Universidad de las Ciencias Informáticas
FACULTAD 6

Subsistema de visualización de información georreferenciada del Sistema Control de Flotas para dispositivos móviles

---

Trabajo de Diploma para optar por el Título de
Ingeniero en Ciencias Informáticas

Autor: Willian Simón Grass
Tutor: Ing. Odiel Estrada Molina

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

Poseemos sin embargo invencibles armas. La principal es la educación....Todo lo transformará y seremos pronto el pueblo más educado y culto del mundo. Ya nadie lo duda dentro y fuera de Cuba".

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los___días del mes de ____ del año _____.

_____	_____
Autor: Willian Simón Grass	Tutor: Ing. Odiel Estrada Molina

DATOS DE CONTACTO

TUTOR: Ing. Odiel Estrada Molina

Graduado de Ingeniería en Ciencias Informáticas en Julio de 2010

Profesor de Subsistemas Organizacionales

Dirección: Universidad de las Ciencias Informáticas (UCI)

Teléfono Oficina: +53 – 7 – 837 Teléfono Apto: +53 – 7 – 835-8503 E-mail: oestrada@uci.cu

Síntesis del Tutor

Profesión: Ingeniero en Ciencias Informáticas. Profesor de Subsistemas Organizacionales de la Facultad 6 de la UCI

Categoría docente: Instructor

Años de graduado: 3

AGRADECIMIENTOS

A Nuestro Comandante en jefe Fidel Castro Ruz por crear este centro de altos estudios y permitir mi formación como profesional.

A mi mamá, mi papá y mi hermano, por apoyarme siempre y luchar tanto para que nunca me faltara nada, por ser mis guías eternos y por apoyarme en cada paso que daba.

A Mis Abuelas Ide y Ne, no solo por ser abuelas, sino, también madres.

A toda mi familia, a mis bisabuelos, tíos, tías y primos que estuvieron siempre pendientes de mí.

A la memoria de mi "abuelo" Mon y mi Tío David al cual estaré eternamente agradecido.

A mi novia Cecilia Esther Hernández Espinosa más por sobre todas las cosas que supo guiarme por el camino correcto.

A mis queridos amigos por darme su apoyo incondicional Lester, José Carlos, Roexcy.

A mi tutor Odiel que siempre estuvo ahí cuando tenía dudas, cuando el camino se tornaba difícil, por ser perfeccionista con cada detalle y hacerme dar lo mejor de mí en cada momento.

A La FEU Y UJC a todos los compañeros que hice en mi estancia como dirigente estudiantil de veras que esa fue mi mayor escuela.

A los profesores que impartieron clases a lo largo de toda la carrera.

A todos muchísimas gracias.

DEDICATORIA

A mi mamá y mi papá por apoyarme siempre y luchar tanto

para que nunca me faltara nada, por ser mis guías eternos.

A mi querido hermano Walter porque al luchar por sus sueños me animaba a

luchar por los míos.

A toda mi familia en especial a mis abuelos,

bisabuelos, tíos, tías y primos que estuvieron siempre pendientes de mí.

A mi compañera, a ti Cili que de veras sin ti no hubiese llegado hasta donde fui capaz de llegar.

RESUMEN

Hoy en día empresas e instituciones hacen uso de soluciones informáticas que ayudan a gestionar los procesos, garantizando de esta forma una mejor toma de decisiones. Un ejemplo de estas soluciones son los llamados Sistemas de Control de Flotas (SCF). Estos sistemas aportan el control exhaustivo de la flota en cuanto a: consumo de combustible, paradas, parámetros de calidad y seguridad con los que transporta su carga y el cumplimiento en la realización de las rutas.

El Centro de Desarrollo Geoinformática y Señales Digitales (GEYSED) de la Universidad de la Ciencias Informáticas (UCI) consta con el proyecto Aplicativos Sistema de Información Geográfica (SIG) y su trabajo está dirigido al desarrollo de Sistema de Información Geográfica de acuerdo al negocio de la empresa o institución que necesite un sistema de este tipo. En este proyecto se implementó un SCF que tiene como objetivo el control de la transportación nacional; pero posee como deficiencia el acceso y visualización a la hora de llevar a cabo cualquier tipo de operación sobre mapas en dispositivos móviles. En el presente trabajo de diploma, se propone un subsistema de visualización de información georreferenciada, para dispositivos móviles, posibilitando a los usuarios acceder y visualizar las funcionalidades en ambientes móviles del SCF del proyecto Aplicativos SIG.

Palabras claves: *dispositivos móviles, información georreferenciada, subsistema de visualización.*

INDICE

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCIÓN

Dada la alta competitividad actual del mercado, toda institución u organización se enfrenta día a día con la necesidad de reducir sus costos y aumentar la calidad del servicio prestado. Los recursos utilizados deben ser administrados de la forma más eficiente para minimizar gastos y maximizar ganancias. El avance de las Tecnologías de la Informática y las Comunicaciones (TIC) ha propiciado el notable desarrollo de soluciones informáticas, que ayudan a gestionar los procesos que se llevan a cabo en cada una de estas instituciones, garantizando una mejor toma de decisiones, que responden a un buen servicio y mayor control sobre los recursos. (Torres, 2006)

Es en este contexto surgen los Sistemas de Control de Flotas (SCF). Estos sistemas son ampliamente utilizados tanto por pequeñas, medianas o grandes empresas. Además son capaces de brindar múltiples servicios tales como: ver los vehículos en el mapa, controlar la actividad de la flota y conocer el vehículo más cercano a una dirección determinada. Por consiguiente, el mantenimiento preventivo de las flotas implica una disminución de fallos mecánicos. La prevención de los mismos permite mantener a las flotas en actividad la mayor parte de su vida útil, facilitando a la empresa realizar una mejor evaluación de su eficiencia.

Algunos de los medios más utilizados para realizar el control por parte de las instituciones son las computadoras personales y la telefonía móvil. Esta última ha tenido gran aceptación por sus disímiles ventajas. Su evolución ha permitido que disminuyan tanto en tamaño como en peso y aunque su principal función es la comunicación, se han integrado otros tipos de servicios tales como datos, audio y video, así como una amplia gama de funcionalidades que aumentan diariamente, posibilitando que sea un producto de preferencia.

Todo esto ha conllevado a la aparición de nuevas formas de negocio basadas en los servicios móviles, los cuales tienen gran potencial económico para la competencia, la innovación y el crecimiento. Este sector ofrece ventajas para la innovación, debido a que implica el uso de nuevas tecnologías, procesos y formas de interactuar con el cliente. La capacidad de obtener la localización geográfica a partir de receptores del Sistema de Posicionamiento Global (GPS), junto a su creciente poder de cálculo, ha marcado el predominio de estos dispositivos sobre las tradicionales computadoras personales. (USA, 2012) Estos equipos están provistos de un sistema operativo que facilita la explotación de sus adelantos de *hardware* por los desarrolladores.

Alcanzar la informatización de todos los sectores sociales es una meta trazada en Cuba para obtener mayores beneficios. Numerosas empresas cubanas no cuentan con las tecnologías y herramientas necesarias para adentrarse en el proceso, por lo que estos son los primeros pasos en los que se está trabajando para lograr el objetivo. En la facultad 6 de la UCI se encuentra el Centro de Desarrollo Geoinformática y Señales Digitales (GEYSED), compuesto por dos departamentos, que a su vez integran diferentes líneas de desarrollo. El proyecto Aplicativos SIG pertenece al departamento Geoinformática, y su trabajo está dirigido al desarrollo de Sistemas de Información Geográfica, de acuerdo a la solicitud de las diferentes empresas o instituciones que necesiten un sistema de este tipo.

Un Sistema de Control de Flotas (SCF) para computadoras personales es una solución existente en el proyecto Aplicativos SIG. Este permite almacenar una gran cantidad de información georreferenciada, que puede ser consultada, modificada o eliminada desde computadoras personales. Teniendo en cuenta el auge de las tecnologías móviles en los últimos años, la disponibilidad y el acceso a redes móviles de datos surge la necesidad de interactuar con el sistema desde estos dispositivos, presentando los mismos problemas de acceso y visualización.

Sobre la base de las consideraciones anteriores se detectaron las siguientes limitaciones: inexistencia de un mecanismo que permita conocer con inmediatez y exactitud la ubicación de la flota mediante el dispositivo móvil, imposibilitando mantener el control sobre la localización y estado de la misma. No existe un procedimiento en el SCF implementado que garantice la visualización en ambientes móviles, de datos referenciados, actualizados sobre los puntos de interés, rutas y paradas en el mapa, afectando la toma de decisiones por parte de los usuarios que en ocasiones consultan la aplicación en busca de información oportuna. No se cuenta con un mecanismo que permita al usuario detectar los movimientos de la flota ni visualizar la información referenciada del SCF mediante servicios WMS y a la vez sea soportada en varias plataformas de dispositivos móviles.

A partir de las limitaciones expuestas se identificó el siguiente **problema a resolver**: ¿Cómo contribuir a la visualización de información georreferenciada del Sistema Control de Flotas del proyecto Aplicativos SIG desde dispositivos móviles?

El **objeto de estudio** está constituido por los Sistemas Control de Flotas en dispositivos móviles.

Se define como **objetivo general** de la investigación: desarrollar un subsistema de visualización de información que permita mostrar en dispositivos móviles la información georreferenciada del Sistema Control de Flotas del proyecto Aplicativos SIG.

El **campo de acción** está enmarcado en el subsistema de visualización de los Sistemas Control de Flotas en dispositivos móviles.

Para cumplir con el objetivo general y dar solución al problema planteado, se definen las siguientes tareas de la investigación:

1. Caracterización de las tendencias actuales, tecnologías y conceptos asociados sobre los subsistemas de visualización de los Sistemas Control de Flotas en dispositivos móviles.
2. Caracterización de los principales subsistemas de visualización de información georreferenciada para dispositivos móviles.
3. Selección de las herramientas, lenguajes de programación y de modelado, tecnologías y metodología de desarrollo de *software* a utilizar para el desarrollo del subsistema de visualización de información georreferenciada para dispositivos móviles.
4. Elaboración de los artefactos asociados a la metodología de desarrollo seleccionada.
5. Construcción del subsistema de visualización de información georreferenciada del Sistema Control de Flotas para dispositivos móviles.
6. Validación del funcionamiento del sistema informático haciendo uso de pruebas ingenieriles.

Se plantea como **idea a defender,** que la implementación del subsistema de visualización de información georreferenciada para el Sistema Control de Flotas del proyecto Aplicativos SIG para dispositivos móviles, permitirá visualizar la información georreferenciada del Sistema Control de Flotas del proyecto Aplicativos SIG.

Para el desarrollo de la investigación se decidió recurrir a los **métodos científicos:**

Métodos teóricos:

Analítico-sintético: se utilizó en el análisis de las bibliografías relacionadas con los Sistemas de Control de flotas para los dispositivos móviles, posibilitando su comprensión y permitiendo la captura de aquellos elementos que se relacionan con el objeto de estudio.

Modelación: se empleó para mostrar las diferentes modelaciones de los artefactos que se construyen como resultado del proceso de ingeniería de *software*.

Métodos empíricos:

Las entrevistas: se realizaron a los usuarios y a especialistas con experiencia en el desarrollo de SCF para recopilar la información referente a los subsistemas de visualización de información georreferenciada.

Observación: se utiliza en la fase de pruebas con el fin de comprobar el cumplimiento de los requisitos.

Los **posibles resultados** del trabajo de diploma están dados:

- ✓ El subsistema de visualización de información georreferenciada del Sistema Control de Flotas para dispositivos móviles.
- ✓ La documentación asociada al proceso de desarrollo del subsistema.

El presente documento se estructura en cuatro capítulos que a continuación se describen:

Capítulo 1: se realiza un análisis del problema, realizándose la fundamentación teórica de la investigación, en el cual serán expuestos los principales conceptos asociados al objeto de estudio para lograr un mayor entendimiento por parte de los usuarios. Así como una síntesis de algunas soluciones existentes para el problema identificado. Se describe el tipo de metodología, lenguaje de modelado y herramienta CASE[1] que son las más adecuadas para la construcción de los artefactos correspondientes a cada uno de los flujos de *RUP*[2].

Capítulo 2: se define los requisitos funcionales y no funcionales del sistema. Teniendo en cuenta los requisitos funcionales, se identifican los casos de uso del sistema, se realizan las descripciones textuales de los mismos y se estructura el modelo de casos de uso del sistema.

Capítulo 3: se definen las clases del diseño y sus relaciones. Se realiza además el Modelo de Implementación y de Despliegue. Se implementarán cada una de las funcionalidades de forma iterativa e incremental hasta lograr el producto completo. Finalmente se realizan las pruebas de sistema que son las pruebas que se realizan cuando el *software* está funcionando como un todo. Se utilizó el método de caja negra, que permite comprobar que cada funcionalidad es operativa.

[1] *Computer Aided Software Engineering*, lo que en español sería Ingeniería de Software Asistida por Ordenador.

[2] Proceso Unificado Racional o de Desarrollo.

CAPÍTULO 1. **FUNDAMENTACIÓN TEÓRICA**

1.1. Introducción

El presente capítulo se compone de las principales definiciones y conceptos que serán de utilidad para lograr un mejor entendimiento de la investigación. Se analizan las soluciones existentes, que lleven a cabo el control de las flotas donde son utilizados los subsistemas de visualización. Se realiza un estudio para obtener una perspectiva de la utilización de funcionalidades, principales tecnologías y herramientas a utilizar durante el desarrollo del subsistema y se precisan las limitantes que imposibilitan su utilización como solución al problema a resolver de la investigación. Además de la metodología a utilizar.

1.2. Conceptos asociados al dominio del problema

Sistema de Información Geográfica (SIG): un Sistema de Información Geográfica (SIG) es una integración organizada de *hardware*, *software* y datos geográficos diseñada para capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada con el fin de resolver problemas complejos de planificación y gestión. (Taylor, 1991)

Subsistema de visualización: un subsistema provee la visualización en capas de información geográfica. Cada una de éstas capas está conformada por imágenes vectorizadas que contienen por ejemplo: espejos de agua, rutas y caminos, actividades humanas, puntos geográficos y curvas de nivel. Con el objeto de brindar una simple interacción al usuario con dicha información, el subsistema contiene herramientas para visualizar una selección de dichas capas y navegación georreferenciada sobre ellas (Ej.: alejar, acercar, trasladar). Además de navegación georreferenciada sobre capas de información geográfica en tiempo real, el sistema permite el almacenamiento de los recorridos y su posterior reproducción. Cabe destacar que cada simulación en tiempo real puede ser almacenada y posteriormente reproducida (N., 2005) , ya sea para análisis y optimización de recorridos, corrección de errores, entre otras aplicaciones.

Dispositivos inalámbricos: son dispositivos que no necesitan estar conectados físicamente mediante cables. Se corresponden con una tecnología mediante la cual todo dispositivo ofrece una conectividad vía radio (ondas). (Morales, y otros, 2002)

1.2.1. Dispositivos móviles

Los dispositivos móviles (también conocidos como computadora de mano, *palmtop* o simplemente *handheld*) son equipos de pequeño tamaño, con algunas capacidades de procesamiento, con conexión permanente o intermitente a una red, con memoria limitada, diseñados específicamente para una función, pero que pueden llevar a cabo otras funciones más generales. (Conde, y otros, Albacete. 2008)

Los dispositivos móviles pueden ser clasificados en:

Dispositivo Móvil de Datos Limitados (*Limited Data Mobile Device*): es una de las clasificaciones de los dispositivos móviles, este es un teléfono móvil clásico que ofrece servicios de datos generalmente limitados a SMS y acceso *Wireless Application Protocol* (WAP).

Dispositivo Móvil de Datos Básicos (*Basic Data Mobile Device*): tiene el menú o navegación basada en íconos y ofrece acceso a correos, listas de direcciones, SMS[3] y en algunos casos a navegadores web clásicos.

Dispositivo Móvil de Datos Mejorados (*Enchaced Data Mobile Device*): tiene la navegación tipo *stylus*, ofrece las mismas características que la clasificación anterior incluyendo además aplicaciones de *Microsoft Office Mobile* (*Word, Excel y Power Point*), versiones móviles de aplicaciones corporativas y además incluyen un Sistema Operativo (SO).

La forma en que interactúan los dispositivos móviles con los usuarios está dada por el SO que tenga instalado. Un SO es un *software* que permite la interacción entre el *hardware* y los usuarios. Facilita al usuario o al programador las herramientas e interfaces adecuadas para realizar sus tareas informáticas, abstrayéndole de los complicados procesos necesarios para llevarlas a cabo. “Los sistemas operativos de los ordenadores personales o computadores de mesa, así como controlan estos aparatos electrónicos, en el caso de los SO móviles la diferencia radica en su orientación a la ''conectividad inalámbrica', los cuales a través de diversas funcionalidades y aplicaciones permiten la usabilidad de los dispositivos.” (Alonso)

[3] Servicio de mensajes cortos o *SMS* por sus siglas en inglés, permite el envío de mensajes cortos de hasta 160 caracteres a teléfonos móviles. El servicio no requiere que el móvil esté activo o en el rango de alcance de la señal celular ya que los *SMS* se almacenan un número de días hasta que la entrega sea posible.

1.2.2. Sistemas de Información Geográfica para dispositivos móviles

Los SIG para móviles son una combinación de las tecnologías SIG con los dispositivos móviles, el acceso inalámbrico a Internet y los sistemas de posicionamiento. Ofrecen la ventaja de poder acceder a la información desde cualquier lugar, de acuerdo a las posibilidades que brindan los dispositivos móviles, en cuanto a su portabilidad y funcionalidad. Existen dos tipos de SIG para móviles los cuales son:

- ✓ **SIG convencional:** en donde el sistema se ejecuta en el dispositivo, centrándose más en los trabajos propios del SIG y en la recolección y edición de datos, el usuario es el operario del SIG. (Gamboa, 1996)
- ✓ **Servicios basados en localización:** son servicios ofrecidos por terceros en función de la posición del dispositivo y del usuario, este pasa a ser el consumidor de un servicio. (Gamboa, 1996)

El acceso y modificación de los datos de un SIG es posible de dos formas principales:

- Dispositivos *on-line*: los usuarios pueden hacer peticiones de datos, modificar los mismos y devolver los resultados en tiempo real. Al utilizar este método se elimina la necesidad de almacenar grandes cantidades de datos en el dispositivo para luego poder actualizar la información. (Descamps-Villa, 2011)
- Dispositivos *off-line*: los usuarios descargan los datos que necesitan en el dispositivo y utilizan una aplicación SIG instalada en el mismo que permite visualizar y manipular los datos en el área de estudio. Este método elimina la necesidad de establecer conexiones desde el área de estudio. (Descamps-Villa, 2011)

Otro rasgo significativo en los SIG para dispositivos móviles es la localización, estos sistemas son generalmente acoplados a Sistemas de Posicionamiento Global (GPS[4]) y comunicación inalámbrica para facilitar el intercambio entre el servidor y los dispositivos móviles.

La determinación de la ubicación de los dispositivos móviles es en tiempo real. Una de las vías para determinar la ubicación de un dispositivo móvil es a través del GPS que es colocado en el teléfono móvil

[4]Sistema de Posicionamiento Global es un sistema global de navegación por satélite (GNSS) que permite determinar en todo el mundo la posición de un objeto, una persona o un vehículo con una precisión hasta de centímetros.

proporcionando una buena precisión y flexibilidad de posicionamiento para la navegación, medición y la captura de datos. (Descamps-Villa, 2011)

El transporte de la información se realiza mediante la red de comunicación inalámbrica, la cual está asociada a la información almacenada en los servidores. Los servidores ofrecen los datos y aplicaciones para el usuario, por ejemplo, ofrece la información de ubicación de los componentes y servicios de procesamiento de información. El tipo de información y servicios varía según el tipo de aplicación que se utiliza.

Algunos de los beneficios que ofrecen los SIG para dispositivos móviles son: la captura y manejo de datos en el área de estudio, el aumento de la eficiencia de la recolección de datos, la sustitución de los mapas en papel en la recogida de datos, la edición de la información geográfica en el dispositivo, el acceso a la información en cualquier lugar y a cualquier hora y una mejor información sobre el punto para tomar decisiones.

1.2.1. Teléfono móvil

Los teléfonos han atravesado diferentes generaciones durante su desarrollo hasta llegar a los actuales dispositivos móviles. Según Roig el teléfono móvil es un dispositivo inalámbrico electrónico para acceder y utilizar los servicios de la red de telefonía celular o móvil. Se denomina celular en la mayoría de países latinoamericanos debido a que el servicio funciona mediante una red de celdas, donde cada antena repetidora de señal es una célula, si bien también existen redes telefónicas móviles satelitales. Su principal característica es su portabilidad, que permite comunicarse desde casi cualquier lugar. La principal función es la comunicación de voz, como el teléfono convencional. (Roig, y otros, 2003.)

La primera generación se caracterizó por ser analógica y estrictamente para la voz. La segunda generación fue la digital. La generación 2.5 fue una alternativa que muchos proveedores de servicio de telecomunicaciones utilizaron para introducirse en la tercera generación. La tercera generación tenía como característica la convergencia de la voz y datos con acceso inalámbrico a Internet, aplicaciones multimedia y altas transmisiones de datos.

1.3. Análisis de soluciones existentes

Con el objetivo de lograr la visualización de la información georreferenciada del sistema existente en dispositivos móviles, se hace necesario el estudio tanto a nivel nacional como internacional de aplicaciones en dispositivos móviles y que proveen este tipo de servicio en aras de tomar decisiones que

favorezcan la construcción de la aplicación. Entre las soluciones existentes a continuación se realiza una caracterización de las más relevantes con especial interés en aquellas que fueron desarrolladas en herramientas libres:

MovilWeb

MovilWeb es una aplicación de localización de vehículos basada en web para el seguimiento de móviles sobre cartografía vectorial y raster, diseñada para controlar flotas dentro de una arquitectura cliente – servidor. Esta herramienta permite el monitoreo de móviles de manera remota sobre una red de comunicaciones, posibilitando reconstruir el comportamiento del vehículo en un determinado periodo de tiempo, reelaborando su trayectoria y analizando su velocidad, detenciones en destinos autorizados o no. A través de la información almacenada en una base de datos histórica, dota al usuario de un grupo de herramientas para el manejo de mapas, similar a las herramientas del *software* profesionales para el manejo de Sistemas de Información Geográfica.

Su arquitectura está orientada a servicios, en ella se produce la integración y encadenamiento de varios servicios de procesamiento y datos geoespaciales distribuidos sobre la web. Está implementada sobre plataformas de *software* libre, utiliza como gestor de base de datos PostgreSQL, *Business Intelligence and Reporting Tools* como herramienta para la generación de los reportes, GeoServer como servicio de mapas, todo se ejecuta sobre Apache Tomcat 6.0 como servidor para Internet, las nuevas utilidades han sido implementadas sobre Java y se nutre de los servicios geoespaciales disponibles en la Infraestructura de Datos Espaciales de la República de Cuba (IDERC), como los servicios de imágenes satelitales y cartografía vectorial. En la actualidad brinda servicio a 193 bases de transporte, con 900 usuarios conectados al sistema y un total de 7 806 vehículos en todas las provincias del país. (Revista Cubana de Ciencias Informáticas.)

Micronav

Es el servicio de gestión y localización de flotas basadas en GPS/GPRS[5] e Internet que permite administrar los recursos móviles con la máxima eficacia, controlarlos y obtener todos los beneficios

[5] *General Packet Radio Service* o servicio general de paquetes vía radio creado en la década de los 80 es una extensión del Sistema Global para Comunicaciones Móviles (*Global System for Mobile Communications* o GSM) para

resultantes de disponer de la información puntual y accesible en todo momento." *E*stá destinado a empresas de sectores tan variados como: ambulancias, grúas, instaladores, servicios de mantenimiento, recogida de residuos, departamentos comerciales, etc. Básicamente trabajan con *Android* y *BlackBerry.* Aunque también tiene este sistema validados algunos modelos de *WindowsMobile* y *Nokia*." (micronav, 2010)

MobileFleet

Es una plataforma destinada a llevar el control y la localización de cualquier objeto móvil. La gestión se puede efectuar desde cualquier terminal móvil y de Internet. La comunicación entre el vehículo y MobileFleet se realiza a través de GPS/GPRS. Es una herramienta de gestión cómoda, rápida, fácil y fiable, que aporta toda la información necesaria sobre la flota de vehículos y maquinaria, optimizando tiempos y ahorrando costes. Aplicación de *software* accesible a través de internet y disponible las 24 horas al día, 365 días del año. Acceso restringido mediante usuario y contraseña habilitada para cada cliente. En ella es reflejada y tratada toda la información de los activos móviles/vehículos/objetos que el usuario desee controlar como son: rutas realizadas, informes, alertas de actividad. Su principal característica es la facilidad e intuición para aprender a usarlo.

Aplicación de *software* que da nombre al sistema y hecha para dispositivos móviles. Válido para prácticamente cualquier móvil con Java o J2ME y con aplicaciones específicas *en iPhone, Blackberry, Android, Palm y Windows Mobile.* (mobilefleet, 2010)

1.3.1. Valoración del análisis de las soluciones existentes

La caracterización de las soluciones existentes estuvo enfocada en los SCF para dispositivos móviles que soportan servicio WMS. Se identificó además el servicio de mapas *MapServer* como el más factible de utilizar en aplicaciones de este tipo en el país. Permitió reconocer el conjunto de funcionalidades que debe poseer el subsistema, las cuales son: cargar el mapa desde el servidor o del dispositivo móvil, *zoom* más, *zoom* menos, paneo, recentrar, localizar flota, habilitar capas en el mapa,interactuar con el mapa.

---

la transmisión de datos mediante conmutación de paquetes. Existe un servicio similar para los teléfonos móviles, el sistema IS-136. Permite velocidades de transferencia de 56 a 144 kbps.

Además, se constató que debido a la necesidad de la investigación se deben incluir otras funcionalidades como por ejemplo, cargar mapas guardados y mostrar datos del acelerómetro.

1.4. Metodología de desarrollo

Por lo general si se quiere construir un *software* de calidad, desarrollado en el tiempo planificado y con los costes establecidos, pero que además satisfaga la necesidad de ser elaborado de una forma más acelerada, es necesario enfocarse en trabajar de forma organizada, donde se controle y documente todo lo relacionado con el proyecto en cuestión y puedan eliminarse los riesgos que podrían presentarse durante el desarrollo del mismo, lo cual no podría lograrse sin el empleo de una metodología eficaz que se adapte a las características propias del *software* que se esté desarrollando.

"Las metodologías se basan en una combinación de los modelos de proceso genéricos (cascada, evolutivo, incremental.). Adicionalmente una metodología debería definir con precisión los artefactos, roles y actividades involucrados, junto con prácticas y técnicas recomendadas, guías de adaptación de la metodología al proyecto, guías para uso de herramientas de apoyo, etc. Habitualmente se utiliza el término "método" para referirse a técnicas, notaciones y guías asociadas, que son aplicables a una (o algunas) actividades del proceso de desarrollo, por ejemplo, suele hablarse de métodos de análisis y/o diseño". (Gómez, 2012)

La clasificación de las metodologías es una tarea bien difícil por la gran cantidad de propuestas y diferencias en el grado de detalle, información disponible y alcance que poseen. A grandes rasgos, considerando su filosofía de desarrollo se pueden agrupar en: metodologías ágiles o ligeras y metodologías pesadas o tradicionales.

Las metodologías tradicionales o pesadas se guían expresamente por una fuerte planificación durante todo el proceso de desarrollo, en el cual se establecen estrictamente las actividades involucradas, los roles definidos, los artefactos que se deben producir, las herramientas y notaciones que serán utilizadas así como el modelado y documentación detallada.

Las metodologías ágiles se enfocan en evadir las burocráticas guías de las metodologías tradicionales. Centran sus procesos en heurísticas provenientes de prácticas de producción de código, donde el cliente es parte del equipo, con ciclos muy cortos de desarrollo y preparadas para cambios durante el proyecto. Los grupos de trabajo están diseñados con poco personal e interactúan en el mismo sitio, generan pocos artefactos y poseen pocos roles, además de no hacer énfasis en la arquitectura del *software*.

Posteriormente de realizar un estudio detallado de un conjunto de metodologías se decidió el uso de una metodología tradicional porque se necesita la existencia de una documentación detallada de todo el proceso de desarrollo, para que pueda ser reutilizada en futuras versiones o en el desarrollo de sistemas similares. También porque se caracterizan por centrar su atención en cumplir con un plan de proyecto definido en la fase inicial. Se cuenta con un equipo de desarrollo numeroso divido por roles. El cliente no forma parte del equipo de desarrollo. Con respecto a la curva de aprendizaje, los modelos ágiles, ofrecen una mayor ventaja pero con ciertas limitaciones, ya que aún no han sido explotadas a gran escala como sucede con *Rational Unified Process* (*RUP* metodología tradicional) que posee alto soporte y herramientas integrales que guían al equipo de desarrollo, facilitando aplicar con mayor efectividad esta metodología, permitiendo aprovecharla al máximo.

1.4.1. Proceso Unificado Racional

El Proceso Unificado Racional o de Desarrollo (*RUP*) constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Es el resultado de varios años de trabajo y uso práctico en el que se han unificado técnicas de desarrollo. Utiliza como lenguaje de modelado el *UML* (Lenguaje Unificado de Modelado). Define como sus principales elementos: Trabajadores (“quién”), Actividades (“cómo”), Artefactos (”qué”), Flujo de actividades (“Cuándo”).

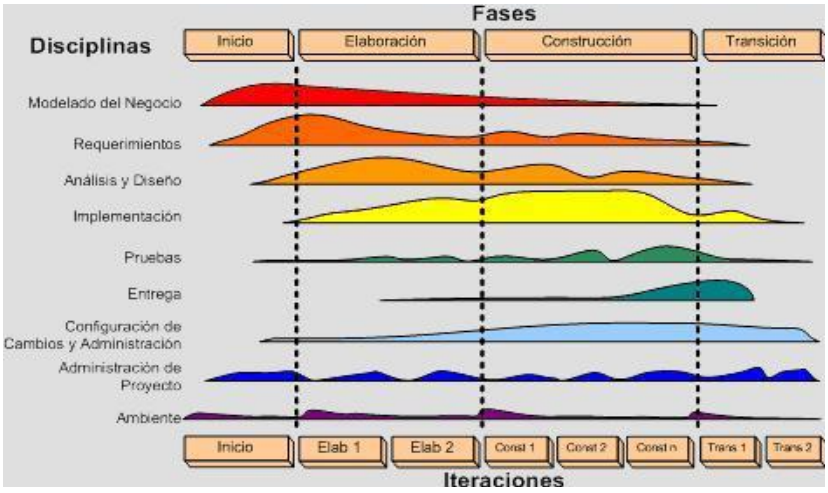


Figura 1 Proceso de desarrollo según *RUP*. [6]

[6] Tomado de (Prieto Alvarez, 2009)

En *RUP* se han agrupado las actividades en grupos lógicos definiéndose nueve flujos de trabajo, los seis primeros conocidos como flujos de ingeniería *(Modelamiento del Negocio, Requisitos, Análisis y Diseño, Implementación, Prueba e Instalación)* y los tres últimos de apoyo (*Administración del proyecto, Administración de configuración y cambios y Ambiente)* (Jacobson, 2000). Cada flujo de trabajo cumple con algunas actividades específicas. El Proceso Unificado de Desarrollo posee trabajadores específicos que producen y consumen artefactos también definidos, según se muestra en la **Figura 1**. Cada fase representa un estado del proyecto, y produce un hito que sirve de entrada a la próxima fase. Todas las disciplinas se aplican en todas las fases, algunas tienen más carga de trabajo que otras en algunas fases específicas.

Las fases en las que se dividirá el proyecto son:

1. **Inicio:** permite desarrollar el análisis del negocio hasta un determinado punto para justificar de esta forma la puesta en marcha del proyecto. Al finalizar la fase todos los involucrados en el proyecto deben tener una comprensión bastante buena de la idea del mismo y de que este es factible llevarlo a cabo.
2. **Elaboración:** permite recopilar la mayor parte de los requisitos que aún quedan pendientes y establece la línea base de la arquitectura. Al finalizar la fase se obtiene generalmente un modelo completo de negocio que describe el contexto del sistema y una arquitectura robusta que da cumplimiento a todos los requisitos.
3. **Construcción:** desarrolla un producto de *software* inicial listo para ser usado por los usuarios, llamado comúnmente versión beta. Al finalizar la fase se obtiene un producto de *software* "completo" que al usuario interactuar con él, el mismo puede develar algunos errores y estos pueden ser modificados.
4. **Transición:** permite implantar el producto en el entorno del cliente. Al finalizar esta fase se obtiene el producto de *software* final libre de errores y con toda su documentación asociada.

Para el desarrollo del subsistema de visualización, se definió el uso de *RUP* como metodología porque permite de forma disciplinada asignar tareas y responsabilidades en una empresa de desarrollo (quién hace qué, cuándo y cómo). "Además es el proceso de desarrollo más general de los existentes actualmente, es adaptable a la mayoría de los proyectos de desarrollo del *software*; característica que lo ha convertido en uno de los más utilizados actualmente" (Raquel, 2012). *RUP* se desarrolla mediante

iteraciones, comenzando por los CU[7] relevantes desde el punto de vista de la arquitectura. El modelo de arquitectura se representa a través de vistas en las que se incluyen los diagramas de *UML*. Facilita el desarrollo del proyecto porque divide el trabajo en partes más pequeñas o miniproyectos. Hace que la programación sea más sencilla. Garantiza la reutilización de componentes. Provee al equipo de la documentación detallada de todo el proceso de desarrollo. Además, al ser una metodología iterativa e incremental permite realizar de forma organizada, refinamientos sucesivos del producto de *software*, por tanto facilita la continuidad de la investigación realizada.

1.5. Lenguaje y herramientas de Modelado

Uno de los resultados esperados de la presente investigación es la documentación asociada a la misma. Esta incluye, entre otros elementos, diagramas que facilitarán el entendimiento del sistema por el personal encargado de proporcionarle mantenimiento o desarrollar nuevas funcionalidades. Por estos motivos es importante la definición de un estándar para la representación de los mismos.

La notación gráfica ayuda al desarrollo de *software* visualizando el problema sin recurrir de forma prematura a la implementación. El uso del modelado, que es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar parte de un diseño de *software* orientado a objetos. Muchas organizaciones los usan en combinación de una metodología de desarrollo para avanzar de una especificación inicial a un plan de implementación para todo el equipo de desarrolladores. Luego de la conclusión de la metodología a utilizar no cabe duda que el lenguaje de modelado por el que se va a guiar el desarrollo del *software* es el Lenguaje Unificado de Modelado (*UML* por sus siglas en ingles).

1.5.1. UML

Se utiliza Lenguaje Unificado de Modelado (*UML* por sus siglas en ingles) en su versión 2.0 ya que permite especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimientos sobre los sistemas que se deben construir. Se usa para entender, diseñar, configurar, mantener, y controlar la información sobre tales sistemas. Admite realizar una verificación y validación del modelo realizado. *UML* se puede usar para modelar distintos tipos de sistemas: sistemas de *software*, sistemas de *hardware*, y organizaciones del mundo real. También ofrece nueve diagramas en

[7] Casos de Uso.

los cuales modelar sistemas. Soporta automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos), lo cual garantiza que el modelo y el código estén actualizados, manteniendo la visión en el diseño de más alto nivel, de la estructura de un proyecto. Además, este lenguaje de modelado es el que utiliza la metodología seleccionada para la solución.

1.6. Herramienta CASE

Para el desarrollo de aplicaciones y la automatización de *software* son utilizadas las herramientas CASE (*Computer Aided Software Engineering*, Ingeniería de *Software* Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de *software* reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del *software* en tareas como el proceso de realizar un diseño del proyecto, dado el diseño se puede obtener automáticamente la implementación de parte del código, compilación automática, documentación o detección de errores entre otras.

Una herramienta CASE se utiliza para ayudar a las actividades del proceso de *software* que es utilizado para diseñar y para implementar otro *software*. Facilitan mejoras en la calidad y productividad del diseño y desarrollo. Por estas razones se ajusta perfectamente a los requisitos de la investigación.

1.6.1. Visual Paradigm

Visual Paradigm (VP) para *UML* es una herramienta CASE profesional que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El *software* de modelado *UML* ayuda a una más rápida construcción de aplicaciones de calidad. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta *UML* CASE también proporciona abundantes tutoriales de *UML*, demostraciones interactivas de *UML* y proyectos *UML*.

Las principales características que presenta el VP es que soporta *UML* versión 2.0, genera diagramas de procesos de negocio - proceso, decisión, actor de negocio, documento. Permite un modelado colaborativo

con CVS[8] y subversión, interoperabilidad con modelos *UML*2 (meta modelos *UML* 2.x para plataforma Eclipse) a través de XMI. Proporciona además una ingeniería inversa - código a modelo, código a diagrama, ingeniería inversa Java, C++, Esquemas XML[9], XML.NET exe/dll, CORBA[10] IDL y la generación de código - modelo a código, diagrama a código. (Rivas, 2012)

También posee un editor de detalles de casos de uso - Entorno todo-en-uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso. Diagramas EJB[11] - visualización de sistemas EJB. Generación de código y despliegue de EJB´s - generación de beans para el desarrollo y despliegue de aplicaciones. Diagramas de flujo de datos.

También incluye otras herramientas y *plugins* de modelado *UML* como es la plataforma Java (Windows/Linux/Mac OS X): SDE para Eclipse, SDE para NetBeans, SDE para Sun ONE, SDE para Oracle JDeveloper, SDE para JBuilder, SDE para IntelliJ IDEA, SDE para WebLogic Workshop, Y para la Plataforma Windows: SDE para Microsoft Visual Studio. (Rivas, 2012)

Se propone el uso de Visual Paradigm *UML* 2.0 en su versión 8.0 como herramienta CASE porque tiene *UML* 2.1 habilitado y este es un estándar ampliamente utilizado para el modelado del *software*. VP para *UML* es un producto que facilita a las organizaciones la diagramación visual y el diseño de sus proyectos. Proporciona código y es compatible con hasta diez lenguajes de programación. Brinda una alta interoperabilidad, además de ser multiplataforma.

1.7. Lenguaje de programación

Los lenguajes de programación son el vínculo de comunicación entre el hombre y la computadora. Un lenguaje de programación es una técnica estándar de comunicación que permite expresar las instrucciones que han de ser ejecutadas en una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen un programa informático.

---

[8] *Concurrent Versioning System.*

[9] *e**X**tensible **M**arkup **L**anguage*.

[10] *Common Object Request Broker Architecture.*

[11] *Enterprise Java Beans.*

1.7.1. Lenguaje de programación web

En la actualidad existen diferentes lenguajes de programación para desarrollar en la web, estos han ido surgiendo debido a las tendencias y necesidades de las plataformas. Desde los inicios de Internet, fueron brotando diferentes demandas por los usuarios y se dieron soluciones mediante lenguajes estáticos. A medida que pasó el tiempo, las tecnologías fueron desarrollándose y surgieron nuevos problemas a dar salida. Esto dio lugar a desarrollar lenguajes de programación dinámicos para la web, que permitieran interactuar con los usuarios y utilizaran Sistemas de Base de Datos.

1.7.2. HTML5 HyperText Markup Language

HTML, siglas de *HyperText Markup Language* (Lenguaje de Marcado de Hipertexto), es el lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML actualmente ha venido marcando un acelerado desarrollo teniendo como resultado el desarrollo de HTML 5 el cual nació en 2004, cuando se fundó el grupo de trabajo que con miembros de *Apple*, la Fundación *Mozilla* y *Opera Software*. Dos años después consiguieron uno de sus principales apoyos, W3C (*World Wide Web Consortium*). (Van Lancker, 2012.)

HTML5 es un lenguaje diseñado para organizar contenido web. Tiene por objeto facilitar el diseño y el desarrollo web, mediante la creación de una IU estandarizada e intuitiva para lenguaje de marcación. Proporciona además los medios para diseccionar y compartimentar sus páginas, y le permite crear componentes discretos que no sólo están diseñados para organizar su sitio lógicamente, sino también para darle a su sitio capacidades de sindicación. El HTML5 podría llamarse el “enfoque de correlación de información al diseño de sitios web" porque incorpora la esencia de la correlación de la información, dividiendo y etiquetando la información para hacerla fácil de entender y de utilizar. Este es el fundamento de la dramática utilidad semántica y estética del HTML5. (Van Lancker, 2012.) El HTML5 da a diseñadores y desarrolladores de todos los niveles la capacidad para publicar cualquier cosa al mundo, desde simple contenido de texto, hasta rica en interactiva multimedia.

El HTML5 ofrece herramientas para la administración efectiva de datos, dibujo, video y audio. Facilita el desarrollo de aplicaciones para diferentes navegadores para la web, así como para dispositivos portátiles. HTML5 es una de las tecnologías que está impulsando los avances de los servicios de computación móvil en nube, gracias a que permite mayor flexibilidad, permitiendo así el desarrollo de sitios web

emocionantes e interactivos. También introduce nuevas etiquetas y mejoras, incluyendo una elegante estructura, controles de formulario, APIs, multimedia, soporte de base de datos, y una velocidad de procesamiento significativamente más rápida.

El HTML5 crea una experiencia de usuario más atractiva: las páginas diseñadas utilizando HTML5 pueden proporcionar una experiencia similar a la de las aplicaciones de escritorio. El HTML5 también ofrece un desarrollo mejorado de múltiples plataformas al combinar la capacidad de las APIs con la ubicuidad del navegador. Usando HTML5 los desarrolladores pueden ofrecer una experiencia de aplicación moderna que cruza plataformas sin problemas.

HTML5 proporcionan nuevas etiquetas, nuevas metodologías y una infraestructura de desarrollo general que descansa en la interacción del HTML5 y sus dos contrapartes, CSS3 y JavaScript. Esta es la esencia del fenómeno de procesamiento de aplicaciones centrado en el cliente. Además de las muchas implementaciones de escritorio de las tecnologías y métodos de la tecnología HTML5, el TML5 puede implementarse en navegadores web de teléfonos móviles ricos en recursos—un mercado en crecimiento, como se ha visto con la popularidad y omnipresencia del *Apple iPhone*, *Google Android* y los teléfonos que operan el *Palm WebOS*. (Van Lancker, 2012.)

1.7.3. CSS3

Damián De Luca (2010) menciona las hojas de estilo en cascada (Cascading Style Sheets o CSS) son las que nos ofrecen la posibilidad de definir las reglas y estilos de representación en diferentes dispositivos, ya sean pantallas de equipos de escritorio, portátiles, móviles, impresoras u otros dispositivos capaces de mostrar contenidos web. Las hojas de estilo nos permiten definir de manera eficiente la representación de nuestras páginas y es uno de los conocimientos fundamentales que todo diseñador web debe manejar a la perfección para realizar su trabajo. La primera versión de CSS fue publicada a fines del año 1996 y fue logrando popularidad y aceptación hasta llegar a la versión 2.1, estándar actual que ofrece gran compatibilidad con la mayoría de los navegadores del mercado.

A partir del año 2005 se comenzó a definir el sucesor de esta versión, al cual se lo conoce como CSS3 o *Cascading Style Sheets Level 3*. Actualmente en definición, esta versión nos ofrece una gran variedad de opciones muy importantes para las necesidades del diseño web actual. Desde opciones de sombreado y redondeado, hasta funciones avanzadas de movimiento y transformación, CSS3 es el estándar que dominará la web por los siguientes años. El usar el CSS es como darle vida a nuestra aplicación, ya que si solo usamos lo que es HTML y JavaScript no tendrá un diseño agradable para el usuario.

1.7.4. JavaScript

JavaScript es un lenguaje que está basado en objetos. No es, como Java, un lenguaje de programación orientado a objetos (OOP). JavaScript emplea clases y herencia, típicas de la OOP lo que no de la misma forma que otros lenguajes de programación. No es necesario declarar los tipos de variables que van a utilizarse. Las referencias a objetos se comprueban en tiempo de ejecución, por lo tanto no se compila. La ventaja que presenta JavaScript sobre el HTML es que permite crear páginas más dinámicas, lo que las hace más atractivas para el usuario. Es prerrequisito indispensable saber HTML para utilizar y dominar el JavaScript.

Con JavaScript se pueden utilizar varias librerías para facilitar el trabajo de los desarrolladores, tanto para el diseño como para crear funcionalidades. Para obtener un diseño más dinámico y agradable para el usuario se propone el uso de la librería jQuery Mobile es un framework JavaScript, pensado para crear aplicaciones web para móviles. El framework permite crear fácilmente aplicaciones web móviles que puedan ser accedidas de varias plataformas de dispositivos móviles dispone de diversas herramientas CSS, también de manera automática.

Creado sobre jQuery con arquitectura de *jQueryUI*: los propios creadores de jQuery usaron su experiencia para desarrollar el framework para móviles y además implementaron la arquitectura diseñada para las librerías de interfaces de usuario *jQueryUI.* Está desarrollado para trabajar con HTML5 para aprovechar todas las características del framework. (Colomila Pardo, 2011)

Preparado para dispositivos táctiles: los dispositivos táctiles tienen cambios en la gestión de eventos y jQuery Mobile nos facilita la labor de adaptarnos a ellos.

1.8. Bibliotecas

Para la visualización de los mapas se desarrolló un subsistema de visualización de información georreferenciada utilizando dos bibliotecas de clases en JavaScript. Estas dos bibliotecas fueron JQuery Mobile en su versión 1.0.1 min y 1.6.4 min y OpenLayers 2.12 para móviles.

El marco de trabajo de la librería jQuery Mobile incluye todos los componentes de la interfaz del usuario necesarios para desarrollar aplicaciones móviles y sitios web móviles completos: páginas, diálogos, barras de herramientas, diferentes tipos de listas de elementos, una variedad de elementos de formato y botones y demás. JQuery Mobile se desarrolla por encima del centro de jQuery, entonces usted tiene acceso a las instalaciones claves, incluidas: acción y manipulación del modelo del objeto del documento HTML y XML

(DOM); manipulación de eventos; comunicación con el servidor a través de Ajax; y animación y efectos de imagen para las páginas web. Por su parte OpenLayers es una biblioteca de clases destinada a la visualización y manipulación de información geoespacial en web. Esta biblioteca es OpenSource y tiene una amplia comunidad de desarrolladores y productos dependientes lo que garantiza su desarrollo.

1.8.1. OpenLayers-2.12

Es un conjunto de librerías escritas en varios lenguajes entre los que sobresalen JavaScript, XML, CSS los cuales permiten tener un producto de calidad. OpenLayers hace que las páginas web sean dinámicas con el objeto de mostrar una interfaz atractiva y novedosa para el usuario, entre las principales características se puede mencionar las siguientes:

- Facilidad de uso.
- Facilidad al momento de llamar al servidor de mapas ya sea de tipo local o algún servidor externo como google map, yahoo map entre otros. (Campoverde Rivera, 2010)

Permite:

- ✓ La navegación de un mapa dentro una página web/wap.
- ✓ Visualizar hasta un punto mínimo y máximo del mapa, mediante la manipulación del zoom.
- ✓ Añadir iconos dentro de un mapa para un mayor entendimiento y fácil Localización o ubicación.
- ✓ Utilizar capas de tipo shp que están dentro de archivos (.map) y la trata con si fuesen marcas.
- ✓ Brinda facilidad de utilizar varios objetos o capas de diferentes servidores y los muestra en una sola aplicación.

1.8.2. JQueryMobile

Es un Framework JavaScript para el desarrollo rápido y fácil de sitios webs optimizados para teléfonos móviles. Con este framework, aceleramos la velocidad de desarrollo de aplicaciones, encapsulando muchas tareas comunes que se realizan cuando usamos el lenguaje JavaScript. Agrega una capa más a JQuery e intenta suplir algunas necesidades que los programadores de dispositivos móviles padecen.

En el pasado, un desarrollador tenía que programar según para qué dispositivo concreto, lo que alargaba los tiempos de desarrollo y mantenimiento de los sitios webs. Ahora con JQueryMobile, evitamos conocer la lógica específica de cada dispositivo y nos centramos en la programación para un solo fin, el navegador de un teléfono móvil.

JQueryMobile, es un framework bastante joven, desde el 13 de Agosto de 2010 (Colomila Pardo, 2011) aunque promete bastante como framework de desarrollo para web para móviles. El marco de trabajo de la librería jQuery Mobile incluye todos los componentes de la interfaz del usuario necesarios para desarrollar aplicaciones móviles y sitios webs móviles completos: páginas, diálogos, barras de herramientas, diferentes tipos de listas de elementos, una variedad de elementos de formato y botones y demás. JQuery Mobile se desarrolla por encima del centro de jQuery, entonces usted tiene acceso a las instalaciones claves, incluidas: acción y manipulación del modelo del objeto del documento HTML y XML (DOM); manipulación de eventos; comunicación con el servidor a través de Ajax; y animación y efectos de imagen para las páginas web.

El autor decide utilizar esta framework pues, permite el uso de temas personalizados y da la posibilidad de crear nuevos temas y trabajar con ellos. Este framework comprimido es de tamaño reducido solo pesa 12 K. Destaca la facilidad para el desarrollo de interfaces de usuario de dispositivos móviles. Es compatible en múltiples plataformas*: IOS, Android, Blackberry, Palm WebOS, Symbian, Windows Mobile*. (Colomila Pardo, 2011).Soporte HTML5 y las nuevas etiquetas HTML5.Para la implementación del subsistema se propone el uso de JQuery Mobile en su versión 1.0.1 min y 1.6.4 min.

1.9. Entorno de Desarrollo Integrado (IDE)

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. Los IDEs proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, C#, Delphi, Visual Basic, etc. Es posible que un mismo IDE pueda funcionar con varios lenguajes de programación. Este es el caso de Eclipse, al que mediante plugins se le puede añadir soporte de lenguajes adicionales.

1.9.1. IDE Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Eclipse fue desarrollado originalmente por IBM Canadá como el sucesor de su familia de herramientas para VisualAge. Actualmente es desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de Código abierto y un conjunto de

productos complementarios, capacidades y servicios. En noviembre del 2001, se formó un consorcio para el desarrollo futuro de Eclipse como Código abierto. En 2003, la fundación independiente de IBM fue creada.

Esta plataforma, típicamente ha sido usada para desarrollar entornos de desarrollo integrados (del inglés IDE), como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se entrega como parte de Eclipse (y que son usados también para desarrollar el mismo Eclipse). Sin embargo, también se puede usar para otros tipos de aplicaciones cliente, como BitTorrent Azureus.

1.10.Servidor de mapas

Los servidores de mapas permiten a los usuarios interactuar con la información geográfica. Las primeras versiones de los servidores de mapas sólo permitían realizar funciones básicas de visualización y consultas alfanuméricas simples. En las versiones recientes es posible realizar funciones mucho más avanzadas. El servidor de mapas es personalizable, es decir, se pueden preparar o programar las herramientas de manera que sean intuitivas para el usuario no experto en SIG.

1.10.1. MapServer

Es una plataforma de código abierto para la publicación de datos espaciales y aplicaciones de cartografía interactiva para la web.

Su funcionamiento básico está configurado en un fichero de texto, que generalmente tiene la extensión ".map". En este fichero, los datos del mapa se organizan en capas, a su vez dividida en una o más clases, donde en cada una de las cuales se pueden definir diferentes estilos visuales. Esta estructura permite la generación de mapas con una definición de estilos muy flexible, que también puede depender de la escala del mapa. El formato salida de MapServer, dependiendo de la solicitud, puede ser gráfico (mapa, leyenda, escala, métricas, visión general) o alfanumérico (el resultado de una consulta de datos alfanuméricos o espacial). El archivo ".map" también incluye la posibilidad de fusionar la producción de una plantilla de HTML MapServer, para generar una página web de lectura fácil y agradable.

Simbología avanzada, los múltiples lenguajes de scripting (PHP, Python, Perl, Ruby, Java, C#) y la ejecución multiplataforma (Linux, Windows, MacOS X, Solaris, etc.)

Admite múltiples formatos de datos vectoriales (ESRI shapfiles, PostGIS, ESRI ArcSDE, Oracle Spatial, MySQL y otros a través de OGR) y Raster (TIFF/GeoTIFF, EPPL7, y otros a través de GDAL).

Soporta más de 1000 proyecciones diferentes "al vuelo" a través de la librería Proj.4. Se propone el uso de la versión 2.6.

1.11.Framework

En el desarrollo de *software*, un Framework es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de *software* concretos, con base en la cual otro proyecto de *software* puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de *software* que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

Actualmente existen algunas soluciones disponibles en el mercado para el desarrollo de aplicaciones multiplataforma, capaces de ejecutar en *Android, iOS, BlackBerry*, entre otros. El principal problema de este tipo de operaciones son la incompatibilidad de *hardware* y los lenguajes de programación que cada uno utiliza para el desarrollo de sus aplicaciones. Existen algunas alternativas para lograr que un solo esfuerzo de desarrollo permita ser portado a los distintos sistemas operativos para dispositivos móviles existentes. Entre las tecnologías actuales para esto tenemos:

1.11.1. Appcelerator Titanium

Es un lenguaje basado en el modelo vista controlador MVC, hace uso de las habilidades de las tecnologías web como son HTML, JavaScript, css, python, ruby y php. Permitiendo el desarrollo de aplicaciones para dispositivos móviles, *tablets* y aplicaciones de escritorio de una manera nativa. *Titanium* tiene su propio API lo que le permite desarrollar cualquier tipo de aplicaciones para este tipo de equipos, el resultado será un código generado nativamente en Objetive-C o en Java dependiendo del dispositivo y sistema operativo que se quiera utilizar.

1.11.2. Ansca Corona

Esta plataforma integra HTML5 con *OpenGL* para crear aplicaciones enriquecidas principalmente utilizadas para el desarrollo de video juegos para dispositivos móviles, es la principal competencias de *Titanium*, también tienen su propio API para hacer uso de los recursos de los sistemas operativos, para al final obtener una aplicación nativa para *iPhone o Android*.

1.11.3. PhoneGap

Es otra de las plataformas de las que hace uso HTML5 y CSS3 programando en JavaScript para el desarrollo de las aplicaciones, permite el acceso a ciertas características nativas para desplegar en algunos sistemas operativos, es bastante extensible a través de plugins y soportar algunos de los sistemas operativos para dispositivos móviles como: *Android, iOS, Windows Phone, BlackBerry, WebOS, Symbian* y *Bada*. Gracias a los plugins de JavaScript JQuery Mobile y Sencha Mobile han podido integrar páginas web para móviles dentro de aplicaciones nativas.

Como se ha podido apreciar existen algunas plataformas y tecnologías que permiten desarrollar aplicaciones nativas para los distintos sistemas operativos para dispositivos móviles que existen actualmente, estos hacen uso de APIs propios y de pseudolenguajes en JavaScript para el desarrollo de sus aplicaciones.

Para portar esta aplicación a otros sistemas operativos lo más conveniente sería utilizar PhoneGap para desarrollar los servicios como páginas web móviles así poder desplegarlas en el navegador de cualquier dispositivo y cualquier sistema operativo. Permitiendo así explotar de una mejor manera los recursos disponibles dentro del teléfono, como el GPS y los recursos visuales para la construcción de la interfaz gráfica, siendo el sistema operativo de mayor crecimiento y el que lidera el mercado mundial de los dispositivos móviles.

1.12.Servidor web

Los servidores web proporcionan un servicio al cliente y devuelven los resultados. La plataforma computacional asociada con los servidores es más poderosa que la de los clientes. Por esta razón se utilizan PCs poderosas, estaciones de trabajo, minicomputadores o sistemas grandes. Además deben manejar servicios como administración de la red, mensajes, control y administración de la entrada al sistema ("login"), auditoría y recuperación y contabilidad.

Ejemplo de aplicaciones servidoras:

- Apache Web Server
- FTP Server-U
- Samba Server
- Microsoft Exchange Server

Para que los clientes y los servidores puedan comunicarse se requiere una infraestructura de comunicaciones o protocolo de comunicación, la cual proporciona los mecanismos básicos de direccionamiento y transporte.

Se denomina protocolo al conjunto de reglas que controlan la secuencia de mensajes que ocurren durante una comunicación entre entidades que forman una red. En este contexto, las entidades de las cuales se habla son programas de computadora o autómatas de otro tipo, tales como dispositivos electrónicos capaces de interactuar en una red.

1.13. Apache

Apache es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos (HTTP 1.1). Apache HTTP es de tecnología *Open Source*[12] sólido y para uso comercial desarrollado por la Apache *Software* Foundation (Rosen, 2004). Se propone el uso de este servidor web en su versión 2.0.6 porque es personalizable, la arquitectura modular de Apache permite construir un servidor hecho a la medida.

En cuanto a la administración los archivos de configuración de Apache están en ASCII, por lo que tiene un formato simple, y pueden ser editados tan solo con un editor de texto. Estos son transferibles, lo que permite la clonación efectiva de un servidor. El servidor puede ser administrado vía línea de comandos, lo que hace la administración remota muy conveniente.

Por otra parte se trata de un servidor muy eficiente. Mucho esfuerzo se ha puesto en optimizar el rendimiento del código “C” de Apache. Como resultado, este corre rápido y consume menos recursos de sistema en comparación a otros servidores. Además, Apache corre en una amplia variedad de sistemas operativos, incluyendo varias versiones de UNIX, Windows9x/NT, MacOS (Sobre Power PC), y varios otros.

El soporte de Apache es provisto por “*The Apache Group*” o “La Fundación Apache”, una gran cantidad de usuarios muy dedicados a su comunidad, así como compañías que ofrecen versiones pagadas de Apache (Rosen, 2004).

1.14. Estructura del uso de las herramientas y tecnologías utilizadas

[12] código abierto.

Una vez definido las diferentes herramientas y tecnologías a utilizar, para el desarrollo de del subsistema de visualización de información georreferenciada del Sistema Control de Flotas para dispositivos móviles. Se muestra una estructura de su utilización.

Figura 2 Herramientas y bibliotecas utilizadas en SVIGSPM.

La estructura del uso de las diferentes bibliotecas y herramientas utilizadas:

- ✓ **Interfaz**

  La interfaz está concebida con la biblioteca JQuery *Mobile* es un producto que está basado en el propio framework JavaScript jQuery esta biblioteca incluye todos los componentes de la interfaz del usuario necesarios para desarrollar la aplicación web. Otro rasgo característico es su compatibilidad tanto con dispositivos de alta gama como con dispositivos con menor capacidad, como aquellos que no son compatibles con JavaScript. Las mejoras progresivas, que es una filosofía de diseño, consisten en los siguientes principios centrales:

  - Todos los contenidos básicos deben ser accesibles para todos los navegadores.
  - Toda funcionalidad básica debe ser accesible para todos los navegadores.
  - El esquema mejorado es proporcionado por CSS vinculados externamente.
  - El esquema mejorado es proporcionado por JavaScript vinculado externamente.
  - Se respetan las preferencias del navegador del usuario final.

Además proporciona soporte a una gran cantidad de dispositivos móviles. JQuery *Mobile* clasifica o agrupa la compatibilidad de los dispositivos en tres categorías basadas en su nivel de compatibilidad:

Grado A: dispositivos con compatibilidad para una experiencia totalmente mejorada con transiciones de páginas animadas basadas en Ajax. JQuery *Mobile* es compatible con más de 20 dispositivos diferentes, incluidos: *iOS 3.2-5.0; Android 2.1-2.3 y 3.0; BlackBerry 6-7 y Playbook; Skyfire 4.1; Opera Mobile*; y los navegadores de escritorio *Chrome, Firefox, Internet Explorer y Opera*.

Grado B: dispositivos compatibles para una experiencia mejorada, pero sin características de navegación de Ajax. Los dispositivos compatibles incluyen: *BlackBerry 5.0, Opera Mini 5.0-6.5, y Nokia Symbian ^3.*

Grado C: dispositivos compatibles con una experiencia HTML no mejorada. Los dispositivos compatibles incluyen: *Smartphone*[13], incluido *BlackBerry 4.x*, *Windows Mobile* y otros.

- ✓ **Visor de mapas**

La utilización en conjunto de JQuery *Mobile* con la biblioteca OpenLayers 2.12, es un conjunto de clases destinada a la visualización y manipulación de información geoespacial en web. Esta biblioteca es *OpenSource* y tiene una amplia comunidad de desarrolladores y productos dependientes lo que garantiza su desarrollo.

- ✓ **Lenguaje de programación**

La utilización del lenguaje Java Script, CSS3, HTML5 en conjunto con PhoneGap que va ser el puente de comunicación entre código Java script y el *hardware* del dispositivo móvil (Ver *Hardware* y Sistemas operativos) dando acceso a través de APIS que están sustentadas bajo los estándares de HTML5 de la W3C.

---

13 Conocidos también como teléfonos inteligentes, son teléfonos móviles con alta capacidad de procesamiento. Presentan características que originalmente no estaban asociadas con los teléfonos: sistema operativo, navegador web, la habilidad de instalar y ejecutar aplicaciones de *software*, poder conectarse a redes inalámbricas, cámara digital y sensor *GPS* (*SearchMobileComputing*, 2007).

- ✓ ***Hardware* y sistemas operativos**

Las APIS son:

Acelerómetro: brinda acceso al acelerómetro del dispositivo si es que cuenta con él.

Cámara: brinda acceso a la aplicación de la cámara para tomar una foto u obtenerla de la galería.

Capturar: brinda acceso a aplicaciones de capturas de audio y video.

Brújula: esta API es útil para hacer verificación en cambio de la orientación del dispositivo, también depende del *hardware* del dispositivo.

Conexión: útil para trabajar con las conexiones de red que cuenta el dispositivo, desde redes WiFi, redes 3G, redes 4G entre otras.

Contactos: proporciona acceso a los contactos almacenados en el dispositivo.

Dispositivo: con esta se pueden obtener datos del dispositivo como el sistema operativo, el nombre y algunos otros datos relevantes.

Eventos: con esta APIS es posible manejar eventos de teclas físicas del dispositivo, además de manejar los diferentes eventos generados en el ciclo de vida de una aplicación.

File: su implementación facilita el acceso a los archivos del dispositivo, con esta API se puede crear, editar y leer archivos binarios.

Geolocalización: útil para obtener la posición geográfica del dispositivo, ya bien sea a través de redes o del GPS satelital si cuenta él cuenta el dispositivo con uno.

Media: proporciona acceso a reproductores multimedia como sonido y video.

Notificaciones: además de ser útil para crear cuadros de diálogos como alertas nativas del sistema, también brinda acceso al vibrador si el dispositivo lo posee.

Storage: facilita el uso de base de datos basadas en el estándar de W3C y el uso de local *Storage*.

- ✓ Sistemas operativos que son compatibles con el uso de cada uno de las bibliotecas framework antes mencionados.

- *Android*
- *iOS, Windows*
- *Phone*
- *BlackBerry OS*
- *Web OS*
- *Symbian, Bada.*

1.15. Sistema Gestor de Base de Datos (SGBD)

Los Sistemas de Gestión de Base de Datos (en inglés Database Management System, abreviado DBMS) son un tipo de *software* muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. El propósito general de los sistemas de gestión de base de datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante para una organización.

1.15.1. Base de datos espacial

El sistema de Base de datos espacial (spatial database) es un sistema administrador de base de datos que maneja datos existentes en un espacio o datos espaciales. En este tipo de base de datos es imprescindible establecer un cuadro de referencia (un SRE, Sistema de Referencia Espacial) para definir la localización y relación entre objetos, ya que los datos tratados en este tipo de base de datos tienen un valor relativo, no es un valor absoluto. Los sistemas de referencia espacial pueden ser de dos tipos: georrefenciados (aquellos que se establecen sobre la superficie terrestre. Son los que normalmente se utilizan, ya que es un dominio manipulable, perceptible y que sirve de referencia) y no georreferenciados (son sistemas que tienen valor físico, pero que pueden ser útiles en determinadas situaciones).

1.15.2. PostgreSQL

PostgreSQL es un SGBD relacional orientado a objetos de software libre, liberado bajo la licencia BSD[14]. Su desarrollo no es manejado por una sola compañía, como otros proyectos *Open Source*, es dirigido y desarrollado por la comunidad PGDG (PostgreSQL *Global Development Group*). Está caracterizado por su alta concurrencia, pues mediante su sistema denominado Acceso Concurrente Multi Version (MVCC) permite el acceso a las tablas que están bajo un proceso que se está ejecutando en ellas sin necesidad de bloqueos. Por otra parte posee una amplia variedad de tipos nativos, entre otros son los casos de:

[14] *Berkeley Software* Distribución, Distribución de *Software Berkeley*, que identifica un sistema operativo derivado del *Unix* y nacido a partir de las aportaciones realizadas a ese sistema por la Universidad de California en *Berkeley.*

.

- ✓ Texto de largo ilimitado.
- ✓ Números de precisión arbitraria.
- ✓ Figuras geométricas.
- ✓ Direcciones IP.
- ✓ Arrays.
- ✓ Otros tipos de datos creados por los usuarios (Ejemplo: GIS (Sistema de Información Geográfica).

PostGIS

PostGIS es un módulo que añade soporte de objetos geográficos al gestor objeto-relacional PostgreSQL, convirtiéndola en una base de datos espacial para su utilización en Sistema de Información Geográfica. Se publica bajo la licencia pública general de GNU. PostGIS ha sido desarrollado por la empresa canadiense Refraction Research, especializada en productos "*Open Source*". PostGIS es hoy en día un producto veterano que ha demostrado versión a versión su eficiencia. En relación con otros productos, PostGIS ha demostrado ser muy superior a la extensión geográfica de la nueva versión de MySQL, y a juicio de muchos, es muy similar a la versión geográfica de la archiconocida Oracle. Un aspecto que hay que tener en cuenta es que PostGIS ha sido certificado en el 2006 por el *Open Geospatial Consortium* (OGC) lo que garantiza la interoperabilidad con otros sistemas también interoperables. PostGIS almacena la información geográfica en una columna del tipo GEOMETRY. Se utiliza esta base de datos espacial porque presenta una gestión de datos centralizada. Fortaleza fundamental para el equipo de desarrollo que necesita trabajar con los datos organizados de esa forma. Permite una edición multiusuario. Admite el almacenamiento de los datos en una SGBD como PostgreSQL. Además de permitir el acceso remoto a datos y aplicaciones externas.

1.16.Conclusiones del capítulo

A lo largo de todo el capítulo se han caracterizado los conceptos asociados a los SCF en dispositivos móviles, los cuales ayudan a una mejor comprensión del objeto de estudio; al mismo tiempo se ha realizado un análisis detallado de los factores que influyen en la situación problemática. El análisis de las soluciones existentes y el estudio de las diferentes herramientas, tecnologías, lenguajes de programación y de modelado y metodología permitieron arribar a las siguientes conclusiones:

- ✓ El análisis de las soluciones existentes, permitió revelar que Android, iOS, Windows Phone, BlackBerry, WebOS, Symbian y Bada eran los SO más utilizados en dispositivos móviles por los SCF.
- ✓ El estudio de soluciones similares permitió identificar algunas de las herramientas y tecnologías que podían ser empleadas en la construcción de la solución como por ejemplo *PhoneGap*, *Titanium, Android*. Otro rasgo significativo es que se pudieron identificar un conjunto de funcionalidades básicas que debe poseer el subsistema: cargar el mapa desde el servidor de mapas o del dispositivo móvil, *zoom* más, *zoom* menos, paneo, recentrar, interactuar con el mapa. Además, se constató que debido a la necesidad de la investigación se deben incluir otras funcionalidades como por ejemplo, localizar la flota mediante el dispositivo móvil, mostrar datos del acelerómetro.
- ✓ El análisis de los diferentes framework de desarrollo multiplataforma orientado a dispositivos móviles, permitió definir la utilización de PhoneGap como framework de desarrollo y además se utilizaran las bibliotecas *JQuery Mobile* para la realización de la interfaces y *OpenLayers* para la visualización de los mapas.
- ✓ Se definió *RUP* como metodología de desarrollo de software empleando *UML* como lenguaje de modelado.
- ✓ Se definió utilizará *PostgreSQL* como *SGBD* junto con su extensión *PostGIS* para el almacenamiento de la información georreferenciada.

CAPÍTULO 2. **PRESENTACIÓN DE LA SOLUCIÓN PROPUESTA.**

2.1. Introducción

En el presente capítulo se describe la propuesta de solución en correspondencia con la metodología *RUP* y lenguaje modelado *UML*. Se muestra el modelo de dominio que identifica los conceptos significativos en el contexto del problema. Se enumeran y especifican los Requisitos Funcionales (RF) y los Requisitos No Funcionales (RNF) que debe cumplir el sistema, se identifican los actores y casos de uso del sistema para conformar el Diagrama de Casos de Uso del Sistema (DCUS). Además se realiza una descripción detallada de los casos de uso que forman parte de la solución.

2.2. Modelo de dominio en el desarrollo de s*oftware*

El Modelo de dominio es empleado fundamentalmente cuando los flujos de información son difusos, es decir, que tengan múltiples orígenes y cuando son solo eventos o sucesos. También la imposibilidad de realizar subsistemas en el proceso de desarrollo del *software* dado por las múltiples interconexiones, es uno de los factores que influyen en la decisión de realizar un modelo de este tipo. Influye además en esta decisión, la existencia del solapamiento de responsabilidades y la dificultad en el establecimiento de las reglas del funcionamiento del producto a implementar.

Un Modelo del dominio captura los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema, además de que no incluyen las responsabilidades de las personas que ejecutan las actividades. Según *RUP* este tipo de modelo representa un subconjunto del Modelo de Objeto del Negocio, lo cual no significa que siempre que haya un Modelo de dominio tenga que existir obligatoriamente un Modelo de Negocio. Por lo que el Modelo de dominio no es más que la representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés.

2.2.1. Modelo de dominio

En el actual subepígrafe se expone un análisis detallado de los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema.

Teniendo en cuenta que no se tienen bien definidos los procesos del negocio se realizará una modelación del dominio y se procederá a explicar cada uno de los conceptos que forman parte del mismo. Todo ello para tener una mejor comprensión de la estructura y dinámica de la organización.

Se muestra el modelo de dominio correspondiente al entorno donde estará el sistema teniendo en cuenta el proceso de representación de la información geográfica, conceptos, actividades, personas involucradas en ello y las reglas del negocio.

2.2.1. Diagrama de clases del Modelo de dominio

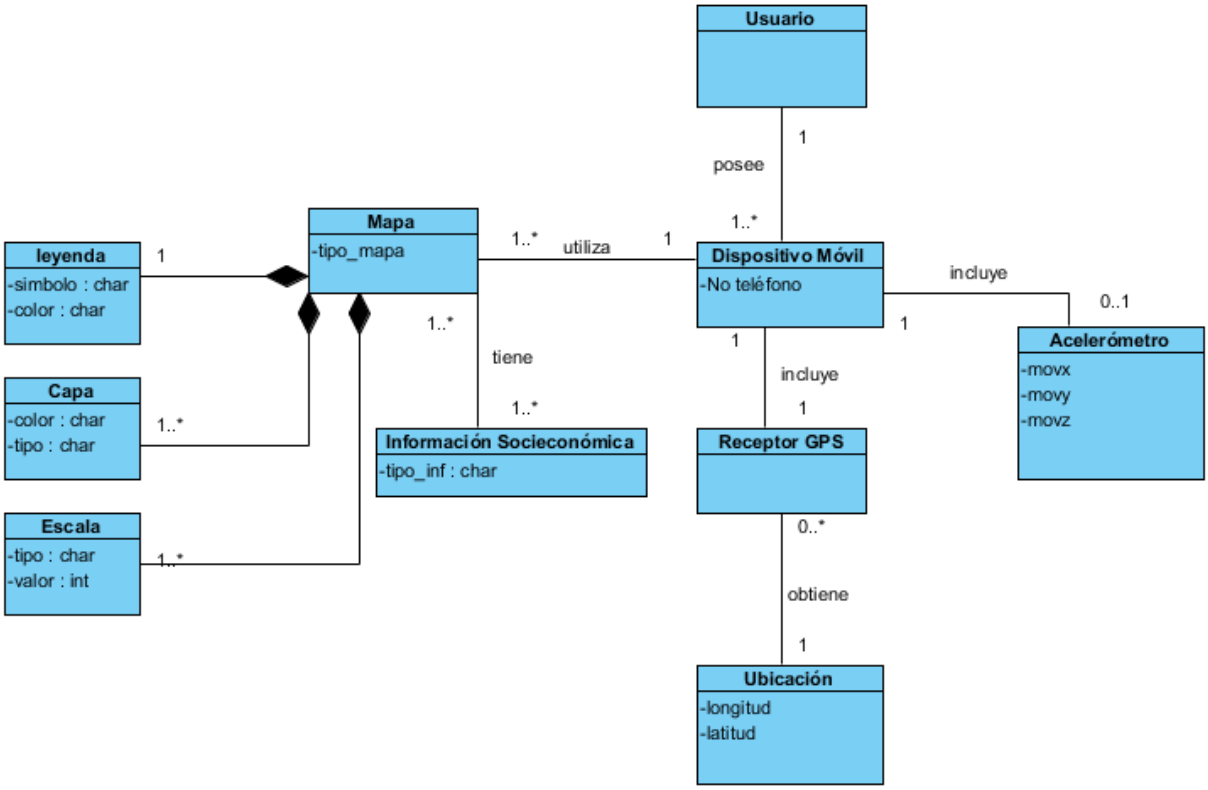


Figura 3 Diagrama de clases del Modelo de dominio de SVIGSPM.

A continuación se relacionan los términos que aparecen en el diagrama y se proporciona una pequeña descripción de los mismos.

Definición de las clases del modelo de dominio

Usuario

Sujeto que necesita conocer información sobre las rutas, paradas y puntos de interés más cercanos. Su interacción con el sistema se realiza al aire libre y generalmente en movimiento. Dispone de un dispositivo móvil con conectividad.

Dispositivo Móvil

Dispositivo desde donde se accederá al subsistema de visualización de información georreferenciada del Sistema control de Flotas del proyecto Aplicativo SIG.

Receptor GPS

Permite conocer con exactitud la ubicación del dispositivo móvil en un sistema de coordenadas WGS-84[15].

Acelerómetro

Permite medir la aceleración en el eje x, y, z.

Ubicación

Permite representar un punto físico sobre la superficie terrestre del mapa. Está compuesta por la latitud y la longitud en un sistema coordenadas WGS-84.

Mapa

Es una representación gráfica y métrica de una porción de territorio sobre una superficie bidimensional, generalmente plana, pero que puede ser también esférica como ocurre en los globos terráqueos. El que el mapa tenga propiedades métricas significa que ha de ser posible tomar medidas de distancia, ángulos o superficies sobre él y obtener un resultado aproximadamente exacto.

[15] **E**s un sistema de coordenadas geográficas mundial que permite localizar cualquier punto de la Tierra (sin necesitar otro de referencia) por medio de tres unidades dadas. WGS84 son las siglas en inglés de *World Geodetic System* 84 (que significa Sistema Geodésico Mundial 1984).

Escala

Relación entre la distancia que separa dos puntos en un mapa y la distancia real de esos dos puntos en la superficie terrestre. En los mapas, la escala puede expresarse de tres modos distintos: en forma de proporción o fracción, con una escala gráfica o con una expresión en palabras y cifras. Cuanto mayor es la escala, más se aproxima al tamaño real de los elementos de la superficie terrestre. Los mapas a pequeña escala generalmente representan grandes porciones de la Tierra y, por tanto, son menos detallados que los mapas realizados con escalas más grandes.
La relación matemática entre las dimensiones en el mapa, carta o plano y la superficie terrestre que representa. Por extensión puede referirse a la mayor o menor profundidad del enfoque en un tema geográfico.

Leyenda

Explicación de los símbolos, los colores, las tramas y los sombreados empleados en un mapa; suele encontrarse a pie de página o en un recuadro, situado en sus márgenes o bien en su dorso. Los símbolos empleados en los mapas pueden llegar a contener un gran volumen de información, que por su facilidad de lectura permiten una rápida interpretación.

Información Socioeconómica

Es un conjunto organizado de datos procesados referentes al aspecto social y económico de cualquier lugar de interés del país.

2.3. Requisitos

El proceso de construcción de un *software* está dado por disímiles actividades que han de ser monitoreadas por alguna guía metodológica, garantizando así la calidad del producto y la satisfacción de los clientes y desarrolladores al mismo tiempo. Técnicas y procedimientos que se lleva a cabo en la Ingeniería del *Software* de cada producto, arrojando así la documentación relacionada con cada uno de los flujos de actividades realizadas durante la construcción del sistema.
Una fase inicial y muy importante dentro del proceso de la Ingeniería de *Software*, es la Ingeniería de Requisitos, donde se realiza el proceso de descubrir, analizar, escribir y verificar los servicios y restricciones, del sistema de *software* que se desea producir; este proceso se realiza mediante la obtención, el análisis, la especificación, la validación y la administración de los requisitos del *software*.

Los requisitos del *software* son las características y cualidades que el sistema debe tener. Estos se dividen en dos grupos, los funcionales y los no funcionales. Una vez definidos e identificados los mismos se tiene la visión general de lo que se quiere hacer en el sistema. (Jacobson, 2000)

Para la obtención de los requisitos de los subsistemas se hizo necesario acudir a varias técnicas relacionadas con dicho proceso como es la entrevista a varios especialistas del departamento de Geoinformática del Centro GEYSED, departamento que se encarga del desarrollo de SIG para distintas empresas. Unido a esto se utilizó la técnica llamada *Lluvia de Ideas* así como la del *Enfoque Grupal*, donde cada uno de los integrantes del equipo de desarrollo expresa su punto de vista e ideas relacionadas con las funcionalidades que requiera el producto. Contando además con el previo estudio e investigación de *software* en los que de una forma u otra se desarrollan estos subsistemas de visualización de información georreferenciada.

Al lograr identificar los requisitos iniciales del *software* y realizar el análisis de los mismos mediante la técnica de verificación, donde se limaron las ambigüedades y la falta de consistencia, se arrojaron los requisitos funcionales y no funcionales que se presentan en los su epígrafes que se relacionan a continuación.

2.3.1. Requisitos Funcionales

Son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema de desarrollo y las restricciones en las que debe operar. Estas actividades no son exactamente los requisitos funcionales, pero si son el punto de partida para identificar qué debe hacer el sistema. Los requisitos funcionales no alteran la funcionalidad del producto, esto quiere decir que los requisitos funcionales se mantienen invariables sin importarle con qué propiedades o cualidades se relacionen.

RF1 Cargar mapa desde el móvil.

- ✓ Permite mostrar la imagen de un mapa previamente guardada en el dispositivo móvil, esta imagen no contiene ningún tipo de información georreferenciada.

RF2 Cargar mapa de servicio WMS[16].

[16] WMS (El servicio *Web Map Service* (WMS)) definido por el OGC (*Open Geospatial Consortium*) produce mapas de datos referenciados espacialmente, de forma dinámica a partir de información geográfica. Este estándar

- ✓ Permite mostrar un mapa conectándose a un servidor que soporte el servicio WMS, este mapa contiene información georreferenciada.

RF3 Guardar mapa en el móvil desde WMS.

- ✓ Permite guardar la imagen de un mapa en un dispositivo móvil desde el servidor de mapas, esta imagen del mapa a guardar no contiene ningún tipo de información georreferenciada.

RF4 Realizar zoom[17] más a un mapa.

- ✓ Permite aumentar la escala de un mapa, realizando una petición al servidor de mapas que soporta el servicio WMS.

RF5 Realizar zoom menos a un mapa.

Permite disminuir la escala de un mapa, realizando una petición al servidor de mapas que soporta el servicio WMS.

RF6 Realizar paneo hacia la derecha.

- ✓ Permite desplazarse hacia la derecha en el mapa, visualizando así la región deseada, sin modificar el tamaño del mapa.

RF7 Realizar paneo hacia la izquierda.

- ✓ Permite desplazarse hacia la izquierda en el mapa, visualizando así la región deseada, sin modificar el tamaño del mapa.

RF8 Realizar paneo hacia arriba.

- ✓ Permite desplazarse hacia arriba en el mapa, visualizando así la región deseada, sin modificar el tamaño del mapa.

RF9 Realizar paneo hacia abajo.

---

internacional define un "mapa" como una representación de la información geográfica en forma de un archivo de imagen digital conveniente para la exhibición en una pantalla de ordenador.

[17] Zoom (del inglés *zoom*): Teleobjetivo especial cuyo avance o retroceso permite acercar o alejar la imagen (RAE, 2001).

- ✓ Permite desplazarse hacia abajo en el mapa, visualizando así la región deseada, sin modificar el tamaño del mapa.

RF10 Habilitar capas del mapa.

- ✓ Permite habilitar varias capas en el mapa, solo si se está trabajando con el servidor de mapas.

RF11 Deshabilitar capas del mapa.

- ✓ Permite deshabilitar varias capas en el mapa, solo si se está trabajando con el servidor de mapas.

RF12 Recentrar el mapa.

- ✓ Permite calcular de acuerdo a la posición del cursor en el mapa la coordenada sobre la que este se encuentra y posiciona dicha coordenada en la parte inferior derecha.

RF13 Mostar características del móvil.

- ✓ Permite mostrar el SO[18] del dispositivo y su versión con el cual el usuario ha accedido al subsistema.

RF14 Localizar flota.

- ✓ Permite mostrar las coordenadas del dispositivo en el mapa además de mostrar la ubicación.

RF15 Mostar datos del acelerómetro

- ✓ Permite mostrar los niveles de aceleración de la flota y mostrar su estado (PANICO, NORMAL).

2.3.2. Requisitos No Funcionales

Son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Los requisitos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto. El subsistema podrá ser usado por personas con conocimientos básicos en el manejo de dispositivos móviles.

Usabilidad

[18] Sistema operativo.

Para ser utilizado el subsistema el usuario deben de tener en su poder un dispositivo móvil que cuente con los requisitos de *hardware* para soportarlo. El subsistema debe tanto dispositivos táctiles como no táctiles y trabajar con imágenes cargadas desde el móvil. Las funcionalidades principales del sistema estarán orientadas a iconos para un mayor reconocimiento por parte del usuario.

Interfaz

Contar con un botón que permita regresar a la interfaz anterior. El menú que sea creado en la página principal que permita desplazarse por los diferentes servicios que brinda. Contar con un botón que permita salir de la aplicación, además de poseer una interfaz nativa propia de los dispositivos móviles que sea operable desde las diferentes plataformas.

Eficiencia

El período de respuesta del subsistema implementado estará proporcionado por el cúmulo de información a procesar, entre mayor cantidad de información mayor será el período de procesamiento. En el caso del tiempo de respuesta será de la misma forma, la velocidad de procesamiento de la información, la actualización y la recuperación dependerán de la cantidad de información que tenga que procesar el subsistema.

Requisitos de *software*

La construcción de la aplicación funcionará bajo los conceptos de arquitectura cliente/servidor. Por tanto, los servidores tanto el de aplicaciones como el de mapas deben tener como requisitos mínimos de *software*:

Servidor de mapas:

- ✓ Tener instalado un servidor de mapas que soporte el servicio WMS.

Servidor web:

- ✓ Tener instalado un servidor web Apache.

Cliente:

- ✓ Debe tener conectividad y algunos de los siguientes sistemas operativos instalados: *Android, iOS, Windows Phone, BlackBerry, WebOS, Symbian y Bada*.

Requisitos de *hardware*

Servidor web:

- ✓ Capacidad mínima de 2GB de RAM y 40GB de disco duro.

Servidor de mapas:

- ✓ Capacidad mínima de 2GB de RAM y 40GB de disco duro.

Cliente:

- ✓ El tipo de dispositivo móvil debe ser smartphone.

Soporte

El equipo de desarrollo de la línea Aplicativos SIG es el encargado de corregir los errores que puedan surgir en el sistema.

2.4. Descripción del sistema

En el presente capítulo, luego de haber realizado la descripción de los requisitos, se le da paso a la siguiente etapa de conceptualización de un *software*, que es la descripción del Modelo de Casos de Uso del Sistema (DCUS), que contiene actores, casos de uso y sus respectivas relaciones.

Los requisitos del *software* se agrupan según sus características, utilidad y usabilidad conformando así los llamados casos de uso, los cuales no son más que fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores.

Estos casos de uso describen cómo se comporta y funciona un sistema específico, mediante un documento que narra de manera secuencial las acciones relevantes que realiza un actor en interacción con el sistema hasta completar un proceso, sin darle importancia a los detalles de la implementación.

Un actor no es parte del sistema en desarrollo, es un agente externo que interactúa con el mismo en pos de obtener un resultado esperado. El subsistema de visualización de información georreferenciada del SCF para dispositivos móviles cuenta con un actor el cual se especifican a continuación.

Actor	Descripción
Usuario	Son las personas que interactúan con el sistema.

Tabla 1 Descripción de los actores del sistema.

2.4.1. Diagrama de casos de uso del sistema

La solución al problema planteado de la presente investigación de los subsistemas de visualización de información georreferenciada cuenta con un DCUS, el cual se presenta a continuación. Este diagrama representa gráficamente a los procesos y su interacción con los actores.

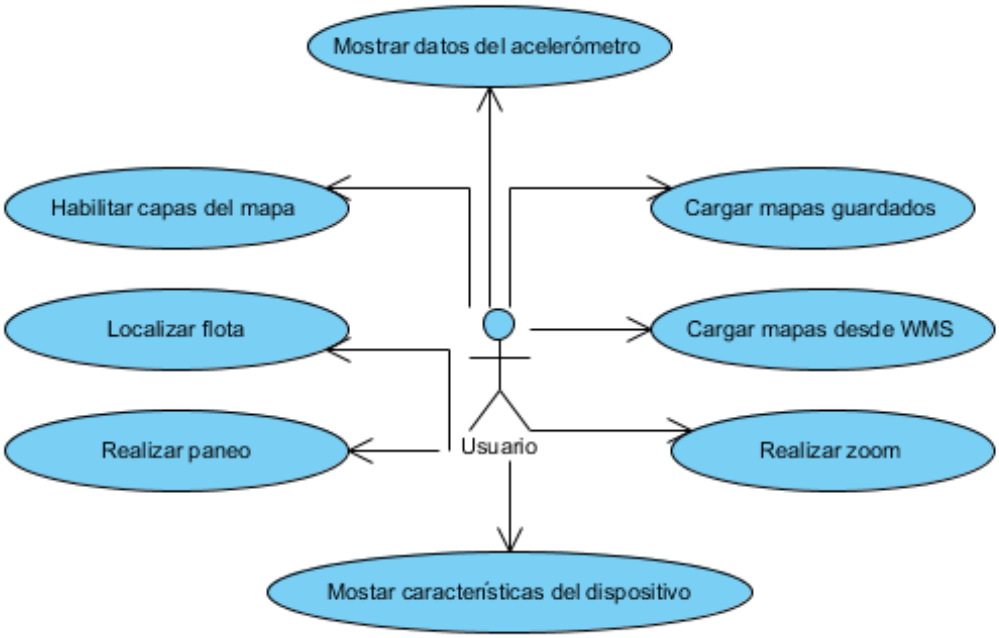


Figura 4 Diagrama de casos de uso del sistema.

2.4.2. Descripción textual de casos de uso del sistema

A continuación se describe el caso de uso Cargar mapa desde WMS. Las descripciones de los casos de uso restantes pueden consultarse en el documento SVIGSPM Modelo de Sistema.

Cargar mapa desde WMS

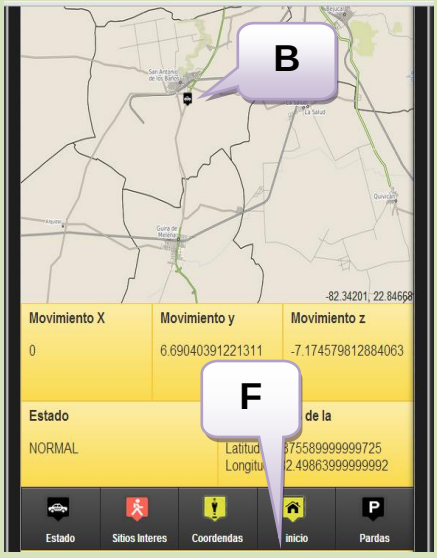
Caso de Uso:	Este caso de uso permite cargar un mapa de servicio WMS en un dispositivo móvil.

Actores:	Usuario
Propósito	Este caso de uso se lleva a cabo para que el usuario pueda visualizar la información georreferenciada asociada a un mapa mediante el servicio WMS.
Resumen:	Este caso de uso se inicia cuando el usuario accede a la Interfaz principal y selecciona la opción Cargar mapa.
Precondiciones:	Poseer un dispositivo móvil con conectividad.
Referencias	RF2, RF3, RF4, RF5, RF6, RF7, RF8, RF9, RF10, RF11, RF13.
Prioridad	Crítico

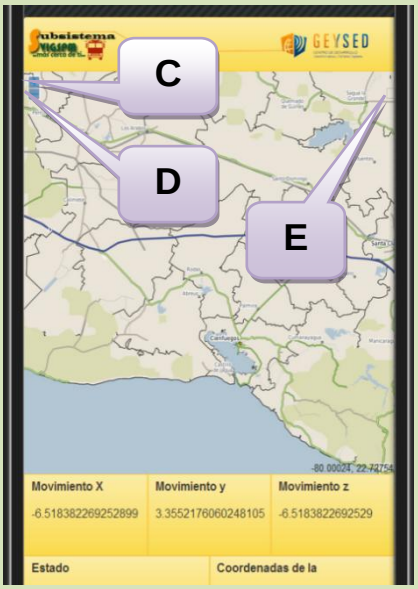
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción “Mapa”. **(Ver A interfaz 1.1)**	**2.** El sistema muestra la interfaz de “Mapa” muestra el mapa con la ubicación **(Ver B interfaz 1.2)** de la flota en la parte inferior con una serie de acciones a realizar.
3. El usuario elige la acción a realizar.	**4.** Si elige: **A.** La parte superior del mapa **(Ver interfaz 1.3)** ir a la Sección Mapa acciones**.** **B.** Las acciones sobre el mapa **(Ver interfaz 1.2)** y elige : a) Mapa ir a la sec*ción* “Mapa”. b) Guardar Mapa en el móvil ir a la sección “Guardar Mapa en el móvil”.

	6. Finaliza el caso de uso.
Sección "*Mapa acciones*"	
2. El usuario elige la acción a realizar.	**1.** El sistema muestra en la parte superior de la pantalla del móvil el mapa con una serie de acciones a realizar. a) + "Zoom más". **(C Ver interfaz 1.3)** b) - "Zoom menos". **(D Ver interfaz 1.3)** c) Habilitar capas. **(E Ver interfaz 1.3)**
	2. El sistema actualiza el mapa de acuerdo a las acciones realizadas por el usuario.
	3. Ir al flujo normal de los eventos (6).
Sección "Guardar Mapa en el móvil"	
	1. El sistema guarda la imagen proyectada en ese instante en el móvil. **(F Ver interfaz 1.2)** ***2.*** Ir al flujo normal de los eventos (6).
Flujo Alterno	
Postcondiciones	**-**
Prototipo de Interfaz	

Interfaz 1.1



Interfaz 1.2



Interfaz 1.3

Tabla 2 Descripción textual del casos de uso "Cargar mapa desde WMS"

2.5. Conclusiones del capítulo

En el presente capítulo se ha tratado de manera específica los artefactos generados en el flujo de trabajo del análisis correspondiente según lo planteado por *RUP* tales como:

- ✓ El modelo de dominio realizado esclareció los elementos que intervienen en el problema así como sus interacciones.
- ✓ Los requisitos funcionales y no funcionales definidos establecen las características del sistema que se implementará.
- ✓ Acorde a la metodología definida, se agruparon los requisitos funcionales en ocho casos de uso. Se describieron los casos de uso en el Modelo del Sistema, definiendo las respuestas que ha de proporcionar la aplicación ante las acciones del usuario.

CAPÍTULO 3. CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA

3.1. Introducción

Para alcanzar una exitosa implementación del subsistema de visualización de información georreferenciada del SCF para dispositivos móviles ; es necesario realizar un análisis en cuanto a la estructura que ha de tener, diseñando de forma explícita las relaciones existentes entre cada una de las entidades que forman parte del análisis. En el capítulo que comienza se propone como parte de la solución a la problemática existente, el Modelo de Diseño que se corresponde con el flujo de trabajo de Análisis y Diseño propuesto por *RUP*.

En el actual capítulo se ha de mostrar una vista general del sistema a construir, a partir del diagrama de paquetes organizado, así como algunos de los artefactos que forman parte del modelo de diseño en piezas manejables. Además de realizar cada uno de los diagramas de clases del diseño correspondientes a cada uno de los casos de uso con que cuenta la aplicación, teniendo en cuenta la arquitectura y los patrones de diseño definidos para la misma.

3.2. Arquitectura de *Software*

La Arquitectura de *Software* (AS) se vincula con el diseño, pues la AS es una forma de diseño de *software* que se manifiesta tempranamente en el proceso de creación de un sistema; pero este diseño ocurre a un nivel más abstracto que el de los algoritmos y las estructuras de datos. En el que muchos consideran un ensayo seminal de la disciplina, Mary Shaw y David Garlan “sugieren que dichas cuestiones estructurales incluyen organización a grandes rasgos y estructura global de control; protocolos para la comunicación, la sincronización y el acceso a datos; la asignación de funcionalidad a elementos del diseño; la distribución física; la composición de los elementos de diseño; escalabilidad y rendimiento; y selección entre alternativas de diseño” (Reynoso, 2009).

Se puede concluir que el concepto más completo de AS es el que brinda la IEEE cuando define como AS a la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución. IEEE 1471-2000 (Jaén, y otros, 2009) Una arquitectura define varios estilos arquitectónicos, entre los más conocidos se destacan:

- Arquitectura centrada en los datos.

- Arquitectura centrada en los flujos de datos.
- Arquitectura llamada y respuesta (call and return).
- Arquitectura Orientada a Objetos. - Arquitectura en capas.

3.2.1. Modelo cliente servidor

La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma.

Sommerville define el modelo arquitectónico cliente-servidor como un modelo de sistema en el que dicho sistema organiza como un conjunto de servicios y servidores asociados, más unos clientes que acceden y usan los servicios. (Sommerville, 2005) Los principales componentes de este modelo son:

- ✓ Un conjunto de servidores que ofrecen servicios a otros sistemas. Ejemplo de servidores de impresoras que ofrecen servicios de impresión, servidores de ficheros que ofrecen servicios de gestión de ficheros y servidores de compilación.
- ✓ Un conjunto de clientes que llaman a los servicios ofrecidos por los servidores. Éstos son normalmente subsistemas en sí mismos. Puede haber varias instancias de un programa cliente ejecutándose concurrentemente.
- ✓ Una red que permite a los clientes acceder a estos servidores. Esto no es estrictamente necesario ya que los clientes y los servidores podrían ejecutarse sobre una única máquina.

3.2.2. Modelo cliente-servidor en SVIGSPM

La arquitectura utilizada en la aplicación una arquitectura cliente- servidor tres capas o tres niveles.

Sommerville platea:

En esta arquitectura la presentación, el procesamiento de la aplicación y la gestión de los datos son procesos lógicamente separados que se ejecutan sobre procesadores diferentes. (Sommerville, 2005)Ver ***Figura 5***

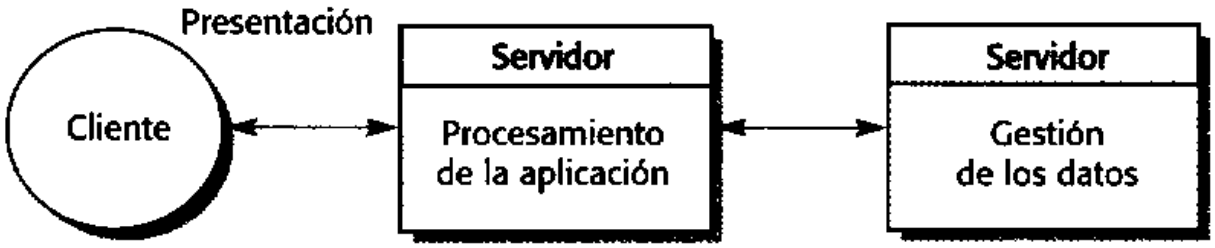


Figura 5 Modelo cliente/servidor tres capas. [19].

En la arquitectura en 3 niveles, existe un nivel intermediario. Esto significa que la arquitectura generalmente está compartida por:

- ✓ Un cliente, es decir, el equipo que solicita los recursos, equipado con una interfaz de usuario (generalmente un navegador web) para la presentación.
- ✓ El servidor de aplicaciones (también denominado *software* intermedio), cuya tarea es proporcionar los recursos solicitados, pero que requiere de otro servidor para hacerlo.
- ✓ El servidor de datos, que proporciona al servidor de aplicaciones los datos que requiere.

[19] (Sommerville, 2005)

En la **Figura 6** se muestra una representación de la arquitectura del subsistema.

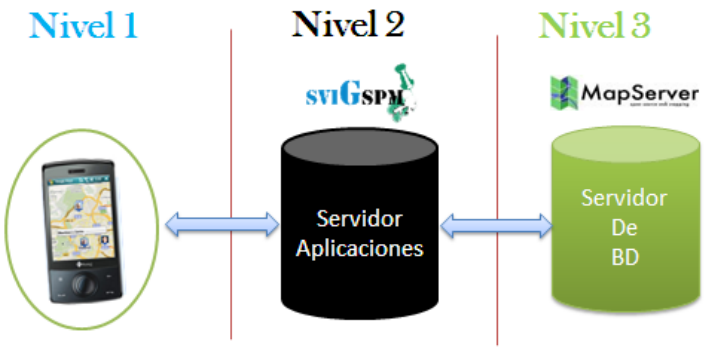


Figura 6 Modelo cliente/servidor en SVIGSPM

3.2.3. Estilo arquitectónico

Luego de un análisis de cada uno de los estilos arquitectónicos el autor decide la utilización del estilo llamada y retorno, y dentro del mismo la utilización del patrón Model-View-Controller (MVC).La principal aportación de este estilo arquitectónico es Soporte de vistas múltiples. Dado que la vista se halla separada del modelo y no hay dependencia directa del modelo con respecto a la vista, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos simultáneamente. Por ejemplo, múltiples páginas de una aplicación de web pueden utilizar el mismo modelo de objetos, mostrado de maneras diferentes. Adaptación al cambio. Los requerimientos de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas de negocios. Los usuarios pueden preferir distintas opciones de representación, o requerir soporte para nuevos dispositivos como teléfonos celulares. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación generalmente no afecta al modelo. Este patrón sentó las bases para especializaciones ulteriores, tales como *Page Controller y Front Controller*.

3.2.4. Patrón arquitectónico Modelo Vista Controlador

El Modelo Vista Controlador es un patrón o modelo de abstracción de desarrollo de *software* que separa los datos de una aplicación, la interfaz de usuario, y la lógica de negocio en tres componentes distintos. Es mayoritariamente usado en aplicaciones web, dónde la vista es la página HTML, el modelo es el Sistema de Gestión de Base de Datos y la lógica interna, y el controlador es el responsable de recibir los eventos y

darles solución. La finalidad del modelo es mejorar la reusabilidad por medio del desacople entre la vista y el modelo. Los elementos del patrón son los siguientes:

- **Modelo**: es la representación de la información en el sistema. Trabaja junto a la vista para mostrar la información al usuario y es accedido por el controlador para añadir, eliminar, consultar o actualizar datos.
- **Vista**: es la presenta al modelo en un formato adecuado para que el usuario pueda interactuar con él, casi siempre es la interfaz de usuario.
- **Controlador**: es el elemento más abstracto. Recibe, trata y responde los eventos enviados por el usuario o por la propia aplicación. Interactúa tanto con el modelo como con la vista.

3.2.5. Patrones de diseño

Los patrones de diseño de *software* ofrecen soluciones a problemas de arquitectura de *software* en ingeniería de *software*. Dan una descripción de los elementos y el tipo de relación que tienen junto con un conjunto de restricciones sobre cómo pueden ser usados.

Para comprender mejor qué es un patrón arquitectónico, hay que tener en cuenta que (Pressman, 2001) plantea que:

Cada patrón describe una categoría del sistema que contiene:

- ✓ Un conjunto de componentes (por ejemplo, una base de datos, módulos computacionales) que realizan una función requerida por el sistema.
- ✓ Un conjunto de conectores que posibilitan la comunicación, la coordinación y la cooperación entre los componentes.
- ✓ Restricciones que definen como se pueden integrar los componentes que forman el sistema.
- ✓ Modelos semánticos que permiten al diseñador entender las propiedades globales de un sistema para analizar las propiedades conocidas de sus partes constituyentes.

Los patrones de diseño se encuentran se encuentran divididos en dos grupos: patrones GRASP[20] (experto, bajo acoplamiento y alta cohesión) y patrones GOF[21] (patrón utilizado: Singleton).

[20] GRASP es un acrónimo que significa *General Responsibility Asignment Software Patterns* (patrones generales de *software* para asignar responsabilidades)

A continuación se describen los patrones de diseño utilizados para el desarrollo del subsistema según se define en (Larman, 1999), los cuales son:

- **Patrón creacional *singleton***: está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase solo tenga una instancia y proporcionar un punto de acceso global a ella. Este patrón se ve evidenciado en la clase controladora del sistema propuesto.

- **Patrón alta cohesión**

Define que las responsabilidades asignadas a una misma clase deben guardar relación y estar enfocadas al mismo objetivo. Se aplica para realizar un diseño que evite contener clases con un alto grado de abstracción, que asuman responsabilidades que podían haber delegado a otros objetos o que tengan complejas responsabilidades. A las clases cargar mapas desde WMS, mostrar datos del acelerómetro, cargar mapas guardados, le son asignadas responsabilidades con el objetivo de que trabajen en una misma área de aplicación y no tengan mucha complejidad.

- **Patrón bajo acoplamiento:**

Define la fuerza con que una clase está conectada con otra, de esta forma una clase debe tener un mínimo de dependencia con otras clases. Es importante para realizar un diseño de clases independientes que puedan soportar los cambios de una manera fácil y que a su vez faciliten la reutilización. A las clases mostrar datos del acelerómetro, cargar mapas guardados, localizar flota, le son asignadas responsabilidades de forma tal que solo se comuniquen con la clase que se encarga de controlar.

- **Patrón experto**

Define que se debe asignar la responsabilidad a la clase que posee la información necesaria para cumplirla. Las clases cargar mapas desde WMS, mostrar datos del acelerómetro, localizar flota poseerán la información necesaria para cumplir con cada una de las responsabilidades que le corresponden.

---

[21] GOF es un acrónimo de *Gang of Four* (La banda de los cuatro).

3.2.6. Vista lógica de la arquitectura de la solución

La vista lógica es una abstracción de acuerdo a criterios diferentes que pueden usar su propia notación y definir de forma independiente el significado de los componentes, relaciones, argumentos, principios y pautas; es la base sobre la cual los arquitectos construyen modelos de aplicación que representan la perspectiva lógica de la arquitectura de una aplicación. **(Ver Anexo 2)**

3.3. Diseño

El diseño es el flujo de trabajo que permite la comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, y tecnologías de interfaz de usuario y es el que da la entrada a las actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.

3.3.1. Modelo de diseño

Según la definición Ivar Jacobson, Grady Booch y James Rumbaugh en su libro “El Proceso Unificado de Desarrollo de *Software*”: “Una clase de diseño y sus objetos, y de ese modo también los subsistemas que contienen las clases de diseño, a menudo participan en varias realizaciones de casos de uso. También puede darse el caso de algunas operaciones, atributos y asociaciones sobre una clase específica que son relevantes para sólo una realización de caso de uso. Esto es importante para coordinar todos los requisitos que diferentes realizaciones de casos de uso imponen a una clase, a sus objetos y a los subsistemas que contiene. Para manejar todo esto, utilizamos diagramas de clases conectados a una realización de caso de uso, mostrando sus clases participantes, subsistemas y sus relaciones. De esta forma podemos guardar la pista de los elementos participantes en una realización del caso de uso” (Jacobson, 2000)

Es en el modelo de diseño donde se definen las clases del diseño que conformarán el SVIGSPM que se va a implementar. **(Ver Anexo 3)**

- Páginas clientes: son las páginas encargadas de permitir a los usuarios interactuar con el sistema tanto para hacer solicitudes como para que sean mostradas las respuestas a las mismas.
- Páginas servidoras: son las encargadas de la construcción de forma dinámica de las páginas clientes y sirven de enlace entre estas y el resto de las clases.
- Páginas controladoras: son las responsables de realizar las operaciones que responden a los procesos de negocio y dar respuestas a las solicitudes hechas por el usuario.

- Clases entidad: son las responsables de la persistencia de los datos físicamente.

3.4. Diseño de la base de datos

Una base de datos correctamente diseñada permite obtener acceso a información exacta y actualizada. Puesto que un diseño correcto es esencial para lograr los objetivos fijados para la base de datos, parece lógico emplear el tiempo que sea necesario en aprender los principios de un buen diseño ya que, en ese caso, es mucho más probable que la base de datos termine adaptándose a sus necesidades y pueda modificarse fácilmente. **(Ver Anexo 1)**

3.5. Modelo de despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. El modelo de despliegue se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño. Consiste en:

- *Nodos:* elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos.
- *Dispositivos*: nodos estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela.
- *Conectores:* expresa el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo.

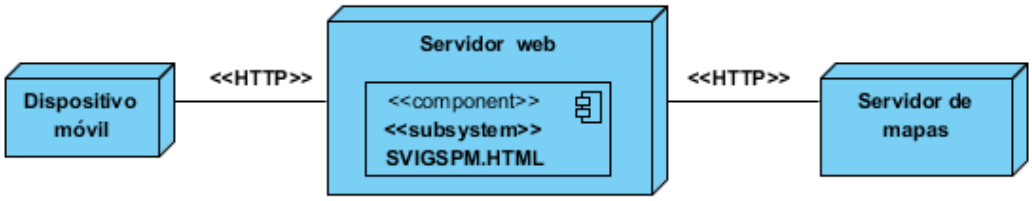


Figura 7 Modelo de Despliegue de SVIGSPM

Explicación del diagrama

En el modelo de despliegue que se presenta en la **Figura 7**, se especifica la existencia por separado del servidor web y del servidor de mapas. Presenta además, el nodo correspondiente a la Dispositivo móvil cliente desde la cual accederán mediante el navegador del dispositivo para así de esta forma interactuar con la aplicación. Además se muestra los protocolos de comunicación.

3.6. Modelo de implementación

3.6.1. Diagrama de componentes

Un diagrama de componentes representa cómo un sistema de *software* es dividido en componentes y muestra las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de *software* pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema.

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. Es un diagrama que muestra un conjunto de elementos del modelo tales como componentes, subsistemas de implementación y sus relaciones. (**Figura 9).**

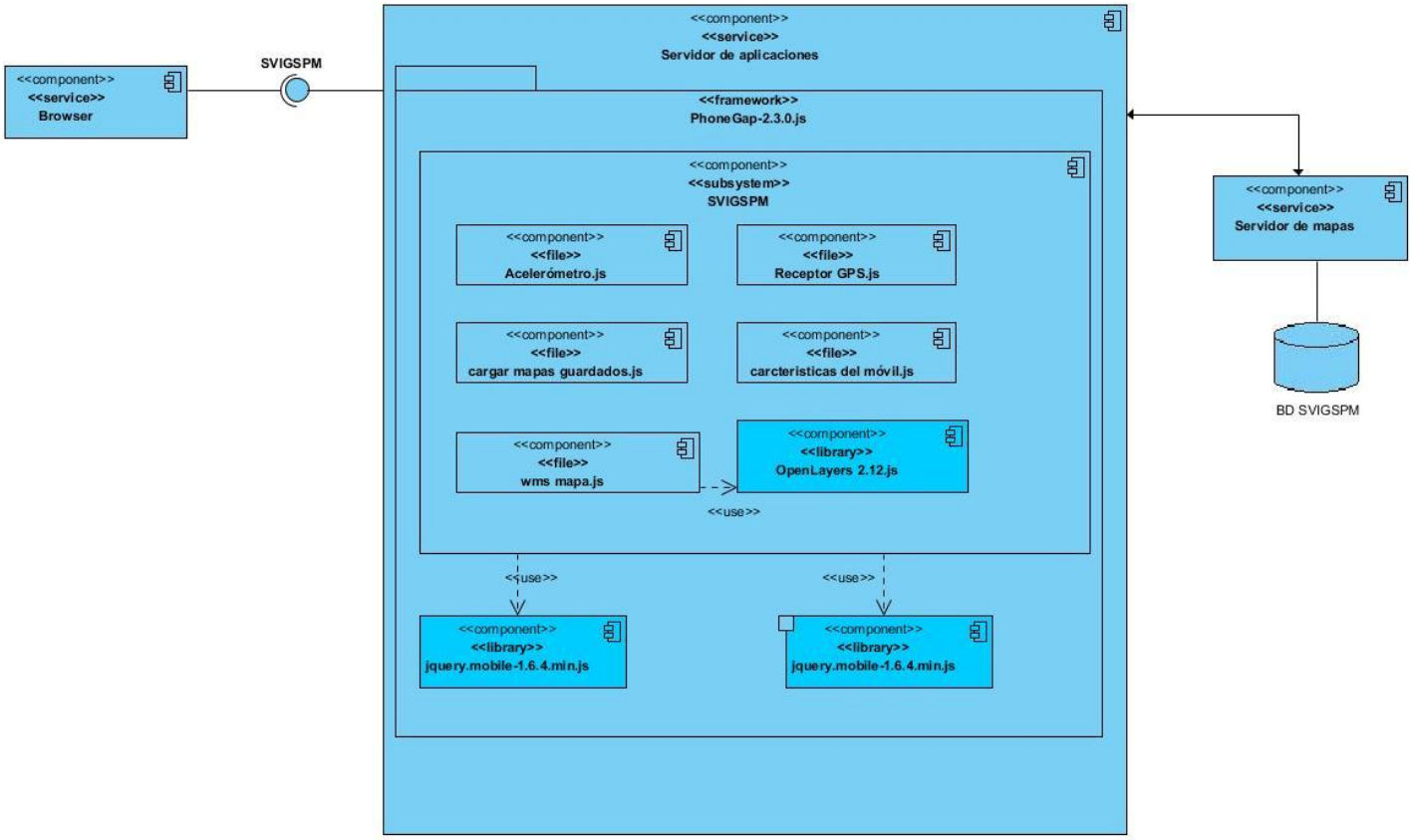


Figura 7 Diagrama de Componentes de SVIGSPM.

3.7. Pruebas

Las pruebas de *software* consisten en la dinámica de la verificación del comportamiento de un programa en un conjunto finito de casos de prueba, debidamente seleccionados de por lo general infinitas ejecuciones de dominio, contra la del comportamiento esperado. Son una serie de actividades que se realizan con el propósito de encontrar los posibles fallos de implementación, calidad o usabilidad del *software*.

Por lo que se hace necesario probar lo que se hizo en aras de comprobar que cumple con todo lo que se precisó en el inicio de su elaboración; el *software* no está ajeno a las pruebas y son estas un elemento crítico para la garantía de la calidad. Definitivamente la prueba de *software* representa una revisión final de las especificaciones del diseño y de la codificación.

El subsistema implementado SVIGSPM fue sometido a la fase de prueba, de los niveles de prueba se utilizó el nivel de prueba de sistema que son las pruebas que se realizan cuando el *software* está funcionando como un todo. Además, de los métodos de pruebas conocidos como caja blanca y caja negra se utilizó el método de caja negra que permite comprobar que cada funcionalidad es operativa. (Pressman, 2001) Dentro de las técnicas de prueba de caja negra se utilizó la técnica partición de equivalencia porque es una de las más efectivas ya que permite examinar los valores válidos e inválidos de las entradas existentes en el *software*, 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, 2001) La partición equivalente se dirige a la definición de casos de pruebas que descubran clases de errores, reduciendo así en número de clases de prueba que hay que desarrollar.

3.7.1. Resultado

Fueron probados ocho casos de uso, los que representan el cien por ciento del total de requisitos funcionales. Para cada caso de uso se tuvo en cuenta los diferentes escenarios que pudieran existir, los cuales implican cada una de las posibles interacciones del usuario o combinaciones de situaciones posibles con el fin de evaluar el comportamiento del sistema ante cada funcionalidad. El resultado de cada prueba coincide con la especificación de los requisitos lo que demuestra la correcta implementación de los mismos. Como ejemplo que muestra lo dicho anteriormente se refleja la sección y el escenario probado para el caso de uso Visualizar mapa desde WMS.

Descripción general

El caso de uso se inicia cuando el usuario desea visualizar el mapa desde el servidor WMS.

Condiciones de ejecución:

No presenta

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC1 Cargar mapa desde servicio WMS.	EC1.1: Mapa cargado con éxito.	Permite cargar un mapa desde servicio WMS mostrando la ubicación del dispositivo.	✓ Clic en la opción “Mapa “
	EC 1.2: No hay conexión con el servidor.	Se intenta cargar un mapa pero no hay conexión con el servidor.	✓ Clic en la opción “Mapa “

Tabla 4: Diseño de caso de prueba de Caja Negra del caso de uso: Cargar mapa de servicio WMS.

ID escenario	Escenario	Respuesta del Sistema	Resultado de la Prueba
EC1.1	Mapa cargado con éxito.	Se carga el mapa.	Satisfactorio
EC1.2	No hay conexión con servidor.	Muestra mensaje de error señalando que no hay conexión con el servidor.	Satisfactorio

Tabla 3: Matriz de Datos. Cargar mapa desde servicio WMS.

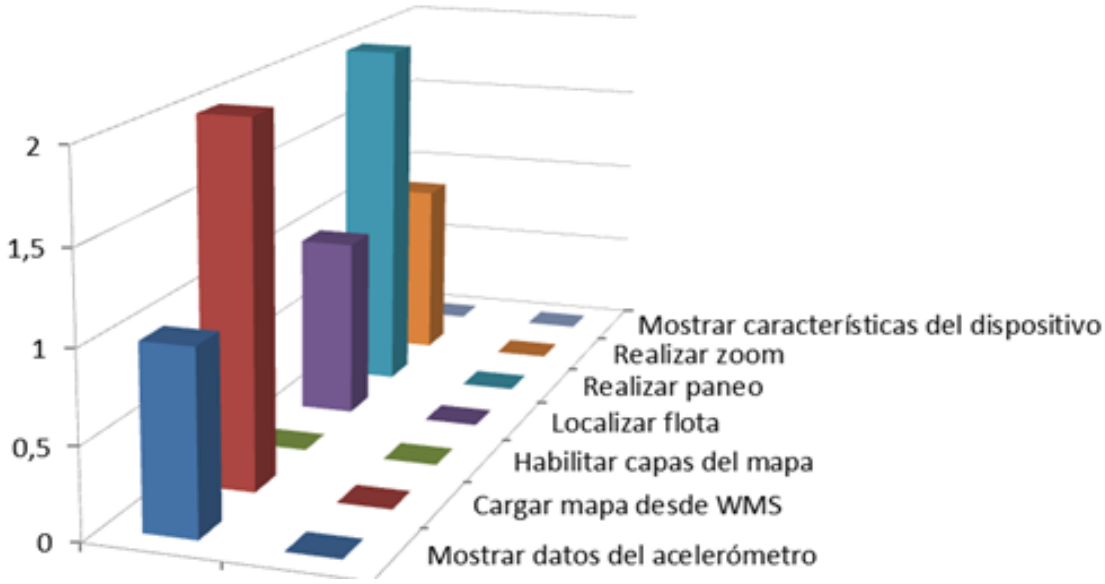


Figura 8 No conformidades por iteración.

Se realizaron dos iteraciones en la primera iteración se encontraron siete no conformidades (NC) según se muestra en la **Figura 8** luego de darle solución se realizó una segunda iteración, en la que no se detectaron NC.

3.8. Conclusiones del capítulo

En este capítulo fueron tratados los aspectos más importantes en cuanto a la construcción del *software*. Se describieron los principales patrones GRASP así como GOF que han sido utilizados en el subsistema. Se determinó el uso de patrón arquitectónico modelo-vista-controlador permitió obtener una visión de cómo estaría estructurado el sistema y de cómo sus componentes se relacionan. Se exponen los principales modelos que describen la implementación y se ofrecen los resultados obtenidos en las pruebas realizadas. Finalmente se concluye lo siguiente:

- ✓ El estudio y aplicación de los patrones de diseño propició la creación de un modelo de diseño más flexible y escalable.
- ✓ Los diagramas de diseño e implementación mostrados describen el sistema de modo que se facilite el trabajo con el mismo si fuese necesario.
- ✓ El modelo de despliegue garantiza que se pueda tener una idea de cómo estará distribuido físicamente el sistema para su correcto funcionamiento.
- ✓ Es apropiado realizarle pruebas de caja negra al subsistema y se comprobó que el resultado de su aplicación valida el cumplimiento de los requisitos definidos.

.

CONCLUSIONES

Una vez concluida la presente investigación en la cual se realizó la implementación del subsistema de visualización de información georreferenciada del Sistema control de Flotas para dispositivos móviles perteneciente al proyecto Aplicativos SIG, se puede arribar a las siguientes conclusiones:

- Las herramientas, tecnologías, metodología de desarrollo de *software* y lenguaje de modelado empleados permitieron implementar un subsistema de visualización de información en el cual se muestra en dispositivos móviles la información georreferenciada del SCF del proyecto Aplicativos SIG.

- La definición del patrón arquitectónico modelo-vista-controlador permitió obtener una visión de cómo estaría estructurado el sistema y de cómo sus componentes se relacionan. Unido a esto la utilización de los patrones de diseño para la construcción de los diagramas de las clases del diseño garantizaron el diseño de un *software* orientado a objetos. Estos últimos permitieron además que existiera poca dependencia entre las clases permitiendo así una mejor reutilización.

- El sistema fue construido con herramientas libres lo que redujo el costo de su desarrollo y se evitan los problemas asociados con el pago de licencias de *software*.

- La solución propuesta resuelve el problema planteado ya que el subsistema SVIGSPM proporciona que la aplicación sea utilizada en la mayoría de los Sistemas Operativos para móviles como *Android, iOS, Windows Phone, BlackBerry, WebOS, Symbian y Bada.*

RECOMENDACIONES

Una vez vencido el objetivo de la investigación, y teniendo en cuenta las experiencias obtenidas, el autor recomiendan:

- ✓ La implementación de un módulo que permita que la imagen que se encuentra contenida dentro del móvil contenga información asociada a la misma.
- ✓ Agregar nuevas funcionalidades como son el cálculo de rutas y permitir al usuario agregar lugares de interés.

BIBLIOGRAFÍA

- **Alonso, Arturo, et al.** *Dispositivos móviles. EPSIG.* Ingeniería de Telecomunicación Universidad de Oviedo : s.n.
- **Bertoltti, María I y Cabut, Diego A. 1986.** *Flota de altura-breve reseña de la evolución histórica y operatividad durante el período 1981/1982.* 1986.
- **Booh, Grady, Rumaugh, James y Jacobson, Ivar. 2000.** *El lenguaje unificado de modelado.* Addison Wesley : Ed. Jesús García Molina, 2000. Vol 1.
- **Cámara, E. Martínez.** *Aplicación cliente/servidor multiplataforma en internet para laboratorio virtual de robótica.*
- **Campoverde, Paola, et al. 2010.** *Análisis,diseño, construcción e implementación de un portal WAP para georreferenciación del patrimonio cultural de la ruta de los yumbos.* Ecuador,Quito : s.n., 2010.
- **Casado, Francis. 2005.** *Una herramienta para la creación de interfaces multiplataforma con UIML. En Proceedings on the VI Congreso de Interacción Persona Ordenador.* Granada : ISBN, 2005. 84-9732..
- **Catalán, A. 2011.** *Curso Android Desarrollo de Aplicaciones Móviles.* España : s.n., 2011.
- **Christ, Adam M. 2011.** *Bridging the mobile app gap. Connectivity and the User Experience,.* 2011. 1.
- **Colomila Pardo, Otto, et al. 2011.** *Tema 6, parte 3 Frameworks web para dispositivos móviles.* 2011.

- **Conde, Miguel, Muños, Carlos y García, Francisco . Albacete. 2008.** *Sistemas de Adaptación de contenidos para dispositivos móviles. En Proc. Actas del congreso de IX Congreso Internacional de Interacción Persona-Ordenador.* s.l. : p. 143-147., Albacete. 2008.
- **Descamps-Villa, Laial. 2011.** *Cómo introducir semántica en las aplicaciones SIG móviles: expectativas, teoría y realidad.* s.l. : et a, 2011.
- **Gamboa, Fallas. 1996.** *Jorge. Sistema de información geográfica: una visión integral.* s.l. : Revista geográfica de América Centra, 1996.
- **Gil, David, Robles, Job S. y Villareal, Verónica. 2010.** *Servicio Web administrador de usuarios multiplataforma.* 2010.
- **Gómez, J. Cervantes. 2012.** *Taxonomía de los modelos y metodologías de desarrollo de software más utilizados.* 2012.
- **Iglesias, Rafael. 2008.** *MovilWeb: aplicación para el control de flota basada en la infraestructura de datos espaciales de la República de Cuba.* Cuba : s.n., 2008. 123.
- **Iñesta, Luis, Sánchez, Juan y Aquino, Nathalie.** *Descripción de una Herramienta de Autor para una Extensión del Lenguaje UIML.*
- **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** *El Proceso Unificado de Desarrollo de Software.* España : Addison Wesley : ISBN:0-201-57169-2., 2000.
- **Jaén, Juan Ignacio, Romero, José Raúl y Vallecillo, Antonio. 2009.** *Especificación de descripciones arquitectónicas multivista basada en modelos. Actas de los Talleres de las Jornadas de Ing. del Software y BBDD.* 2009. 2.
- **Larman, Craig. 1999.** *UML y patrones. Pearson.* 1999.

- **Llano, Marinés Alemán, Cáceres, Yoandy Pérez y Góngora, Nilber Barbán.** *Desarrollo de un componente que facilite la evaluación del aprendizaje en la plataforma educativa ZERA.*
- **Lunny, Andrew. 2012.** *PhoneGap Beginner's Guide. Packt Publishing Ltd, 2011.* s.l. : Apress, 2012.
- **Marinacci, Joshua. 2012.** *uilding Mobile Applications with Java: Using the Google Web Toolkit and PhoneGap. O'Reilly Media, Inc.* 2012.
- **Meier, Reto. 2012.** *Professional Android 4 application development. Wrox.* 2012.
- **micronav. 2010.** www.micronav.com. [En línea] 2010. [Citado el: 7 de 11 de 2012.]
- **mobilefleet. 2010.** www.mobilefleet.es. [En línea] 2010. [Citado el: 12 de 11 de 2012.]
- **Morales, Antonio Fernández y Toledano, María Cruz Mayorga. 2002.** *Micromódulos didácticos basados en un entorno inalámbrico WAP para los estudios de turismo.* TURITEC : s.n., 2002.
- **Mota Olmos, Sotero. 2013.** *Programación de aplicaciones móviles en Android para la evaluación del conocimiento.* 2013.
- *MovilWeb:Aplicación para el control de flotas basada en PostgreSQL.* **Suárez, Guillermo González. Revista Cubana de Ciencias Informáticas, 2011.** 1, Cuba : s.n., Revista Cubana de Ciencias Informáticas, 2011, Vol. 5.
- **Myer, Thomas. 2011.** *Beginning PhoneGap. Wrox.* 2011.
- **N., Acostan N. y Mosca. 2005.** *En VII Workshop de Investigadores en Ciencias de la computación - WICC 2005.* Cordova,Argentina : s.n., 2005.
- **Pérez Lovelle, Sonia. 2010.** *Automatización de la arquitectura de componentes genéricos usando UML.* 2010. 1.

- **Pressman, Roger S. 2001.** *Ingeniería del Software. Un enfoque práctico..* Madrid : Quinta Edición., 2001. s.n.
- **Raquel, Anaya. 2012.** *Una visión de la enseñanza de la ingeniería de software como apoyo al mejoramiento de las empresas de software.* s.l. : Revista Universidad EAFIT, 2012. 60-76.
- *Revista Cubana de Ciencias Informáticas.* Cuba : Ediciones Futuro. ISSN:1994-1536.
- **Reynoso, Carlos. 2009.** *Introducción a la Arquitectura de Software. Documento de la Universidad de Buenos Aires.* 2009. 1.
- **Rincón, Andrés. 2008.** *Herramienta de software para la localización geografica de terminales en redes móviles celulares.* 2008.
- **Rivas, Lornel, et al. 2012.** *Criterios para la selección de herramientas de ingeniería de software en PYMES.* Venezuela : Revista de la Facultad de Ingeniería Universidad Central, 2012.
- **Roig, Oriol, González, José Luis y Comes, Ramón . 2003..** *Principios de comunicaciones móviles.* Univ. Politèc. de Catalunya : s.n., 2003.
- **Rojas, Margarita. 2004.** *Curso de UML Multiplataforma Adaptativo basado en la Teoría de Respuesta al Item. Ingeniería informática.* 2004. 5.
- **Rosen, Lawrence. 2004.** *Open source licensing.* s.l. : Prentice Hall PTR, 2004.
- **Sánchez, Juan. 2002.** *Un método para validar requisitos y generar interfaces de usuario multiplataforma.* 2002.
- **Sánchez, Juan.** *Proyecto Multiplataforma para dispositivos móviles Y Smartphone "PickUP".*
- **SearchMobileComputing. 2007.** SearchMobileComputing. [En línea] junio de 2007. [Citado el: 2013 de 2 de 2.] http://searchmobilecomputing.techtarget.com/definition/smartphone..

- **Singh, Inderjeet y Palmieri, Manuel. 2011.** *Comparison of cross-platform mobile development tools. IDT: Malardalen University.* 2011.
- **Sommerville, Ian. 2005.** *Ingeniería del software.* Pearson : Educación, 2005.
- **Taylor, David. 1991.** *Geographic information systems: The microcomputer and modern cartography.* Oxford : UK: Pergamon Press, 1991.
- **Toloza, Juan Manuel y Acosta, Nelson. 2009.** *Proyecto de desarrollo de un sistema de control de flotas de vehículos. En XI Workshop de Investigadores en Ciencias de la Computación.* 2009.
- **Torres, Jairo. 2006.** *Procedimiento jerárquico basado en optimización y simulación para la gestión de vehículos en sistemas automatizados de manufactura.* 2006. pág. 5. 1.
- **USA, IDC Corporate. 2012.** *Worldwide Quarterly Tablet Tracker.* Massachusetts : IDC Corporate USA, 2012.
- **Valverde, Francisco,. 2007.** *Hacia un Modelo de Interacción Abstracto para la Definición de Interfaces Multiplataforma. En VIII Congreso Internacional de Interacción Persona-Ordenador, Interacción.* 2007.
- **Van Lancker, Luc. 2012..** *Los fundamentos del lenguaje.* s.l. : Ediciones EN, 2012.
- **Wargo, John M. 2012.** *PhoneGap Essentials: Building Cross-platform Mobile Apps. Addison-Wesley Professional.* 2012.
- **Yañez, María Rebeca y Villatoro , Pablo. 2005.** *Las nuevas tecnologías de la información y de la comunicación (TIC) y la institucionalidad social: hacia una gestión basada en al conocimiento.* s.l. : United Nations Educational, 2005. Vol. 108. 9213226853.

ANEXOS

Anexo 1. Diagrama Entidad Relación

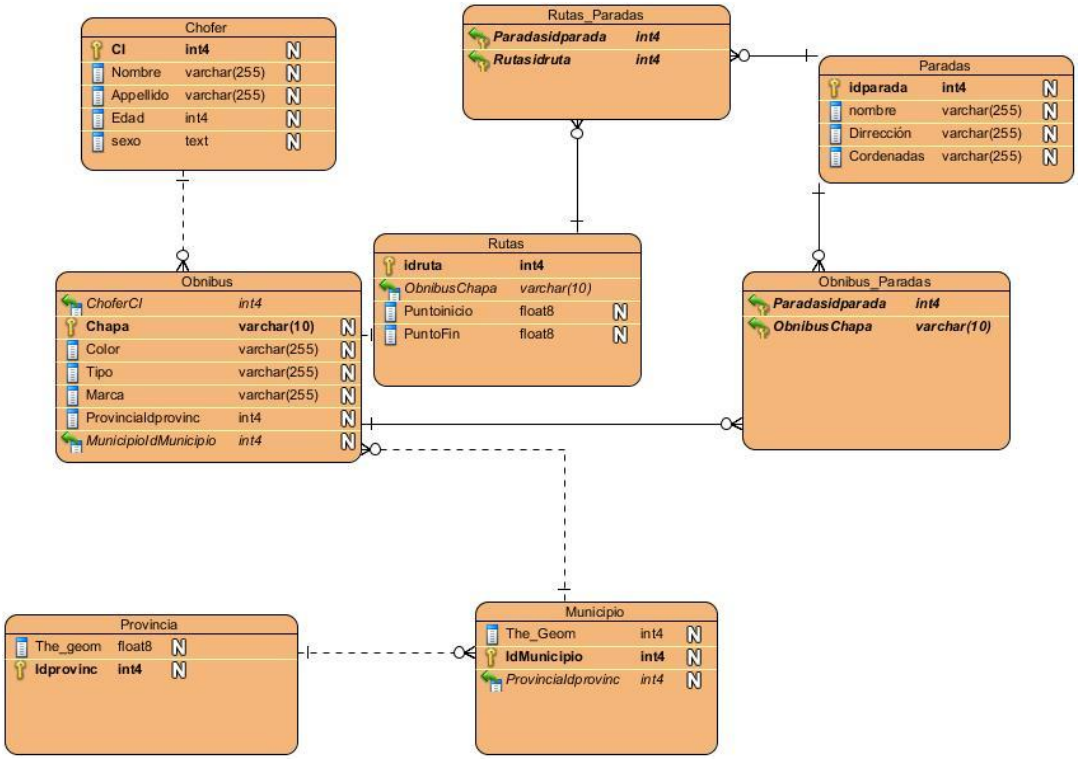


Figura 9 DER de SVIGSPM.

Anexo 2. Vista Lógica de la Arquitectura

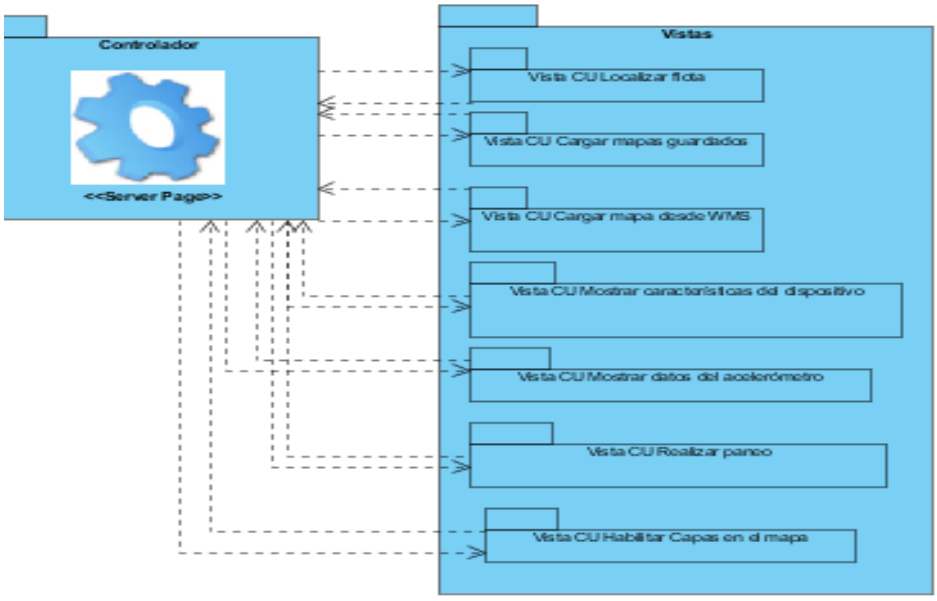


Figura 10 Vista lógica de la arquitectura de SVIGSPM.

Anexo 3. Diagrama de Clases del Diseño

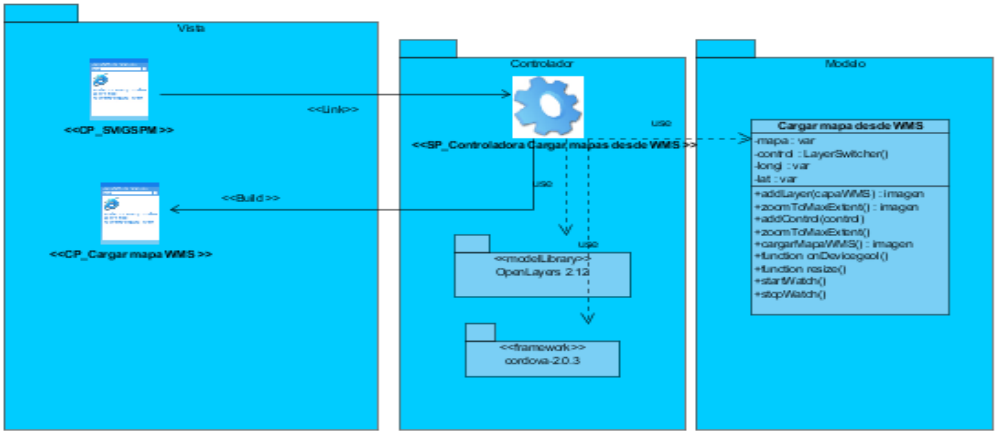


Figura 11 Diagrama de clase del diseño CU "Cargar mapa WMS".

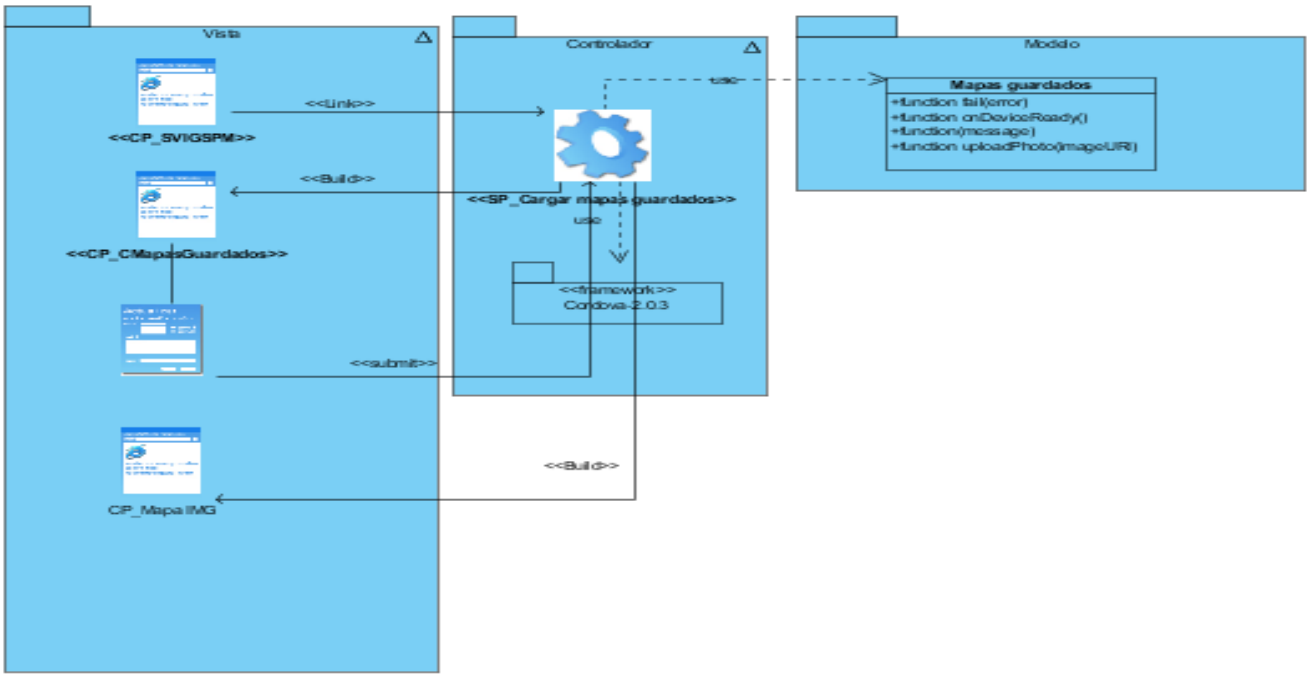


Figura 12 Diagrama de clase del diseño CU "Cargar mapas guardados".

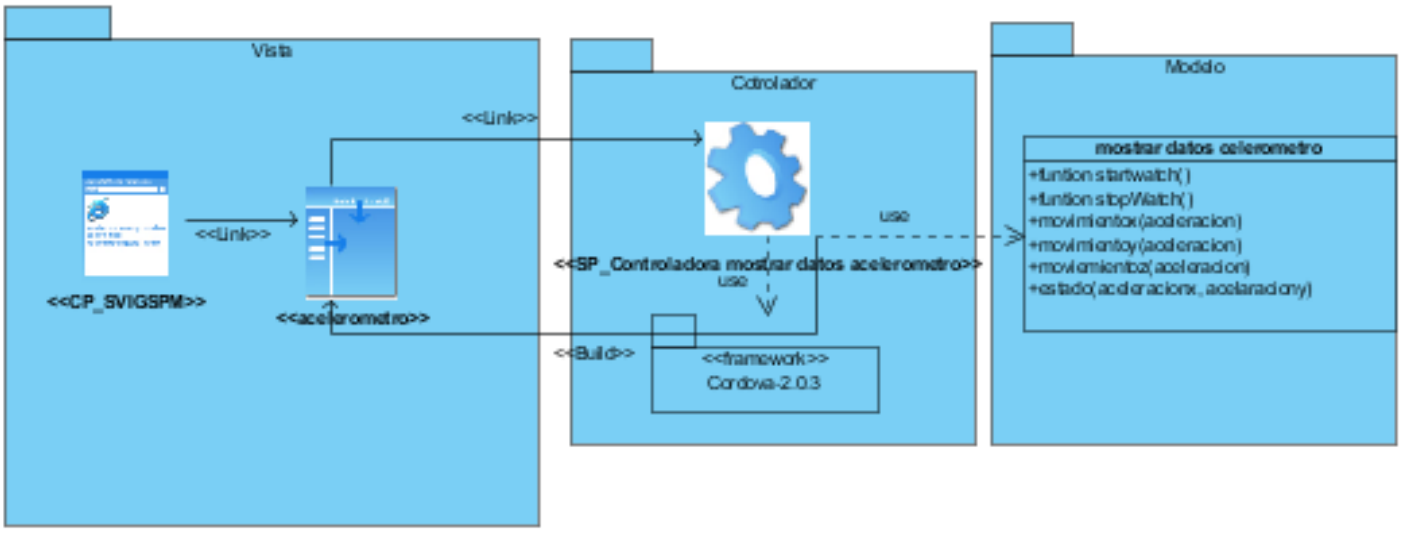


Figura 13 Diagrama de clase del diseño CU "Mostrar datos del acelerómetro".

GLOSARIO DE TÉRMINOS

A

- **API.**

Una interfaz de programación de aplicaciones o API (del inglés *Application Programming Interface*) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro *software* como una capa de abstracción.

B

-Base de Datos:

Conjunto de datos no redundantes, almacenados en un soporte informático, organizado de forma independiente de su utilización y accesible simultáneamente por distintos usuarios y aplicaciones. La diferencia de una Base de Datos respecto a otro sistema de almacenamiento de datos es que estos se almacenan de forma que no cumplan con tres requisitos básicos: no redundancia, independencia y concurrencia.

F

- **Framework.**

Es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de *software* concretos, con base en la cual otro proyecto de *software* puede ser organizado y desarrollado.

H

- **Herramienta CASE.**

(*Computer Aided/Assisted Software*/*System Engineering*) o (Ingeniería de *Software* Asistida por Computadora) son herramientas utilizadas en el modelamiento visual de sistemas.

-HTML

Significa "*Hipertext Markup Language*" y es el lenguaje de programación utilizado para crear las páginas de Internet. Con él se definen la posición, forma y funcionamiento de las imágenes, textos e hipervínculos incluidos en la página. En HTML, se permite utilizar dos formatos gráficos: GIF y JPEG.

-HTTP

Hiper Text Transmision Protocol. Las normas comunes que deben respetar los ordenadores que forman la red para comunicarse entre sí.

- **IDE**

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI).

-Interfaz

Una conexión e interacción entre *hardware*, *software* y usuario, es decir como la plataforma o medio de comunicación entre usuario o programa.

R

- ***RUP*.**

El Proceso Unificado Racional o de Desarrollo (*Rational Unified Process* en inglés, tradicionalmente abreviado como *RUP*) constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

S

- **SIG**

Sistema de Información Geográfica (GIS en Inglés).

- **SVIGSPM**

Subsistema de Visualización de Información Georreferenciada sobre Plataformas Móviles.

U

- ***UML***

Un lenguaje para modelar objetos es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar parte de un diseño de *software* orientado a objetos.