y no por su llave foránea, de esta forma se tienen relaciones de asociación y el trabajo con los objetos es transparente al medio de almacenamiento.
- Patrón Table Data Gateway (Tabla de acceso a datos): en sistemas medianos y complejos, es bueno mantener el código de acceso a datos separado del resto

del sistema. Este patrón recomienda mantener todas las consultas SQL para acceder a los datos en clases separadas, el resultado de la consulta se transforma en colecciones de objetos para el caso de las consultas SELECT y para las consultas tipo INSERT, DELETE y UPDATE se trabaja directamente con una instancia del objeto; de este modo no se tiene que recurrir a otras estructuras para almacenar los datos, sino que se reutilizan los mismos objetos del dominio con el cual trabaja el sistema. En el sistema desarrollado este patrón se evidencia en las clases tipo repositorios donde cada clase maneja y manipula su correspondiente clase del dominio, ejemplo: Chofer y ChoferRepository.

#### 4.1.2 Diagrama de clases del diseño

Los diagramas de clases son diagramas de estructura estática que muestran las clases del sistema y las relaciones entre ellas; muestran lo que el sistema puede hacer y cómo puede ser construido. Cuando se crea un diagrama de clases, se está modelando una parte de los elementos y relaciones que configuran la vista de diseño del sistema.

A continuación se muestra una vista global de clases y paquetes más significativos del diseño y se muestra también el diagrama de clases del diseño correspondiente al caso de uso “Gestionar planificación”. Los restantes diagramas de clases del diseño pueden ser consultados en el documento *GeoQ-EON-Modelo de Diseño.docx*.

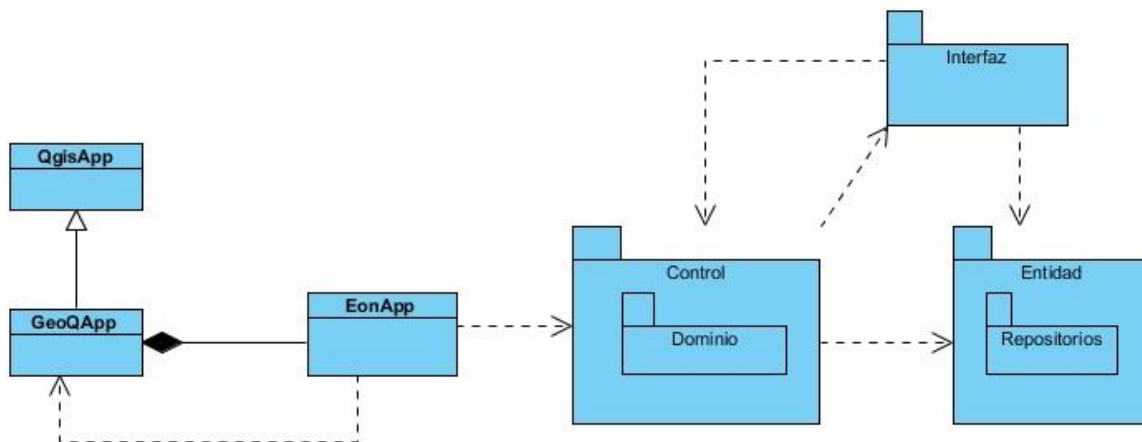


Figura 8 Vista global de clases y paquetes más significativos del diseño

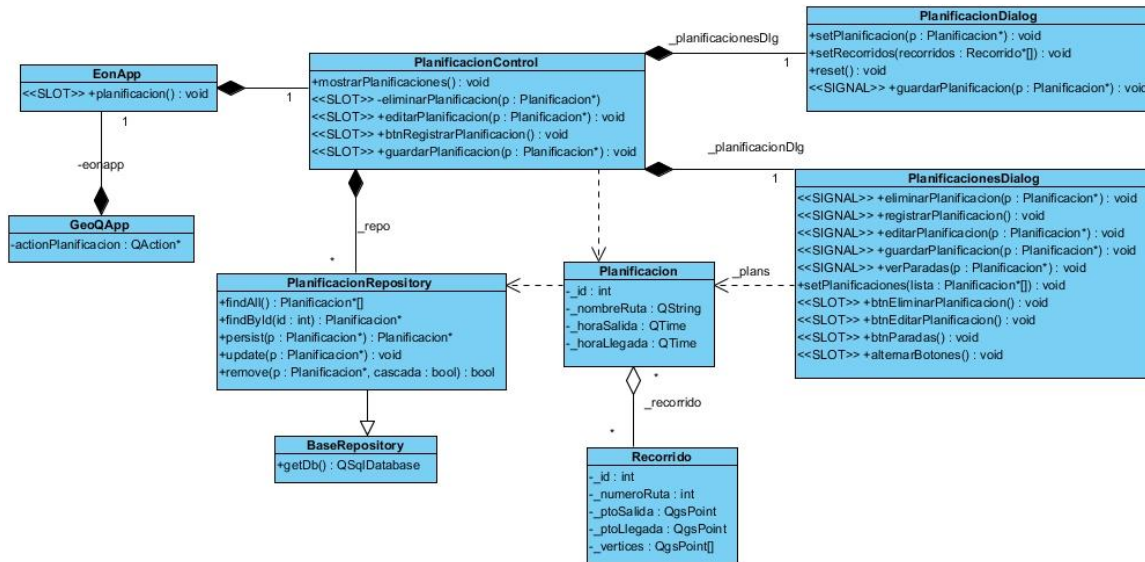


Figura 9 Diagrama de clases del diseño del caso de uso "Gestionar planificación"

## 4.2 Diseño de la base de datos

El primer paso para la construcción de una base de datos es definir su estructura, de forma tal que permita un adecuado mecanismo para almacenar los datos y posteriormente recuperarlos. Para lograr un buen diseño de la base de datos es necesario seguir un conjunto de pasos que comienzan con definir las clases persistentes, refinarlas y clasificarlas junto a sus atributos, lo que permitirá realizar el diagrama de clases persistentes y posteriormente la conversión de las clases al medio de almacenamiento.

### 4.2.1 Diagrama de clases persistentes

Todas las clases identificadas durante el desarrollo del diseño no tienen que ser necesariamente persistentes, la persistencia de una clase está dada por la capacidad de la misma para mantener su valor en el espacio y en el tiempo; contrario a las clases temporales que son manejadas y almacenadas por el sistema en tiempo de ejecución, dejando de existir al finalizar el programa. En la siguiente figura se muestra el diagrama de clases persistentes correspondiente al sistema planteado.

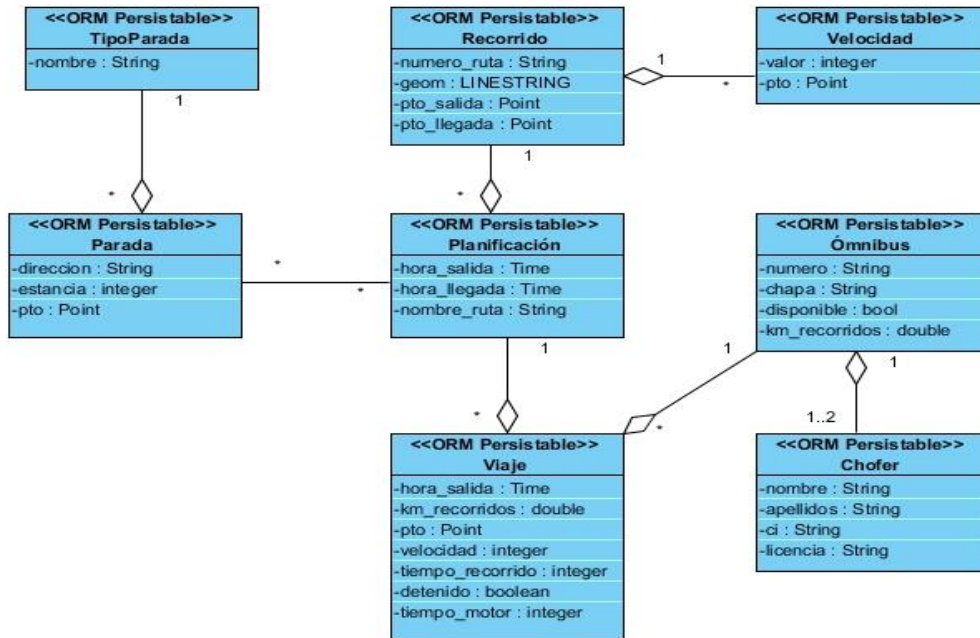


Figura 10 Diagrama de clases persistentes

#### 4.2.2 Modelo Entidad-Relación

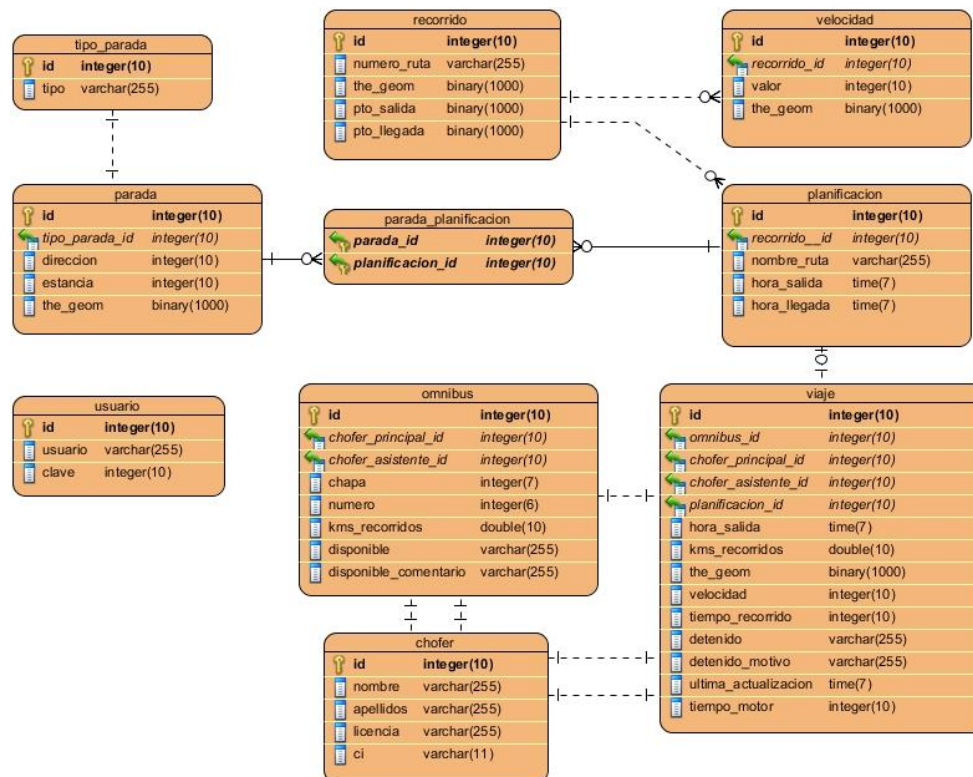


Figura 11 Modelo Entidad - Relación (modelo físico)

## 4.3 Implementación

En la fase de construcción se implementa el sistema, es decir, se crea el código correspondiente al resultado de la fase de diseño, siguiendo los patrones y la arquitectura escogida.

La implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes. Define una jerarquía de subsistemas de implementación que contiene componentes e interfaces **(Rumbaugh, 2004)**.

### 4.3.1 Modelo de Despliegue

El modelo de despliegue describe la distribución física del sistema en términos de cómo se distribuyen la funcionalidades entre los nodos de cómputo. Este modelo incluye **(Rumbaugh, 2004)**:

- Los nodos, sus características, y sus conexiones.
- Una correspondencia inicial de clases activas sobre los nodos.

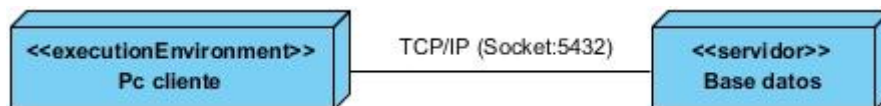


Figura 12 Modelo de despliegue

De acuerdo a la figura anterior se tienen 2 nodos de cómputos donde los componentes del sistema se distribuyen en el nodo PC Cliente cuyas características están detalladas en los requisitos no funcionales; y en el nodo Base de datos se incluye un script SQL generado a partir del modelo física (Entidad – Relación) presentado con anterioridad; las características de este nodo también fueron descritas en los requisitos no funcionales.

### 4.3.2 Diagrama de componentes

Los Diagramas de Componentes modelan la vista estática de un sistema. Se representa como un grafo de componentes software unidos por medio de relaciones de dependencia (compilación, ejecución), pudiendo mostrarse las interfaces que estos soporten. Estos tienen en consideración los requisitos relacionados con la facilidad de

desarrollo, la gestión del software, la reutilización y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo (Pressman, 2005).

A continuación se ilustra el diagrama de componentes para el caso de uso del sistema Gestionar planificación:

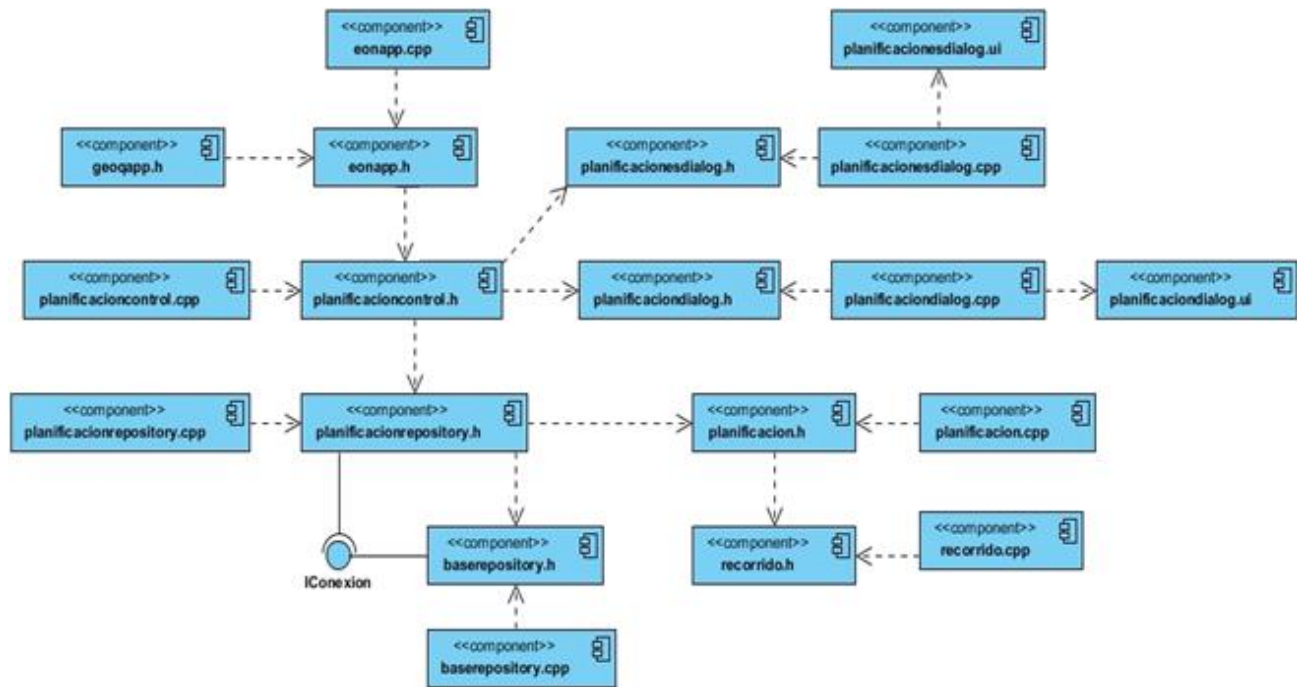


Figura 13 Diagrama de componentes del caso de uso "Gestionar planificación"

Como bien se mencionó anteriormente, el traspaso y encapsulación de los elementos del diseño en componentes depende principalmente de la arquitectura seleccionada y el lenguaje de programación. A los efectos de esta investigación una clase de diseño se traduce a componente siguiendo el siguiente criterio:

Tipo de clase	Componentes que genera	Dependencias
<b>Interfaz</b>	<b>Genera 3 componentes:</b>	.cpp → .h
Ej.: ViajeDialog	<ul style="list-style-type: none"> <li>➤ Definición (viajedialog.h)</li> <li>➤ Implementación (viajedialog.cpp)</li> <li>➤ Elementos de diseño (viajedialog.ui)</li> </ul>	.cpp → .ui
		Las dependencias se traducen en una operación (#include)

<b>Controladora</b>  Ej.: ViajeControl	<b>Genera 2 componentes:</b>  ➤ Definición (viajecontrol.h) ➤ Implementación (viajecontrol.cpp)	.cpp → .h .cpp → .ui  Las dependencias se traducen en una operación (#include)
<b>Entidades, Repositorio, Dominio y Auxiliares</b>	<b>Generan 2 componentes del siguiente modo:</b>  ➤ Definición (<nombreclase>.h) ➤ Implementación (<nombreclase>.cpp )	.cpp → .h .cpp → .ui  Las dependencias se traducen en una operación (#include)
<b>Relaciones de clases: Dependencia, Asociación, Composición, Agregación y Herencia</b>		
Todas las relaciones entre las clases del diseño se traducen a relaciones de dependencia entre sus correspondientes componentes. Estas dependencias se cumplen con el uso de operaciones tipo #include.		

**Tabla 6 Trazabilidad y encapsulamiento de los elementos del diseño en componentes**

#### 4.4 Prueba

Las pruebas de software son un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones, del diseño y la codificación (Pressman, 2005) .

- **Pruebas de Verificación:** Se comprueba el cumplimiento de las especificaciones del diseño.
- **Pruebas de Validación:** Se encargan de velar por el cumplimiento de los requisitos del análisis.
- **Pruebas de Caja Blanca:** Se conoce el código y se trata de ejecutar cada uno de los elementos del mismo.
- **Pruebas de Caja Negra:** Solamente se conoce la interfaz y se trata de probar cada uno de los elementos que componen a la misma.

Cualquier tipo de prueba tiene el objetivo principal de encontrar errores o defectos en el software.



Para que las pruebas aplicadas a un software tengan éxito es necesario realizar casos de pruebas con probabilidad de descubrir los errores en el sistema, a través de técnicas que guíen el proceso de la prueba.

### 4.4.1 Pruebas de caja negra

Para probar el sistema propuesto se utiliza la prueba de Caja Negra, la cual se refiere a las pruebas que se llevan a cabo sobre la interfaz del software, por lo que los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que la salida producida es correcta, ya que no es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar. Esta prueba permite encontrar (**Pressman, 2005**):

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

Para el desarrollo de las pruebas de caja negra existen varias técnicas, entre las que se encuentran:

- **Técnica de la Partición de Equivalencia:** esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del sistema.
- **Técnica del Análisis de Valores Límites:** esta técnica prueba la habilidad del sistema para manejar datos que se encuentran en los límites aceptables.
- **Técnica de Grafos de Causa-Efecto:** es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

Dentro del método de Caja Negra la técnica de la Partición de Equivalencia es una de las más efectivas pues permite examinar los valores válidos e inválidos de las entradas existentes en el sistema, descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico (**Pressman, 2005**).

A continuación se realizarán las pruebas de caja negra utilizando la técnica de partición de equivalencia.

**Caso de uso: Gestionar planificación.**

**Descripción general del caso de uso:** El caso de uso inicia cuando el usuario selecciona la opción Planificación. El sistema muestra una interfaz con la lista de planificaciones y permite registrar, eliminar y editar los datos de la planificación. El usuario selecciona los datos de la planificación y selecciona la opción deseada. El sistema guarda los cambios y concluye el caso de uso.

**Condiciones de ejecución:** El usuario debe estar autenticado en el sistema.

**Secciones a probar en el caso de uso.**

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: "Registrar nueva planificación"	EC 1.1: Registrar nueva planificación exitosamente.	El usuario hace clic izquierdo y selecciona la opción Registrar. El sistema muestra una interfaz con los datos a registrar. El usuario introduce los datos y hace clic en el botón "OK". El sistema valida los datos y permite registrar una nueva planificación, se guardan los cambios y termina el caso de uso.	<ol style="list-style-type: none"> <li>1. Página principal.</li> <li>2. Clic barra de herramientas.</li> <li>3. Clic en la opción Planificación.</li> </ol>
	EC 1.2: Registrar nueva planificación sin éxito.	El usuario hace clic izquierdo en el botón "Cancelar". El sistema no realiza ningún cambio y termina el caso de uso.	<ol style="list-style-type: none"> <li>1. Página principal.</li> <li>2. Clic barra de herramientas.</li> <li>3. Clic en la opción Planificación.</li> </ol>
SC 2: "Editar una planificación"	EC 2.1: Editar una planificación exitosamente.	El usuario hace clic izquierdo y selecciona la opción Editar. El sistema muestra una interfaz con los	<ol style="list-style-type: none"> <li>1. Página principal.</li> <li>2. Clic barra de herramientas.</li> </ol>

		datos que desea editar. El usuario realiza cambios en los datos que desee y hace clic en el botón "OK". El sistema permite editar los datos asociados a una planificación, se guardan los cambios y termina el caso de uso.	3. Clic en la opción Planificación.
	EC 2.2: Editar una planificación sin éxito.	El usuario hace clic izquierdo en el botón "Cancelar". El sistema no realiza ningún cambio y termina el caso de uso.	1. Página principal. 2. Clic barra de herramientas. 3. Clic en la opción Planificación.
SC 3: "Eliminar una planificación seleccionada"	EC 3.1: Eliminar una planificación seleccionada exitosamente.	El usuario hace clic izquierdo y selecciona una planificación determinada, luego selecciona la opción Eliminar. El sistema muestra una interfaz para eliminar la planificación. El usuario hace clic izquierdo en el botón "OK". El sistema permite eliminar una planificación seleccionada, se guardan los cambios y termina el caso de uso.	1. Página principal. 2. Clic barra de herramientas. 3. Clic en la opción Planificación.
	EC 3.2: Eliminar una planificación sin éxito.	El usuario hace clic izquierdo en el botón "Cancelar". El sistema no realiza ningún cambio y termina el caso de uso.	1. Página principal. 2. Clic barra de herramientas. 3. Clic en la opción Planificación.

Tabla 7 Secciones a probar en el caso de uso "Gestionar planificación"

Descripción de variables

No.	Nombre del campo	Clasificación	Valor nulo	Descripción
-----	------------------	---------------	------------	-------------

1	Ruta	Campo de Texto	No	Permite escribir el nombre de la ruta de la planificación.
2	Recorrido	Lista de selección	No	Permite seleccionar el recorrido que contempla la ruta.
3	Hora de Salida	Campo de Texto	No	Permite escribir la hora de salida del ómnibus.
4	Hora de Llegada	Campo de Texto	No	Permite escribir la hora de llegada del ómnibus.

Tabla 8 Descripción de las variables para el caso de uso "Gestionar planificación"

### Matriz de datos

#### SC 1: "Registrar nueva planificación"

Id del escenario	Ruta	Recorrido	Hora Salida	Hora Llegada	Respuesta del sistema	Resultado de la prueba
EC 1.1	V	V	V	V	El sistema permite que el usuario registre una nueva planificación.	Satisfactorio
EC 1.2	N/A	N/A	N/A	N/A	El sistema cierra el formulario para los datos y retorna a la interfaz anterior.	Satisfactorio

Tabla 9 Matriz de datos. SC1: "Registrar nueva planificación". CU "Gestionar planificación"

### Resultados de las pruebas

Se le realizaron pruebas de calidad al sistema, tanto internas como externas, con el fin de garantizar la calidad del software. Para una primera iteración se detectaron 11 no conformidades en cuanto a la interfaz del software y 8 de lógicas, se corrigieron estos errores dándoles solución a los mismos. Luego para una segunda iteración se detectaron 4 no conformidades de interfaz y ya en una tercera no se detectaron no conformidades en el sistema.

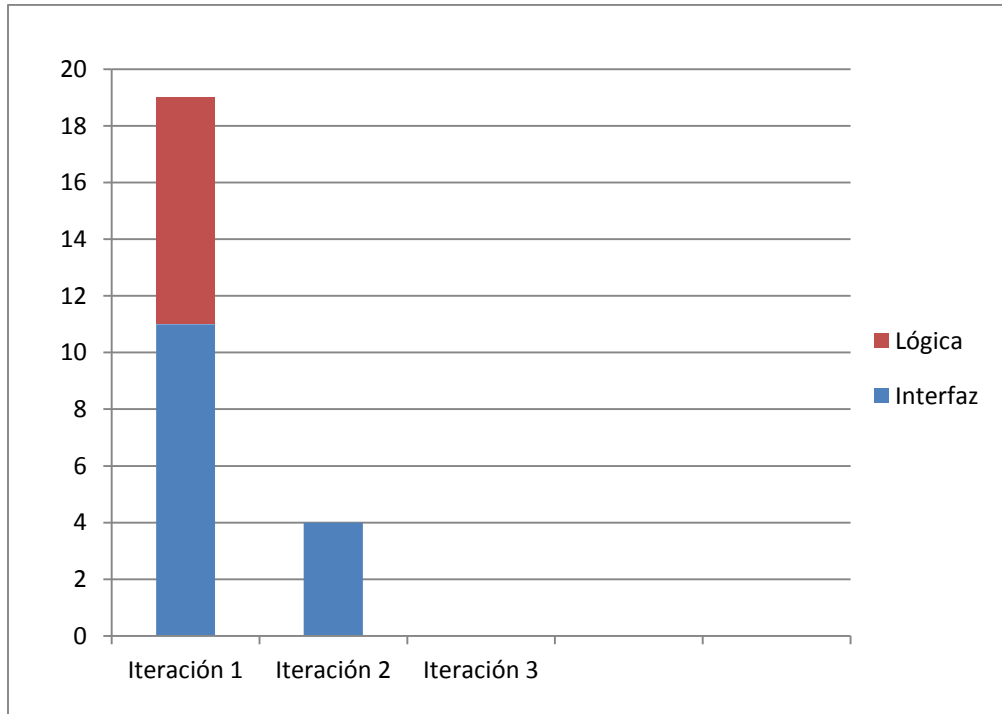


Figura 14 Cantidad de no conformidades

### Descripción de la solución propuesta

La solución propuesta permite gestionar un viaje el cual se realiza de la siguiente forma: El caso de uso inicia cuando el usuario selecciona de la barra de herramientas la opción Viajes. El sistema muestra una interfaz con las siguientes opciones de: Registrar, Eliminar, Detener, Reanudar un viaje deseado. Para el usuario realizar cada una de las opciones antes mencionadas hace clic sobre una de la misma, luego el sistema muestra una interfaz para introducir los datos necesarios en correspondencia de la opción seleccionada por el usuario.

Para Registrar nuevo viaje.

Ejemplo: El usuario introduce los datos: Ruta planificada: "Habana-Corralillo", Ómnibus:"1123", Chofer principal: "Carlos Oviedo", Chofer asistente: "Alberto Menendez", Hora de salida: "4:50 PM". El sistema guarda los cambios realizados, y así para cada opción.

Además el sistema permite visualizar la flota de ómnibus, esta opción inicia cuando el reloj del sistema actualiza la hora, el sistema muestra una interfaz con la flota de ómnibus representada en la hora determinada.

Otras de las opciones que brinda la solución es que permite gestionar paradas y gestionar velocidad. Para realizar la gestión de una parada, el usuario selecciona la opción "Paradas". El sistema muestra una interfaz para seleccionar los datos de la parada y permite registrar una parada nueva, además de eliminar y editar una parada determinada, afectando a las planificaciones que tendrán dicha parada. El usuario introduce los datos de la parada y selecciona la opción deseada. El sistema guarda los cambios y así para cada opción que el usuario desee realizar.

Luego para realizar la gestión de una velocidad, el usuario selecciona la opción "Velocidades". El sistema muestra una interfaz con todas las velocidades y permite registrar una velocidad nueva, además de eliminar y editar una velocidad determinada, afectando los recorridos que tendrán dicho cambio de velocidad. El usuario introduce los datos de la velocidad y selecciona la opción deseada. El sistema guarda los cambios y así se realizan las demás opciones.

### **4.5 Conclusiones parciales**

En el presente capítulo se arribó a un sistema completamente diseñado y construido en términos de clases del diseño. Se generaron además cada uno de los artefactos y diagramas referentes al flujo de trabajo de implementación culminando la modelación completa de la solución propuesta.

### CONCLUSIONES

Una vez finalizada la investigación y luego de obtenerse los resultados, es posible resaltar una serie de conclusiones que se enumeran a continuación:

1. El contar con una herramienta de simulación para controlar la flota de ómnibus permite hacer cálculos de distancia y tiempos de ejecución de los viajes de la EON. Esto es beneficioso en el sentido que se pueden trazar nuevas rutas, administrar recursos como el combustible y regular los horarios de llegada y planificación por paradas.
2. Las tecnologías, herramientas, lenguajes y framework seleccionados para el desarrollo de la aplicación que centra la investigación obedecen a criterios de selección de tecnologías libres y multiplataforma, de acuerdo con las políticas que impulsa la universidad y el país en sentido general.
3. La selección del patrón Modelo-Vista-Controlador permitió controlar la lógica del sistema en una capa bien definida de controladores y objetos de dominio; de este modo varias funciones del sistema pudieron utilizarse desde interfaces diferentes sin necesidad de realizar cambios en la lógica de la aplicación.
4. El uso los patrones de diseño Table Data Gateway y Foreign Key Mapping posibilitaron que las entidades y objetos del dominio se manipularan como objetos en sí y no como referencia a una estructura de la base de datos, lo cual abstrae toda la capa de datos y el centro de atención gira entorno a la lógica de estos datos.
5. El diseño y la arquitectura de información de la aplicación se presentan de manera sencilla, fácil de usar, intuitiva y con balance visual de colores que proporcionan un entorno agradable al usuario final.
6. Todos los requisitos funcionales y no funcionales capturados en el momento correspondiente fueron debidamente implementados.
7. El diseño de las pruebas de caja negra permitió validar los requisitos funcionales capturados y comprobar que el sistema realiza las acciones que debe realizar en el momento y lugar en el adecuado.

## RECOMENDACIONES

Los autores de la presente investigación recomiendan:

1. Establecer un periodo de capacitación a los usuarios del departamento de operaciones de la EON que utilizarán la herramienta construida para gestionar la flota.
2. Incluir técnicas, componentes o subsistemas de almacenamiento en memoria como memcached o redis con el objetivo de aliviar la carga de la base de datos y almacenar cálculos y/o resultados de operaciones que con frecuencia se utilizan. Ejemplo de ello: la lista de paradas por planificación, los puntos de velocidad a lo largo del recorrido, los ómnibus de la base y la lista de choferes.
3. Hacer uso del patrón de diseño Serialized LOB (variante BLOB o CBLOB) con el propósito de una vez calculado los vértices y tramos (paradas, velocidad, recorrido) de cada recorrido, guardar la estructura jerárquica resultado de esta operación en memoria para evitar futuros cálculo y de este modo reducir los tiempos de respuestas del sistema.
4. Agregar nuevas funcionalidades al sistema como son:
  - a. Cuando el usuario regrese el indicador de tiempo a una hora anterior a la actual, el sistema brinde la posibilidad de asumir dos variantes: considerar que el día actual concluyó; o regresar la simulación a la hora indicada del mismo día.
  - b. Guardar configuraciones internas como: frecuencia con que varía el tiempo y última hora simulada.
5. Hacer distinciones de roles de acuerdo al nivel de privilegios del usuario; de este modo se puede definir: ¿qué operaciones puede realizar cada tipo de usuario?
6. Incluir soporte para analizar tramas geo-localizadas por GPS y de este modo refinar los valores de tiempo simulados, haciendo comparaciones entre los valores obtenidos producto a la simulación y los valores reales de tiempo empleados.



### REFERENCIAS BIBLIOGRÁFICAS

1. **Águila, Adrián Gracia. 2011.** Plataforma Tecnología Soberana, para el almacenamiento, representación y análisis de información espacial en interés para la defensa. 2011.
2. **Álvarez, Gonzálo. 1999.** ¿Qué es Java? Características del lenguaje. [En línea] 1999. [Citado el: 8 de Enero de 2013.] <http://www.iec.csic.es/cryptonicon/java/guesjava.html>.
3. **Álvarez, Marcos Manuel. 2012.** Tecnologías de la Información. 2012.
4. **autores, Colectivo de. 2006.** Preparación pedagógica integral para profesores integrales. La Habana : Félix Varela, 2006.
5. **Ayala, Claudia P. 2005.** Análisis Comparativo de Lenguajes de Modelado Orientados a Objetivos basados en i\*. Barcelona : s.n., 2005.
6. **Belmonte, Oscar. 2004.** Introducción al lenguaje de programación Java: Una guía básica. 2004.
7. **Bosque, Sendra J. 1992.** Sistemas de Información Geográfica. Madrid : s.n., 1992.
8. **Carrillo, Isaías. 2008.** Metodología del Desarrollo del Software. 2008.
9. **Castro, Fidel. 1998.** Ministerio de Cultura de la República de Cuba. Dirección de Seguridad y Protección. [En línea] Gaceta Oficial de la República de Cuba, 14 de Junio de 1998. [Citado el: 1 de Diciembre de 2012.] <http://www.min.cult.cu/loader.php?sec=instituciones&cont=direccionseguridad>.
10. **Chavez, Juan Manuel. 2010.** GESTION INFORMÁTICA CON SOFTWARE LIBRE. Cádiz : s.n., 2010.
11. **Companioni, Alain Leon. 2010.** Sistema de Información Geográfica para la representación de objetivos petroleros. La habana : s.n., 2010.
12. **Consejería de Agricultura, Pesca y Medio Ambiente. 2007-2013.** Consejería de Agricultura, Pesca y Medio Ambiente. [En línea] 2007-2013. [Citado el: 5 de octubre de 2012.] <http://www.juntadeandalucia.es/agriculturaypesca/portal/servicios/sig/pesca-y-acuicultura/la-herramienta-sigaqua.html>.
13. **Consejería de Agricultura, Pesca y Medio Ambiente. 2007-2013.** Consejería de Agricultura, Pesca y Medio Ambiente. [En línea] 2007-2013.

- <http://www.juntadeandalucia.es/agriculturaypesca/portal/servicios/sig/pesca-y-acuicultura/la-herramienta-sigaqua.html>.
14. **Cornejo, José Enrique González.** 2012. DocIRS. [En línea] 2012. <http://www.docirs.cl/uml.htm>.
  15. **Crompovoets, Joep y Rajabifard, Tatiana Delgado y Abbas.** 2006. Introducción a las Infraestructuras. La Habana : s.n., 2006.
  16. **Cuesta, Sandra Milena.** 2008. Definiciones de sistema. Bogotá : s.n., 2008.
  17. **Cueva, Juan Manuel.** 1998. CONCEPTOS BÁSICOS DE PROCESADORES DE LENGUAJE. España : SERVITEC, 1998.
  18. **CyMSat.** 2005. Control y Monitoreo Satelital. [En línea] 2005. [Citado el: 14 de 12 de 2012.] <http://www.cymsat.com.ar/soluciones.htm>.
  19. **De Nobrega, María.** 2012. Herramientas CASE: Rational Rose. [En línea] 2012. [Citado el: 10 de Diciembre de 2012.] [http://curso\\_sin2.blogia.com/2005/060401-herramientas-case-rational-rose.-por-maria-de-nobre](http://curso_sin2.blogia.com/2005/060401-herramientas-case-rational-rose.-por-maria-de-nobre).
  20. **definicion.de.** 2012. Definicion.de. [En línea] 2012. <http://definicion.de/modelo/>.
  21. **Departamento de Sistemas Informáticos y Computación.** 2010. Rational Unified Process (RUP). Valencia : s.n., 2010.
  22. **Díaz, Alvaro A. Penroz.** 2005. Graphical User Interface (GUI) para el programa servidor de mapas MapServer 4.6.1. 2005.
  23. **Diccionario, Online de Español.** 2007. Diccionario.com. [En línea] 2007. [Citado el: 14 de 12 de 2012.] <http://www.1diccionario.com/buscar/flota>.
  24. **ebisframe.** 2006. Arquitectura en tres capas. 2006.
  25. **EMS Database Management.** 2012. EMS SQL Manager. 2012.
  26. **Entorno Virtual de Aprendizaje.** 2007-2008. Entorno Virtual de Aprendizaje. Entorno Virtual de Aprendizaje. [En línea] 2007-2008. <http://eva.uci.cu/course/view.php?id=161>.
  27. **Escobar, Sandor.** 2011. Sistema de Información Geográfica para automatizar la gestión de la distribución de las redes eléctricas en la Unión Nacional Eléctrica. La Habana : s.n., 2011.
  28. **Espinosa, A.M.** 2010. Fundamentos del Mapserver, Mapscript, PostGIS y su integración con el Cartoweb. 2010.
  29. **Espinosa, Antonio Membrides.** 2000. Fundamentos del Mapserver, Mapscript, PostGIS y su integración con el Cartoweb. Ciudad de la Habana : s.n., 2000.

30. **Farré, José David y González, Guillermo. 2010.** Aplicación para analizar y procesar tramas GPS utilizando software libre. Villa Clara. Cuba : Agencia de Software GEOMIX. Empresa GEOCUBA, 2010.
31. **Fernández, Humberto. 2011.** Módulo de Control de Flotas para el Sistema de Información Geográfica GeoQ. La Habana : s.n., 2011.
32. **Francisca Losavio, Jean Carlos Guzmán, Alfredo Matteo. 2011.** Correspondencia Semántica entre los lenguajes BPMN y GRL. Venezuela : Revista Venezolana de Información, Tecnología y Conocimiento, 2011.
33. **Gallego, Juan Pablo Gómez. 2007.** Fundamentos de la Metodología RUP. s.l. : UNIVERSIDAD TECNOLÓGICA DE PEREIRA, 2007.
34. **González, Guillermo, Cruz Iglesias, Rafael y Capote Fernández, José Luis. 2011.** MovilWeb: Aplicación para el control de flotas basada en PostgreSQL. La Habana : Revista Cubana de Ciencias Informáticas, 2011. 1994-1536.
35. **González, Herrera y Iglesias, Cruz . 2008.** Mapping Interactivo. MovilWeb: Aplicación para el control de Flotas basada en la Infraestructura de Datos . 2008.
36. **González, Pedro José. 2010.** Implementación de los módulos de Facturación y Cobro del Sistema de Facturación y Cobro para la Empresa de Gas Manufacturado de Ciudad de la Habana. Habana. Ciudad Habana : s.n., 2010.
37. **Guía de Ubuntu. 2008.** Guía Documentada para Ubuntu. [En línea] 2008. [http://www.guia-ubuntu.com/index.php?title=PgAdmin\\_III](http://www.guia-ubuntu.com/index.php?title=PgAdmin_III).
38. **Guillaumet, A., Cellini, Andres y Coen, Fernando. 2000.** Minería de Datos Espaciales. 2000.
39. **Hernández, Orallo. 2006.** El Lenguaje Unificado de Modelado (UML). 2006.
40. **IBM. 2011.** Introducción a Business Process Management (BPM). [En línea] 29 de Abril de 2011. [Citado el: 28 de Febrero de 2013.] <http://www.ibm.com/developerworks/ssa/local/websphere/introduccion-bpm/index.html>.
41. **IGNPerú. 2005.** Instituto Geográfico Nacional Peruano. Instituto Geográfico Nacional Peruano. [En línea] 2005. [Citado el: 4 de diciembre de 2012.] [www.ign.gob.pe/](http://www.ign.gob.pe/).
42. **Interflex. 2009.** Software selection. SP-Expert. [En línea] 2009. [Citado el: 7 de Enero de 2013.] <http://www.softwareseleccion.com/sp+expert-p-2755>.
43. **Jurídicas, Ediciones. 2009.** Decreto 3422 de 2009. Colombia : s.n., 2009.

44. **Laguna, Miguel A. 2006.** Modelado de sistemas software. 2006.
45. **Laman, Craig. 1999.** UML y Patrones. Introducción a1 análisis y diseño orientado a objetos. México : PRENTICE HALL, 1999.
46. **Linde, J.M.M. 2012.** ¿Qué es un SIG? [En línea] 2012. [Citado el: 7 de 12 de 2012.] <http://www.mappinginteractivo.com/plantilla-ante.asp?>
47. **Machín, Lisbey Borroto y Rondón, Dayanis Palacio. 2010.** Sistema de apoyo al proceso de relatoría en reuniones de ALBET, S.A. Habana : s.n., 2010.
48. **Mapserver, Sitio Oficial. 2011.** MapServer open source web mapping. MapServer open source web mapping. [En línea] 2011. [Citado el: 8 de enero de 2013.] <http://mapserver.org/trunk/es/about.html#about>.
49. **Martín, Carlos. 2011.** SIG-Rutas: solución informática para el servicio de transporte obrero en la Universidad de las Ciencias Informáticas. La Habana : Serie Científica de la Universidad de las Ciencias Informáticas, 2011. Vol. 4. 2306-2495.
50. **Martínez, Rafael. 2010.** Sobre PostgrSQL. [En línea] 2 de Octubre de 2010. [Citado el: 6 de Febrero de 2013.] [http://www.postgresql.org.es/sobre\\_postgresql](http://www.postgresql.org.es/sobre_postgresql).
51. **Marvin, David. 2008.** Definición de lenguaje de programación. Tipos. Ejemplos. [En línea] 16 de Octubre de 2008. [Citado el: 11 de Diciembre de 2012.] <http://catedraprogramacion.foroactivos.net/t83-definicion-de-lenguaje-de-programacion-tipos->
52. **Menéndez, Alberto. 2010.** Herramienta para la captura de datos de campo del inventario de petróleo y gas perteneciente al Sistema de Gestión de Datos Geológicos. La habana : s.n., 2010.
53. **Mezquita, Julio Cerezal y Rodríguez, Jorge Fiallo. 2005.** ¿Cómo investigar en Pedagogía? Ciudad de la Habana : s.n., 2005.
54. **Micronav. 2009.** Servicio de control de flotas mediante GPS. [En línea] 2009. [Citado el: 20 de 12 de 2012.] <http://www.micronav.net/>.
55. **Microsoft Corporation. 2012.** Las diez ventajas principales de Microsoft Office 2007. [En línea] 2012. [Citado el: 10 de Diciembre de 2012.] <http://office.microsoft.com/es-es/word-help/las-diez-ventajas-principales-de-microsoft-offic>.

56. **Molina, David. 2012.** SiCaTu. Sistema de Calculo de Turnos. [En línea] 9 de Febrero de 2012. [Citado el: 15 de Octubre de 2012.] <http://modal.es>.
57. **Montoya, Caro. 2012.** scribd. scribd. [En línea] 2012. <http://es.scribd.com/doc/52147445/3/Teoria-de-datos-e-informacion>.
58. **Moreta, Orlando Ramiro Erazo. 2009.** Diseño e implementación de Mapa Interactivo utilizando Web Mapping y Base de Datos Espacial: Ciudad de Quevedo. Quito : s.n., 2009. Tesis Doctoral.
59. **Murillo, Félix. 1999.** Herramientas CASE. 1999.
60. **Olaya, Víctor. 2010.** Sistemas de Información Geográfica. Girona : s.n., 2010.
61. **Oracle. 2012.** Bienvenido a NetBeans y [www.netbeans.org](http://www.netbeans.org). ¿Qué es NetBeans? [En línea] Oracle Corporation, 2012. [Citado el: 11 de Diciembre de 2012.] [http://netbeans.org/index\\_es.html](http://netbeans.org/index_es.html).
62. **Pavón, Eduardo León. 2000.** Tutorial Visual Paradigm for UML. España : s.n., 2000.
63. **Pérez, Adrián Fuentenegro. 2011.** Generación de consultas para la manipulación de Geontologías desde la Plataforma GeneSIG. 2011. Tesis.
64. **pgAdmin. 2009.** pgAdmin PostgreSQL tools. [En línea] 26 de Marzo de 2009. [Citado el: 7 de Febrero de 2013.] <http://www.pgadmin.org/licence.php>.
65. **Pockin. 2008.** Modelado de Sistemas com UML. 2008.
66. **Portal Profesional del Medio Ambiente. 2010.** Revista Ambientum formación. [En línea] 2010. <http://www.ambientum.com/revista/2010/febrero/aplicaciones-medioambientales-SIG.asp>.
67. **2012. PostGIS.** [En línea] Project OSGeo, 2012. <http://postgis.refractor.net/>.
68. **PostgreSQL. 2013.** Sitio Oficial de PostgreSQL. [En línea] 2013. [Citado el: 27 de 02 de 2013.] <http://postgresql.org/docs>.
69. **PostgreSQL Sitio oficial del Servidor de Base de datos.** [En línea] [Citado el: 12 de noviembre de 2012.] [http://www.dirphp.com/dir/Software\\_y\\_servidores\\_PHP/PostgreSQL\\_Sitio\\_oficial\\_del\\_Servidor\\_de\\_Base\\_de\\_datos\\_51.html](http://www.dirphp.com/dir/Software_y_servidores_PHP/PostgreSQL_Sitio_oficial_del_Servidor_de_Base_de_datos_51.html).
70. **ProDevelop. 2009.** ProDevelop integración de Tecnologías. [En línea] 2009. <http://www.prodevelop.es/es/tecs/geo/servidoresmapas>.

71. **Qt Project Hosting. 2012.** Qt Project. Qt Creator. [En línea] 5 de Enero de 2012. [Citado el: 13 de Diciembre de 2012.] [http://qt-project.org/wiki/Category:Tools::QtCreator\\_Spanish](http://qt-project.org/wiki/Category:Tools::QtCreator_Spanish).
72. **Real Academia Española.** Real Academia Española. [En línea] [Citado el: 20 de febrero de 2013 .] <http://www.rae.es>.
73. **Reynoso, Carlos y Kiccillof, Nicolás. 2004.** Estilos y Patrones en la Estrategia de Arquitectura de. Buenos Aires : Universidad de Buenos Aires, 2004. V 1.0.
74. **Robaina, Irilys Ledón. 2008.** Propuesta del Diseño Arquitectónico del Simulador de Sistemas Biológicos: BioSyS. Habana : s.n., 2008. Trabajo de Diploma.
75. **Ronny, Yabar Aizcorbe. 2010.** Development with opengl and qt. [En línea] 2010.
76. **Rosas, Gonzalo. 2011.** Modelado del Negocio con UML. 2011.
77. **Rumbaugh, James. 2004.** El proceso unificado de desarrollo de software. La Habana : Félix Varela, 2004.
78. **Sanabria, Javier Manrique. 2011.** Conceptos de SIG y Cartografía. Bogotá : s.n., 2011.
79. **Sanchez, María A. Mendoza. 2002.** Metodologías De Desarrollo De Software. 2002.
80. **Sarabia, Gerardo López. 2008.** Búsqueda y ponderación de información contenida en base de datos espaciales utilizando jerarquías. Mexico : Centro de Investigación en Computación, 2008. Tesis Doctoral.
81. **Solex. 2012.** Solex. Planificación y Control de Turnos de Guardia. [En línea] 8 de Octubre de 2012. [Citado el: 21 de Octubre de 2012.]
82. **Solís, Roberth Figueroa y Camilo J.** Metodologías Tradicionales VS. Metodologías Ágiles. s.l. : Universidad Técnica Particular de Loja.
83. **Soluciones Informáticas Globales. 2007.** SIG Soluciones Informáticas Globales. SIG Soluciones Informáticas Globales. [En línea] 2007. <http://www.sig2k.com.ar/erp-ventajas.html>.
84. **Tabares, Marta Silvia. 2011.** Arquitectura de Software Parte 1. [En línea] 4 de Septiembre de 2011. [Citado el: 10 de Diciembre de 2012.] <http://www.slideshare.net/mstabare/arquitecturas-de-software-parte-1>.

## *Referencias Bibliográfica*

---

85. **tasof. 2005.** De Modelos, Metamodelos y Metametamodelos . De Modelos, Metamodelos y Metametamodelos . [En línea] 13 de octubre de 2005. <http://tasof-ucn.blogspot.com/2005/10/de-modelos-metamodelos-y.html>.
86. **Tobarra, Manuel. 2003.** CMM y RUP: Una perspectiva común. Albacete : s.n., 2003.
87. **Yagüez, Domingo . 2011.** Sistema de Información Geográfica (S.I.G.). [En línea] 2011. [http://www.inta.gov.ar/barrow/info/documentos/SIG/que\\_es\\_sig.htm](http://www.inta.gov.ar/barrow/info/documentos/SIG/que_es_sig.htm).
88. **Zayas, Carlos Alvarez de. 1995.** Metodología de la Investigación Científica. Santiago de Cuba : Centro de Estudios de Educación Superior, 1995. pág. 80.
89. **Zayas, Carlos M. Alvarez de. 1999.** La escuela en la vida: didáctica. La Habana : Pueblo y Educación, 1999.

## BIBLIOGRAFÍA

1. **Associates, O'Reilly &. 1998.** UML en Resumen: Una Rápida Referencia de Escritorio. 1998.
2. **Brazales, Alfonso y Garcia de Jalon, Javier. 1998.** Aprenda C++ como si estuviera en primero. España: s.n., 1998.
3. **Bruno R. Preiss y otros. 1997.** Data Structures and Algorithms with Object-Oriented Design Patterns in C++. Universidad de Waterloo, Canadá.
4. **Bravo, Javier Domínguez. 2000.** Breve Introducción a la Cartografía y a los Sistemas de Información Geográfica (SIG). Madrid: CIEMAT, 2000.
5. **CORNEJO, J. E. G. 2001.** Arquitectura en Capas. Un camino hacia los procesos distribuidos. 2001.
6. **Departamento de Ingeniería de Software. 2007-2008.** Introducción al proceso de desarrollo de software. Habana: s.n., 2007-2008.
7. **Hernández, Rolando Alfredo y González, Sayda Coello. 2011.** El Proceso de Investigación Científica. Ciudad de La Habana: Universitaria del Ministerio de Educación Superior, 2011. pág. 110.
8. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** El Proceso Unificado de Desarrollo de Software. España: Pearson Educación, 2000.
9. **Larman, Craig. UML y Patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado.** Canadá: s.n.
10. **Nicanor, Lisandro Damián. 2007.** Introducción al desarrollo con Qt. 2007.
11. **Pressman, Roger. 2005.** Ingeniería del Software. Un enfoque práctico. La Habana: Félix Varela, 2005.
12. **Reynoso, Carlos. 2004.** Introducción a la arquitectura de software. Buenos Aires: s.n., 2004.
13. **RIORDAN, Rebecca. 1999.** Designing Relational Database Systems. s.l.: Microsoft Press, 1999. ISBN 0-7356-0634-X.
14. **Rumbaugh, J., Booch, G. y Jacobson, L. 2000.** El Proceso Unificado de Desarrollo. 2000.
15. **Allen G. Taylor 2006.** SQL For Dummies, 6th Edition. Wiley Publishing, Inc. Indianapolis, Indiana. ISBN: 978-0-470-04652-4.