UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 6

Proveedor de MipMapping para GeoQ

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

Autores:

Lilibeth Batista García

Roberto Ortiz Borges

Tutor:

M.Sc. David Silva Barrera

La Habana, junio de 2013.

Año 55 de la Revolución.

La magnitud de lo que logramos, no depende de lo que tenemos, sino de lo que seamos capaces de hacer…

Che

Lilibeth

A mis padres: Clara Elena y Enrique: por ser la inspiración de este sueño.

A mis abuelos (Nelida y Héctor): por siempre creer en mí.

A mi novio: Liuver: por mantenerse a mi lado

A mis tíos (Papito, Mandy, Gollo): sin sus ocurrencias mis días fueran muy tristes.

A mi familia (Nidia, Nidiecita y Recito, Puchita y Agustín, Nequitia y Tita Nelisi, Carlos, Luciano y Miriam, Adolfo, Yurima, Adonaisito y Samu, Yudita): por su apoyo incondicional

A mis amigos: Indira y Yoyi: gracias por aparecer en mi vida.

Roberto

Principalmente, con mucho cariño y amor, a mi mamá, Magalis que me ha guiado y ayudado a llegar hasta aquí y mi papá Roberto que aunque no se encuentre físicamente siempre está conmigo, por ser las personas que me dieron la vida y que están orgullosos de mí y eso me hace la persona más feliz del mundo.

A mi hermana y esos dos sobrinos malcriados que tengo y quiero mucho.

A mi abuela Santa por estar siempre presente en mi vida y quererme tanto y a mi abuelo Leyva.

A mis tíos, principalmente a Ania, Jorge, Reini, Lule y mi primo Vania por quererme incondicionalmente y preocuparse tanto por mí.

A mis amistades Arturo, Luis, Arianny, Laritza, Rafa, Polo, Dimelsa, Mahe, Yoarys, Niva, Leandro, Yendry, Javier, Aindamais y especialmente a Yanela que ha sido mi hermana estos 5 años y siempre ha estado en los momentos difíciles para mí; en fin, a todas las personas que me han aguantado y apoyado durante estos 5 años.

A todos los que de alguna manera contribuyeron en el desarrollo de esta tesis.

Agradecimientos

Lilibeth

A mis padres y mi novio: por luchar a mi lado para obtener este sueño.

A mi tutor M.Sc. David Silva: por la confianza y la seguridad que deposito en todo momento, por su ayuda y sus regaños. ¡Muchas gracias!

A mi compañero de tesis: por crecerse en los momentos difíciles.

A todas las personas que de una forma u otra han estado a mi lado y me han apoyado en la realización de este trabajo.

A la Universidad de la Ciencias Informáticas: por mostrar el camino y convertirme en una ingeniera comprometida con la Revolución.

A la Revolución: por darme la posibilidad de conocer este inmenso lugar que es la UCI.

Agradecimientos

Roberto

A todos los que han contribuido a mi formación como ingeniero, especialmente:

A mi mamá, mi papá, mi hermana, mi abuela, mi tía Ania, mis tíos Jorge, Reini y Lule, mi abuelo Leyva y mi primo Vania, por el apoyo incondicional y la confianza que han depositado en mí y por el amor que me han dado en todo momento.

A todos mis compañeros de grupo de primer año, a los del grupo de cuarto y quinto, especialmente a mis buenos amigos Yanela, Arturo, Luis, Arianny, Laritza, Rafa, Polo, por pasar tantos momentos buenos y otros no tanto, pero siempre juntos, y por hacerme saber que siempre puedo contar con ellos.

A mi compañera de tesis Lily que siempre confió que nos podíamos graduar.

A mi tutor y todos los profesores que contribuyeron a mi formación como ingeniero.

A todas estas personas especiales para mí, quiero darle las gracias por brindarme toda su ayuda, apoyo y comprensión para poder lograr hacer mi sueño realidad.

DECLARACIÓN DE AUTORÍA

Declaramos por este medio que, yo Lilibeth Batista García con carnet de identidad 90041104312 y Roberto Ortiz Borges con carnet de identidad 89052137426, somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año 2013.

Lilibeth Batista García **Roberto Ortiz Borges**

Autor **Autor**

_____ _____

M.Sc. David Silva Barrera

Tutor

_____

Resumen

El presente trabajo, titulado: “Proveedor de MipMapping para GeoQ”, se realiza como alternativa de solución a la inexistencia de un módulo que agilice el proceso de consumo de recursos en la plataforma GeoQ, de manera que permita obtener las cartografías ráster de Google Maps.
Está diseñado sobre una arquitectura basada en componentes y orientada a objeto, se implementó sobre plataformas de software libre, sus utilidades están desarrolladas en el lenguaje de programación C++, teniendo como entorno de desarrollo integrado Qt. Con el objetivo de guiar el proceso de desarrollo se utilizó la metodología de software RUP (Proceso Unificado de Racional) y la modelación de la ingeniería se hizo mediante el lenguaje de modelado UML (Lenguaje Unificado de Modelado), asistido por la herramienta CASE Visual Paradigm.
Entre los resultados obtenidos se encuentran el módulo proveedor de MipMapping de la Plataforma de Información Geográfica GeoQ y la documentación UML de los artefactos resultantes del análisis y diseño del sistema de gestión de datos e información.

Palabras claves: sistema de información geográfica, proveedor, consumo de recursos.

Índice

Índice de Ilustración

Índice de Tablas

Introducción

El crecimiento continuo del desarrollo en las Tecnologías de la Información y la Comunicación (en lo adelante TIC), ha permitido informatizar las principales necesidades de la sociedad, tales como: el almacenamiento de grandes volúmenes de datos, el desarrollo de canales de comunicaciones, el fácil acceso a todo tipo de información, la automatización de tareas, entre otras; lo cual ha permitido gestionar grandes fuentes de información y a su vez, ha disminuido la posibilidad de errores que ocurren con el trabajo manual.

El surgimiento de las TIC y su avance acelerado en los últimos años, ha generado grandes cambios sociales, tanto en las estructuras económicas como en las culturales y educativas. Su incorporación en diversos espacios ha provocado una revolución tecnológica que se enmarca como medio alternativo de divulgación y transmisión de información. Además se ha convertido asimismo, en una pieza imprescindible para la informatización de las sociedades a nivel mundial. Cuba no ha quedado exenta de los beneficios que aportan las TIC, sino que se ha apoderado de los mismos, para el desarrollo de la economía nacional, la sociedad y el servicio a los ciudadanos.

La informatización cubana, es el proceso de utilización ordenada y masiva de las TIC para satisfacer las necesidades de información y conocimiento de todas las personas y de las esferas sociales. Este proceso tiene como objetivo principal permitir una mayor generación de riquezas que haga sustentable el aumento sistemático de la calidad de vida de los cubanos.

Formando parte del proceso de informatización de la sociedad cubana se encuentra el sector de la Geoinformática [1] que mantiene un desarrollo considerable y sostenido desde los últimos años, explotando la utilización de Sistemas de Información Geográficas (en lo adelante SIG) en sus principales productos. Un 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 geográfica (**Bosque**, 1999).

Estos sistemas son herramientas que permiten a los usuarios crear consultas interactivas, analizar la información espacial, editar datos, mapas y visualizar los resultados de todas estas operaciones. En la última década el desarrollo de los SIG ha sido ascendente, los mismos pueden ser utilizados para

[1]Geoinformática: Disciplina que surge de unir la informática y la ciencia de la tierra. Es el uso de las matemáticas y las técnicas informáticas para resolver problemas geográficos.

investigaciones científicas, gestión de recursos y activos, evaluación del impacto ambiental, planificación urbana y cartografía por nombrar algunos de los más relevantes (**Farré**, 2010).

Una de las instituciones destacadas en el desarrollo de SIG en Cuba, es la Universidad de las Ciencias Informáticas (en lo adelante UCI). La misma cuenta con el centro de desarrollo GEySED que dirige proyectos para la elaboración de estos sistemas. La dirección del centro manifestó su interés en desarrollar una aplicación que responda a las exigencias de un SIG de escritorio. Este tipo de aplicación tiene como principal ventaja, la edición de cartografías, teniendo mayor potencial en la representación gráfica de las mismas.

Se crea así el proyecto SIG-Desktop con el objetivo general de desarrollar sistemas de escritorio que permitan editar las cartografías en el centro, brindar un conjunto de funcionalidades para la geo-referenciación[2] y localización de recursos sobre los mapas. Se adicionan además funcionalidades nuevas en dependencia de las necesidades que surjan con las oportunidades de negocio, con el fin de lograr que los productos generados a partir del proyecto en cuestión sean una Línea de Producto de Software (LPS).

La aplicación multiplataforma GeoQ es el primer producto del proyecto SIG-Desktop implementada con el framework Qt. La misma se encarga del desarrollo de Sistemas de Información Geográfica en ambiente escritorio. Se toma como base el software libre Quantum GIS [3](QGIS), a este se le hacen mejoras y se personaliza para negocios y clientes específicos. GeoQ permite visualizar y superponer cartografías vectoriales y ráster en diferentes formatos y proyecciones sin necesidad de convertirlos a un formato interno o común.

Posibilita además, el diseño de mapas y la exploración interactiva de los datos espaciales con una interfaz gráfica de usuario amigable. Sin embargo, el proceso de obtención de cartografías ráster de Google Maps se hace engorroso ya que GeoQ tiene la necesidad de utilizar el software MapServer como intermediario para obtenerlas.

MapServer es una plataforma de código abierto para la publicación de datos espaciales y aplicaciones cartográficas interactivas para la web, es liberado bajo una licencia estilo MIT y se ejecuta en todas las

[2] Geo-referenciación: Proceso de definición de un objeto en un espacio físico, mediante el cálculo de su localización en un sistema de coordenadas.

[3] Quantum GIS: SIG de código libre para plataformas GNU/Linux, Unix, Mac OS y Microsoft Windows.

plataformas (Windows y Linux), su objetivo principal es visualizar, consultar y analizar información geográfica a través de la red (**Ballari Daniela**, 2011). GeoQ consume este recurso (MapServer) como un Servicio de Mapa en la Web (por sus siglas en ingles WMS), este servicio no es más que una petición en la forma de Localizadores de Recursos Uniformes (URL; por sus siglas en inglés) que realiza el usuario o aplicación que esté requiriendo el servicio (**Open Geospatial Consortium**, 2013).

Por lo anteriormente planteado se tiene como ***problema a resolver:*** ¿Cómo agilizar el proceso de consumo de recursos de la aplicación GeoQ para obtener cartografías ráster de Google Maps?

Para dar solución al problema detectado se ha planteado como ***objetivo general*** desarrollar un proveedor de MipMapping para GeoQ que permita consumir de forma ágil las cartografías ráster de Google Maps.

El ***objeto de estudio*** de la investigación se encuentra delimitado en los proveedores de datos cartográficos de los SIG.

Enmarcado en el ***campo de acción*****:** Desarrollo e implementación de proveedores de datos cartográficos ráster de Google Maps para GeoQ.

Con el fin de dar cumplimiento del objetivo general se desglosan los siguientes ***objetivos específicos*****:**

- Realizar el modelamiento del negocio y levantamiento de los requisitos.
- Analizar y diseñar los procesos seleccionados.
- Implementar un proveedor de MipMapping para la aplicación GeoQ.

Proponiendo como ***idea a defender:*** Si se implementa un Proveedor de cartografías ráster para el sistema GeoQ, se agilizará el consumo de recurso y favorecerá en la toma de decisiones.

Se diseñaron las siguientes ***tareas de investigación*** llevadas a cabo para el logro de los objetivos:

1. Caracterizar los temas relacionados con los diferentes sistemas que utilicen proveedores de cartografías ráster en Cuba y el mundo.
2. Seleccionar las herramientas a utilizar en la construcción de la solución.

3. Analizar el funcionamiento de los procesos que se involucran con los proveedores de cartografías ráster.

4. Seleccionar y argumentar la Metodología de Desarrollo de Software a usar en el proceso.

5. Documentar la información referente al análisis, diseño e implementación del sistema.

6. Diseñar e implementar la solución propuesta.

7. Validar la propuesta de solución.

En el desarrollo de la presente investigación se tuvieron en cuenta una serie de métodos científicos los cuales se exponen a continuación:

Métodos Teóricos:

- **Histórico - Lógico:** En la primera fase de la investigación se desarrolla un estudio del estado del arte de la problemática analizada, revisando de forma crítica cada uno de los documentos para lograr un mejor entendimiento de lo que se debe desarrollar.

- **Analítico- Sintético:** Se realiza un estudio con profundidad de la información acerca de las posibles tecnologías, metodologías y herramientas a ser utilizadas, logrando definir con mayor certeza las mismas para un mejor entendimiento de la situación y luego poder sintetizar la confección de la solución propuesta.

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

Métodos Empíricos:

- **Análisis Documental:** Se utilizó para analizar la documentación contenida en los sitios, revistas y libros que aparecen en Internet sobre los procesos de consumo de recursos hacia la aplicación Google Maps, todas actuales y confiables, lo que posibilita llegar a conclusiones certeras del tema en cuestión.

Métodos para hacer las pruebas y validaciones al software:

- Pruebas unitarias: Se probará el correcto funcionamiento de estructuras de código

indistintamente para asegurarse que cada una funcione correctamente por separado.

- Pruebas de integración: Se realizarán una vez que se hayan hecho las pruebas unitarias.
- Pruebas funcionales: Se harán pruebas basadas en la ejecución, revisión y retroalimentación de las funcionalidades diseñadas.

Como ***resultado*** del cumplimiento de las tareas investigativas se espera obtener:

- El informe de la investigación científica.
- La documentación técnica del proceso de desarrollo de software.
- La creación de un proveedor de MipMapping para la aplicación GeoQ.

La investigación consta de 4 capítulos:

Capítulo 1. Fundamentación teórica: Se especifican y se explican los principales conceptos asociados, se detalla la situación problemática actual, se exponen el estudio y las principales conclusiones de las soluciones que existen en la actualidad para gestionar estos procesos.

Capítulo 2. Tendencias y tecnologías actuales a utilizar: Se explican las principales tecnologías, lenguajes de programación y herramientas que se utilizarán para la construcción de la solución, además de las ventajas que supone el utilizarlas.

Capítulo 3. Presentación de la solución propuesta: Se analiza la solución propuesta, se plantea la construcción de la misma en función de diagramas de clases y estándares de diseño, generalidades de la implementación, modelo de despliegue y modelo de implementación.

Capítulo 4. Construcción de la solución propuesta: Se detalla el proceso de construcción de la propuesta de solución. Se realizan las pruebas necesarias para verificar los resultados alcanzados.

Capítulo 1. Fundamentación Teórica

1.1 Introducción

En este capítulo se tratarán los temas referidos al aseguramiento teórico de la investigación; se enunciarán los principales conceptos asociados al dominio o marco del problema para permitir un mejor entendimiento de los mismos. Se realizará una breve descripción del objeto de estudio. Se analizarán los avances tecnológicos asociados a los sistemas existentes, así como la descripción actual del dominio del problema, para dar solución a la situación problemática existente en la aplicación GeoQ del proyecto SIG-Desktop de la UCI.

1.2 Conceptos asociados al dominio del problema

Sistemas de Información Geográfica

Es "…*un sistema de hardware, software y procedimientos diseñado para realizar la captura, almacenamiento, manipulación, análisis, modelación y presentación de datos referenciados espacialmente para la resolución de problemas complejos de planificación y gestión*." NCGIA (**National Centre of Geographic Information and Analysis**, 1990).

Clasificación de los SIG según el modelo de datos

Los Sistemas de Información Geográfica pueden ser clasificados atendiendo al modelo de datos con que procesa la información, de esta forma se pueden encontrar tanto SIG que utilizan modelos Vectoriales, Ráster o ambos.

- **Datos Vectoriales***: S*on aquellos Sistemas de Información Geográfica que para la descripción de los objetos geográficos utilizan elementos compuestos por puntos, líneas y polígonos definidos por coordenadas relativas a algún sistema cartográfico. (Figura 1)
- **Datos Ráster***:* Los Sistemas de Información Ráster basan su funcionalidad en una abstracción de la realidad. Cada superficie a representar se divide en filas y columnas, formando una malla o rejilla regular. Cada celda ha de ser rectangular, aunque no necesariamente cuadrada.(Figura 1)

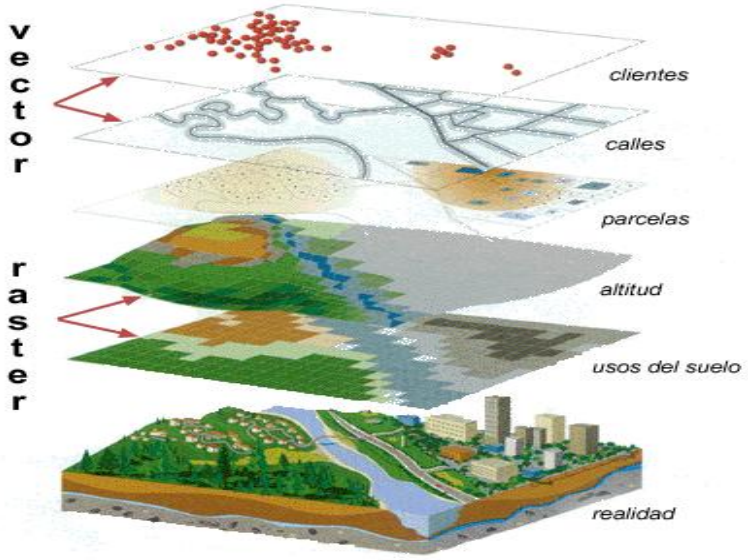


Ilustración 1. Clasificación de capas

Funcionamiento de los SIG

Los SIG generalmente permiten realizar seis tareas fundamentales que apoyan la comprensión de su funcionamiento:

- Ingreso*:* Cuando se van a introducir los datos estos deben ser convertidos internamente a un formato digital adecuado.
- *Manipulación:* Transformaciones que se le realizan a los datos como cambios de escala, proyección y generalización.
- Manejo/Administración*:* Para facilitar el manejo de la información almacenada se requiere de un sistema de manejo de bases de dato (SMBD).
- Consulta*:* Se deben responder las preguntas realizadas por los usuarios al sistema.
- Análisis*:* Para desarrollar el análisis se utilizan dos procesos: proximidad y superposición.
- Visualización*:* Se muestra como un mapa o gráfico.

MapServer

MapServer es una plataforma de código abierto para la publicación de datos espaciales y aplicaciones de cartografía interactiva para la web. Originalmente desarrollado a mediados de la década de 1990 en la Universidad de Minnesota, MapServer es liberado bajo una licencia tipo MIT y funciona en todas las plataformas principales (Windows, Linux, Mac OS X) (**Ballari Daniela**, 2011).

Cartografía

La cartografía viene del griego chartis (mapa) y graphein (escrito), es la ciencia que se encarga del estudio y la confección de los mapas geográficos, territoriales y de diferentes dimensiones lineales (**CALI**, 2011). Otros autores coinciden en que sea el arte de trazar mapas geográficos. Ciencia que los estudia. (DICCIONARIO DE LA LENGUA ESPAÑOLA).

Google Maps

Google Maps es un servicio gratuito de Google. Es un servidor de aplicaciones de mapas en la Web, que ofrece imágenes de mapas desplazables, así como fotos satelitales del mundo e incluso la ruta entre diferentes ubicaciones o imágenes a pie de calle (**Google Maps**, 2013).

MipMapping

Técnica que consiste en aplicar texturas de mayor o menor tamaño a los objetos dependiendo de la distancia a la que aparezcan con el objetivo de hacer el dibujado más rápido y ahorrar memoria.

1.3 Análisis de otras soluciones existentes

De las principales soluciones existentes a nivel mundial, se analizaron las principales funcionalidades y características en aras de ver las semejanzas y diferencias que pudieran tenerse en cuenta en el desarrollo de la propuesta de solución.

EMaps

La aplicación eMaps proporciona algunas de las características más útiles de Google Maps en entorno de escritorio. eMaps es una aplicación pequeña pero efectiva que sigue las reglas de uso muy intuitivas y proporciona una amplia área para navegar a través de los mapas disponibles, mientras que permite al usuario acceder a varias capacidades útiles como obtener direcciones, orientaciones, crear ubicaciones predeterminadas o navegar por el servicio Street View (Vista por calles). Utiliza directamente las cartografías que brinda Google Maps (**Softonic**, 2013).

En cuanto a funcionamiento eMaps proporciona:

Controles básicos de la navegación. Permite ver la representación geográfica de la Tierra, mapas físicos, imágenes de satélite o una combinación de ellas. Puede acceder a representaciones de la

Luna, Martes y del cielo. Permite ver fotos, cámaras web o artículos de Wikipedia con geolocalización [4]para cada ubicación. Permite guardar el mapa actual como JPG o enviarlo a impresoras. Requiere muy pocos recursos (su archivo es inferior a 200 kb) (**Softonic**, 2013). (Ver Figura 2)

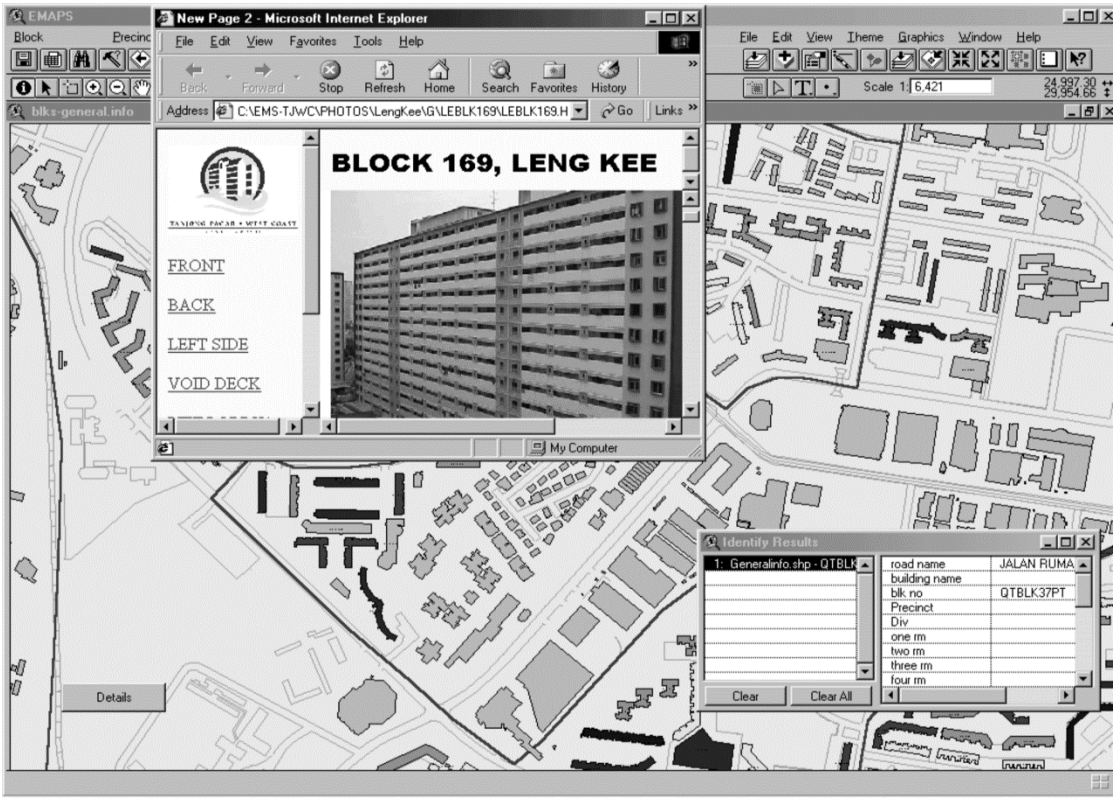


Ilustración 2. Aplicación eMaps

ArcGIS

ArcGIS Desktop es el producto principal que utilizan los profesionales SIG para compilar, usar y administrar la información geográfica. Incluye aplicaciones SIG profesionales y completas que admiten diversas tareas SIG, incluidas la representación cartográfica, la compilación de datos, el análisis, la administración de geodatabase y el uso compartido de información geográfica (**ArcGis Community**, 2011).

Los usuarios de ArcGIS Desktop realizan gran variedad de trabajo de SIG, desde creación de cartografía sencilla y compilación de datos al análisis espacial avanzado. Utilizan Desktop para trabajar

[4] Geolocalización: Hace referencia al conocimiento de la propia ubicación geográfica de modo automático.

en SIG 3D, realizar el análisis espacial, administrar imágenes y realizar procesamiento de imágenes avanzado y para automatizar muchos procedimientos de SIG. Permite el trabajo con mapas, la visualización de resultados analíticos y compilación y edición de entidades (**ArcGis Community**, 2011).

Funciones en línea de ArcGIS Desktop

Los documentos y paquetes de ArcGIS también se pueden publicar como servicios web. Si utiliza ArcGIS Desktop junto con ArcGIS Server, puede convertir cualquier mapa, geodatabase o modelo en un servicio web de SIG para compartirlo en un grupo de trabajo, en toda una empresa o abiertamente en Internet.

Además, ArcGIS Desktop incluye funciones en línea que le permiten compartir información con otros en Internet. Puede utilizarlas para comunicarse con otros usuarios.

ArcGIS Desktop es escalable y puede satisfacer las necesidades de muchos tipos de usuarios. Está disponible en tres niveles funcionales:

- ArcView se centra en el uso exhaustivo de datos, creación de mapas y análisis.
- ArcEditor agrega funciones avanzadas de edición de geodatabases [5]y creación de datos.
- ArcInfo es un escritorio de SIG completo y profesional que contiene funciones de SIG completas, con eficaces herramientas de geoprocesamiento[6].

Posee extensiones opcionales donde cada una permite agregar capacidades, como geoprocesamiento de ráster y análisis de red entre muchas otras (**ArcGis Community**, 2011). (Ver Figura 3)

[5] Geodatabases: Colección de datos actualizados de diversos tipos que se utiliza en ArcGIS y se administra en una carpeta de archivos o una base de datos relacional.

[6] Geoprocesamiento: Conjunto de tecnologías orientadas a la recopilación y tratamiento de informaciones espaciales con un objetivo específico.

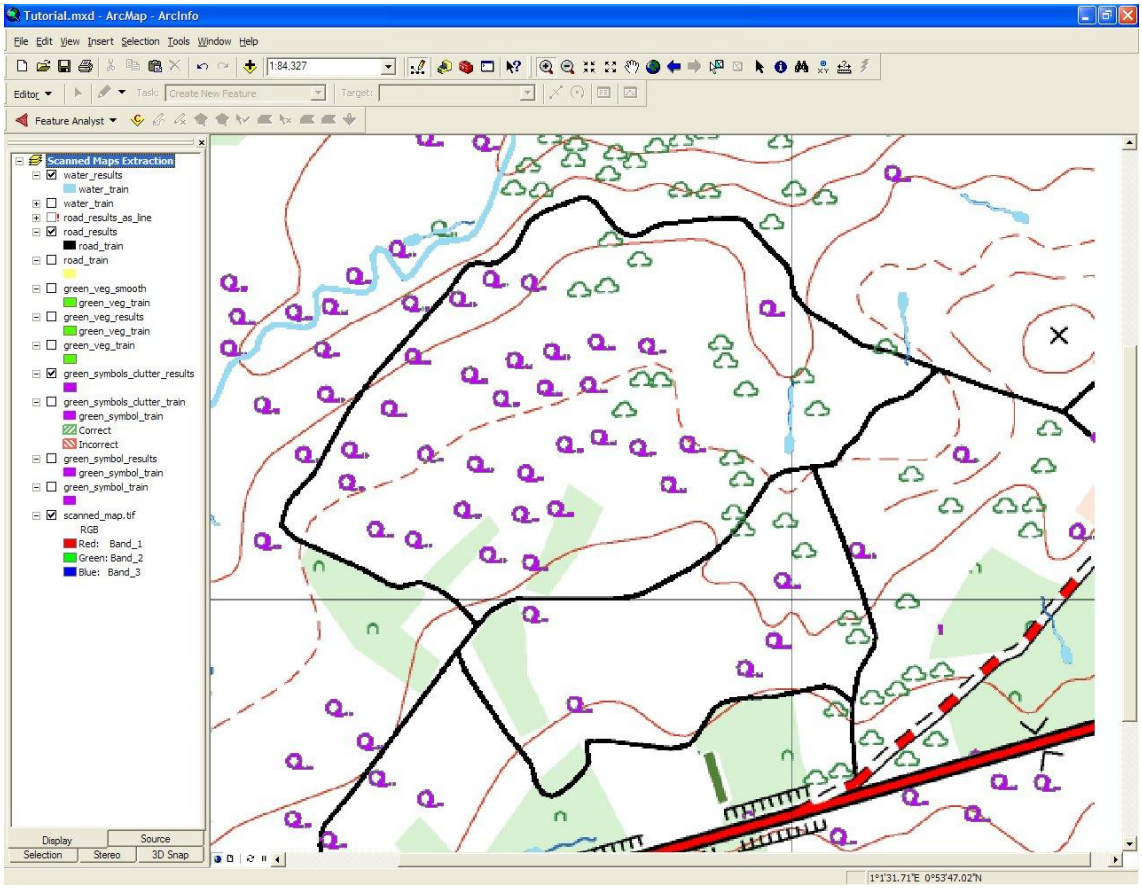


Ilustración 3. Aplicación ArcGIS

GvSIG

GvSIG Desktop es un potente Sistema de Información Geográfica (SIG) libre diseñado para dar solución a todas las necesidades relacionadas con el manejo de información geográfica. Se caracteriza por ser una solución completa, fácil de usar y que se adapta a las necesidades de cualquier usuario de SIG. Es capaz de acceder a los formatos más comunes, tanto vectoriales como ráster, tanto locales como remotos, integra estándares OGC y cuenta con un amplio número de herramientas para trabajar con información de naturaleza geográfica (consulta, creación de mapas, geoprocesamiento, redes, etc.) que lo convierten en una herramienta ideal para usuarios que trabajen con la componente territorial (Geoprocesamiento, 2013).

Algunas de sus características más destacadas son:

- Portable: funciona en distintas plataformas hardware / software, Linux, Windows y Mac OS. El lenguaje de programación es Java.
- Modular: es ampliable con nuevas funcionalidades mediante el desarrollo de extensiones, permitiendo una mejora continua de la aplicación, así como el desarrollo de soluciones a medida.
- De código abierto: licencia GNU/GPL, lo que permite su libre uso, distribución, estudio y mejora.
- Interoperable con las soluciones ya implantadas: es capaz de acceder a los datos de otros programas privativos, como ArcView, AutoCAD o Microstation sin necesidad de cambiarlos de formato.
- Internacionalizable: está disponible en más de una veintena de idiomas (castellano, inglés, alemán, italiano) y permite la incorporación de nuevos idiomas con facilidad.
- Sujeto a estándares: sigue las directrices marcadas por el Open Geospatial Consortium (OGC) (**Geoprocesamiento**, 2013).

Funcionalidades:

En gvSIG Desktop podemos encontrar un amplio abanico de funcionalidades, integrando las más diversas áreas de aplicación de los SIG:

- Vectorial: Acceso a formatos vectoriales, acceso a bases de datos, navegación, consulta, selección, análisis y geoprocesamiento, edición gráfica y alfanumérica, simbología, etiquetado, diseñador de planos, conversión de datos a otros formatos y sistemas de proyección, relaciones entre tablas, estadísticas, normalización.
- Ráster y teledetección: Acceso a formatos ráster, tabla de color y gradientes, recorte de datos y bandas, exportación de capas, procesamiento por píxel, tratamiento de interpretación de color, generación de pirámides, realces radiométricos, histogramas, geolocalización, reproyección de ráster, geo-referenciación, vectorización automática, álgebra de bandas, definición de áreas de interés, clasificación supervisada y no supervisada, árboles de decisión, fusión de imágenes, mosaicos, diagramas de dispersión.
- Infraestructuras de Datos Espaciales y estándares: Acceso a servicios remotos mediante estándares OGC (WMS, WFS, WFS-T, WCS), acceso mediante servicios no estándar (ArcIMS, Ecwp), servicio de búsqueda por catálogo, servicio de localización por nomenclador, acceso a formatos de fichero estándar, extensión de publicación de servicios OGC.
- Redes: Topología de red, gestión de paradas, camino mínimo, área de servicio, evento más cercano, matriz orígenes-destino, árbol de recubrimiento mínimo, conectividad.
- 3D: Vista 3D plana, vista 3D esférica, soporte de todos los formatos y servicios remotos de gvSIG, capas de elevación, capas vectoriales con alturas, capas 3D, posibilidad de rásterizar o visualizar como primitivas gráficas las capas vectoriales, simbología 3D, extrusión de capas vectoriales, geo-referenciación y edición de objetos 3D, encuadres 3D, sistema de Animación 3D, selección, información, visualización estéreo (anaglifo, horizontal split, etc.), visualización a pantalla completa, búsqueda geográfica por nombre (gazeeteer) (**Geoprocesamiento**, 2013) (Ver Figura 4)

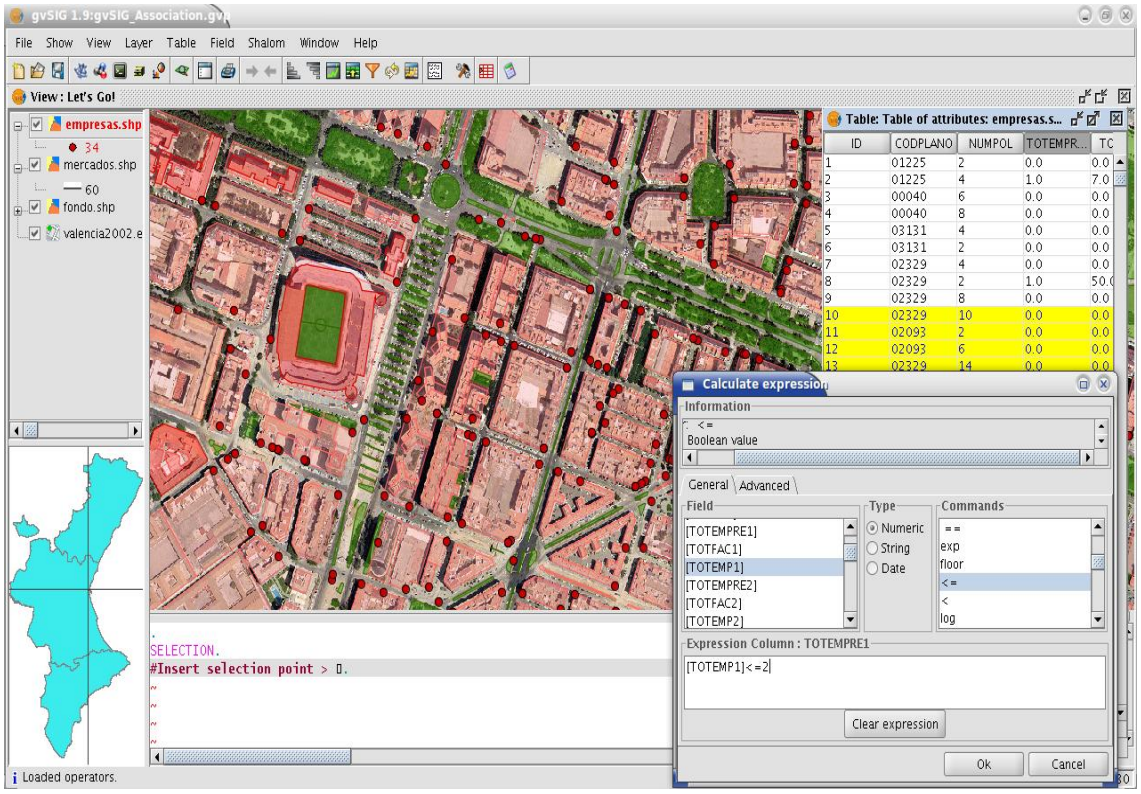


Ilustración 4. Aplicación gvSIG

1.4 Valoración general

Luego de analizar algunas de las principales soluciones existentes, se considera que presentan características similares a lo que se desea desarrollar, en cuanto a utilización de cartografías del servicio Google Maps. De igual manera si GeoQ utilizara estos sistemas para dar solución a su problema, seguiría estando presente el principio de estar utilizando un intermediario para llegar a Google Maps. Por lo que se hace imposible la utilización de forma directa de dichas soluciones, dado las características que los mismos presentan, solo pueden ser utilizados como punto de referencia para el proveedor de cartografías que se desea desarrollar debido a las funcionalidades que los mismos poseen para ser utilizadas con estos fines.

1.5 Conclusiones del capítulo

En el presente capítulo se expusieron todos los contenidos referidos al aseguramiento teórico del resto de la investigación, los principales conceptos y definiciones asociadas a los SIG, analizando la situación actual del dominio del problema, fundamentando así el marco metodológico y garantizando la continuidad del proceso investigativo.

Se evaluaron las diferentes herramientas existentes a nivel mundial y nacional, concluyendo que:

- Actualmente el proyecto GeoQ necesita desarrollar un proveedor que utilice las cartografías del servicio Google Maps para erradicar el problema de la investigación.
- Aunque existen herramientas encargadas de dar solución a la problemática planteada, las mismas no cumplen con las especificidades que se requieren para el desarrollo del problema en cuestión. Por ello se concluye que se requiere adicionar a la plataforma un proveedor de datos cartográficos ráster de Google Maps.

Capítulo 2.Tendencias actuales y tecnologías a utilizar

2.1 Introducción

En este capítulo se realiza un análisis de las tendencias y tecnologías de desarrollo actuales, entorno a la solución propuesta. Se abordan conceptos y estudios que soportan cada propuesta tecnológica y herramienta seleccionada. Se estudiará además el framework de desarrollo, leguaje de programación, Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés) y la herramienta de Ingeniería de Software Asistida por Ordenador (CASE, por sus siglas en inglés) que se utilizará, así como una valoración del porqué de la utilización de estas metodologías y herramientas en el desarrollo de la aplicación.

2.2 Lenguaje Unificado de Modelado (UML)

El Lenguaje Unificado de Modelado (UML, sus siglas en inglés), es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear, configurar, mantener y controlar la información sobre tales sistemas (**Cornejo**, 2001).

UML se utiliza para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas software como para la arquitectura de hardware donde se ejecuten. Otro objetivo de este modelado visual es lograr la independencia del lenguaje de implementación, de tal forma que los diseños realizados usando UML se puedan implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos) (**Cornejo**, 2001).

Los principales objetivos de UML son:

Visualizar: Permite expresar de una forma gráfica un sistema, de forma que otro lo puede entender.

Especificar: Permite especificar cuáles son las características de un sistema antes de su construcción.

Construir: A partir de los modelos especificados se pueden construir los sistemas diseñados.

Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

Herramienta de modelado del software

Las herramientas de Ingeniería de Sistemas Asistido por Computadoras (CASE, por sus siglas en inglés), son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste 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, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras (**Scribd 2**, 2013).

2.2.1 Visual Paradigm

Visual Paradigm es una herramienta profesional que soporta el ciclo de vida del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. UML ayuda a una más rápida construcción de aplicaciones de calidad y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación (**Free Download Manager**, 2013).

Características principales:

- Modelar procesos de negocio
- Administrar requerimientos
- Importar archivos desarrollados con Rational Rose
- Importar y exportar archivos XML

- Generar código e ingeniería inversa
- Generar una capa Objeto- Relacional fiable, escalable y de alto rendimiento.
- Modelar visualmente el diseño lógico y físico de datos
- Automatizar el mapeo entre el modelo de objetos y el modelo de datos.
- Soporta una amplia gama de bases de datos donde se incluyen: Oracle, Microsoft SQL Server, PostgreSQL, MySQL y otros. Es multiplataforma, disponible para los Sistemas Operativos Linux, Windows y Mac OS.

Fundamentación de la selección

Rational Rose es la herramienta por excelencia para RUP, pero la misma está patentada bajo licencia privativa, lo cual no se ajusta a la tecnología deseada, por lo que se hace necesario el uso de una herramienta que cumpla con los requisitos de modelado. Actualmente Visual Paradigm está bajo la licencia Community, la cual plantea que su uso es libre siempre y cuando no sea para uso comercial, constituye una de las herramientas de modelado más utilizadas en el mundo. Soporta los últimos estándares de UML. Provee mecanismos de re-ingeniería o ingeniería inversa y generación de código. Por tales razones se seleccionó dicha herramienta.

2.3 Metodologías de desarrollo de software

Durante la construcción de todo sistema o proyecto de software se corren riesgos y se presentan situaciones que en muchos casos se hacen difíciles controlar. Como respuesta a semejante problema, se hicieron estudios que devinieron en la creación de las metodologías de desarrollo de software y en diversos estándares. Dichas metodologías y estándares proporcionan las guías para un desarrollo controlado y basado en soluciones exitosas, de forma tal, que gane en agilidad el proceso y en calidad el producto. Formalmente se pueden definir estas metodologías como un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software (**Grupo de Soluciones GSInnova**, 2013).

Fundamentación de la selección

Las exigencias de los clientes actuales conllevan a la necesidad de implementar soluciones rápidas y que cumplan con los requerimientos planteados, por lo que el Desarrollo Rápido de Aplicaciones (RAD; sus siglas en inglés) es una de las características más importantes en la actualidad. Para corroborar

esto se deben utilizar herramientas basadas en este enfoque. En el caso particular del Proceso Unificado de Desarrollo (RUP; sus siglas en inglés), por el especial énfasis que presenta en cuanto a su adaptación a las condiciones de este proyecto mediante su configuración previa a aplicarse, realizando una configuración adecuada, podría llevarse a cabo de una forma ágil. Por lo tanto utilizar la metodología RUP, en la que se definan claramente los objetivos de cada fase, los entregables y sus actividades permitiría tener un entorno predictivo, en el que se agilicen las prácticas y técnicas necesarias para llevar a cabo la integración continua manteniendo diseños simples.

Según lo antes expuesto se acordó el uso de la metodología RUP orientándose a un desarrollo configurado que agilice el proceso de desarrollo.

2.3.1 Proceso Unificado de Desarrollo (RUP)

Este es un producto del proceso de ingeniería de software que proporciona un enfoque disciplinado para asignar tareas y responsabilidades dentro de una organización del desarrollo. Su meta es asegurar la producción de software de alta calidad que resuelva las necesidades de los usuarios dentro de un presupuesto y tiempo establecidos.

Los artefactos de la metodología RUP servirán como guía para documentar e implementar de una manera fácil y eficiente, dentro de las respectivas fases con las cuales cuenta. Cada fase en RUP puede descomponerse en iteraciones. Una iteración es un ciclo de desarrollo completo dando como resultado una entrega de producto ejecutable (interna o externa) (**Grupo de Soluciones GS Innova**, 2013).

Es una metodología centrada en:

- Realizar un levantamiento exhaustivo de requerimiento
- Buscar o detectar defectos en las fases iniciales.
- Reducir el número de cambios tanto como sea posible.

- Realizar el análisis y diseño, tan completo como sea posible.
- Diseño genérico, intenta anticiparse a futuras necesidades.
- El cliente no pertenece al grupo de desarrollo.

Los aspectos más importantes que definen este proceso unificado son tres:

- Dirigido por casos de uso: Basándose en los casos de uso, los desarrolladores crean y llevan a cabo una serie de modelos de diseño e implementación. Además, estos modelos se validan para que sean conformes a los casos de uso. Finalmente, los casos de uso también sirven para realizar las pruebas sobre los componentes desarrollados (**Jacobson** y otros, 2000).
- Centrado en la arquitectura: En la arquitectura de la construcción, antes de construir un edificio este se contempla desde varios puntos de vista: estructura, conducciones eléctricas, fontanería, etc. Cada uno de estos aspectos está representado por un gráfico con su notación correspondiente. Siguiendo este ejemplo, el concepto de arquitectura de software incluye los aspectos estáticos y dinámicos más significativos del sistema (**Jacobson** y otros, 2000).
- Iterativo e incremental: Todo sistema informático complejo supone un gran esfuerzo que puede durar desde varios meses hasta años, por lo tanto, lo más práctico es dividir un proyecto en varias fases. Actualmente se suele hablar de ciclos de vida en los que se realizan varios recorridos por todas las fases. Cada recorrido se denomina iteración en el proyecto en la que se realizan varios tipos de trabajo (denominados flujos). Además, cada iteración parte de la anterior incrementado o revisando la funcionalidad implementada (**Jacobson** y otros, 2000).

RUP se divide en 4 fases (Ver Figura 5):

- Conceptualización (Concepción o Inicio): El objetivo de esta fase es determinar la visión del proyecto. Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema (**Pressman**, 1999).
- Elaboración: El objetivo es determinar la estructura óptima. Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los

requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido (**Pressman**, 1999).

- Construcción: El objetivo de esta fase es obtener la capacidad operacional inicial. Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtiene también uno o varias liberaciones del producto que han pasado las pruebas. Se ponen estas liberaciones a consideración de un subconjunto de usuarios (**Pressman**, 1999).

- Transición: El objetivo es obtener la primera versión del proyecto (liberación). Puede implicar reparación de errores (**Pressman**, 1999).

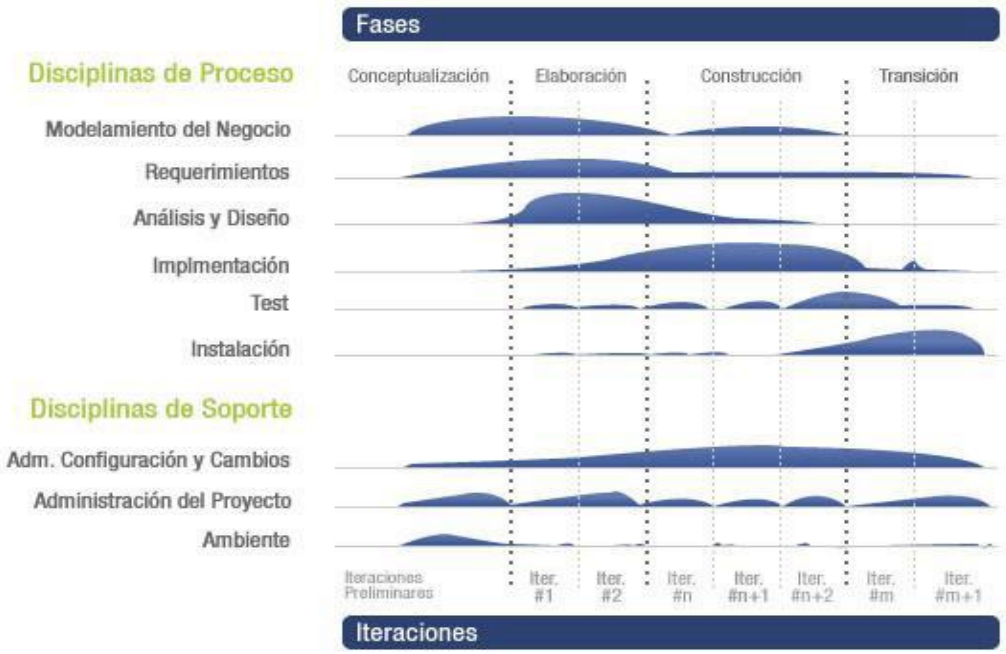


Ilustración 5. Imagen representativa de la estructura de RUP.

2.4 Lenguaje de Programación.

Un lenguaje de programación es un conjunto de símbolos, reglas sintácticas y semánticas, mediante las cuales el programador puede interactuar con el hardware para obtener un resultado observable. Existen diversos lenguajes, pero ninguno de ellos se puede considerar el más capacitado o el más adecuado para la creación de aplicaciones ya que el leguaje no determina lo que se programará. En el presente epígrafe se aborda el lenguaje propuesto para dar solución a los objetivos planteados y los indicadores que posibilitaron su selección.

2.4.1 Lenguaje C++.

El lenguaje de programación C++ es uno de los más empleados en la actualidad. Se puede decir que C++ es un lenguaje híbrido ya que permite programar tanto en estilo procedimental (como si fuese C), como en estilo orientado a objetos, como en ambos a la vez. Además, también se puede emplear mediante programación basada en eventos para crear programas que usen interfaz gráfica de usuario (**González**, 2008).

Las principales ventajas que presenta el lenguaje C++ son:

- Versatilidad: C++ es un lenguaje de propósito general, por lo que se puede emplear para resolver cualquier tipo de problema.
- Eficiencia: C++ es uno de los lenguajes más rápidos en cuanto ejecución.

2.5 Entorno Integrado de Desarrollo

Un Entorno Integrado de Desarrollo (IDE, por sus siglas en inglés), no es más que un editor de código que además puede servir para depurar y facilitar las diferentes tareas necesarias en el desarrollo de cualquier tipo de aplicación. Una de las tareas más notables de cualquier desarrollador, lo constituyen la selección del lenguaje y entorno de desarrollo. En muchos casos esta selección define el propio alcance y calidad del proyecto final. En el presente epígrafe se presentan los fundamentos tomados en

cuenta para la selección del entorno de desarrollo según las capacidades de integración con el lenguaje de implementación propuesto (**Sitio web de la EU de Ingeniería Técnica de Ovideo**, 2013).

2.5.1QtCreator

Qt dispone de cuatro grandes ventajas:

- Qt es completamente gratuito para aplicaciones de código abierto aunque también se utiliza para aplicaciones comerciales.
- Las herramientas, librerías y clases están disponibles tanto para casi todas las plataformas Unix y sus derivados (como Linux, MacOS X, Solaris, entre otros) como también para la familia Windows, por lo que una aplicación puede ser compilada y utilizada en cualquier plataforma sin necesidad de cambiar el código.
- Qt tiene una extensa librería con clases y herramientas para la creación de aplicaciones elegantes.
- La Interfaz de programación de aplicaciones (API; por sus siglas en inglés) de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL, así como uso de XML, gestión de hilos, soporte de red, una API multiplataforma unificada para la manipulación de archivos, de ficheros y de estructuras de datos tradicionales (**Nokia. QT**, 2013).
- Qt Creator es una nueva plataforma de entorno de desarrollo integrado disponible junto con las bibliotecas Qt, bajo licencia LGPL. QtCreator está diseñado para el trabajo con bibliotecas Qt para sus versiones 4.x, en este caso se define utilizar la versión 4.7 (**Gómez**, 2010).

Entre las principales características de Qt Creator se pueden mencionar las siguientes:

- Resaltado de sintaxis y completado de código.
- Control de código estático y consejos de estilo a medida que se escribe.

- Apoyo a la refactorización código fuente.
- Ayuda sensible al contexto.
- Coincidencia de paréntesis y los modos de selección de paréntesis.
- Capacidades de edición avanzada (**Gómez**, 2010).

Además de las características antes mencionadas se debe resaltar que las bibliotecas Qt son multiplataforma y las aplicaciones que se apoyan en ellas tienen buena respuesta y un consumo de recursos aceptable.

2.5.2 QT

Qt es una plataforma de desarrollo que incluye clases, librerías y herramientas para la producción de aplicaciones de interfaz gráfica en C++ entre otras funcionalidades. Está distribuida bajo los términos de GNU Lesser General Public License, es software libre y de código abierto. Qt es multiplataforma e incluye soporte de nuevas tecnologías como OpenGL, XML, bases de datos y programación para redes. QT dispone de una amplia gama de herramientas que facilitan entre otras cosas la creación de formularios, botones y ventanas de diálogo con el uso del ratón. Además provee soporte para la internacionalización de su contenido (**Gómez**, 2010).

2.6 Conclusiones Parciales

La selección de estas herramientas y tecnologías se llevó a cabo teniendo en cuenta el cumplimiento de los principios establecidos por la UCI, debido a que están desarrolladas bajo la licencia GNU GPL, lo que facilita la continuación de su desarrollo, garantizando en gran medida la obtención de la documentación técnica asociada al sistema. Por lo que los autores del presente trabajo concluyen que:

- La utilización de una metodología robusta como RUP permite la generación de los artefactos necesarios para su incorporación a la plataforma GeoQ.

- Visual Paradigm representa una colección premiada de herramientas que facilitan las organizaciones visuales y el diagrama de diseño.
- La utilización de tecnologías libres garantizará mayor rendimiento del producto final.

Capítulo 3.Descripción de la Solución Propuesta

3.1 Introducción

En el presente capítulo se llevará a cabo una descripción de la solución propuesta; teniendo en cuenta la indefinición no significativa de los procesos elementales del negocio, se realizará un modelo de dominio, así como la descripción del mismo. Se identificarán los requisitos funcionales y no funcionales que debe tener el sistema para su correcto funcionamiento. Seguidamente se presentará la solución a través de la vista del sistema, definiendo actores y casos de usos del sistema, así como una breve descripción de los mismos, se mostrará además el Diagrama de Casos de Uso del Sistema (DCUS en lo adelante).

3.2 Modelo del Negocio

El modelado del negocio logra crear una visión más profunda de la organización a la que se le realiza la automatización, permitiendo definir los procesos, roles y responsabilidades en los modelos de casos de uso del negocio. Permite comprender la estructura y la dinámica de la organización en la cual se va a implantar el sistema, además de los problemas actuales de dicha organización e identificar así las mejoras potenciales. Realizar un modelado del negocio tiene como ventaja que se puedan derivar los requerimientos del sistema que va a soportar la organización (**Jacobson**, 2000).

Sin embargo, la investigación no está enmarcada en las necesidades de un cliente específico ni tiene por objetivo la creación de una aplicación independiente en la que deban identificarse procesos a automatizar sino, en la incorporación de una nueva funcionalidad para la aplicación GeoQ. No se concibe la necesidad de definir un Modelo del Negocio por lo que se procederá a trabajar en el desarrollo del Modelo del Dominio.

Se utiliza el término Modelo de Dominio para distinguirlo del Modelo de Negocio ya que este último es mucho más abarcador. El Modelo de Dominio centra su atención en una parte del Modelo del Negocio la relacionada con el ámbito del proyecto.

3.3 Modelo del Dominio

El Modelo de Dominio permite de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio del problema. Contiene las clases conceptuales del mundo real. Puede utilizarse para obtener y expresar el entendimiento de un proceso en análisis como antesala del diseño

de un sistema ya sea de software o de otro tipo. El Modelo de Dominio se presenta en forma de diagrama de clases donde se exponen los principales conceptos y roles del sistema en cuestión (**Jacobson**, 2000).

3.3.1 Definiciones, Acrónimos y Abreviaturas del Modelo de Dominio

- Google Maps*:* Aplicación web Google Maps que proporciona cartografías en formato ráster.
- Cuadrados*:* Representa las losas de tamaño 256x256 que conforman las cartografías que provee Google Maps.
- Modelo Ráster*:* Es el modelo de datos complementario al modelo de datos vectorial, tiene como principal característica el llevar a cabo una representación "discreta" del mundo real, empleando una malla de rejillas regulares.
- Ráster*:* Representa un formato ráster el mismo se compone de filas y columnas de celdas, donde cada celda almacena un valor único.
- Cartografía*:* La cartografía constituye un conjunto de operaciones que permiten, a partir de observaciones y mediciones, la representación de una parte o la totalidad de la Tierra.
- Mapas*:* Representación geográfica de una parte de la superficie terrestre, en la que se da información relativa a una ciencia determinada.
- Capas*:* Elementos que superpuestos conforman un mapa, definen por separado la información a representar del mismo.
- Proyección*:* Define la proyección con que se representará la imagen de salida del mapa.
- Escala*:* Línea recta dividida en partes iguales que representan unidades de medida, sirve para dibujar proporcionadamente las distancias y dimensiones en un mapa, para luego calcular las medidas reales con respecto a lo dibujado.
- Leyenda*:* Texto o símbolo que acompaña a un mapa y explica su contenido, objeto que define la leyenda del mapa.

Las relaciones existentes entre los conceptos asociados al ámbito del sistema se reflejan en el correspondiente Diagrama de Clases del Modelo del Dominio.

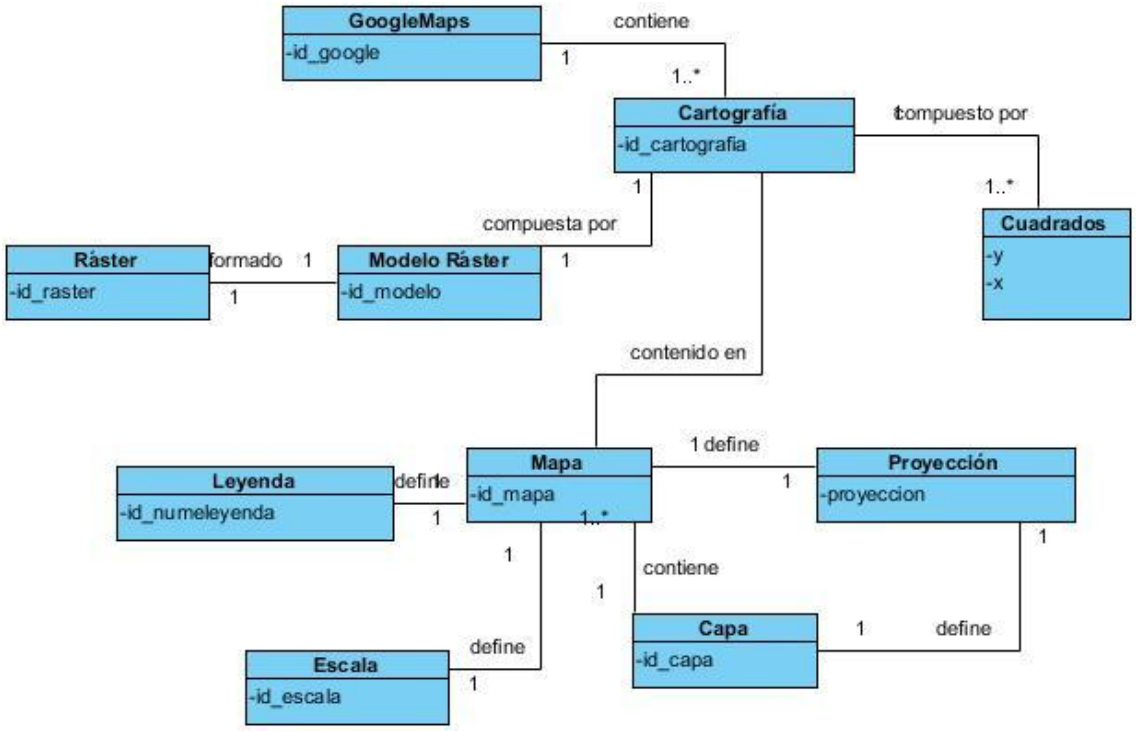


Figura 2.Diagrama de Clases del Modelo de Dominio

3.3.2 Descripción del Modelo de Dominio

El cliente es el usuario que interactúa con la plataforma, esta responde a las necesidades del mismo realizando las operaciones básicas de un SIG: visualización, interpretación, procesamiento y adquisición de los datos contenidos en mapas y cartografías. Los mapas definen leyenda, escala y proyección, mientras que las cartografías están compuestas por modelos ráster que a su vez tienen cuadrados (tiles), estas cartografías son adquiridas desde la aplicación Google Maps.

3.4 Requisitos Funcionales

Los requisitos funcionales establecen el comportamiento del sistema y muestran a su vez cómo los casos de uso serán llevados a la práctica. Son además condiciones o capacidades que el sistema debe cumplir (**Jacobson**, 2000). Los requisitos funcionales propuestos para el proveedor de MipMapping son:

RF1: Crear conexión a Google Maps.
RF2: Descargar los tiles (cuadrados).
RF3: Mostrar el mapa.

RF4: Guardar el mapa

3.5 Requisitos no Funcionales

Un requisito no funcional o atributo de calidad, especifica características del sistema, como: dependencias de la plataforma, facilidad de mantenimiento, extensibilidad y fiabilidad. Se refieren además a todos los requisitos que no describen información a guardar, ni funciones a realizar (**Jacobson**, 2000). Los requisitos no funcionales propuestos para el proveedor de MipMapping son:

Usabilidad

- El sistema podrá ser usado por personas con conocimientos básicos en el manejo de computadoras. Se emplearán componentes que indiquen al usuario el estado de los procesos que por su complejidad requieran de un tiempo de procesamiento apreciable.
- El software tendrá siempre visible la opción de Ayuda, lo que posibilitará un mejor aprovechamiento por parte de los usuarios de sus funcionalidades.
- El Sistema tendrá una correcta Arquitectura de la Información, a partir de un estudio de usuarios para su etiquetado e indexado.
- Las funcionalidades principales del sistema estarán orientadas a íconos para un mayor reconocimiento por parte del usuario.

Fiabilidad

- La herramienta de implementación a utilizar debe tener soporte para recuperación ante fallos y errores.
- Debido a la arquitectura que presenta el sistema, siendo más robusto al no tratarse de un sistema de gestión que requiera mantenimiento y optimización en el almacenamiento, se estima un tiempo promedio de 6 meses entre posibles fallas.

Eficiencia

- El tiempo de respuesta estará dado por la cantidad de información a procesar; entre mayor cantidad de información mayor será el tiempo de procesamiento. Para un mapa con extensión municipal, el tiempo de respuesta promedio es menor que 1 segundo. Para una cartografía de

extensión de datos igual a Cuba, por cada capa podría demorarse hasta 2 y 3 segundos dependiendo necesariamente de la cantidad de objetos de la misma.

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

Soporte

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

Interfaz

Interfaz de Usuario

El sistema debe:

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

Interfaces de Hardware

Para las PC clientes:

- Se requiere al menos 128 MB de memoria RAM.
- Se requiere al menos 40 GB de disco duro. Para instalación y/o trabajo con mapas y almacenarlos en directorios de carpetas.
- Procesador 512 MHz como mínimo.

Interfaces de Software

La construcción de la aplicación funcionará bajo los conceptos de arquitectura basada en componentes y orientada a objetos.

El sistema GeoQ es un sistema multiplataforma que puede ejecutarse en los siguientes sistemas operativos:

- GNU/Linux, en su versión 12.4.
- Wwindows XP
- Windows Server 2000 o superior.

Interfaz de Comunicación

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

Requisitos de Licencia

GeoQ se publica bajo la licencia pública general de GNU (GNU GPL). Desarrollar este producto bajo esta licencia permite la inspección y modificación del código fuente.

Integridad

La información manejada por el sistema es objeto de cuidadosa protección contra la corrupción y estados inconsistentes, de la misma forma es considerada igual a la fuente o autoridad de los datos. Se incluye también mecanismos de chequeo de integridad y realización de auditorías por personal calificado de la entidad.

Disponibilidad

La información y las funcionalidades del sistema estarán disponibles y el usuario podrá acceder a ellas las 24 horas de los 7 días de la semana.

3.6 Descripción de los actores del sistema

El modelo de casos de uso del sistema permite a los desarrolladores de software y a los clientes llegar a un acuerdo sobre las condiciones y posibilidades que debe cumplir el mismo. Este modelo mediante la representación de uno o más actores describe lo que debe hacer el sistema para cada usuario (**Jacobson**, 1998).

3.7 Descripción de los actores del sistema

Los actores son cualquier instancia que intercambia datos con el sistema, puede ser un usuario, hardware externo e incluso otro sistema. Un actor siempre tendrá interacción con el sistema ya sea

inicializando el caso de uso o intercambiando información (**Jacobson**, 1998). A continuación se describe el actor que interactuará con el sistema:

Actores del Sistema	Descripción
Usuario	Es la persona que utiliza y manipula las funcionalidades del sistema.

3.8 Casos de Uso del Sistema

Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema (**Jacobson**, 1998).

Un diagrama de casos de uso del sistema representa gráficamente a los procesos y su interacción con los actores.

CUS1 Cargar proveedor de MipMapping.

CUS2 Guardar Mapa.

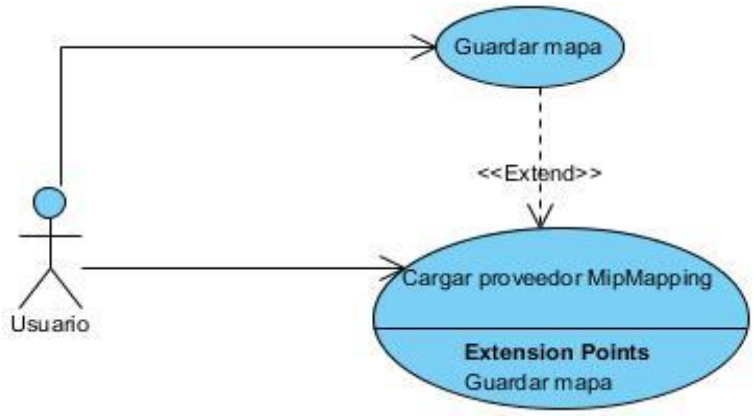


Ilustración 7. Diagrama de Casos de uso del Sistema

3.9 Descripción textual del caso de uso Cargar proveedor de MipMapping

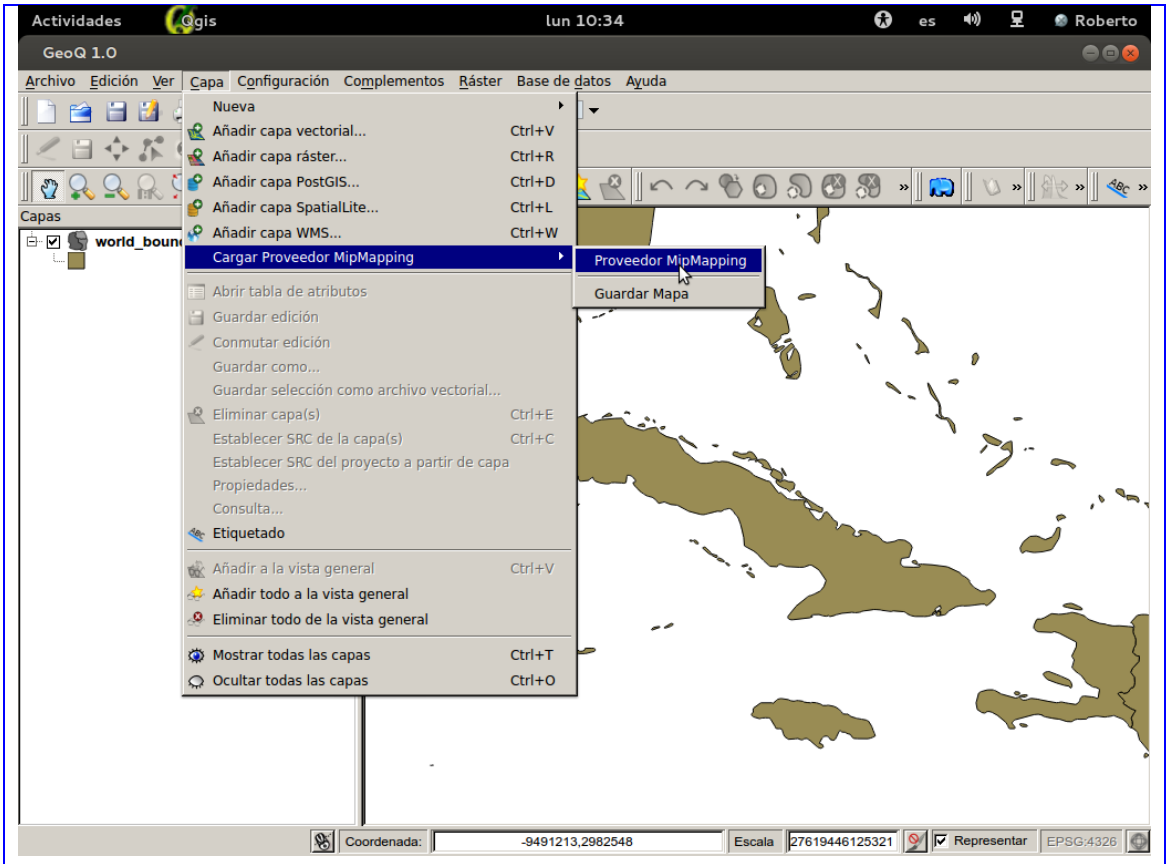
Caso de uso	Cargar proveedor de MipMapping

Actores	Usuario
Resumen	El caso de uso inicia cuando el usuario selecciona, la opción “Cargar proveedor de MipMapping”. Esta funcionalidad permite en la aplicación GeoQ cargar una cartografía ráster, concluyendo así el caso de uso.
Precondiciones	Debe estar cargada al menos una capa.
Referencias	**RF 1, RF 2, RF 3.**
Prioridad	Secundario

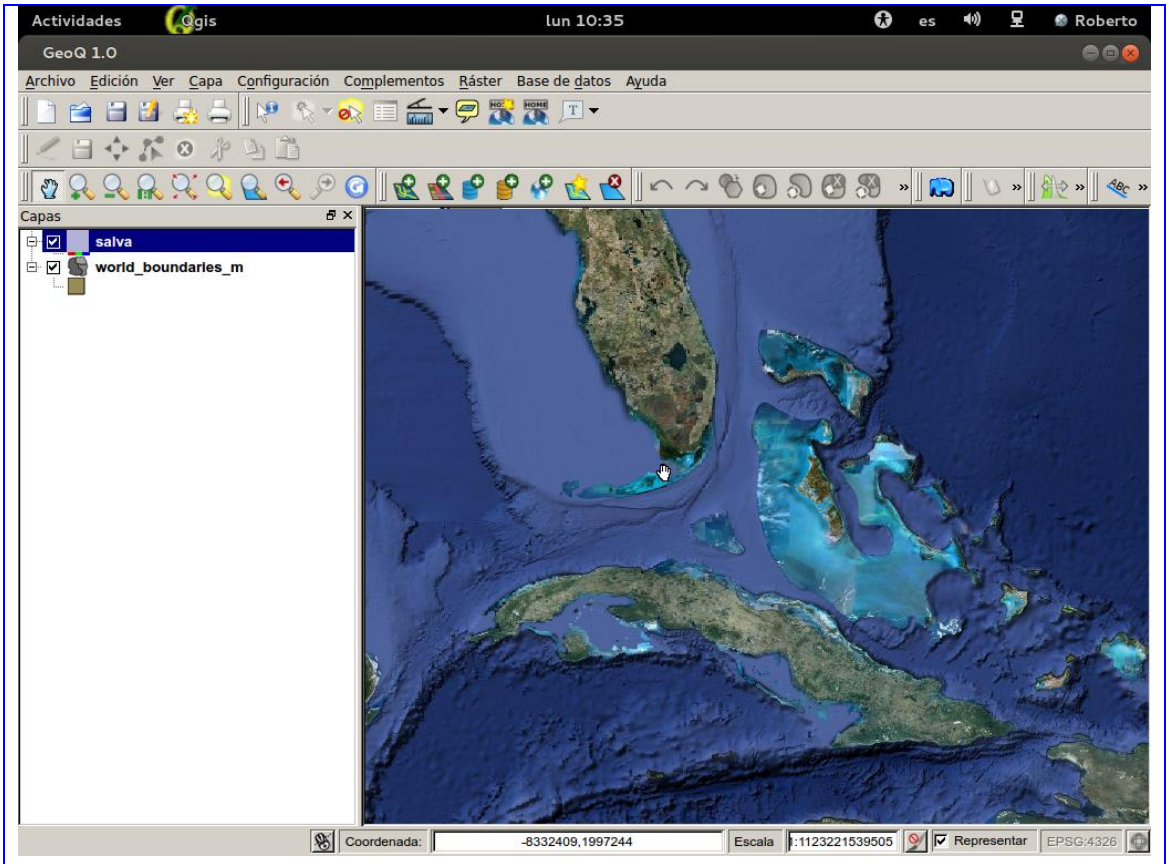
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
El caso de uso se inicia cuando el usuario hace clic izquierdo en el menú ¨Capas¨, selecciona la opción ¨Cargar proveedor de MipMapping¨ y escoge la opción ¨Proveedor MipMapping¨. Ver interfaz 1.	El sistema muestra la nueva cartografía y termina el caso de uso. Ver interfaz 2.

Prototipo de Interfaz

Interfaz 1


Actividades
Qgis
lun 10:34
es
Roberto
GeoQ 1.0
Archivo Edición Ver Capa Configuración Complementos Ráster Base de datos Ayuda
Nueva
Añadir capa vectorial... Ctrl+V
Añadir capa ráster... Ctrl+R
Añadir capa PostGIS... Ctrl+D
Añadir capa SpatialLite... Ctrl+L
Añadir capa WMS... Ctrl+W
Cargar Proveedor MipMapping
Proveedor MipMapping
Guardar Mapa
Abrir tabla de atributos
Guardar edición
Conmutar edición
Guardar como...
Guardar selección como archivo vectorial...
Eliminar capa(s) Ctrl+E
Establecer SRC de la capa(s) Ctrl+C
Establecer SRC del proyecto a partir de capa
Propiedades...
Consulta...
Etiquetado
Añadir a la vista general Ctrl+V
Añadir todo a la vista general
Eliminar todo de la vista general
Mostrar todas las capas Ctrl+T
Ocultar todas las capas Ctrl+O
Capas
world_boun
Coordenada:
-9491213,2982548
Escala
27619446125321
Representar
EPSG:4326

Interfaz 2



Flujo Alterno	
Poscondición	Se muestra el resultado de la búsqueda.

Tabla 1.Descripción del CUS Cargar proveedor de MipMapping

3.10 Descripción textual del caso de uso Guardar Mapa.

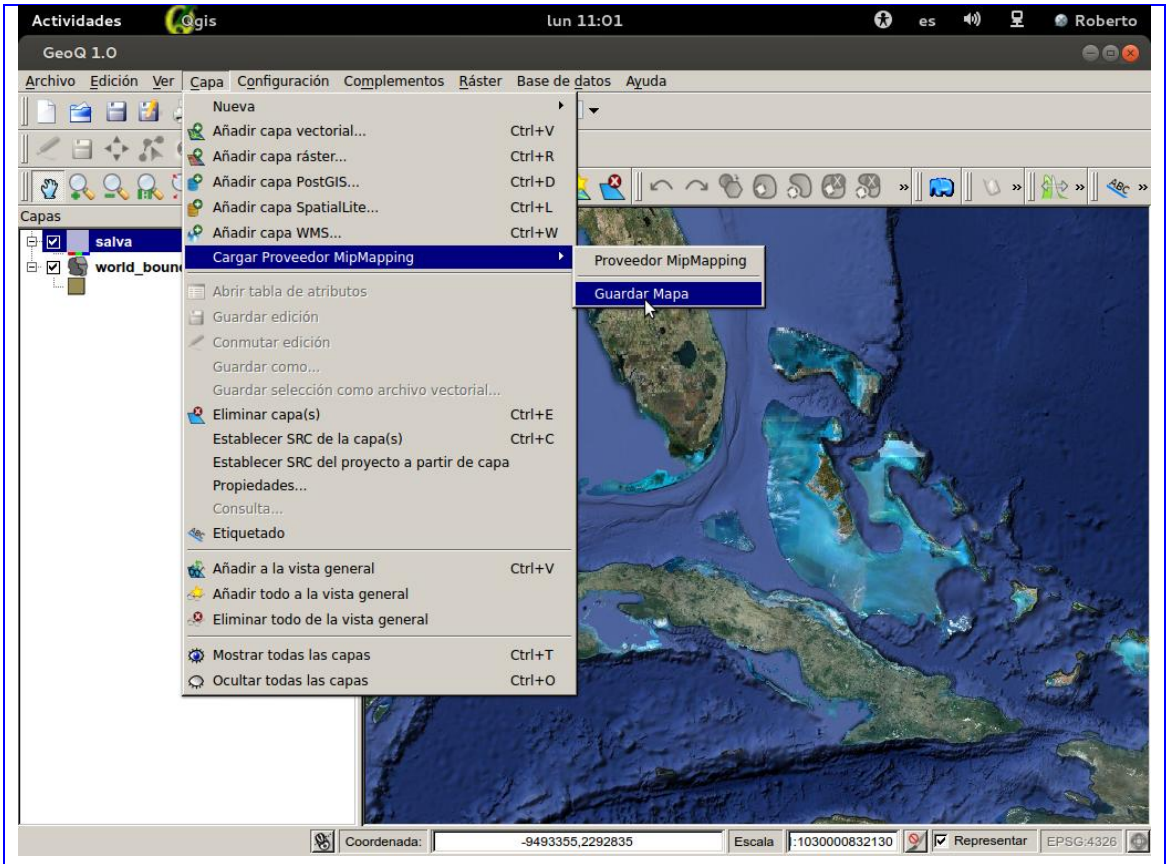
Caso de uso	Buscar activo del sistema de guardia
Actores	Usuario
Resumen	El caso de uso inicia cuando el usuario selecciona, la opción “Guardar Mapa”. Esta funcionalidad permite en

	la aplicación GeoQ guardar el mapa que el usuario está viendo en ese momento, el usuario introduce los datos necesarios para guardar la imagen, concluyendo así el caso de uso.
Precondiciones	Debe estar cargada al menos una capa.
Referencias	**RF 4.**
Prioridad	Secundario

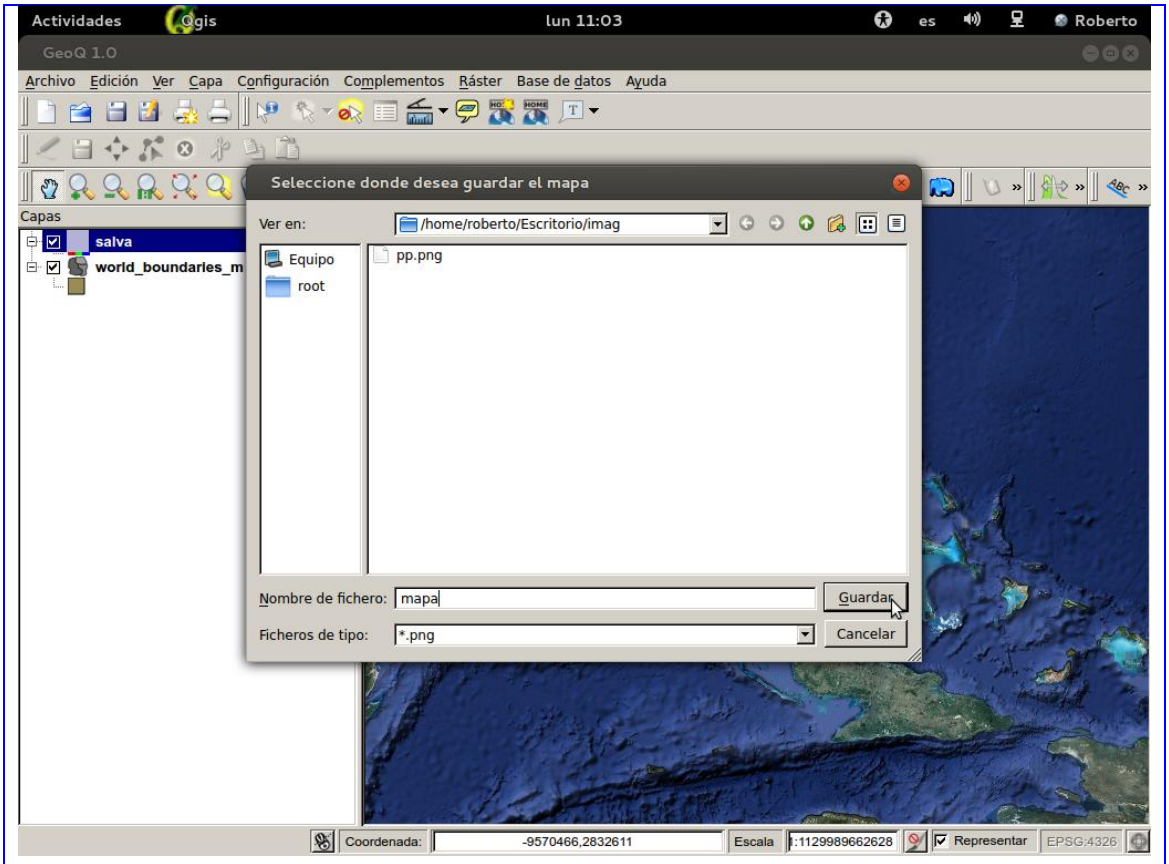
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el usuario hace clic izquierdo en el menú ¨Capas¨, selecciona la opción ¨Guardar Mapa¨ y escoge la opción ¨Guardar Mapa¨. Ver interfaz 1.	El sistema muestra la interfaz para guardar el mapa. Ver interfaz 2.
El usuario introduce el destino para guardar el mapa y hace clic izquierdo en el botón aceptar.	El sistema ejecuta la acción y termina así el caso de uso.

Prototipo de Interfaz

Interfaz 1


Actividades
Qgis
lun 11:01
es
Roberto
GeoQ 1.0
Archivo Edición Ver Capa Configuración Complementos Ráster Base de datos Ayuda
Nueva
Añadir capa vectorial... Ctrl+V
Añadir capa ráster... Ctrl+R
Añadir capa PostGIS... Ctrl+D
Añadir capa SpatialLite... Ctrl+L
Añadir capa WMS... Ctrl+W
Cargar Proveedor MipMapping
Proveedor MipMapping
Guardar Mapa
Abrir tabla de atributos
Guardar edición
Conmutar edición
Guardar como...
Guardar selección como archivo vectorial...
Eliminar capa(s) Ctrl+E
Establecer SRC de la capa(s) Ctrl+C
Establecer SRC del proyecto a partir de capa
Propiedades...
Consulta...
Etiquetado
Añadir a la vista general Ctrl+V
Añadir todo a la vista general
Eliminar todo de la vista general
Mostrar todas las capas Ctrl+T
Ocultar todas las capas Ctrl+O
Capas
salva
world_boun
Coordenada:
-9493355,2292835
Escala
:1030000832130
Representar
EPSG:4326

Interfaz 2

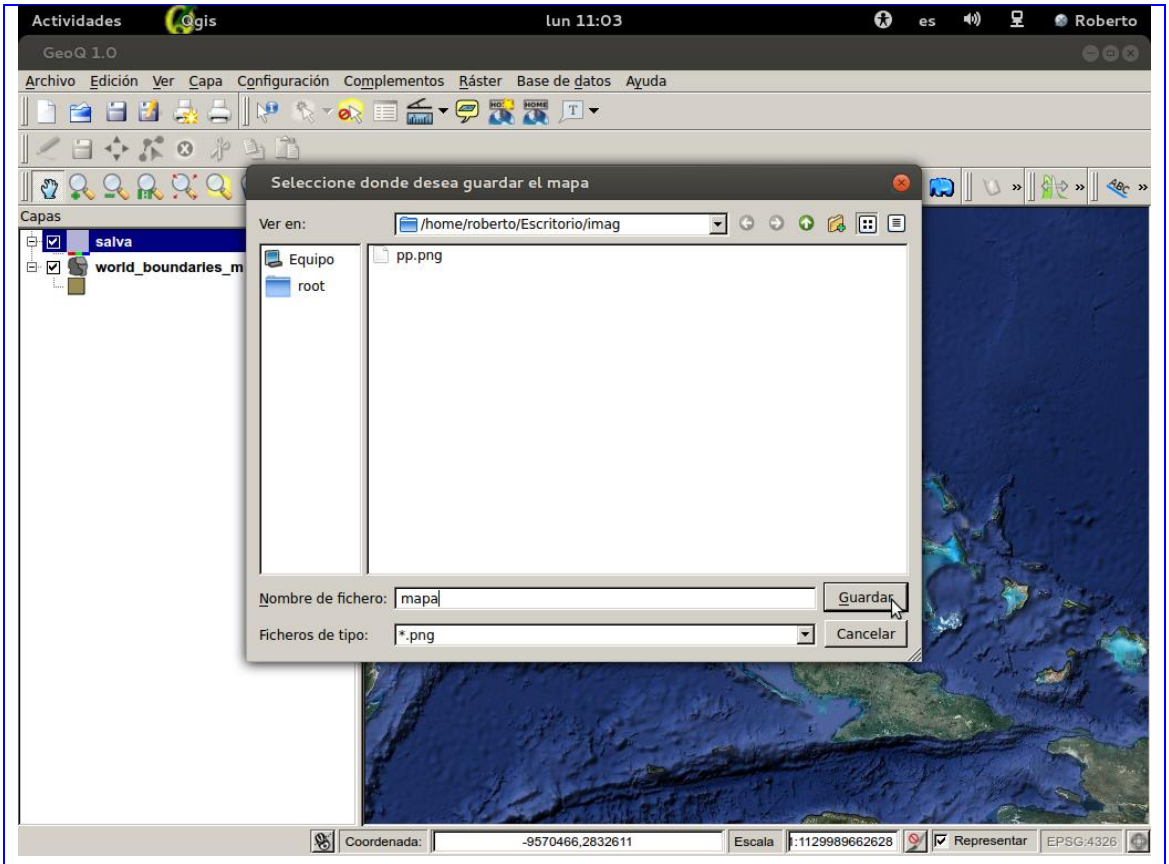


Flujo Alterno

3.1 El usuario hace clic izquierdo en el botón de cancelar.	4.1 El sistema ejecuta la acción y termina el caso de uso.

Prototipo de Interfaz

Interfaz 1



Poscondición	Se muestra el resultado de la búsqueda.

Tabla 2.Descripción del CUS Guardar Mapa.

3.11 Descripción de la Arquitectura

La arquitectura de software es una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se le percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión (**Clements**, 1996).

3.11.1 Arquitectura Orientada a Objetos

"Los componentes de un sistema encapsulan los datos y las operaciones que deben aplicarse para manejar los datos. La comunicación y coordinación entre componentes se consigue mediante el paso de mensajes" (**Pressman**, 2005).

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

3.11.2 Arquitectura Basada en Componentes

"Un componente es una unidad de composición de aplicaciones software, que posee un conjunto de interfaces y un conjunto de requisitos y que ha de poder ser desarrollado, adquirido, incorporado al sistema y compuesto con otros componentes de forma independiente, en tiempo y espacio" (**Fuentes**, et al., 2003).

"Un componente de software, es una unidad de composición con interfaces especificadas contractualmente y dependencias del contexto explícitas. Los componentes en el sentido estilístico son las unidades de modelado, diseño e implementación. Las interfaces están separadas de las implementaciones y las interfaces y sus interacciones son el centro de incumbencias en el diseño arquitectónico. Los componentes soportan algún régimen de introspección, de modo que su funcionalidad y propiedades puedan ser descubiertas y utilizadas en tiempo de ejecución." (**Reynoso** y otros, 2004).

El Desarrollo de Software Basado en Componentes (DSBC), aboga por el principio de reutilización del software, donde la descomposición de la aplicación en componentes funcionales respalda que los componentes sean implementados de forma que se logre su manipulación sobre otros sistemas.

3.12 Patrones de Diseño

Los patrones de diseño representan un esquema de las soluciones a problemas en el desarrollo de software. Brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos comunes. Se debe tener presente los siguientes elementos de un patrón:

nombre, problema (cuando aplicar un patrón), solución (descripción abstracta del problema) y consecuencias (costos y beneficios) (**Tedeschi**, 2011).

Los patrones se dividen en (**Tedeschi**, 2011):

- Patrones GRASP.
- Patrones GOF.

3.12.1 Patrones GRASP

El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el *software* orientado a objetos. GRASP es un acrónimo que significa *General Responsibility Asignment Software Patterns* (patrones generales de software para asignar responsabilidades). Los patrones GRASP describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objetos esenciales y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable.

Patrón Experto

El patrón experto es el principio básico de asignación de responsabilidades. Indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtendrá un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento).

Patrón Alta cohesión y Bajo acoplamiento.

Los conceptos de cohesión y acoplamiento están íntimamente relacionados. Un mayor grado de cohesión implica un menor grado de acoplamiento. Maximizar el nivel de cohesión en todo el sistema resulta en una minimización del acoplamiento.

Alta cohesión

Plantea que la información que almacena una clase debe de ser coherente y debe estar relacionada con la clase.

Bajo acoplamiento

Este patrón plantea mantener las clases lo menos ligadas entre sí, de tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización y disminuyendo la dependencia entre las mismas.

Creador

El patrón creador guía la asignación de responsabilidades relacionadas a la creación de objetos, una tarea muy común en sistemas orientados a objetos. El intento básico del patrón creador es encontrar un creador que necesite estar conectado al objeto creado en un evento en particular. Separa la construcción de un objeto complejo de su representación, de forma que el mismo proceso de construcción pueda crear diferentes representaciones.

3.12.2 Patrones GOF

Los patrones GOF (Gang Of Four) están dirigidos al desarrollo de sistemas orientados a objetos, se encuentran divididos en los siguientes grupos:

Creación: utilizado en la abstracción de cómo es creado un objeto.

Estructura: indican cómo se encuentran compuestas las clases.

Comportamiento: utilizado en la asignación de responsabilidades en las clases.

Patrón creacional Singleton

Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. El patrón es utilizado en la clase GeoQApp.

3.13 Conclusiones Parciales

En el presente capítulo se describió de forma estructural el modelo de dominio. Se obtuvieron los requisitos funcionales y no funcionales, concluyendo que:

- Con la identificación de los requisitos funcionales y no funcionales se logró una breve descripción de las características que presenta el sistema, permitiendo la creación del Diagrama de Caso de Uso del Sistema, el cual presenta un alto grado de información para el sistema que se desea desarrollar.

- Los requisitos funcionales nos brindan de forma gráfica una información más detallada de lo que se desea desarrollar, proporcionando un punto de partida para el desarrollo del sistema.
- Con la realización del Diagrama de Caso de Uso del sistema y correspondiente descripción se garantizará una correcta implementación del sistema deseado en su posterior etapa de desarrollo.

Capítulo 4. Construcción de la Solución Propuesta

4.1 Introducción

En el presente capítulo se construye la propuesta de solución para controlar y gestionar el flujo informativo que se ha generado. Esta propuesta incluye los modelos de análisis y diseño, los cuales dejarán sentadas las bases para comenzar con la implementación de la nueva versión de la solución del sistema. Esto se logrará a través de los diferentes diagramas de clases que incluyen dichos modelos. Además, se explican los patrones de diseño empleados para garantizar la viabilidad y calidad requeridas por el modulo a implementar.

4.2 Modelo de Diseño

El diseño del software es un proceso de varios pasos que se clasifican dentro de uno mismo. En general, la actividad del diseño se refiere al establecimiento de la estructura de datos, la arquitectura general del software y las representaciones de interfaz y algoritmos. El proceso de diseño traduce los requisitos en una representación de software (**Pressman**, 1999).

El modelo de diseño pretende crear un plano del modelo de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases. Procura lograr la descomposición de los trabajos de implementación en partes manejables a través de una comprensión de los aspectos relacionados con los requisitos funcionales y no funcionales unidos a las restricciones de los lenguajes de programación, sistemas operativos y tecnologías de distribución.

4.2.1 Diagrama de clases del diseño

Un diagrama de clases del análisis es un artefacto en el que se representan los conceptos en un dominio del problema. Representa además, el funcionamiento del mundo real, no de la implementación automatizada del mismo (**Pressman**, 1999).

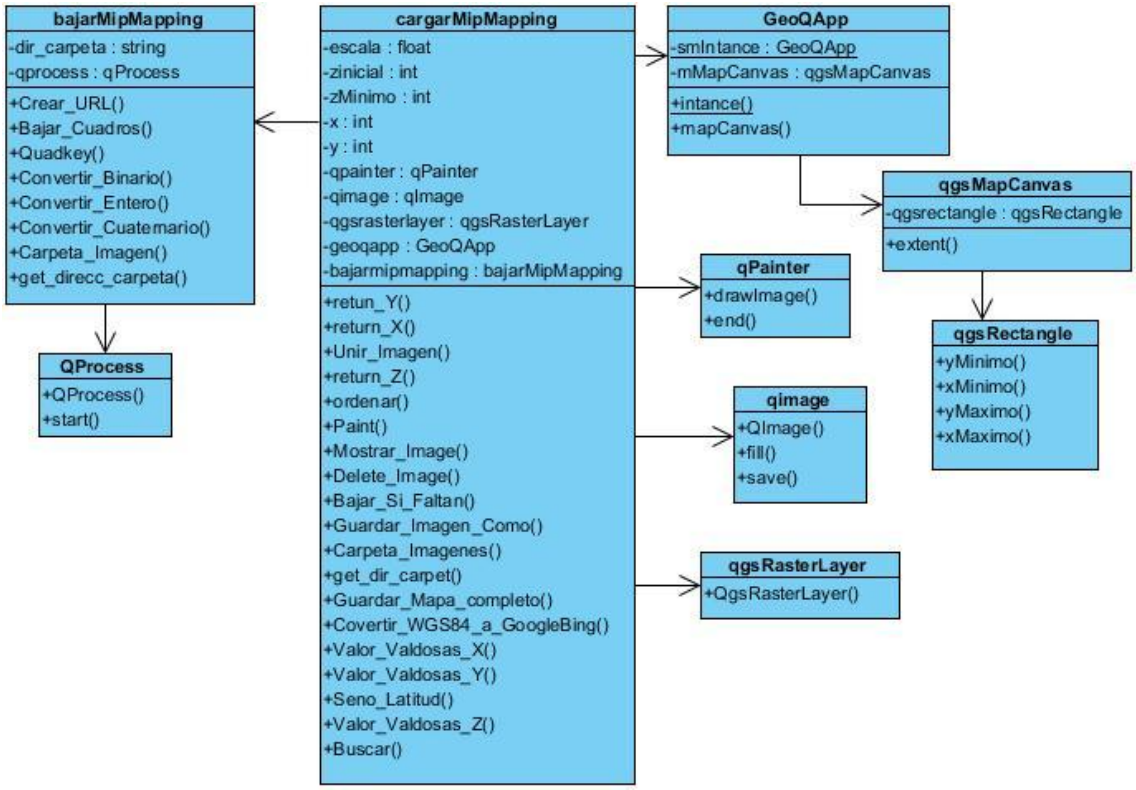


Ilustración 8. Diagramas de clases del diseño.

4.2.2 Descripción del Diagrama de clases del diseño

Las clases expuestas cuentan con los métodos necesarios para lograr enlazar los datos compilados con el constructor de la aplicación mostrando en pantalla la imagen especificada, como complementación del caso de uso en cuestión (Cargar proveedor de MipMapping).

4.3 Modelo de implementación

El modelo de implementación representa la programación de la aplicación en términos de componentes. Describe la organización de los datos, archivos, ejecutables, código fuente y directorios de acuerdo a los mecanismos disponibles para su estructuración en el entorno de implementación. Fundamentalmente, se describen las relaciones que existen entre los paquetes y clases del modelo de diseño a subsistemas y componentes físicos. En este artefacto se describe cómo se implementan los componentes que lo conforman, acoplándolos en paquetes organizados en capas y jerarquías, señalando las dependencias entre los mismos. Durante la fase de construcción se materializan en

forma de código aquellos modelos desarrollados durante las fases de análisis y diseño siguiendo la guía de los patrones y arquitectura escogidos (**Pressman**, 1999).

4.4 Diagrama de Componentes

Los diagramas de componentes se utilizan para modelar la vista estática de un sistema. Muestran la organización y las dependencias lógicas entre un conjunto de componentes de software, sean componentes de código fuente, librerías, binarios o ejecutables. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema. Se representa como un grafo de componentes software unidos por medio de relaciones de dependencia (**Pressman**, 1999). Ver figura 5.

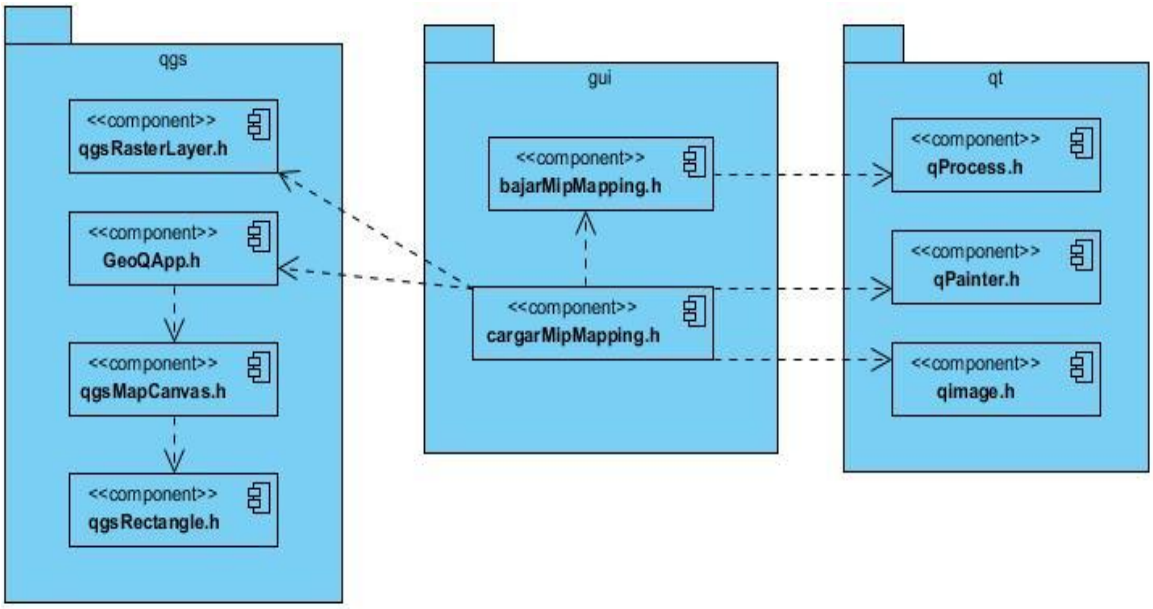


Ilustración 9. Diagrama de Componentes

Descripción del diagrama de componentes

El diagrama de componentes se divide en tres paquetes fundamentales, para lograr la correcta modelación de la vista estática de lo que se desea implementar.

Paquete qgs:

El paquete qgs contará con las clases propias de la aplicación GeoQ, que son utilizadas para lograr la implementación de GeoQ.

Paquete gui:

El paquete gui contará con las clases que dan apoyo a la implementación de GeoQ y que son implementadas por los autores.

Paquete gt:

El paquete gt contará con las clases de gt que son necesarias para la implementación de GeoQ

4.5 Modelo de Despliegue

La nueva funcionalidad incluida dentro de la aplicación GeoQ, no modifica a la misma con nuevos elementos que alteren su despliegue, de ahí que se decida mostrar el diagrama de despliegue oriundo de la aplicación. (Ver figura 6).

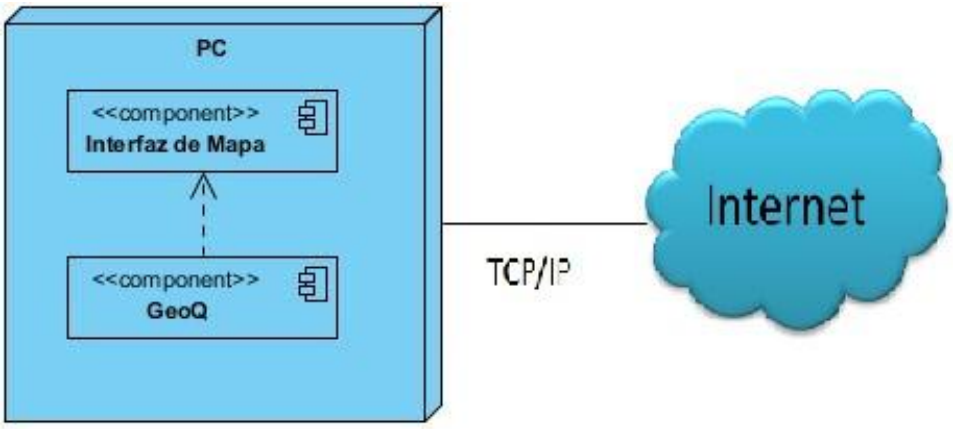


Ilustración 30.Diagrama de Despliegue de GeoQ

4.6 Pruebas del sistema propuesto

El hombre tienen una habilidad innata de provocar errores y sus invenciones no están exentas de estos, de ahí que se requiera realizar varios test o pruebas que garanticen la calidad de los productos de software desarrollados por los mismos.

Los errores dentro del proceso de desarrollo del software pueden venir dados desde su creación durante las fases de inicio y análisis hasta su explotación durante el despliegue. La identificación errónea de requisitos o un mal diseño de clases pueden provocar faltas graves en el buen funcionamiento de una aplicación. De ahí que se realicen a las aplicaciones de software numerosas pruebas con el objetivo de descubrir fallas no detectadas hasta ese momento.

Sin embargo son disímiles los tipos de pruebas existentes y la selección del prototipo de prueba apropiado a determinado negocio debe responder a las características del mismo y a lo que se desee probar.

- Pruebas de Integración: Comprueban la compatibilidad y funcionalidad de los interfaces entre los distintos elementos que componen un sistema, pueden ser módulos, aplicaciones individuales, aplicaciones cliente/servidor, etc.
- Pruebas de Validación: Son realizadas sobre un software completamente integrado para evaluar el cumplimiento con los requisitos especificados.
- Prueba de Caja Blanca: Se basa en el diseño de casos de prueba que usen la estructura de control del diseño procedimental para derivarlos, en otras palabras se analiza la estructura lógica del programa.
- Prueba de Caja Negra: Este tipo de prueba se centra principalmente en los requisitos funcionales del software reflejados en su interfaz sin tener en cuenta el funcionamiento interno de la aplicación, no considera la codificación dentro de los parámetros a evaluar. Se basa en que las entradas sean aceptadas de forma adecuada y se reciba una salida correcta demostrando que cada función es completamente operativa.
- Prueba de Aceptación: Es la prueba final basada en las especificaciones del usuario o basada en el uso del programa por el usuario final. Su objetivo principal es demostrarle el cumplimiento del requisito de software al usuario. Puede estar asociado tanto a requisitos funcionales como no funcionales y cada requisito puede tener una o más pruebas de aceptación asociada.

"La prueba no puede asegurar la ausencia de defectos, sólo puede demostrar que existen defectos en el software" (**Pressman**, 2000).
El tipo de prueba y el método utilizado por ésta puede variar, pero su objetivo fundamental es el mismo, encontrar errores o defectos en el funcionamiento del software.

Realizar un proceso de pruebas que garantice la aptitud del software para satisfacer las necesidades del usuario requiere técnicas que guíen el desarrollo. Establecer una estrategia es un elemento clave, de modo que se pretende realizar casos de pruebas que ayuden a la detección de fallas en el sistema.

La aplicación cuenta con un 70% de código generado por programas auxiliares lo que dificulta en gran medida su análisis lógico de ahí que se descarte el uso del tipo de prueba Caja Blanca. La misma no es un módulo ya que no constituye en esencia una aplicación sino una funcionalidad dentro de un

software probado, lo que elimina las posibilidades de pruebas de Integración y Validación y no cuenta con un cliente que realice las pruebas de Aceptación, por lo antes expuesto se selecciona como método de prueba la técnica funcional de Caja Negra.

4.7 Técnica de Caja Negra

"Las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software" (**Pressman**, 2000).

Mediante las técnicas de prueba de caja negra se obtiene un conjunto de casos de prueba que satisfacen los siguientes criterios:

- Casos de prueba que reducen, en un coeficiente que es mayor que uno, el número de casos de prueba adicionales que se deben diseñar para alcanzar una prueba razonable.
- Casos de prueba que nos dicen algo sobre la presencia o ausencia de clases de errores en lugar de errores asociados solamente con la prueba que estamos realizando.

Para desarrollar la prueba de caja negra existen varias técnicas, entre ellas están: (**Pressman**, 2000).

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

El método de partición equivalente es considerado uno de los más prácticos para la identificación de errores, el mismo divide el campo de entrada de un programa en clases de datos, de los que se pueden derivar casos de prueba.

"El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada…una clase de equivalencia representa un conjunto de estados validos o no válidos para condiciones de entrada. Típicamente, una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica."(**Pressman**, 2000).

Para definir las clases de equivalencia se siguen un conjunto de directrices:

- Si una condición de entrada especifica un rango, se define una clase de equivalencia válida y dos no válidas.
- Si una condición de entrada requiere se define una clase de equivalencia válida y dos no válidas.
- Si una condición de entrada especifica un miembro de un conjunto, se define una clase de equivalencia válida y una no válida.
- Si una condición de entrada es lógica, se define una clase de equivalencia válida y una no válida.

Al aplicar estas directrices en la obtención de clases de equivalencia se ejecutan casos de prueba para cada elemento de datos del campo de entrada, tratando siempre de ejercitar la mayor cantidad de atributos de cada clase de equivalencia a la vez.

Se muestra el caso de prueba para el caso de uso Cargar Proveedor de MipMapping utilizando la técnica de partición equivalente.

Descripción general del caso de uso

El caso de uso comienza cuando el actor desea abrir el proveedor de MipMapping y termina cuando el sistema representa la información contenida en el mismo.

Nombre de la Sección	**Caso de uso:** Cargar Proveedor de MipMapping	
Escenario de la sección	**Descripción de la funcionalidad**	**Flujo Central**
EC 1.1 Seleccionar cargar proveedor de	Al acceder a la opción “Cargar proveedor de MipMapping” del menú “Capas “se muestra la interfaz “Cargar proveedor de MipMapping”. La misma permite la selección	Ventana Principal. Menú Capas. Cargar proveedor de MipMapping.

MipMapping	de la dirección donde el usuario guardara los datos. Complementariamente se tienen los botones “Aceptar” y “Cancelar”.	
EC 1.3 Cancelar en Cargar proveedor de MipMapping	Al seleccionar el botón “Cancelar” el sistema debe cerrar la interfaz “Cargar proveedor de MipMapping” cancelando la operación.	Ventana Principal. Menú Capas. Cargar proveedor de MipMapping. Botón Cancelar.

Tabla 3.Descripción del CUS Guardar Mapa

Descripción de las variables

No.	Nombre del campo	Clasificación	Requerido	Descripción
1	Nombre	Campo de Texto	Si	Especifica el nombre de la dirección donde se guardaran los datos.

Tabla 4.Descripción de las variables para el caso de uso Cargar proveedor de MipMapping.

Matriz de datos (SC 1 Cargar proveedor de MipMapping).

Escenario	Nombre	Respuesta del sistema	Resultado de la prueba
EC 1.1	**V**	Al acceder a la opción “Cargar proveedor de MipMapping” el sistema muestra la ventana que permite la selección del mismo.	

Tabla 5.Matriz de Datos (SC 1 Cargar proveedor de MipMapping).

Descripción general del caso de uso

El caso de uso comienza cuando el actor desea guardar el mapa y termina cuando el sistema ejecuta la acción.

Nombre de la Sección	**Caso de uso:** Guardar mapa	
Escenario de la sección	**Descripción de la funcionalidad**	**Flujo Central**
EC 1.1 Seleccionar Guardar mapa.	Al acceder a la opción “Guardar mapa” del menú “Capas “se muestra la ventana “Guardar mapa”. La misma permite guardar el mapa todo el mapa con el que el usuario está trabajando. Complementariamente se tienen los botones “Aceptar” y “Cancelar”.	Ventana Principal. Menú Capas. Guardar mapa.
EC 1.2 Cancelar en Guardar mapa.	Al seleccionar el botón “Cancelar” el sistema debe cerrar la ventana “Guardar mapa.” cancelando la operación.	Ventana Principal. Menú Capas. Guardar mapa. Botón Cancelar.

Tabla 6.Secciones a probar en el caso de uso Guardar mapa.

Descripción de las variables

No.	Nombre del campo	Clasificación	Requerido	Descripción
1	Nombre	Campo de Texto	Si	Especifica el nombre de la dirección donde se guardaran los datos.

Tabla 7.Descripción de las variables para el caso de uso Guardar mapa.

Matriz de datos (SC 1 Guardar mapa)

Escenario	Nombre	Respuesta del sistema	Resultado de la prueba
EC 1.1	V	Al acceder a la opción "Guardar mapa" el sistema muestra la ventana que permite la selección del mismo.	

Tabla 8.Matriz de Datos (SC 1 Guardar mapa).

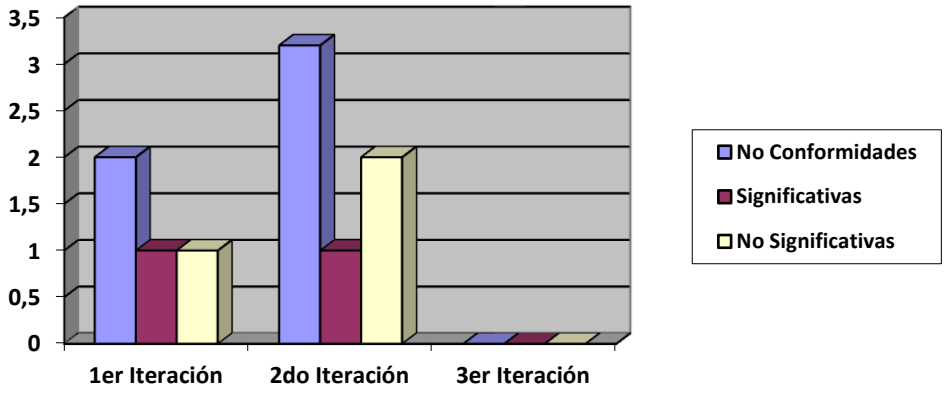
4.8 Resultados

Tras realizarse las pruebas a la aplicación desarrollada se obtuvieron No Conformidades, las mismas se clasificaron en No Conformidades significativas y no significativas; se representaron en una gráfica los resultados para una mejor visualización de los mismos. Estos conceptos se trabajaron en la gráfica a través de números reales.

En la 1ra iteración se obtuvieron 2 No Conformidades, 1 significativa y 1 no significativa.

En la 2da iteración, una vez corregidas las No Conformidades anteriores se aplicó nuevamente el proceso de pruebas y se obtuvieron 3 No Conformidades, 1 significativa y 2 no significativa.

En la 3ra iteración no se obtuvieron No Conformidades, por lo que se decidió no realizar una 4ta iteración.



4.9 Conclusiones Parciales

Durante el desarrollo del capítulo fueron expuestos temas de gran importancia para la construcción del sistema, de los cuales se obtuvieron las siguientes conclusiones:

- Los diagramas de clases correspondientes al diseño y al análisis, contribuyeron a una adecuada implementación.
- Se abordaron las generalidades para el desarrollo de la aplicación teniendo en cuenta los componentes utilizados, así como las relaciones entre ellos a través del modelo de implementación.
- El uso de los patrones de diseño favoreció el trabajo en equipo y garantizó la viabilidad y calidad requeridas por el sistema.
- La solución implementada es correcta, lo cual se demuestra con los resultados de las pruebas aplicadas.

Conclusiones

Con el diseño del proveedor de MipMapping para GeoQ se cumplen los objetivos propuestos para este trabajo en función de las tareas trazadas para llevar a cabo el proceso investigativo; por lo que se arribó a las siguientes conclusiones:

- Con la documentación técnica generada se logró describir claramente todos los artefactos generados, convirtiéndose en una guía para obtener un mejor entendimiento de los procesos automatizados y una base para mejoras futuras.
- Los requerimientos funcionales y no funcionales identificados durante el levantamiento de requisitos permitieron la creación del Diagrama de Caso de Uso del Sistema y un mejor entendimiento para el desarrollo del sistema. Los mismos fueron implementados en su totalidad y debidamente probados.
- La representación en el mapa sin utilizar la aplicación MapServer permite integrar cartografías ráster y vectoriales, así como lograr una vista en forma satelital.
- El sistema diseñado garantizará obtener las cartografías ráster de Google Maps para la aplicación GeoQ, lo que favorecerá la toma de decisiones.
- Permitirá además guardar e importar las cartografías ráster de la aplicación Google Maps, dándole la oportunidad al usuario de re-utilizar las cartografías consultadas.
- Se disminuye el acceso a internet, al reutilizarse las cartografías previamente descargadas.

Recomendaciones

Realizado el análisis de una significativa cifra de elementos que favorecieron la elaboración de la presente investigación, es necesario que se enumeren algunas recomendaciones para la mejora continua de la propuesta en cuestión:

1. El diseño de un repositorio central que permita a varios usuarios compartir los tiles (cuadros) previamente descargados.

Referencias Bibliográficas

ArcGis Community ArcGis Online [Online] // ArcGis Online. - Esri Community, 2011. - 1 5, 2011. - http://www.arcgis.com/home/.

Ballari Daniela Instalación de MapServer como WMS, WFS y WCS [Report]. - Universidad Politécnica de Madrid : [s.n.], 2001.

Nokia Corporation Qt Creator [Online] // Qt Creator. - 2008-2011. - 3 8, 2011. - http://qt.nokia.com/products/developer-tools.

Regents of the University of Minnesota MapServer [Online] // MapServer. - Regents of the University of Minnesota, 2011. - 1 5, 2011. - http://www.mapserver.org/.

Reynoso Carlos and Kiccillof Nicolás Estilos y Patrones en la Estrategia de Arquitectura de Microsoft [Book]. - 2004.

RUMBAUHG J JACOBSON I El Lenguaje Unificado de Modelado. Manual de Referencia [Book]. - Madrid : Pearson Educación, 2000.

Sendra Bosque Los Sistemas de información Geográfica [Report]. - 1999.

Pressman, R. 1999. Software Engineering.A Practitioner's Approach. USA: s.n., 1999.

González, Carlos D. [En línea] febrero de 2008. [Citado el: 15 de enero de 2013.] http://www.usabilidadweb.com.ar/cpp.php.

Scribd 1. *Metodologías de Desarrollo de Software* [En línea] [Citado el: 15 de febrero de 2012.] http://www.scribd.com/doc/2050925/metodologias-de-desarrollo-software.

Grupo de Soluciones GSInnova. [En línea] [Citado el: 15 de febrero de 2013.] http://www.rational.com.ar/herramientas/rup.html.

Cornejo, José Enrique González. DocIRS. [En línea] [Citado el: 16 de enero de 2013.] http://www.docirs.cl/uml.htm.

Scribd 2. *Metodologías de Desarrollo de Software* [En línea] [Citado el: 16 de febrero de 2013] http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.

Free Download Manager. [En línea] [Citado el: 16 de febrero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

Nokia. QT. [En línea] [Citado el: 17 de febrero de 2013.] http://qt.nokia.com/products/developer-tools.

Softonic. [En línea] [Citado el: 22 de enero de 2013.] http://emaps.softonic.com/mac

Geoprocesamiento [En línea] [Citado el: 23 de octubre de 2012.] http://mundogeo.com/

José David Farré, Guillermo González. 2010. APLICACIÓN PARA ANALIZAR Y PROCESAR TRAMAS GPS UTILIZANDO SOFTWARE LIBRE. Villa Clara: AGENCIA DE SOFTWARE GEOMIX, EMPRESA GEOCUBA LA HABANA., 2010.

Fuentes Lidia, Troya Jose M y Vallecillo Antonio Desarrollo de Software Basado en Componentes [Informe]. - Málaga : [s.n.], 2003.

Open Geospatial Consortium [En línea] [Citado el: 14 de marzo de 2013.] http://www.opengeospatial.org/standards/wms

Río, Víctor G. Sánchez y Homero V. 1995. *Metodología CASE para el desarrollo de sistemas.* 1995.

Referencia de Figuras

(1)Tomado del sitio web (http://www.aulati.net/?tag=google-maps,2011)

Bibliografía Consultada

ArcGis Community ArcGis Online [En línea] // ArcGis Online. - Esri Community, 2011. - 5 de 11 de 2012. - http://www.arcgis.com/home/.

Ballari Daniela Instalación de MapServer como WMS, WFS y WCS [Informe]. - Universidad Politécnica de Madrid : [s.n.], 2001.

Barbosa Dr. José Seguinot Pasado, presente y futuro de los SIG [Informe]. - Puerto Rico : [s.n.], 2007.

Booch G, Rumbaugh J y Jacobson I El Proceso Unificado de Desarrollo de Software [Libro]. - 2000.

Carrillo Pérez Isaías, Pérez González Rodrigo y Rodríguez Martín David Aureliano Metodología de Desarrollo de Software [Informe]. - 2008.

Case CASE Tools [En línea] // CASE Tools. - 2011. - 4 de 12 de 2012. - http://case-tools.org/.

Case.Herramientas Case www.elprisma.com [En línea] // www.elprisma.com. - 3 de enero de 2011. - http://www.elprisma.com/apuntes/curso.asp?id=13324...

Cubillo Darío Rodríguez Proyecto Final del Máster en Tecnologías de la Información Geográfica [Informe]. - Universidad Autònoma de Barcelona : [s.n.], 2000.

Fuentes Lidia, Troya Jose M y Vallecillo Antonio Desarrollo de Software Basado en Componentes [Informe]. - Málaga : [s.n.], 2003.

Fundación Plone Portal GvSig [En línea] // Portal GvSig. - Fundación Plone, 2010-2011. - 3 de 11 de 2012. - http://www.gvsig.org/web/.

GRASS Development Team GRASS GIS [En línea] // GRASS GIS. - GRASS Development Team, 1999-2011. - 4 de 11 de 2012. - http://grass.osgeo.org/.

Hernández Enrique Orallo El UML [Informe].

Herramientas Case Visual Paradigm [En línea] // Visual Paradigm. - 2011. - 3 de 12 de 2012. - http://www.visual-paradigm.com/.

Jalón Javier García de Aprenda C++ como si estuviera en primero [Libro]. - Navarra : [s.n.].

Joven Club de Computación y Electrónica Debate en el Ciberespacio [En línea] // Debate en el Ciberespacio. - 2009-2010. - 5 de 11 de 2012. - http://foro.jovenclub.cu/index.php?PHPSESSID=69606834cf5c1b334688a49d8594f714&topic=7792.msg48450#msg48450.

Martínez Alejandro y Martínez Raúl Guía a Rational Unified Process [Libro]. - Castilla : [s.n.].

Nokia Corporation Qt Creator [En línea] // Qt Creator. - 2008-2011. - 8 de 11 de 2012. - http://qt.nokia.com/products/developer-tools.

Open Source Geospatial Foundation Quantum Gis [En línea] // Quantum Gis. - Open Source Geospatial Foundation(OSGeo), 2011. - 5 de 11 de 2012. - http://www.qgis.org/.

OSGeo Guia de Usuario QGis [Informe]. - 2004-2009.

Prado Elena Raja Casi todas las pruebas del software [Informe]. - 2007.

Pressman Roger S Ingeniería de Software un Enfoque Práctico [Libro]. - 2000. - Vol. 6ta Edición.

Regents of the University of Minnesota MapServer [En línea] // MapServer. - Regents of the University of Minnesota, 2011. - 5 de 11 de 2012. - http://www.mapserver.org/.

Reynoso Carlos y Kiccillof Nicolás Estilos y Patrones en la Estrategia de Arquitectura de Microsoft [Libro]. - 2004.

Rienzi Bruno Estándares y Tecnologías de base para la construcción de un Sistema de Información Geográfica Empresarial [Informe]. - 2010.

RUMBAUHG J JACOBSON I El Lenguaje Unificado de Modelado. Manual de Referencia [Libro]. - Madrid : Pearson Educación, 2000.

Sendra Bosque Los Sistemas de información Geográfica [Informe]. - 1999.

Sepúlveda Carlos Gustavo Infante Aplicaciones Geográficas Enriquecidas para Internet utilizando componentes [Informe]. - 2008.

Zayas Dr. Cs. Carlos Alvarez de Metodología de la Investigación Científica [Libro]. - Santiago de Cuba : [s.n.], 1995.

Glosario de Términos

-A-

API: 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. Son usadas generalmente en las bibliotecas.

-C-

Cartografía: La cartografía constituye un conjunto de operaciones que permiten a partir de observaciones y mediciones, la representación de una parte o la totalidad de la superficie terrestre.

Código: Combinación de letras o números que identifican un producto o persona. Permiten realizar determinadas operaciones o manejar algunos aparatos.

Caso de uso (CU): Secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema.

-E-

Edición de la cartografía: Conjunto de transformaciones de que es objeto la cartografía con el objetivo de resaltar e identificar aspectos relevantes sobre el contenido que la misma posee.

-G-

GNU GPL: Licencia que posibilita la modificación y redistribución del *software*.

GEySED: Centro de Desarrollo de Geoinformática y Señales Digitales.

-M-

Mapfile: Fichero de configuración de MapServer, donde se especifican la estructura y composición de los datos cartográficos que serán representados.

-R-

Requerimiento: Petición de una acción que se considera necesaria.

-S-

Sistema de información geográfico: Herramienta informática que integra hardware, software y datos geográficos que permitiendo capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada.

Software: Equipamiento lógico o soporte lógico de una computadora digital.

Anexos

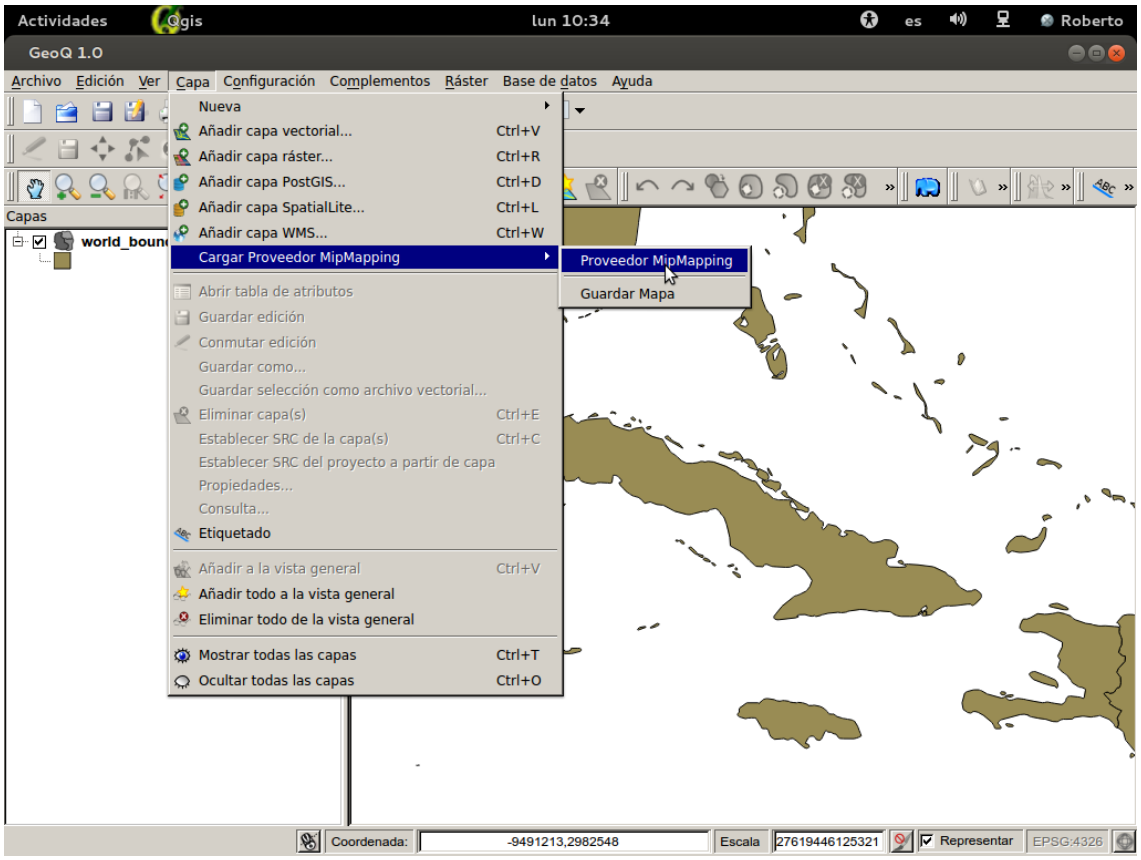


Ilustración 11. Aplicación GeoQ. Caso de Uso Cargar Proveedor MipMapping.

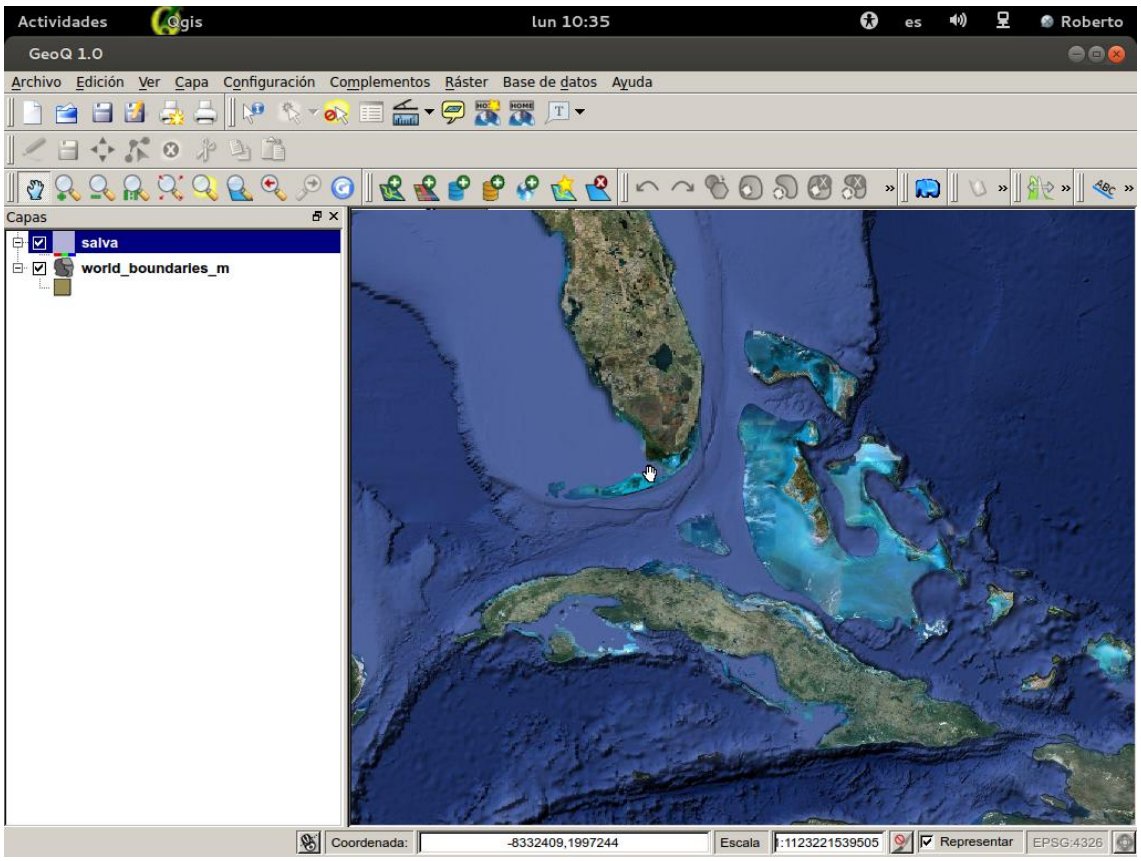


Ilustración 12. Aplicación GeoQ. Caso de Uso Cargar Proveedor MipMapping.

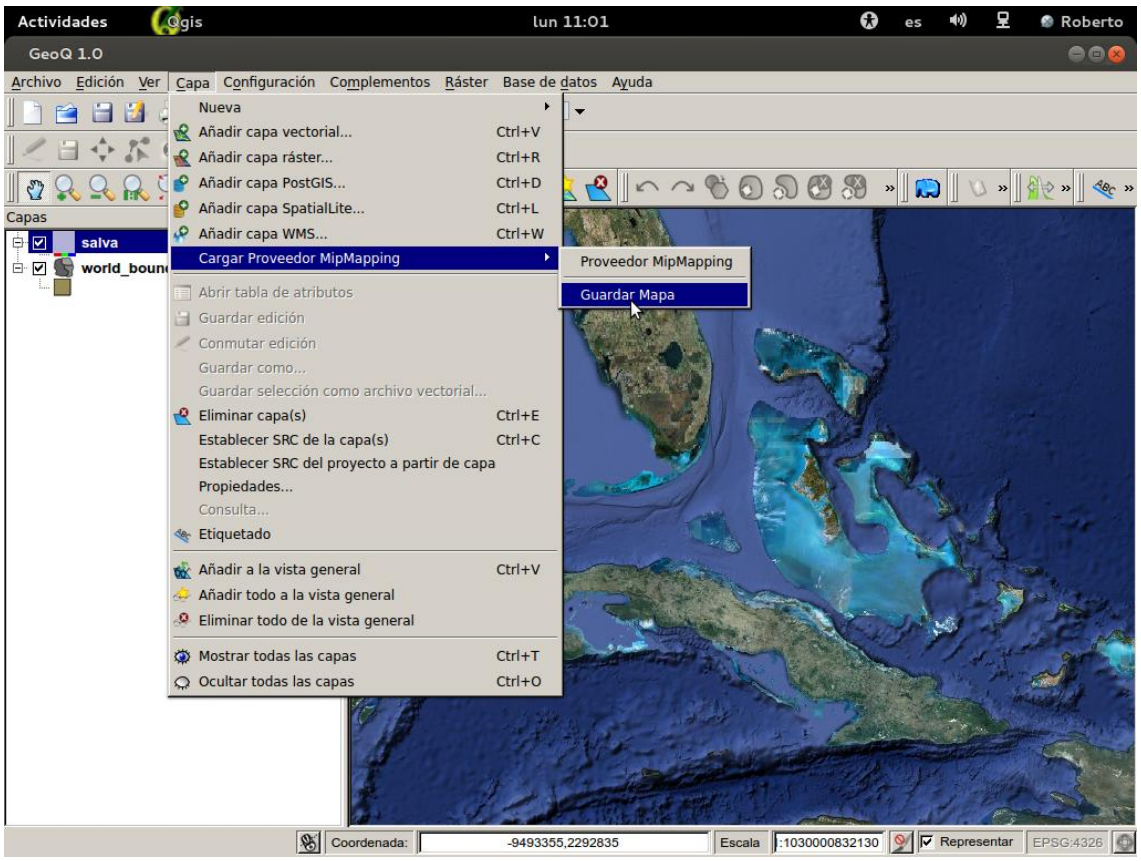


Ilustración 13. Aplicación GeoQ. Caso de Uso Guardar Mapa.

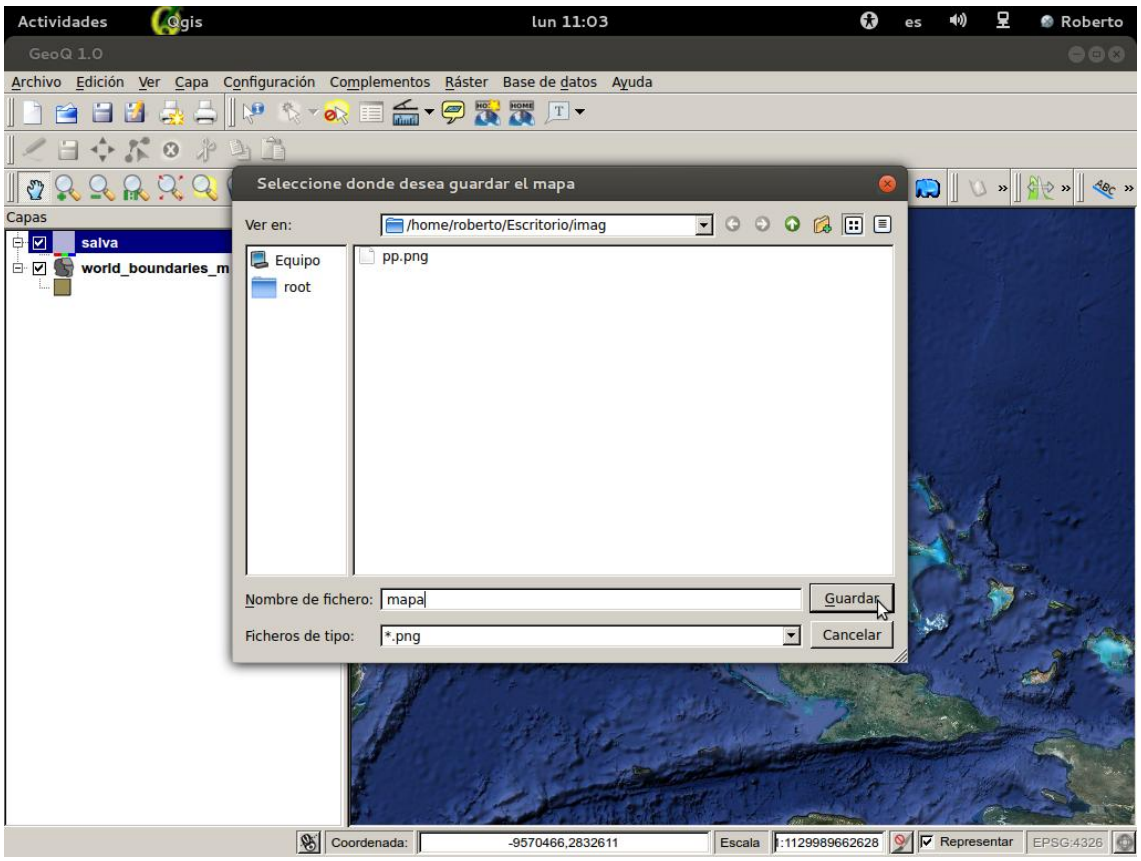


Ilustración 14. Aplicación GeoQ. Caso de Uso Guardar Mapa.