Universidad de las Ciencias Informáticas

Facultad 9

ANÁLISIS DE UN SISTEMA DE INFORMACIÓN GEOGRÁFICA SOBRE GVSIG EN EL DEPARTAMENTO GEOINFORMÁTICA

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

AUTOR: Alejandro Luis Fernández Muñoz

TUTOR: Ing. Armando Batista Piñeda

COTUTOR: Ing. Yordanys Piñeiro Gómez

Ciudad de La Habana, Cuba

julio de 2010

DEDICATORIA

Le dedico este trabajo:

A mi mamá Leticia, mi abuela Telma y mi hermanito Robertico por ser las personas que más amo en la vida.

A mi abuelo Andrés por ser un padre para mí, te quiero grande viejo.

A mis tíos: Yanel, Alfredo y Yamir, mis primos y toda la familia que tanto me ha apoyado siempre en todos los aspectos.

A mi tía Virgen que siempre vivirá en mi corazón.

AGRADECIMIENTOS

A Dios por haberme hecho de la forma que soy, por darme la fe y la sabiduría necesarias para vivir, por darme lo que necesito y no lo que deseo.

A la Revolución, porque sin ella no pudiera haberme graduado de ingeniero.

A mi tutor por haberme apoyado en todo momento, por entenderme y darme ánimo cuando las cosas se pusieron difíciles.

A mi oponente Dayris y al tribunal por la ayuda brindada y su preocupación por mi tesis.

A mi mamá por ocuparse de mí y mis problemas, por permitirme el lujo de sólo ocuparme de mi tesis y por ser la fuente de apoyo y cariño más cercana.

A mi abuelita por su preocupación, apoyo y comprensión.

A mi familión.

A Yordanys por ser mi compañera sentimental durante toda la carrera, ayudarme en todo lo que pudo y por los momentos bonitos, por hacerme más fácil el camino.

A la gente del proyecto que siempre respondieron a mis dudas en los momentos que recurrí a ellos, en especial a Membrides y Karel.

A mis compañeros y amigos de aula y de la universidad, que estuvieron conmigo en los años de la carrera, en especial a Mariano y Leonel.

A la gente del cuarto: Yanoino, Elyony, El mello, Daniel, Pepe, Jean, Yosiel, Yumar, Olivier, Cuza, Reinier (BadBoy), Adrian (Jason), Sevilla, Friki, Dayron, Santiaguero, Luisdey, Chenike, Gilberto (Hermit), El Chino, Yasmany, Pacheco, José Eugenio, Gustavo y Falero, por la convivencia, los chistes, las conversaciones y los momentos de despeje que siempre ayudan a hacer más ligeros los problemas.

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas y la Oficina Nacional de Recursos Minerales a hacer uso del mismo en su beneficio.

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

_____	_____
Alejandro Luis Fernández Muñoz	Ing. Armando Batista Piñeda

NOTA ACLARATORIA: Este documento es un extracto de 80 páginas del documento original.

RESUMEN

El progresivo avance de las Tecnologías de la Información y la Comunicación ha permitido el uso de los Sistemas de Información Geográfica (muy conocidos como SIG) en casi todas las esferas de la sociedad. Las aplicaciones de estos programas informáticos son incontables y sus funcionalidades se han ido incrementando cada vez más. La producción de esta tecnología requiere, entre otras necesidades, que se obtengan correctamente los requerimientos de software y la modelación del análisis, máxime cuando el desarrollo se realiza en un ambiente dinámico como es el caso del proyecto GENESIG, perteneciente al Departamento Geoinformática de la Universidad de las Ciencias Informáticas. Este trabajo versa precisamente sobre las primeras etapas del proceso de desarrollo de un Sistema de Información Geográfica sobre la herramienta libre gvSIG en el proyecto antes mencionado.

Palabras clave:

Análisis, gvSIG, Proceso de Desarrollo de Software, Sistema de Información Geográfica.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En la actualidad, el progresivo avance de las tecnologías computacionales y la necesidad del manejo de cada vez mayores cantidades de información, conllevan a la necesidad de un mayor uso de los sistemas informáticos en diversas actividades de la sociedad. La agricultura, la meteorología, el turismo, la medicina, son algunas de las tantas ramas de aplicaciones de una herramienta informática muy popular y de gran impacto en los últimos tiempos: el Sistema de Información Geográfica.

Existe una gran variedad de definiciones que responden a la conceptualización de un Sistema de Información Geográfica (en lo adelante SIG). Sin embargo, una forma simple de describirlo sería como una tecnología que permite integrar, manipular y visualizar una gran variedad de datos capaces de crear una imagen de la geografía, medioambiente y características socioeconómicas de una zona determinada estudiada.

Cuba se encuentra inmersa en un proceso de transformaciones tecnológicas, donde los SIG por sus aplicaciones juegan un papel importante. El hecho de ser un país del tercer mundo bloqueado y los altos precios de las patentes para adquirir los derechos de los software propietarios, conllevan a trazar planes y estrategias para lograr una independencia tecnológica, lo que sólo es posible mediante el uso del software libre. Para lograr este objetivo se han creado diversas instituciones que dentro de su marco de trabajo radica el estudio y desarrollo estas tecnologías bajo los principios mencionados.

Desde hace algunos años, la Universidad de las Ciencias Informáticas (UCI) se ha unido a los programas nacionales vigentes para alcanzar un logro superior al existente en el sector informático y contempla dentro de sus principales actividades la producción, donde tiene incluido dentro de sus perfiles el desarrollo de sistemas de software y soluciones SIG.

En la universidad, existe el proyecto GENESIG que se especializa en el desarrollo de SIG y su arquitectura contempla la construcción de familias temáticas para sus productos: aplicaciones Web, de tipo escritorio y para móviles.

Actualmente se trabaja en la construcción de una plataforma Web, bajo tecnologías libres, que brinda un conjunto significativo de funcionalidades de gran valor en este ámbito; sin embargo, las exigencias del

mercado ya indican que es necesario incursionar en las otras líneas definidas. Para el caso de las soluciones de tipo escritorio se han realizado estudios preliminares que señalan a gvSIG como una de las opciones a considerar.

Se ha pensado, estratégicamente, en realizar una aplicación genérica con esta herramienta que cumpla con un conjunto de requisitos básicos de interés para la dirección del proyecto. La intensión es utilizar esta aplicación, más la documentación asociada al proceso de construcción de la misma, para acelerar y poder enfrentar solicitudes de clientes satisfactoriamente.

El equipo de desarrollo con que cuenta GENESIG para enfrentar sus necesidades, está compuesto por especialistas, profesores y estudiantes bajo un modelo que relaciona tres componentes fundamentales: docencia, investigación y producción. Bajo este modelo, la estabilidad del personal implicado en un proyecto es limitada. Con relativa frecuencia entran nuevos miembros, se realizan movimientos y otras actividades que generan un contexto bastante dinámico. Por tal motivo, el inicio de un proceso de desarrollo demanda que sean correctamente capturadas y documentadas las funcionalidades que debe cumplir un producto de software, de manera que conduzca a su correcto diseño e implementación.

Partiendo de la situación expuesta se ha formalizado el siguiente **problema científico**:

¿Cómo lograr el entendimiento entre clientes y desarrolladores para facilitar la construcción de un producto de software sobre la herramienta gvSIG?

Para lograr resolver el problema en cuestión se considera que es necesario realizar el proceso de modelación del análisis para el desarrollo de un producto de software sobre la herramienta gvSIG en el proyecto GENESIG, utilizando las técnicas y herramientas de la Ingeniería de Software, lo cual figura como el **objetivo general** de esta investigación.

Para darle cumplimiento al objetivo trazado se hace necesario centrar la investigación en el Proceso de Desarrollo de SIG en el proyecto GENESIG, lo cual constituye el **objeto de estudio**.

El **campo de acción** que enmarca la investigación es la modelación del análisis de un SIG sobre la herramienta gvSIG en el proyecto GENESIG.

Para lograr el objetivo planteado se trazan las siguientes **tareas**:

- Caracterizar el proceso de desarrollo de los SIG.
- Argumentar el uso de la Metodología de Desarrollo de Software, el Lenguaje de Modelado y la Herramienta Case.
- Documentar el Modelo de Dominio de la aplicación SIG.
- Identificar las funcionalidades que debe brindar el sistema.
- Documentar el Modelo de Casos de Uso del Sistema de la aplicación SIG.
- Documentar el Modelo de Análisis de la aplicación SIG.
- Aplicar métricas para la validación de los artefactos.

El desarrollo exitoso de las tareas expuestas anteriormente contribuirá al cumplimiento de la **idea a defender** de esta investigación: La realización de un correcto proceso de modelación del análisis, usando una metodología de desarrollo adecuada, facilitará la posterior construcción de una aplicación de software sobre la herramienta gvSIG en el proyecto GENESIG.

En el desarrollo de este trabajo se utilizaron métodos teóricos y empíricos. Dentro de los métodos teóricos se encuentran el de análisis y síntesis, el histórico-lógico y el causal y dentro de los métodos empíricos serán utilizadas las entrevistas.

El método de **Análisis y Síntesis**, se define con el objetivo de analizar los diversos documentos relacionados con el proceso de desarrollo de SIG, del análisis de la gestión de información geográfica sobre la herramienta gvSIG y las metodologías existentes para la construcción de los artefactos necesarios, así como para realizar una síntesis de los mismos.

El método **Causa**l se emplea para determinar y analizar los factores que provocan la necesidad de desarrollo de una aplicación SIG de tipo escritorio, específicamente en el proyecto GENESIG.

El método de **Modelación** se emplea para mostrar los diversos diagramas que se construyen como resultado del proceso de Ingeniería de Software.

Las **Entrevistas** se realizan a líderes de proyectos y especialistas de GeoCuba para recopilar información referente al proceso de análisis de la gestión de información geográfica sobre la herramienta gvSIG y para identificar los requisitos funcionales y no funcionales con que contará la aplicación a construir.

El método **Histórico-Lógico** se aplica para investigar la existencia de alguna aplicación de tipo escritorio desarrollada con gvSIG, que tenga implementado un grupo de funcionalidades que sirvan de punto de partida para el software que se desea construir.

Se espera obtener como **posibles resultados** de este trabajo de diploma, la documentación técnica correspondiente a la modelación del análisis del proceso de desarrollo de un SIG de tipo escritorio sobre la herramienta gvSIG.

El documento está estructurado en 4 capítulos, que son descritos a continuación:

El *Capítulo 1* contempla la fundamentación teórica de esta investigación, en la cual son expuestos los principales conceptos que contribuyen al mejor entendimiento del problema en cuestión, se especifican detalladamente los argumentos que esclarecen el objeto de estudio.

Por su parte el *Capítulo 2* trata acerca de las tendencias y las tecnologías que se utilizan en la actualidad a nivel internacional para el desarrollo de SIG. Paralelamente a esto se estudian un conjunto de metodologías (ágiles y pesadas) que se manejan en el mundo para el desarrollo de software de forma general, con el objetivo de seleccionar de ellas, la más adecuada para guiar el proceso de desarrollo de este tipo de software en el proyecto GENESIG.

En el *Capítulo 3* se presenta la solución propuesta a partir de la descripción del Modelo de Dominio y la identificación de requisitos funcionales y no funcionales del sistema a construir, que finalmente se agrupan en casos de uso del sistema para conformar el Modelo de Casos de Uso del Sistema, así como la descripción textual de estos casos de uso.

Por último en el *Capítulo 4* se realiza la validación de la solución propuesta, para esto se aplican un conjunto de métricas a los requisitos y a los casos de uso.

1

FUNDAMENTACIÓN TEÓRICA

En el presente capítulo se realiza una investigación sobre el objeto de estudio, describiendo algunos de los conceptos asociados al tema analizado; además, se realiza una descripción de las principales herramientas SIG escritorio, centrándose principalmente en gvSIG para así tener un mayor conocimiento del tema y puntos de comparación.

1.1 Conceptos asociados al dominio del problema

Para que se tenga una comprensión mayor de los temas que serán abordados en este capítulo, directamente relacionados con el objeto de estudio de la investigación, se describen a continuación una serie de conceptos asociados al dominio del problema.

1.1.1 Datos Espaciales

Los datos espaciales son informaciones sobre la localización y las formas de un objeto geográfico y las relaciones entre ellos. Los datos geográficos son entidades que cuantifican la distribución, el estado y los vínculos de los distintos fenómenos u objetos naturales o sociales.

Los datos espaciales refieren a entidades o fenómenos que cumplen los siguientes principios básicos:

- Tienen posición absoluta sobre un sistema de coordenadas (x, y, z).
- Tienen una posición relativa frente a otros elementos del paisaje.
- Tienen una figura geométrica que las representan (punto, línea, polígono).

- Tienen características que los describen.

En resumen se puede decir que los datos espaciales contienen información referente a la localización y características propias de las formas de un objeto geográfico como pueden ser el color, tamaño, ubicación respecto a otros objetos entre otras. Los datos espaciales son los encargados de soportar la información geográfica.

1.1.2 Información Geográfica

Es cualquier información que pueda ser geográficamente referenciada, es la que describe un sitio especifico o está asociada al mismo. Entiéndase por información como el conjunto de datos que al relacionarse adquieren sentido o un valor de contexto.

En (YAGÜEZ y LANGHI, 2002) se denomina Información Geográfica (IG) a aquellos datos espaciales georreferenciados requeridos como parte de las operaciones científicas, administrativas o legales. Dichos datos espaciales suelen llevar una información alfanumérica asociada. Se estima que el 80% de los datos corporativos existentes en todo el mundo poseen esta componente geográfica.

1.1.3 Sistema de Información

Un Sistema de Información es el conjunto de elementos (personas, datos, actividades, recursos) que interactúan entre sí, procesando los datos y dando lugar a una información más elaborada y organizada con un propósito determinado. Un Sistema de Información no está obligatoriamente ligado a la Informática, los Sistema de Información tienen un conjunto de componentes relacionados que se utilizan para recolectar, procesar, administrar, mover, almacenar, obtener, manipular, controlar, desplegar, intercambiar, transmitir o recibir y distribuir información para apoyar la toma de decisiones y el control en una organización.

Un Sistema de información realiza tres actividades básicas:

- *Entrada*: es el proceso de recolectar los datos necesarios para procesar la información de una forma automática o manual y de un origen externo o interno.

- *Procesamiento*: es el conjunto de procedimientos previamente establecidos que transforman los datos obtenidos en la entrada en información útil y significativa.

- *Salida*: retorna la información ya elaborada y lista para las personas que la utilizarán y/o a las actividades que la requieren.

Cuando la información manejada por un Sistema de Información es del tipo geográfica entonces estos sistemas pasan a llamarse Sistema de Información Geográfica.

1.1.4 Sistema de Información Geográfica

Existen muchas definiciones que conceptualizan a un SIG como la que plantea que : “Es un sistema de hardware, software y procedimientos diseñado para realizar la captura, almacenamiento, manipulación, análisis, modelización y presentación de datos referenciados espacialmente para la resolución de problemas complejos de planificación y gestión”. (NCGIA, 1990).

Otros autores plantean que un SIG es un: “conjunto de métodos, herramientas y datos que están diseñados para actuar coordinada y lógicamente para capturar, almacenar, analizar, transformar y presentar toda la información geográfica y de sus atributos con el fin de satisfacer múltiples propósitos. Los SIG son una tecnología que permite gestionar y analizar la información espacial, que surgió como resultado de la necesidad de disponer rápidamente de información para resolver problemas y contestar a preguntas de modo inmediato” (RODRÍGUEZ, y otros, 1998).

“Un SIG es una integración organizada de hardware, software, datos geográficos y personal, diseñado 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. También puede definirse como un modelo de una parte de la realidad referido a un sistema de coordenadas terrestre y construido para satisfacer unas necesidades concretas de información”. (YAGÜEZ y LANGHI, 2002).

A continuación, algunos conceptos que están vinculados a los SIG que son necesarios dominar.

1.1.5 Otros conceptos importantes

Formato raster

El formato raster, es una forma de tratamiento y representación espacial de la información digital mediante la disposición de celdas o pixeles. Se pueden obtener a través de imágenes de satélite, fotografía aérea, cámara fotográfica por solo decir algunos ejemplos.

Formato vectorial

El formato vectorial, es una forma de tratamiento y representación espacial de las entidades mediante la asignación de dos o tres datos para cada punto, cuyo valor es el de sus propias coordenadas espaciales que pueden ser dos o tres dependiendo de que la representación sea en 2D o en 3D. Es el formato más conveniente para representar objetos cartográficos con coordenadas exactas de precisión. Los archivos vectoriales son imágenes formadas por la combinación de líneas, polígonos o puntos.

Mapa

Representación en tamaño menor y en una superficie plana o esférica mediante signos, símbolos gráficos y colores de la totalidad o parte de la superficie terrestre.

1.2 Situación problemática

En la actualidad existe una gran demanda de los SIG a nivel internacional. Cuba hace algunos años ha venido incursionado en este campo y en los últimos tiempos se ha intensificado la investigación y producción de los mismos a raíz de diversas peticiones que se generan por variadas empresas.

La UCI, centro de altos estudios donde se vincula la docencia, la producción y la investigación no se ve exenta de esta gran responsabilidad, y ha incluido dentro de sus perfiles investigativos este campo que hoy día tiene gran trascendencia por el impacto de su solución. Con el objetivo de organizar de forma eficiente el proceso productivo se crean un gran número de perfiles que son organizados en diferentes Polos Productivos asociados a cada una de las Facultades que integran la Universidad. Surge así el Polo de Geoinformática perteneciente a la Facultad 9 que luego pasó a llamarse Departamento de

Geoinformática, los cuales son los encargados de darle respuesta a los diversos pedidos que sean presentados por los clientes.

Las peticiones son cada vez mayores y se ha demostrado que resulta muy complejo dar solución a todas estas solicitudes en periodos cortos y paralelos de desarrollo, además que se pudo identificar que la mayoría de estos productos tienen funcionalidades que son comunes a cualquier SIG. Por otra parte, la arquitectura del proyecto GENESIG cuenta con tres puntos fundamentales: Desarrollo de SIG para web, escritorio y móviles, en el caso del desarrollo de SIG para la web el proyecto cuenta con una Plataforma entorno Web, sobre la cual se desarrollan hoy día los productos de este tipo por los equipos del proyecto GENESIG lo que representa una ventaja en términos de ahorro de tiempo y de poder garantizar la calidad de los productos; no se tiene una aplicación base para SIG de tipo móvil o escritorio lo que representa un atraso en este ámbito y dificulta el desarrollo de estas áreas en el proyecto.

Muchas de las peticiones que hoy se realizan deben contener funcionalidades que conllevan gran análisis, lo cual no puede ser solucionado desde la Web, por lo que en esos casos se debe implementar alguna solución de tipo escritorio, las cuales son las que están preparadas para darle respuesta a estos requisitos de gran complejidad. Hoy el equipo de proyecto no cuenta con una herramienta definida para dar respuesta ante la aparición de solicitudes de SIG de tipo escritorio, por lo que se han realizado estudios a estas herramientas , estos estudios preliminares señalan a gvSIG como una de las principales opciones a considerar; se ha pensado, estratégicamente, en realizar una aplicación genérica con esta herramienta que cumpla con un conjunto de requisitos básicos de interés para la dirección del proyecto, para esto se hace necesario estudiar esta herramienta para incorporar las funcionalidades que la misma no contiene y que sean necesarias al proyecto así como también obviar aquellas que no son de interés que la herramienta contenga, con la intensión de utilizar esta aplicación y la documentación asociada al proceso de construcción de la misma, para acelerar y poder enfrentar solicitudes de clientes de forma satisfactoria.

El equipo de desarrollo lo componen especialistas, profesores y estudiantes bajo un modelo que relaciona tres componentes fundamentales: docencia, investigación y producción. Bajo este modelo, la estabilidad del personal implicado en un proyecto es limitada. Con relativa frecuencia entran nuevos miembros, se realizan movimientos y otras actividades que generan un contexto bastante dinámico. Por tal motivo, el

inicio de este proceso demanda que sean correctamente capturadas y documentadas las funcionalidades que debe cumplir la aplicación de software, que conduzca a su correcto diseño e implementación.

1.3 Desarrollo de Sistemas de Información Geográfica

El desarrollo de un SIG en cualquier organización es una tarea siempre progresiva, compleja, laboriosa y continua, considerando las características especiales de los datos que utiliza y sus correspondientes procesos de actualización. Los SIG están compuestos por 5 partes fundamentales: el hardware, el software, la información, el personal capacitado y los métodos de desarrollo, para el correcto desarrollo de un SIG no se pueden descuidar ninguna de estas partes, aunque evidentemente el mayor peso lo tienen la información con que se cuenta y el personal capacitado, ya que un SIG carente de cartografía y datos relevantes es inservible, de ahí que sea la información geográfica el elemento de más valor en un SIG, lo mismo sucede con el personal capacitado, sin las personas con los conocimientos necesarios para trabajar sobre un SIG la información se vuelve inútil, no importa con cuanta se cuente.

Las soluciones para muchos problemas frecuentemente requieren acceso a varios tipos de información que sólo pueden ser relacionadas por geografía o distribución espacial. Sólo la tecnología SIG permite almacenar y manipular información usando geografía, analizar patrones, relaciones, y tendencias en la información, todo con el interés de contribuir a la toma de mejores decisiones. Un SIG puede mapear cualquier información almacenada en plantillas o bases de datos, que tenga un componente geográfico que permita ver patrones, relaciones y tendencias, que no pueden verse en un formato de tabla o lista.

Los SIG representan los datos verdaderos del mundo que usan datos digitales. Si los datos existen en forma ordinaria entonces puede ser utilizado en convertir coordenadas dentro del formato digital. Mientras que los datos de entrada dentro del SIG no necesitan especificar las identidades de los objetos en el mapa así como sus relaciones espaciales.

En la actualidad existen muchas herramientas a nivel internacional propietarias y libres, que son utilizadas para la construcción de este tipo de sistemas, algunas con un enfoque de escritorio otra sobre la web. Tradicionalmente las aplicaciones de escritorio han sido los grandes representantes de las herramientas para la gestión de los SIG, las cuales permiten la manipulación de todo tipo de información de esta índole, así como su edición, análisis y explotación.

A nivel internacional existen muchas de estas herramientas, las cuales tienen sus particularidades. Precisamente el desarrollo de este epígrafe se centra en el análisis de un conjunto de soluciones que existen a nivel internacional. A continuación se presentan de forma sintetizada algunas de ellas con sus respectivas características.

Open JUMP

Open Jump es un SIG programado en Java y que basa su funcionalidad en plugins. Permite la conexión a servidores de cartografía WMS y existen plugins para numerosos formatos tanto de archivo como de servidores. Dentro de sus opciones más interesantes se encuentran las herramientas de edición de las que dispone, para modificar datos vectoriales, así como herramientas básicas de geoprocesamiento. Carece de opciones de georreferenciación. Pose una interfaz de usuario muy intuitiva, buen número de formatos soportados a través de plugins, incluyendo conexión a servidores, buen punto de partida para la creación de proyectos personalizados debido a la documentación existente y a la facilidad de implementación de nuevas funciones. Por otra parte, una debilidad pudiera ser la ausencia de algunas funcionalidades básicas como por ejemplo la impresión de cartografía y cuadrículas de las cuales muchas están en vías de solución y que existe cierta descoordinación en la generación de versiones, aunque actualmente se ha creado un comité para coordinar el desarrollo de las futuras versiones. (BERNI et al., 2005)

Kosmo

Kosmo es una de las primera Plataformas SIG Libre Corporativa, distribuidas bajo licencia GNU/GPL. En (MARGALEF y POLO) se plantea que se trata de una herramienta similar a JUMP puesto que está desarrollada sobre esta plataforma. Es fácilmente extensible y ha aprovechado esta potencialidad para añadir al proyecto algunas mejoras como un editor avanzado de simbología Styled Layer Descriptor (SLD), acorde con los estándares de la OGC. También permite la conexión a diversas bases de datos, dispone de un constructor de consultas y añade una herramienta para crear composiciones para imprimir. No obstante, no incluye la posibilidad de reproyectar capas.

Tiene integradas hoy día, en fase de análisis o en fase de integración diversos componentes de los principales proyectos libres existentes: PostgreSQL/PostGIS, MySQL, Geoserver, Deegree, Geonetwork,

Openlayers, GRASS, gvSIG, ImageJ, Openoffice, JTS, GDAL y Geotools. Es un proyecto de software libre aunque su entorno no está limitado a ese ámbito, estando diseñado para implantarse dentro de entornos corporativos en los que puede integrarse con alternativas comerciales: Oracle, ArcSDE.

Entre sus características generales se pueden destacar que es un cliente SIG de escritorio de funcionalidades avanzadas y que es una herramienta capaz de visualizar y procesar datos espaciales, que se caracteriza por poseer una interfaz de usuario amigable. Tiene la capacidad de acceder a múltiples formatos de datos, tanto vectoriales en fichero, como Shapefile o DXF, o en base de datos, como PostgreSQL, MySQL u Oracle, como raster: TIFF, ECW, MrSID, BMP, GIF, JPG, PNG, con capacidad de edición y, en general, ofreciendo numerosas utilidades al usuario SIG.

Posee funcionalidades de almacenamiento y gestión de la información geográfica en Sistemas de Gestión de Bases de Datos Relacionales (SGBDR), edición, consulta y explotación de la información geográfica desde el cliente de escritorio Kosmo, junto con funciones de geoprocesamiento, entre muchas otras, así como publicación de la información geográfica a través de protocolos estándar del OGC (WMS, WFS) y publicación de información y funcionalidades específicas en Internet, y acceso mediante el uso de los clientes ligeros Kosmo.

Su diseño y arquitectura está basado en la gestión y análisis de la información territorial a través de bases de datos espaciales. El proyecto está en pleno desarrollo, y con el primero de sus componentes, Kosmo-Desktop, en continua en evolución, y ya disponible para aquellos que requieren de avanzada funcionalidad en un SIG de escritorio potente.

Quantum GIS

En (MARGALEF y POLO) se expresa que Quantum GIS se trata de un SIG con una apariencia muy cuidada y que posee algunas características muy interesantes, tales como soporte directo para edición en PostGIS, conexión con GRASS para tareas como edición de topología, y buen número de formatos soportados, tanto vectoriales como matriciales. Además, añadir datos y cambiar la simbología es tan fácil y fiable como se podría esperar de un SIG competente. Es interesante el hecho de poder acceder a los metadatos de las capas cargadas.

Es un SIG que se ejecuta en Linux, Unix, Mac OSX y Windows, licenciado bajo la GNU General Public License, es software de fuente abierta y libre, que es muy fácil de usar. Está basado en plugins lo cual permite que se lleven a cabo tareas como la conversión de archivos shape de ESRI a PostGIS o conectarse a un GPS y mostrar su posición, pero no cuenta con herramientas de análisis. Soporta formatos vectoriales y raster.

1.3.1 GVSIG

Es una herramienta SIG sobre software libre que surge por el año 2002 a partir de la necesidad de la Consellería de Infraestructuras y Transporte (CIT, Valencia España) de migrar toda la informática de su organización a software libre. Una de las áreas de esa organización es el área de SIG y en ese momento según un análisis que ellos hicieron no existía una aplicación que pudiera sustituir a los programas comerciales utilizados, pero que sí existe la posibilidad de desarrollar un SIG libre que cumpliera con sus expectativas por lo que se dan a la tarea de financiar el desarrollo de esta herramienta a través de un concurso lanzado a la comunidad, la empresa ganadora de dicho concurso fue IVER Tecnologías de la Información, S.A., de la cual se tuvo la oportunidad de recibir su visita en la UCI donde impartieron conferencias sobre gvSIG e intercambiaron con estudiantes y profesores así como con los integrantes del proyecto GENESIG. Cuando el producto estuvo listo se publicó su primera versión en el año 2004 y hasta el momento han ido surgiendo nuevas versiones cada vez más perfeccionadas.

La herramienta gvSIG está orientada al manejo de información geográfica. Su interfaz es amigable y de fácil uso, es capaz de acceder a los formatos raster y vectoriales de forma rápida. Integra en una vista datos tanto locales como remotos a través de un origen WMS (Web Map Service), WCS (Web Coverage Service) o WFS (Web Feature Service). Cuando se construyó se tuvo en cuenta la extensibilidad del proyecto para permitir la incorporación de nuevas funcionalidades a la aplicación, así como desarrollar aplicaciones totalmente nuevas a partir de las librerías utilizadas en gvSIG, su licencia es de tipo GPL.

En (MARGALEF y POLO) se dice que gvSIG: “Se trata de un producto muy afianzado y orientado al usuario final, tanto a nivel de interfaz de usuario como de funciones implementadas. Soporta los formatos más populares de todas las tipologías de datos y permite trabajar con estándares del OGC. Se trata de un software con buenas capacidades vectoriales. No hace mucho ha incluido funcionalidades 3D, una

herramienta de optimización de rutas, así como un módulo de gestión de sistemas de referencia. Además, es fácilmente internacionalizable, hecho que explica la variedad de idiomas con los que está disponible".

Este Framework está desarrollado sobre JAVA, su interfaz se caracteriza por ser amigable y de fácil entendimiento. Permite cargar datos locales y remotos, entre los formatos que usa están: .SHP, .DXF, DWG, y DGN todos estos de tipo vectorial y .ECW, el MrSID, el GeoTIFF y otros de tipo raster. GvSIG posibilita cargar datos remotos a través de un origen WMS, WCS o WFS, y también de bases de datos espaciales como PostGIS y MySQL.

Las principales funcionalidades que incorpora gvSIG (GRUPO GVSIG, 2010):

- **Acceso a formatos vectoriales:** SHP, GML, DXF, DWG, DGN.
- **Acceso a formatos ráster:** BMP, GIF, TIF, TIFF, JPG, JPEG, PNG, VRT, DAT de ENVI, ERDAS (LAN, GIS, IMG), PCI Geomatics (PIX, AUX), ADF de ESRI, ILWIS (MPR, MPL), MAP de PC Raster, ASC, PGM, PPM, RST de IDRISI, RMF, NOS, KAP, HDR, RAW.
- **Acceso a servicios remotos**: OGC (WMS, WFS, WCS), ArcIMS, Ecwp.
- **Acceso a bases de datos y tablas:** PostGIS, MySQL, ArcSDE, Oracle, JDBC, CSV.
- **Navegación:** zooms, desplazamiento, gestión de encuadres, localizador.
- **Consulta:** información, medir distancias, medir áreas, hiperenlace.
- **Selección:** por punto, por rectángulo, por polígono, por polilínea, por círculo, por área de influencia, por capa, por atributos, invertir selección, borrar selección.
- **Búsqueda:** por atributo, por coordenadas.
- **Geoprocesos:** área de influencia, recortar, disolver, juntar, envolvente convexa, intersección, diferencia, unión, enlace espacial, translación 2D, reproyección, geoprocesos Sextante.
- **Edición gráfica:** añadir capa de eventos, rejilla, pila de comandos, deshacer/rehacer, copiar, simetría, rotar, escalar, desplazar, editar vértice, polígono interno, matriz, explotar, unir, partir,

autocompletar polígono, insertar punto, multipunto, línea, arco, polilínea, polígono, rectángulo, cuadrado, círculo, elipse.

- **Edición alfanumérica:** modificar estructura tabla, editar registros, calculadora de campos.
- **Representación vectorial:** símbolo único, cantidades (densidad de puntos, intervalos, símbolos graduados, símbolos proporcionales), categorías (expresiones, valores únicos), múltiples atributos, guardar/recuperar leyenda, editor de símbolos, niveles de simbología, bibliotecas de símbolos.
- **Representación raster:** brillo, contraste, realce, transparencia por píxel, opacidad, tablas de color, gradientes.
- **Etiquetado:** etiquetado estático, etiquetado avanzado, etiquetado individual.
- **Tablas:** estadísticas, filtros, orden ascendente/descendente, enlazar, unir, mover selección, exportar, importar campos, codificación, normalización.
- **Constructor de mapas:** composición de página, inserción de elementos cartográficos (Vista, leyenda, escala, símbolo de norte, cajetín, imagen, texto, gráfico), herramientas de maquetación (alinear, agrupar/desagrupar, ordenar, enmarcar, tamaño y posición), grid, plantillas.
- **Impresión:** impresión, exportación a PDF, a Postcript, a formato de imagen.
- **Redes:** topología de red, gestor de paradas, costes de giro, camino mínimo, conectividad, árbol de recubrimiento mínimo, matriz orígenes-destinos, evento más cercano, área de servicio.
- **Raster y teledetección:** estadísticas, filtrado, histograma, rango de escalas, realce, salvar a raster, vectorización, regiones de interés, componentes generales, georreferenciación, geolocalización, clasificación supervisada, cálculo de bandas, perfiles de imagen, árboles de decisión, componentes principales, fusión de imágenes, diagramas de dispersión, mosaicos.
- **Publicación:** WMS, WFS, WCS de MapServer, WFS de Geoserver.

- **3D y animación:** Vista 3D plana y esférica, capas 3D, simbología 3D, extrusión, edición de objetos 3D, encuadres 3D, animación 2D y 3D, visualización estéreo.
- **Topología:** construcción topológica, edición topológica, generalizar, suavizar, invertir sentido de líneas, convertir capa de líneas/polígonos a puntos, convertir capa de polígonos a líneas, triangulación de Delaunay/Poligonación de Thiessen.
- **Otros:** gestión de Sistemas de Referencia Coordenados, exportar/importar WMC, scripting, gestión de traducciones.

Vale destacar que gvSIG es uno de los SIG libres más completos de la actualidad por el amplio conjunto de funcionalidades que contiene y además la posibilidad de poder agregar más funciones a las ya existentes, de las funcionalidades que gvSIG presenta actualmente se destacan la funcionalidad 3D que permite la representación y el trabajo con cartografía en 3D, Constructor de mapas que permite la elaboración de mapas a los especialistas en cartografía y el conjunto de funcionalidades de geoprocesamiento entre las cuales están análisis de proximidad, área de influencia e intersección las cuales permiten realizar análisis lógicos que pueden servir para la toma de decisiones.

Permite la agregación de nuevas funcionalidades mediante extensiones sin necesidad de poseer un gran conocimiento del core de la aplicación gracias a su estructura, la cual está constituida por tres subsistemas: Andami, Fmap, gvSIG extensión.

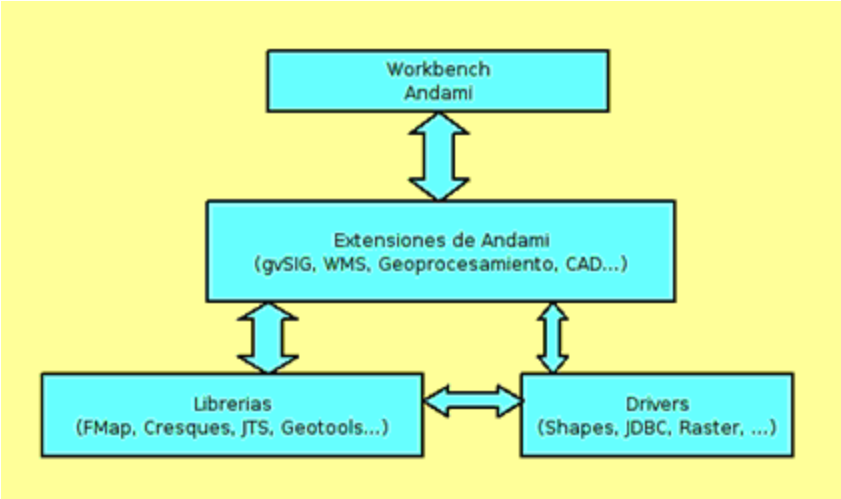


Figura 1.1 Estructura gvSIG

Andami: Representa el esqueleto en el que se sustenta gvSIG. Es como un armazón en el que van encajando las distintas extensiones que conformarán la aplicación. Además de esto se encarga de dotar de aspecto a las ventanas de la aplicación (CONSELLERIA DE INFRAESTRUCTURAS Y TRANSPORTE, 2008)

Fmap: Es el corazón SIG de la plataforma. Incluye todas las clases necesarias para manejar objetos SIG, así como drivers y adaptadores para utilizar los formatos más usados para el almacenamiento de los datos cartográficos. Dentro de esta librería se encuentran clases para leer y escribir los formatos soportados, dibujar los mapas a las escalas adecuadas, asignar leyendas, definir simbologías, realizar búsquedas, consultas, análisis, etc. (CONSELLERIA DE INFRAESTRUCTURAS Y TRANSPORTE, 2008)

gvSIG extensión: Se trata de una extensión que contiene la parte de interface de usuario que presenta los datos geográficos manejados por *Fmap*. En este subsistema se encuentran las clases que implementan la mayor parte de cuadros de diálogo que utiliza la aplicación final, así como las clases de soporte a esos cuadros de diálogo. Por ejemplo, aquí se encuentran los formularios para asignar leyendas, crear mapas y vistas, definir escalas, etc. (CONSELLERIA DE INFRAESTRUCTURAS Y TRANSPORTE, 2008)

1.4 Observaciones sobre el estudio de los SIG escritorio de la actualidad

Antes de la creación de este trabajo se realizó un estudio de las principales herramientas SIG a nivel mundial por un equipo de trabajo del proyecto GENESIG que arrojó varias opciones a considerar entre las que están las expuestas anteriormente y otras, de este estudio se tomaron las opciones más interesantes y estas fueron asignadas a grupos de trabajo para profundizar en su investigación y desarrollo si procedían. Entre esas asignaciones se encuentra este trabajo, por lo que aunque desde antes de empezar este trabajo ya se había escogido a gvSIG se ha querido compararlo con otros sistemas y plantear las ventajas que presenta el mismo.

Entre las ventajas que posee gvSIG está el hecho de ser multiplataforma, su licencia es GPL lo que asegura la modificación y redistribución sobre la misma licencia GPL. Siendo una aplicación de grandes potencialidades y complejas funciones no deja su interfaz de ser amigable y su aprendizaje es muy sencillo, lejos de ser gvSIG un software rígido y susceptible a los cambios es extensible lo que permite agregar nuevas funcionalidades, quitar aquellas que no son necesarias así como desarrollar nuevas funciones lo que claramente es una gran ventaja a la hora de personalizar un software para determinada entidad. Integra servicios web de WFS, WCS y WMS. Permite utilizar tipos de datos raster y vectoriales.

1.5 Conclusiones Parciales

Este capítulo comprende un conjunto de conceptos básicos como el de información geográfica y el de datos espaciales que contribuyen a un mejor entendimiento del marco conceptual que rodea al objeto de estudio, así como un conjunto de elementos que lo caracterizan. Unido a esto se realiza un análisis de la situación problemática por la cual surge la necesidad de realizar esta investigación y se realiza entonces un estudio de algunas herramientas similares a la que le concierne a este trabajo entre las que se trataron Kosmo, Open Jump, Quantum GIS y gvSIG. Se plantean algunas ventajas de gvSIG que hacen a esta herramienta una de las más populares de la actualidad y sin dudas una buena opción a la hora de desarrollar un SIG.

2

TENDENCIAS Y TECNOLOGÍAS ACTUALES

En el presente capítulo se realiza un estudio de las tendencias y tecnologías actuales para el desarrollo de software, dentro de las cuales se hace un análisis a las metodologías, herramientas CASE más relevantes a nivel mundial y al Lenguaje Unificado de Modelado (UML). Entre las metodologías que se abordan están metodologías ágiles y pesadas, algunas de ellas son: Microsoft Solution Framework (MSF), Rational Unified Porches (RUP), Xtreme Programing (XP), Dynamic Systems Development Method (DSDM), Feature Driven Development (FDD). Son muchas las herramientas CASE existentes en el mundo debido a que estas han gozado de mucha popularidad, en el presente capítulo solo se han tratado algunas de las más importantes como son: Enterprise Architect, Visual Paradigm, Rational Rose. Para finalmente hacer un análisis de cuales se adecuan más al Análisis de una solución de SIG sobre gvSIG en el Departamento GeoInformática.

2.1 Metodologías de desarrollo de software

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

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

actividades del proceso de desarrollo, por ejemplo, suele hablarse de métodos de análisis y/o diseño". (VALENCIA)

Por su parte, (PIATTINI 1996) define a la metodología de desarrollo de software como "un conjunto de procedimientos, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software".

Cada una de las metodologías existentes tienen características específicas que las hacen únicas; la gran cantidad de propuestas y diferencias en el grado de detalle, información disponible y alcance que poseen hacen que su clasificación sea muy difícil. A grandes rasgos, considerando su filosofía de desarrollo se pueden agrupar en: metodologías ágiles o ligeras y metodologías tradicionales o pesadas.

2.1.1 Metodologías Tradicionales de Desarrollo

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

"Este esquema tradicional para abordar el desarrollo de software ha demostrado ser efectivo y necesario en proyectos de gran tamaño (respecto a tiempo y recursos), donde por lo general se exige un alto grado de ceremonia en el proceso. Sin embargo, este enfoque no resulta ser el más adecuado para muchos de los proyectos actuales donde el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad. Ante las dificultades para utilizar metodologías tradicionales con estas restricciones de tiempo y flexibilidad, muchos equipos de desarrollo se resignan a prescindir del "buen hacer" de la ingeniería del software, asumiendo el riesgo que ello conlleva." (LETELIER, 2003).

A continuación se realiza una breve caracterización de algunas metodologías tradicionales que fueron estudiadas:

Microsoft Solution Framework (MSF)

Esta es una metodología que se centra en los modelos de proceso y de equipo. MSF es adaptable, escalable, puede organizar equipos pequeños de 3 ó 4 personas, así como también proyectos que requieren 50 personas o más. Es flexible, es utilizada en el ambiente de desarrollo de cualquier cliente. Puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.

MSF comprende cuatro fases dentro de sus procesos.

Fase 1 - Estrategia y alcance

- Elaboración y aprobación del Documento de Alcance y Estrategia definitivo.
- Formación del Equipo de Trabajo y distribución de competencias y responsabilidades.
- Elaboración del Plan de Trabajo.
- Elaboración de la matriz de Riesgos y Plan de Contingencia.

Fase 2 - Planificación y Prueba de Concepto

Deben alcanzarse tanto el Documento de Planificación y Diseño de Arquitectura, como el Documento de Plan de Laboratorio - Prueba de Concepto.

Fase 3 - Estabilización

La solución implantada en la maqueta se pasa a un entorno real de explotación, restringido en número de usuarios y en condiciones tales que se pueda llevar un control efectivo de la situación. Los hitos y objetivos fundamentales de esta fase son:

- Selección del entorno de prueba piloto.
- Gestión de Incidencias.
- Revisión de la documentación final de Arquitectura.
- Elaboración de la documentación de Formación y Operaciones.

- Elaboración del Plan de Despliegue.
- Elaboración del Plan de Formación.

Fase 4 - Despliegue

Se llevan a cabo en esta fase los planes diseñados en la anterior, principalmente el de despliegue y el de formación. Los principales trabajos e hitos a conseguir son implantación de la plataforma, puesta en servicio de todas las funciones, formación a los usuarios y administradores, continuación con las labores de recepción de incidencias, clasificación, tratamiento y resolución. El registro de mejoras y sugerencias, funcionalidades no cubiertas y novedades a incorporar en sucesivas versiones de la plataforma, incluyendo mejoras aportadas por los fabricantes de software. La revisión de las Guías y manuales de usuario, rectificación de errores y obtención de los documentos de formación definitivos.

MSF se compone de varios modelos encargados de planificar las diferentes partes implicadas en el desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el Modelo de Aplicación.

Rational Unified Process (RUP)

“El Proceso Unificado de Rational (RUP), es un proceso de ingeniería de software planteado por Kruchten (1996) cuyo objetivo es producir software de alta calidad, es decir, que cumpla con los requerimientos de los usuarios dentro de una planificación y presupuesto establecido. Cubre el ciclo de vida y desarrollo de software”. (DÍAZ-ANTÓN et al. 2004)

RUP toma en cuenta las mejores prácticas en el modelo de desarrollo de software en particular las siguientes (JACOBSON et al. 1999):

1. Desarrollo de software en forma iterativa.
2. Manejo de requerimientos.
3. Utiliza arquitectura basada en componentes.

4. Modela el software visualmente (modela con UML).
5. Verifica la calidad del software.
6. Controla los cambios.

RUP Proceso Unificado Racional es una metodología cuyo fin es entregar un producto de software. Se estructura todos los procesos y se mide la eficiencia de la organización. Es un proceso de desarrollo de software el cual utiliza el lenguaje unificado de modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

El RUP es un conjunto de metodologías adaptables al contexto y necesidades de cada organización. Describe como aplicar enfoques para el desarrollo del software, llevando a cabo unos pasos para su realización. Se centra en la producción y mantenimiento de modelos del sistema además de servir como guía de cómo usar UML.

El Proceso Unificado es un proceso de desarrollo de software cuyo ciclo de vida se caracteriza por:

- Dirigido por casos de Uso
- Centrado en arquitectura
- Iterativo e incremental

El RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al culminar cada una de ellos, estos a la vez se dividen en fases y donde se debe tomar una decisión importante. El ciclo de vida organiza las tareas en fases e iteraciones.

RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en las distintas actividades. En la siguiente figura se muestra cómo varía el esfuerzo asociado a las disciplinas según la fase en la que se encuentre el proyecto RUP.

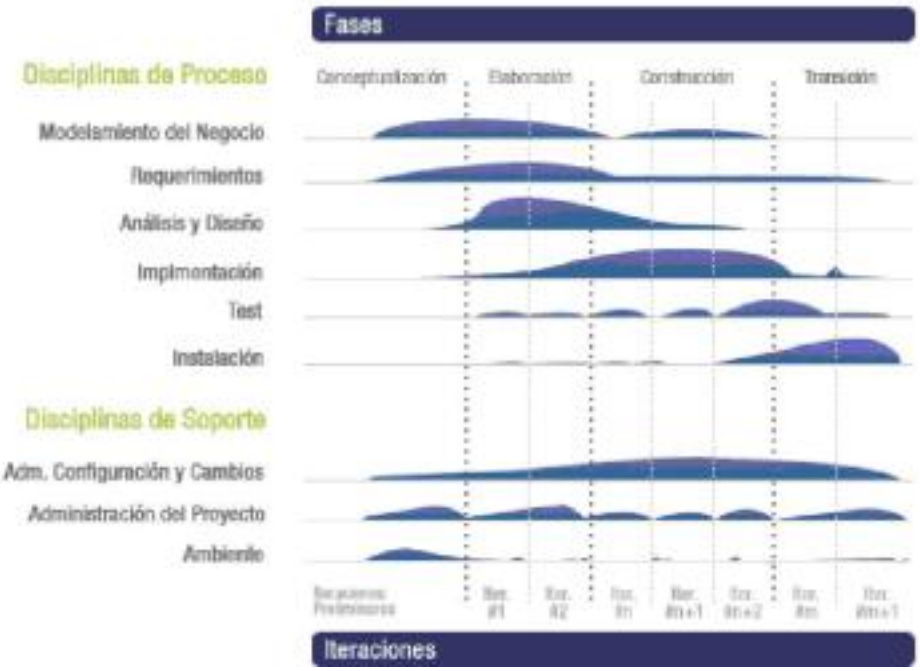


Figura 2.1 Fases y Flujos de Trabajo

Inicio: Se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos.

Elaboración: Se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos.

Construcción: Se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario.

Transición: Se Instala el producto en el cliente y se entrena a los usuarios. Surgen nuevos requisitos a ser analizados.

RUP en cada una de sus fases pertenecientes a la estructura estática realiza una serie de artefactos que sirven para comprender mejor tanto el análisis como el diseño del sistema.

RUP brinda una forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo). Pretende implementar las mejores prácticas en Ingeniería de Software, posee un desarrollo iterativo, administración de requisitos. Usa una arquitectura basada en componentes que posibilita el

control de cambios. También ofrece un modelado visual del software así como la verificación de la calidad del mismo. El RUP se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso.

La metodología RUP es más apropiada para proyectos grandes, dado que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas. En proyectos pequeños, es posible que no se puedan cubrir los costos de dedicación del equipo de profesionales necesarios.

2.1.2 Metodologías Ágiles de Desarrollo

En (PRESSMAN 2002) se dice que a principios de la década del ’90, surgió un enfoque que fue bastante revolucionario para su momento ya que iba en contra de la creencia de que mediante procesos altamente definidos se iba a lograr obtener software en tiempo, costo y con la requerida calidad. El enfoque fue planteado por primera vez en 1991 y se dio a conocer en la comunidad de ingeniería de software con el mismo nombre que su libro, RAD o Rapid Application Development. RAD consistía en un entorno de desarrollo altamente productivo, en el que participaban grupos pequeños de programadores utilizando herramientas que generaban código en forma automática tomando como entradas sintaxis de alto nivel.

Las metodologías ágiles están basadas en heurísticas provenientes de prácticas de producción de código, preparadas para cambios durante el proyecto y el cliente es parte del equipo de desarrollo. Los grupos de trabajo están diseñados con poco personal trabajando todos en el mismo sitio, generan pocos artefactos, poseen pocos roles, además de no hacer énfasis en la arquitectura del software.

Este tipo de metodología tiene entres sus principales características (MARTÍNEZ, 2005):

- La resistencia a las metodologías burocráticas
- Hicieron un postulado en lugar de una metodología
    - Individuos y sus interacciones son más importantes que procesos y herramientas
    - Software que funcione es más importante que documentación exhaustiva
    - Colaboración con el cliente en lugar de negociación de contratos

- Respuesta ante el cambio en vez de seguir un plan cerrado

Entre las metodologías ágiles se encuentran:

Extreme Programming (XP).

- SCRUM.
- Crystal Clear.
- Feature -Driven Development (FDD).
- Dynamic Systems Development Method (DSDM).
- Adaptive Software Development (ASD).

A continuación se realiza un estudio de varias de las Metodologías Ágiles para el desarrollo de Software referenciadas con anterioridad.

Extreme Programing (XP)

Esta metodología está centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP está guiada por una rápida programación y se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, en la reutilización de código, en la realización de pruebas a los principales procesos con el objetivo de tratar de obtener los posibles errores futuros, esto conocido como pruebas unitarias, en la simplicidad en las soluciones implementadas. Impone un alto nivel de disciplina entre los programadores, lo cual permite mantener un mínimo nivel de documentación que a su vez se traduce en una gran velocidad de desarrollo, además de proponer que el trabajo de los programadores sea en pares de forma tal que uno realice lo que el otro no hace en ese instante. XP se define como especialmente adecuada para proyectos de corto plazo con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico. Está formada por cuatro partes fundamentales que encierran sus características fundamentales, las cuales son: historia de usuarios, roles, procesos y prácticas.

XP se basa principalmente en tres aspectos:

- Pruebas Unitarias: se basa en las pruebas realizadas a los principales procesos.
- Re fabricación: se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- Programación en pares: una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento.

Esta metodología empieza en pequeño y añade funcionalidad con retroalimentación continua, donde el manejo del cambio se convierte en parte sustantiva del proceso y el costo del cambio no depende de la fase o etapa. No introduce funcionalidades antes que sean necesarias además que el cliente o el usuario se convierte en miembro del equipo

El cliente tiene derechos como decidir qué se implementa, saber el estado real y el progreso del proyecto, añadir, cambiar o quitar requerimientos en cualquier momento, obtener lo máximo de cada semana de trabajo y obtener un sistema funcionando cada 3 ó 4 meses.

Y además el desarrollador decide cómo se implementan los procesos, crea el sistema con la mejor calidad posible, pide al cliente en cualquier momento aclaraciones de los requerimientos, estima el esfuerzo para implementar el sistema y cambia los requerimientos en base a nuevos descubrimientos

En esta plataforma XP construye un proceso de diseño evolutivo que se basa en refactorizar un sistema simple en cada iteración. Todo el diseño se centra en la iteración actual y no se hace nada anticipadamente para necesidades futuras. El resultado es un proceso de diseño disciplinado, lo que es más, combina la disciplina con la adaptabilidad de una manera que indiscutiblemente la hace la más desarrollada de entre todas las metodologías adaptables.

Feature Driven Development (FDD)

Se propuso por Jeff De Luca y Peter Coad. Esta metodología fue desarrollada alrededor del año 1998.

FDD es un método ágil, iterativo y adaptativo. A diferencia de otras más, no cubre todo el ciclo de vida sino sólo las fases de diseño y construcción y se considera adecuado para proyectos grandes y de misión crítica.

FDD no requiere un modelo específico de proceso y se complementa con otras metodologías. Enfatiza cuestiones de calidad y define claramente entregas tangibles y formas de evaluación del progreso. Los principios de FDD son pocos y simples:

- Se requiere un sistema para construir sistemas si se pretende escalar a proyectos grandes.
- Un proceso simple y bien definido trabaja mejor.
- Los pasos de un proceso deben ser lógicos y su mérito inmediatamente obvio para cada miembro del equipo.
- Los buenos procesos van hasta el fondo del asunto, de modo que los miembros del equipo se puedan concentrar en los resultados.
- Los ciclos cortos, iterativos, orientados por rasgos son mejores.

Su ciclo de vida está compuesto por cinco procesos, estos procesos son: Desarrollar un Modelo Global, Construir una Lista de Features, Planificar por Feature, Diseñar por Feature y Construir por Feature. Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe reunir el software.

2.2 Lenguajes de Modelado

El lenguaje de modelado es usado por muchas organizaciones en combinación con una metodología de desarrollo de software para avanzar de una especificación inicial a un plan de implementación y para comunicar dicho plan a todo un equipo de desarrolladores. El uso de un lenguaje de modelado es más sencillo que la auténtica programación, pues existen menos medios para verificar efectivamente el funcionamiento adecuado del modelo.

2.2.1 Lenguaje Unificado de Modelado (UML)

Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir.

UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

Los Diagramas de Estructura enfatizan en los elementos que deben existir en el sistema modelado como son los Diagrama de clases, Diagrama de componentes, Diagrama de objetos, Diagrama de estructura compuesta (UML 2.0), Diagrama de despliegue, Diagrama de paquetes.

Los Diagramas de Comportamiento enfatizan en lo que debe suceder en el sistema modelado por ejemplo: Diagrama de actividades, Diagrama de casos de uso, Diagrama de estados.

Los Diagramas de Interacción son un subtipo de diagramas de comportamiento, que enfatiza sobre el flujo de control y de datos entre los elementos del sistema modelado como es el Diagrama de secuencia, Diagrama de comunicación, que es una versión simplificada del Diagrama de colaboración (UML 1.x), Diagrama de tiempos (UML 2.0), Diagrama de vista de interacción (UML 2.0).

UML permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos. También documenta todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, etc.). Este lenguaje de modelado es independiente del proceso, aunque para utilizarlo óptimamente se debería usar en un proceso que fuese dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental.

2.3 Herramientas Case

CASE significa Computer Aided Software Engineering, lo que viene a ser en español Ingeniería de Software Asistida por Computación. Las Herramientas CASE son las aplicaciones informáticas que dan asistencia a los profesionales durante todo el transcurso del ciclo de vida de desarrollo de un software, las etapas de este ciclo están determinadas en el siguiente orden: Investigación Preliminar, Análisis, Diseño, Implementación e Instalación. Las Herramientas CASE fueron desarrolladas para automatizar el proceso de desarrollo de un sistema contribuyendo a mejorar la calidad y la productividad.

2.3.1 Enterprise Architect

Enterprise Architect es una herramienta comprensible de diseño y análisis UML que permite modelar y gestionar información compleja, diseñar y visualizar software y construir y distribuir sistemas. Abarca el ciclo de vida completo del desarrollo de software. Enterprise Architect es una herramienta multi-usuario, basada en Windows, diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad.

Enterprise Architect soporta la ingeniería inversa y la generación de código de un amplio rango de lenguajes incluyendo C#, Java, PHP, Delphi, C++, VB.Net, Visual Basic y otros. El editor de código fuente posee un resaltador de sintaxis incorporado, Enterprise Architect permite navegar y explorar su modelo de código fuente en el mismo ambiente. Enterprise Architect le ayuda a administrar la complejidad con herramientas para rastrear las dependencias, soporte para modelos muy grandes, control de versiones, Líneas Base por cada punto del tiempo, la utilidad de comparar para seguir los cambios del modelo, interfaz intuitiva y de alto rendimiento con vista de proyecto.

2.3.2 Visual Paradigm

Es una herramienta CASE desarrollada por la Visual Paradigm Internacional una de las principales compañías de herramientas CASE. Es una herramienta CASE que utiliza UML como lenguaje de modelado para la construcción de los sistemas software. Es multiplataforma ya que se puede ejecutar sobre diferentes sistemas operativos. Además, presenta una fuerte integración con el lenguaje de programación Java gracias a su herramienta Enterprise JavaBeans que permite el despliegue distribuido,

transaccional, portable y seguro del uso. Soporta el ciclo de vida completo de desarrollo de software y permite realizar generación de código e ingeniería inversa de los datos.

Ofrece:

- 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.
- Soporte ORM - Generación de objetos Java desde la base de datos
- Generación de bases de datos - Transformación de diagramas de Entidad-Relación en tablas de base de datos
- Ingeniería inversa de bases de datos - Desde Sistemas Gestores de Bases de Datos (DBMS) existentes a diagramas de Entidad-Relación
- Generador de informes para generación de documentación
- Distribución automática de diagramas - Reorganización de las figuras y conectores de los diagramas UML
- Editor de figuras.

- Soporta una amplia gama de bases de datos donde se incluyen: Oracle, Microsoft SQL Server, PostgreSQL, MySQL.
- Se puede integrar a los principales IDEs: Eclipse, Borland JBuilder, NetBeans, Sun ONE, IntelliJ IDEA, Oracle JDeveloper.
- Es multiplataforma, disponible para los Sistemas Operativos Linux, Windows, y Mac OS.

2.3.3 Rational Rose

Rational Rose es una herramienta desarrollada por Rational, una compañía que se dedica exclusivamente al desarrollo de herramientas de desarrollo de software de alto nivel. Esta herramienta se encarga tanto de llevar a cabo la automatización de los sistemas para la generación de código como de las labores de ingeniería inversa. Ayuda a una mejor comprensión del sistema y de sus distintos componentes, se puede aplicar ingeniería inversa a una serie de códigos distintos siempre y cuando estos sean orientados a objeto.

Rational es un agrupamiento de metodologías y herramientas que abarca todos los aspectos del desarrollo de software, desde su concepción hasta la elaboración del producto. Sus productos están centrados en la metodología del Proceso Unificado de Desarrollo de Software (RUP). Propone el desarrollo de software basado en las mejores prácticas recopiladas en innumerables proyectos exitosos y proporciona un lenguaje común de modelado para el equipo que facilita la creación de software de calidad rápidamente. No es multiplataforma y además necesita de mucha memoria para poder ser manejado de forma rápida y eficiente lo que representan desventajas de su uso.

El navegador UML de Rational Rose permite establecer una trazabilidad real entre el modelo y el código ejecutable. Facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información: Vista de Casos de Uso, Vista Lógica, Vista de Componentes y Vista de Despliegue, pero utilizan un lenguaje común para comprender y comunicar la estructura y la funcionalidad del sistema en construcción.

Ofrece:

- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo desarrollo que facilita la comunicación.
- Capacidades de ingeniería inversa.
- Modelo y código que permanece sincronizado ciclo de desarrollo.

2.4 Observaciones del estudio de las Tendencias y Tecnologías.

Luego del estudio realizado a varias metodologías ágiles y pesadas, teniendo en cuenta que existen muchas y no existe una universal sino que las particularidades de cada proyecto orientan la utilización de una u otra metodología. Teniendo en cuenta que RUP es altamente configurable y adaptable a las exigencias del software en construcción lo que permite reducir el gran volumen de documentación que posee a solo los artefactos que interesan al equipo de desarrollo. RUP es una metodología estándar a nivel mundial, por su característica de ser iterativo e incremental brinda una alta retroalimentación, posee una alta capacidad de generar documentación relevante del negocio, además existe un dominio de esta metodología del personal del proyecto y define claramente las actividades que se realizan por roles. Esta investigación constituye una etapa dentro de un flujo de procesos, la información generada será utilizada como entrada de otros procesos de desarrollo de software por lo que se concluye que la metodología a usar sea RUP ya que es la que más se acerca a las exigencias del equipo de desarrollo del proyecto GENESIG.

Como herramienta CASE se escoge a Visual Paradigm ya que soporta muy bien el lenguaje de modelado que se utiliza (UML), Visual Paradigm es independiente de la metodología que se usa, es una herramienta que goza de gran popularidad en el mundo y su utilidad y calidad están probadas. También soporta el ciclo de vida completo de desarrollo de software y posibilita la generación e ingeniería inversa de código. Además de ser multiplataforma, posee la herramienta Enterprise JavaBeans que permite una fuerte integración con el lenguaje Java el cual es precisamente el lenguaje sobre el cual está desarrollado el framework gvSIG.

Por lo que se concluye que la metodología de desarrollo a utilizar en el Análisis de una solución de SIG sobre gvSIG en el Departamento Geoinformática es RUP, así como la herramienta CASE para modelar los artefactos generados será Visual Paradigm, teniendo como lenguaje de modelación UML.

2.5 Conclusiones Parciales

En este capítulo se trataron las tendencias y tecnologías actuales a utilizar, indagando en cada uno de los conceptos fundamentales. Se realiza un estudio de las metodologías de desarrollo de software más relevantes de la actualidad. Se hace referencia al lenguaje de modelado UML, así como también se hace un estudio de las herramientas CASE más populares y sus principales potencialidades a la hora de desarrollar un software.

Posterior al análisis de las distintas tendencias y tecnologías propuestas en el presente capítulo, se concluyen cuales han de utilizarse en el Análisis de una solución de SIG sobre gvSIG en el Departamento Geoinformática. Concluyendo que se utilizará Visual Paradigm como herramienta CASE, como metodología a RUP y UML de lenguaje de modelado.

3

DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA

En este capítulo primeramente se realiza el modelamiento del negocio, dentro del mismo se realiza Modelo de Dominio. Posteriormente se exponen los requisitos funcionales y no funcionales con los cuales el sistema deberá cumplir, también se efectúa la descripción del sistema, donde se obtienen los Actores del Sistema, el Modelo de Casos de Uso del Sistema y la descripción textual de cada caso de uso. Por último se realiza el análisis, dentro del mismo se exponen los diagramas de clases del análisis y los diagramas de colaboración.

3.1 Modelo de Dominio en el Desarrollo de Software

Un Modelo de Dominio es un caso especial de modelo de negocio, es decir, este modelo es un subconjunto del Modelo de Objetos del Negocio, hay que aclarar que no siempre que haya Modelo de Dominio existe el Modelo de Negocio. El objetivo del Modelo de Dominio es comprender y describir las clases más importantes dentro del contexto del sistema. Se emplea fundamentalmente cuando la información que se tiene no es suficiente como para hacer el Modelo del Negocio, también cuando se hace imposible realizar subsistemas en el proceso de desarrollo del software debido a las múltiples interconexiones.

Algunas veces, como en los dominios de negocio muy pequeños, no es necesario desarrollar un modelo de objetos para el dominio; en su lugar, puede ser suficiente un glosario de términos.

El Modelo de Dominio (Modelo Conceptual) es una representación visual de los conceptos u objetos significativos del mundo real para un problema o área de interés. Representa clases conceptuales del dominio del problema, conceptos del mundo real, no de los componentes de software. En este modelo no

se muestra comportamiento. Las clases conceptuales pueden tener atributos pero no métodos. (PRESSMAN 2002)

El Modelo de Dominio se describe mediante diagramas UML, específicamente mediante diagramas de clases conceptuales significativas en el dominio del problema, donde se muestran a los clientes, usuarios, revisores y a otros desarrolladores las clases del dominio y cómo se relacionan unas con otras a través de asociaciones.(JACOBSON et al. 1999)

Al no tener bien definidos los procesos del negocio, se procede a realizar una modelación del dominio, posteriormente se explican los conceptos que forman parte del diagrama de clases del Modelo de Dominio para tener una mejor comprensión de la estructura y dinámica de la organización y finalmente se da una explicación breve de este diagrama.

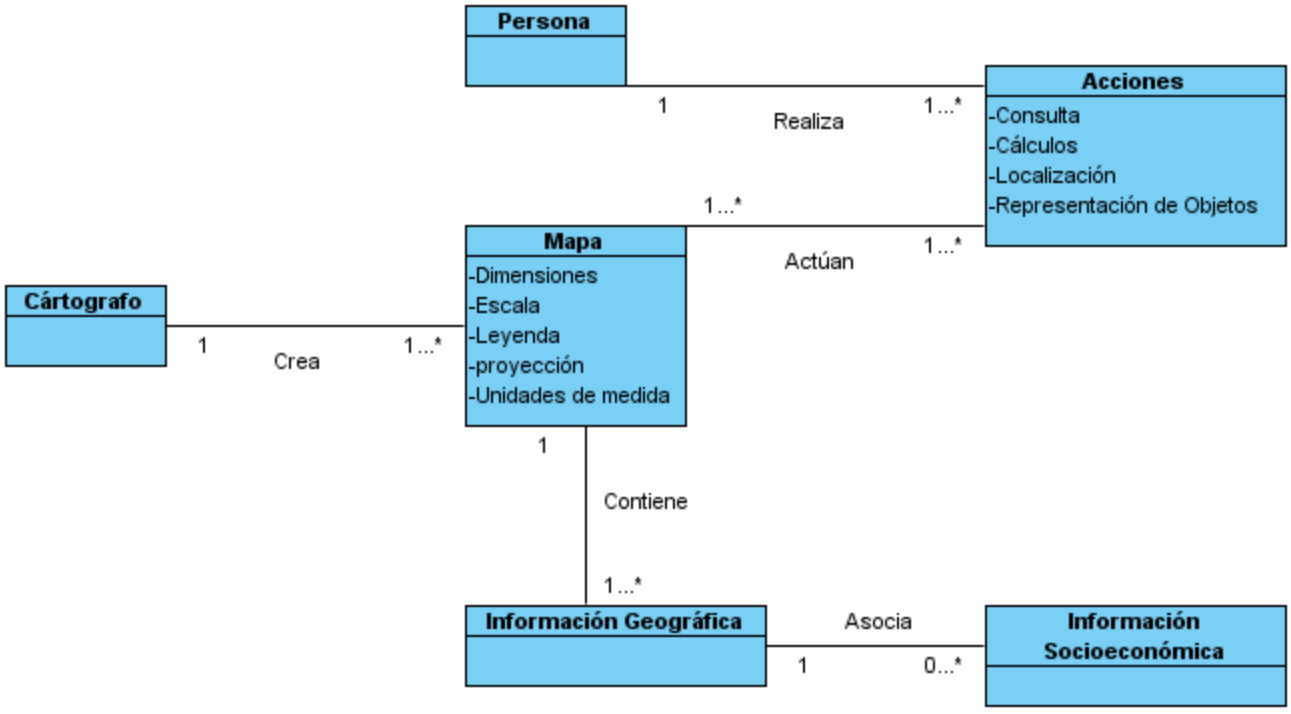


Figura 3.1 Diagrama de Clases del Dominio

3.1.1 Definición de las clases del Modelo del Dominio

Cartógrafo

Persona que ejerce la cartografía como profesión, la cartografía es la ciencia que estudia la secuencia de etapas y procesos para la visualización de un espacio geográfico mediante la creación de mapas, cartas y planos, es el encargado de crear los mapas.

Mapa

Es un modelo gráfico de la superficie terrestre donde se representan las características naturales o artificiales de una superficie y sus propiedades, mediante un mapa es posible tomar medidas de distancia, ángulos o superficies sobre él y obtener un resultado aproximadamente exacto.

Persona

Cualquiera que necesite trabajar con la información que brindan los mapas.

Acciones

Son aquellas actividades que pueden ejercer las personas sobre un mapa para sacar beneficios de estos. Entre estas acciones están la consulta, cálculos, representación de objetos y localización.

Información Geográfica

Es cualquier información que pueda ser geográficamente referenciada, es la que describe un sitio especifico o está asociada al mismo. Entiéndase por información como el conjunto de datos que al relacionarse adquieren sentido o un valor de contexto.

Información Socioeconómica

Es toda la información referente a lo social y a lo económico que se asocia a las localidades geográficas que se representan en los mapas. Son los datos de interés de cualquier índole que pueden asociarse a una localidad, por ejemplo los aeropuertos, lugares de interés y puertos son algunos que se encuentran

habitualmente en un mapa, también la densidad poblacional, los lugares de riesgo de incendio y otros son datos socioeconómicos que se pueden representarse en un mapa.

3.1.2 Breve descripción del diagrama

El cartógrafo se encarga de realizar uno o varios mapas, un mapa está constituido por dimensiones, escala, leyenda, proyección y unidades de medidas, estos mapas contienen información geográfica de interés, esta información geográfica puede tener asociada o no información socioeconómica. Las personas que poseen los conocimientos básicos para trabajar sobre estos mapas pueden realizar varias acciones sobre estos con el objetivo de obtener algún beneficio de estos.

3.2 Requerimientos

Dentro de la etapa de concepción la especificación de requisitos es una de las tareas más importantes para el desarrollo de un software. Es muy importante que se haga un levantamiento de requisitos con calidad para que el equipo de desarrollo conozca bien que se quiere exactamente de la aplicación antes de comenzar a desarrollar la misma con el objetivo de no trabajar de más y de no omitir alguna funcionalidad o capacidad de importancia.

Un requisito (requerimiento) es una característica de diseño, una propiedad o un comportamiento de un sistema. Los requisitos constituyen la descripción de los deseos o de las necesidades de un producto. (JACOBSON et al. 1999)

La obtención de los requisitos se basó principalmente en el estudio previo del desarrollo de SIG y de las funcionalidades de la herramienta gvSIG, además se utilizaron un conjunto de técnicas asociadas con este proceso como la entrevista, que se les realizó a personal del proyecto GENESIG y la técnica Lluvia de Ideas, donde cada uno de los integrantes del equipo de desarrollo expresa su punto de vista e ideas relacionadas con las funcionalidades que requiere el producto.

Los requerimientos se dividen en dos grupos, requerimientos funcionales y requerimientos no funcionales. Estos deben ser necesarios, concisos, completos, consistentes, verificables y no ambiguos.

Al lograr identificar los requerimientos iníciales del software y realizar el análisis de los mismos mediante la técnica de verificación, donde se limaron las ambigüedades y la falta de consistencia, se arrojaron los requisitos funcionales y no funcionales presentados a continuación.

3.2.1 Requisitos Funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Son el punto de partida para identificar qué debe hacer el sistema. Los requerimientos funcionales definen las funciones que el sistema será capaz de realizar. (JACOBSON et al. 1999)

El sistema a construir servirá de base para la posterior implementación de aplicaciones SIG específicas a un negocio determinado, por tanto, se pretende que incluya solo un conjunto de requerimientos básicos, funcionalidades generales para este tipo de aplicación.

RF 1. Autenticar usuario.

RF 2. Gestionar usuario.

RF 2.1 Adicionar usuario

RF 2.2 Modificar usuario

RF 2.3 Eliminar usuario

RF 3. Crear Vista.

RF 4. Abrir Proyecto.

RF 5. Salvar Proyecto.

RF 6. Salvar el proyecto bajo otro nombre y/o en otra localidad.

RF 7. Salir de la aplicación.

RF 8. Visualizar mapa (a partir de información geográfica almacenada y según los estándares de representación).

RF 9. Mostrar referencia al área que se visualiza respecto al área total del mapa.

RF 10. Mostrar las capas de información que componen el mapa en el orden jerárquico correspondiente.

RF 11. Cambiar el estado de visualización de una capa determinada.

RF 12. Cambiar el orden de las capas.

RF 13. Navegar en el mapa (explorar).

RF 13.1. Mover (panear) el mapa (pan)

RF 13.2. Aumentar el nivel (escala) de visualización (zoom in).

RF 13.3. Disminuir el nivel (escala) de visualización (zoom out).

RF 13.4. Ver el mapa completo (zoom extend)

RF 13.5. Ir al nivel anterior de visualización, si existe.

RF 13.6. Ir al nivel siguiente de visualización, si existe.

RF 14. Gestionar capas. Incluye Agregar, Modificar y Eliminar capa, permite realizar estas acciones sobre un proyecto y con cada tipo de capa.

RF 14.1. Gestionar capas vectoriales.

RF 14.1.1 Agregar capa vectorial

RF 14.1.2 Modificar capa vectorial

RF 14.1.3 Eliminar capa vectorial

RF 14.2. Gestionar capas raster.

RF 14.2.1 Agregar capa raster

RF 14.2.2 Modificar capa raster

RF 14.2.3 Eliminar capa raster

RF 14.3. Gestionar capas WMS.

RF 14.3.1 Agregar capa WMS

RF 14.3.2 Eliminar capa WMS

RF 14.4. Gestionar capas WFS.

RF 14.4.1 Agregar capa WFS

RF 14.4.2 Eliminar capa WFS

RF 14.5. Gestionar capas WCS.

RF 14.5.1 Agregar capa WCS

RF 14.5.2 Eliminar Capa WCS

RF 14.6. Gestionar capas GeoBD.

RF 14.6.1 Agregar capa GeoBD

RF 14.6.2 Modificar capa GeoBD

RF 14.6.3 Eliminar capa GeoBD

RF 15. Seleccionar elementos sobre una o varias capas.

RF 15.1. Seleccionar a partir de la definición de un punto.

RF 15.2. Seleccionar a partir de la definición de un rectángulo.

RF 15.3. Seleccionar a partir de la definición de un polígono.

RF 16. Borrar selecciones realizadas. Se quiere que cuando se realice una selección de las que se exponen en el RF 15 el usuario tenga la opción de borrar la selección realizada.

RF 17. Medir distancia entre dos puntos o entre varios (distancia acumulada).

RF 18. Calcular área. El usuario tendrá la posibilidad de contar con esta funcionalidad para realizar cálculos de área sobre el mapa que se encuentre trabajando.

RF 19. Localizar un punto en el mapa dadas sus coordenadas. Dadas determinadas coordenadas (X, Y) centrar el mapa en el punto correspondiente a esas coordenadas.

RF 20. Visualizar ayuda. El usuario en todo momento podrá consultar la ayuda en busca de respuestas, la misma debe ser lo más detallada posible.

RF 21. Navegar en una de las direcciones cardinales o en una determinada por un ángulo establecido, con un factor de desplazamiento dado.

Al comparar estos requerimientos con la aplicación gvSIG original podemos encontrar que los requerimientos RF1, RF2 y RF21 no se encuentran reflejados en la misma, y también que algunas funcionalidades que presenta esta aplicación no se toman como requerimientos aquí, ya que por su complejidad y uso no cumplen objetivo en una aplicación base, en el caso de los requerimientos funcionales RF1 y RF2 se decidió incorporarlos por el hecho de que como el objetivo de la aplicación que se realizará está dirigido a servir de base para realizar personalizaciones a empresas interesadas, se cree que en todos los casos seria un punto clave garantizar la seguridad de los datos de estas empresas. En el caso del RF21 es una funcionalidad que no está contemplada en el framework gvSIG a pesar de que este es muy completo ya que tiene un conjunto amplio de funcionalidades (ver epígrafe 1.2.2 para más información). Esto se debe a que es requerida fundamentalmente por usuarios especializados, como por ejemplo el personal de perfil militar, pero es el caso de que estos usuarios son potenciales en el uso de las tecnologías desarrolladas por el proyecto GENESIG pues básicamente el surgimiento del mismo fue por necesidades del país en este ámbito por lo que se hizo necesario incluirla como funcionalidad básica.

3.2.2 Requisitos No Funcionales

Los requerimientos no funcionales, son propiedades o cualidades que el producto debe tener. Representan las características que hacen al producto atractivo, usable, rápido o confiable.

Los requisitos no funcionales especifican propiedades del sistema, como restricciones del entorno o de la implementación, rendimiento, dependencias de la plataforma, facilidad de mantenimiento, extensibilidad y fiabilidad. (JACOBSON et al. 1999)

Usabilidad

- El sistema podrá ser usado por cualquier persona con los conocimientos informáticos básicos.
- El software tendrá siempre visible la opción de Ayuda.
- El usuario tendrá la posibilidad de consultar el manual de usuario con las especificaciones de cómo trabajar con la aplicación.
- La interfaz no debe ser compleja de tal forma que los usuarios puedan interactuar con la misma de forma sencilla.

Seguridad

- Los usuarios deben autenticarse antes de entrar al sistema.
- Identificar al usuario antes de que pueda realizar cualquier acción sobre el sistema.
- Proteger la información manejada por el sistema de accesos no autorizados.
- Las funcionalidades del sistema se muestran de acuerdo al nivel de usuario que esté activo.

Requerimientos del Software

- PostgreSQL como Sistema Gestor de Base de Datos.
- PostGis como extensión de PostgreSQL para el soporte de datos espaciales.

Requerimientos del Hardware

- 256 MB RAM como mínimo, se recomienda 512 MB RAM.
- Se requiere al menos 500 MB de disco duro para su instalación.

Confiabilidad

- La información que se maneja por el sistema debe estar protegida de acceso no autorizado y divulgación.

Rendimiento

- El funcionamiento de esta aplicación debe ser óptimo en cuanto a tiempo de respuesta y velocidad de procesamiento.

Diseño

- La aplicación debe ser altamente configurable y extensible, permitiendo agregar y quitar funcionalidades de acuerdo a las necesidades del cliente.
- Se debe lograr un diseño sencillo buscando siempre la comodidad de los usuarios finales, así como se deben tener en cuenta los estándares para un correcto diseño.

Portabilidad

- Multiplataforma, sistemas operativos Windows y Linux.

Legales

- El sistema debe ajustarse y regirse por la ley, en este caso la licencia de gvSIG es GPL y todos los demás software se usan de acuerdo a la ley.
- Como producto se distribuye amparado bajo las normativas legales establecidas en el registro comercial emitido por las entidades jurídicas de la UCI.

Ayuda

- La aplicación tiene que contar con un manual de ayuda.

3.3 Descripción del sistema

Para continuar con las actividades que propone la metodología RUP se procede a realizar la descripción del Modelo de Casos de Uso del Sistema (MCUS) correspondiente al flujo de trabajo requerimientos, dentro de esta se agrupan lo requerimientos funcionales en Casos de Uso del Sistema (CUS), se identifican los Actores del Sistema y se describen los CUS. Para conformar los casos de uso se agrupan los requerimientos del software según sus características los cuales son fragmentos de funcionalidades que el sistema ofrece para aportar un resultado de valor para sus actores.

3.3.1 Actores del Sistema

Los Actores del Sistema son agentes externos al sistema que colaboran con el mismo para obtener un beneficio determinado.

Tabla 3.1 Actores del Sistema

Actor	Descripción
Usuario	Persona que una vez autenticado utilizará, según el nivel de acceso que tenga, las funcionalidades que ofrece la aplicación.
Administrador del Sistema	Persona que se encarga de la seguridad del sistema.

3.3.2 Modelo de Casos de Uso del Sistema

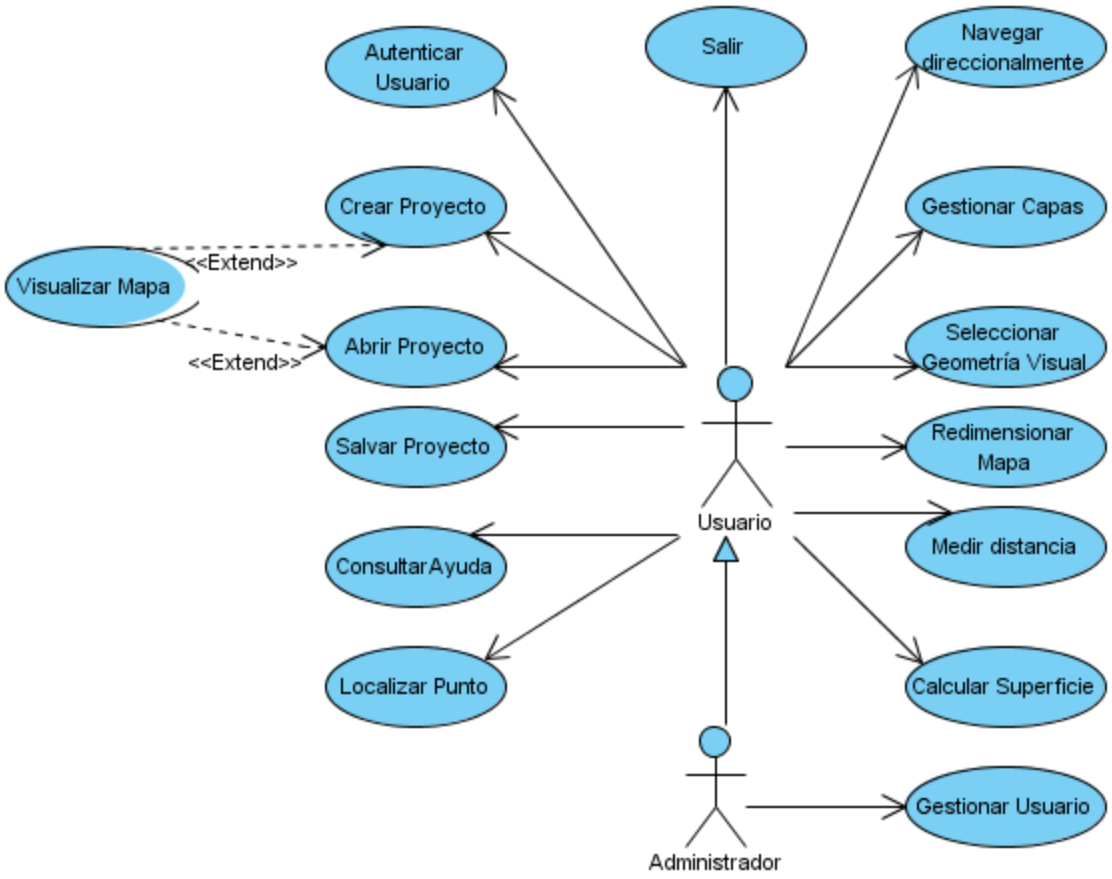


Figura 3.2 Modelo de Casos de Uso del Sistema

3.4 Descripción textual de Casos de Uso del Sistema

3.4.1 Navegar direccionalmente

Tabla 3.2 Descripción CUS: Navegar direccionalmente

Caso de Uso:	Navegar direccionalmente
Actores:	Usuario

Propósito	Este caso de uso se lleva a cabo con el objetivo de que el usuario tenga la posibilidad de navegar en una de las direcciones cardinales (N, S, E, O, NE, NO, SE, SO), así como también especificar el ángulo y la distancia a desplazarse.
Resumen:	El caso de uso se inicia cuando el actor necesita mover el encuadre (región visible del mapa) hacia una dirección determinada, esta dirección puede ser especificada por el usuario o el mismo puede escoger una de las predeterminadas (N, S, E, O, NE, NO, SE, SO) y finaliza cuando el sistema ejecuta la acción y la visualiza.
Precondiciones:	Debe estar abierta al menos una vista, la cual no puede ser vacía (esta vista debe estar compuesta por al menos una capa).
Referencias	RF 21
Prioridad	Auxiliar

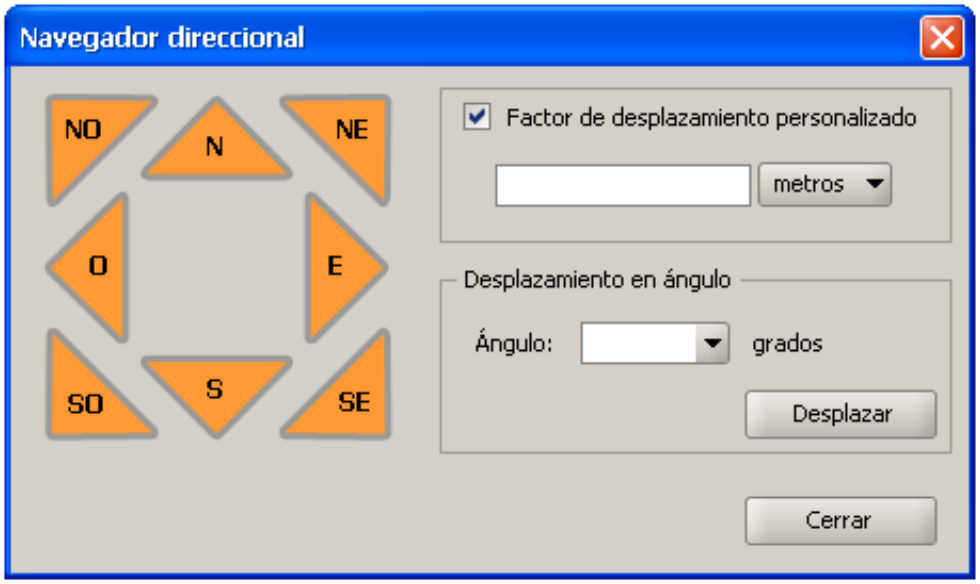
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El actor selecciona la opción navegador direccional dando clic sobre el menú “vista” y posteriormente dando clic sobre esta opción.	**2.** El sistema muestra la ventana Navegador direccional, la misma cuenta con los botones desplazar hacia el N, S, E, O, NE, NE, NO, SE, SO, también se encuentra la opción de especificar el factor de desplazamiento mediante un cuadro de texto (debe permitir escribir sólo números positivos) y una lista de selección para las

	unidades de medida; una lista de selección editable (que permita escribir un valor además de escoger) para un ángulo para el desplazamiento (debe permitir escribir sólo números enteros positivos). Ver prototipo de interfaz.
3. El actor puede realizar una de las siguientes acciones: **a)** Desplazar el encuadre (región visible del mapa) en una dirección cardinal (N, S, E, O, NE, NE, NO, SE, SO) **b)** Cambiar factor de desplazamiento. **c)** Desplazar el encuadre según un ángulo especificado.	**4.** El sistema ejecuta una acción: a) Si el actor realiza la acción "Desplazar el encuadre en una dirección cardinal" ir a sección "Desplazar en una dirección cardinal" b) Si el actor realiza la acción "Cambiar factor de desplazamiento" ir a sección "Cambiar factor de desplazamiento" c) Si el actor realiza la acción "Desplazar el encuadre según un ángulo" ir a sección "Desplazar según un ángulo"
Sección "Desplazar en una dirección cardinal"	
1. El actor selecciona una de las direcciones cardinales por defecto dando clic sobre el botón	**2.** El sistema desplaza el encuadre hacia la dirección especificada por el actor con un factor (distancia) de desplazamiento (uno por defecto si

correspondiente.	no se ha establecido uno por el actor).
Sección "Cambiar factor de desplazamiento"	
1. El actor marca la casilla de verificación correspondiente a la opción de especificar el factor de desplazamiento.	**2.** El sistema habilita el cuadro de texto para que el usuario introduzca el factor de desplazamiento y la lista de selección para las unidades de distancia.
3. El actor introduce en el cuadro de texto el número correspondiente a la distancia con la que desea que se desplace el encuadre, y escoge la unidad de medida (metros, kilómetros, millas) de la lista de selección.	**4.** El sistema establece el nuevo factor de desplazamiento introducido que tendrá efecto la próxima vez que se ejecute el flujo de la sección "Desplazar en una dirección cardinal" o "Desplazar según un ángulo".
Sección "Desplazar según un ángulo"	
1. Para realizar un desplazamiento con un ángulo diferente de los básicos, el actor tiene dos opciones: **a)** Escoge uno de los ángulos que se encuentran en la lista de selección por defecto y da clic sobre el botón "Desplazar". **b)** Escribe el número del ángulo deseado en la lista de selección	**2.** El sistema desplaza el encuadre hacia la dirección determinada por el ángulo especificado por el actor, con un factor (distancia) de desplazamiento (uno por defecto si no se ha establecido uno por el actor).

(que debe permitir la escritura) y da clic sobre el botón “Desplazar”.	

Interfaz 1



Flujos Alternos

Sección “Cambiar factor de desplazamiento”

Acción del Actor	Respuesta del Sistema
1.2 El actor introduce el número correspondiente a la distancia con la que se desplazará el encuadre.	**2.1** El número es mayor que el soportado por el sistema y se muestra un mensaje de error.
1.2 La casilla de verificación correspondiente a la opción de especificar el factor de desplazamiento se encuentra	**2.2** El sistema deshabilita el cuadro de texto para introducir un factor de desplazamiento personalizado y la lista de selección para las unidades

marcada y el actor procede a desmarcarla.	de medida.
Sección "Desplazar según un ángulo"	
1.1 El actor escribe el número del ángulo (un valor mayor que 360) y da clic sobre el botón desplazar.	**2.1** El sistema calcula el resto de la división entera por 360 y es el ángulo que se procesa y además sustituye el valor del campo escrito por el actor por el calculado.

3.5 Análisis

3.5.1 Modelo de análisis

El modelo de análisis es la primera aproximación al modelo de diseño, permite razonar sobre los aspectos internos del sistema y contribuye a refinar los requerimientos.

En (JACOBSON, I et al., 1999) se plantea que durante el análisis, se analizan los requisitos que se describen en la captura de requerimientos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que nos ayude a estructurar el sistema entero, incluyendo su arquitectura.

3.5.2 Diagramas de Clases del análisis

Las clases del análisis representan abstracciones de una o varias clases del sistema, las cuales se caracterizan por centrarse en el tratamiento de los requisitos funcionales. Un diagrama de clases del análisis es un artefacto que se genera en el análisis. Las clases del análisis pueden ser de tipo interfaz, controladora y entidad, cada una posee un estereotipo distintivo en UML.

A continuación se muestra uno de estos diagramas, el DC Navegar Direccionalmente, que por constituir un aporte al framework gvSIG es uno de los más importantes. Los diagramas correspondientes a los restantes casos de uso pueden ser consultados en el Anexo 1 (en la versión digital).

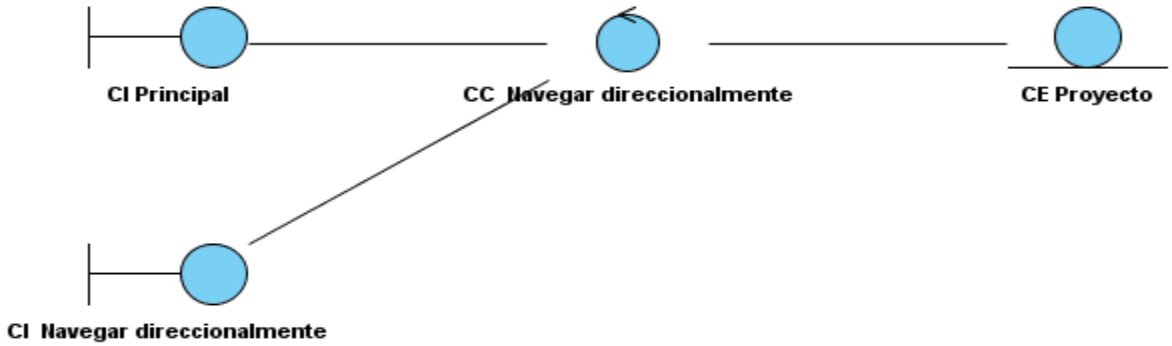


Figura 3.3 DC Navegar Direccionalmente

3.5.3 Diagramas de Colaboración

En (JACOBSON et al. 1999) se plantea que los diagramas de colaboración muestran las interacciones entre objetos creando enlaces entre ellos y añadiendo mensajes a esos enlaces. El nombre de un enlace debería denotar el propósito del objeto invocante en la interacción con el objeto invocado. Esta es más o menos la misma información que la mostrada por los diagramas de secuencia, pero destacando la forma en que las operaciones se producen en el tiempo. En los diagramas de colaboración los mensajes enviados de un objeto a otro se representan mediante flechas, mostrando el nombre del mensaje, los parámetros y la secuencia del mensaje. Los diagramas de colaboración están indicados para mostrar una situación.

En el siguiente diagrama de colaboración “DC Navegar Direccionalmente”, se pueden observar las interacciones entres las clases del CU “Navegar Direccionalmente”, lo que ayuda a conocer mejor la forma en que el mismo funciona y sus pasos lógicos. Los demás diagramas se encuentran en el anexo 2 (en la versión digital).

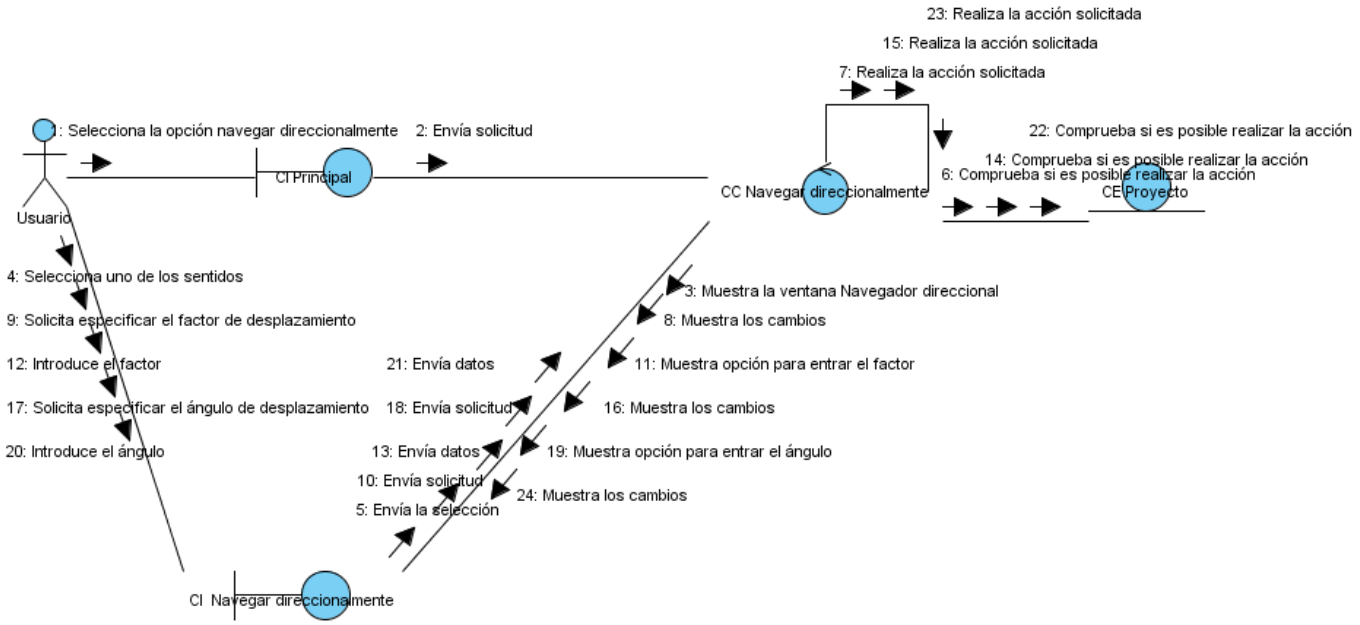


Figura 3.4 DCO Navegar Direccionalmente

3.6 Conclusiones Parciales

Tras la realización de este capítulo quedo conformada la solución propuesta, la cual servirá de guía para el diseño y posterior implementación del sistema en cuestión. Se realizó el modelamiento del negocio, dentro del mismo se realizó Modelo de Dominio. Se especificaron los requisitos funcionales y no funcionales del sistema con los cuales el sistema deberá cumplir, también se realizó la descripción del sistema, dentro de la misma se obtuvieron los Actores del Sistema, el Modelo de Casos de Uso del Sistema y la descripción textual de cada Caso de uso. El análisis fue otro punto abordado, dentro del mismo se realizaron los diagramas de clases del análisis y los diagramas de colaboración.

4

VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

En este capítulo se realiza la validación de algunos de los artefactos generados en el capítulo anterior mediante el uso de métricas, el uso de las métricas es necesario para valorar la calidad de los productos de la ingeniería que en este trabajo se desarrollaron. Después de la aplicación de las métricas se exponen los resultados y se hace un análisis de los mismos para medir la calidad del trabajo realizado.

4.1 Definiciones fundamentales de las métricas

La palabra métrica es frecuentemente relacionada con las palabras medida y medición, pero estas tienen un significado diferente, a continuación se exponen los mismos para llegar a una mejor comprensión del tema.

Medición: Es el proceso por el cual los números o símbolos son asignados a atributos o entidades en el mundo real tal como son descritos de acuerdo a reglas claramente definidas. (PRESSMAN 2002)

Medida: Proporciona una indicación cuantitativa de extensión, cantidad, dimensiones, capacidad y tamaño de algunos atributos de un proceso o producto. (PRESSMAN 2002)

Métrica: Es una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado. (PRESSMAN 2002)

Clasificación de las métricas

Son muchas las métricas existentes y también sus propósitos, entre los aspectos que pueden medir están la calidad, complejidad, desempeño y de productividad. Aunque estas no son las únicas y tampoco se utilizarán todas, a continuación se exponen algunas de ellas.

Métricas de calidad: Como su nombre lo indica estas métricas están dirigidas a medir la calidad del software; algunas variables que podemos encontrar son: reusabilidad, estructuración o modularidad, pruebas, mantenimiento, exactitud, entre otras.

Métricas de complejidad: Son utilizadas para medir la complejidad del software, entre las variables que podemos encontrar están: volumen, tamaño, anidaciones, estimación de costo y configuración.

Métricas de desempeño: Son las métricas que miden la conducta de módulos y sistemas de un software. Tienen que ver con la eficiencia de ejecución, tiempo, almacenamiento y complejidad de algoritmos computacionales.

Métricas de productividad: Se centran en el rendimiento del proceso de la ingeniería del software.

4.2 Métricas a utilizar en los artefactos generados

4.2.1 Aplicación de Métricas a la especificación de requisitos

Para realizar la validación de los requisitos existe toda una lista de características que sugieren que pueden emplearse para valorar la calidad del modelo de análisis y la correspondiente especificación de requisitos: especificidad (ausencia de ambigüedad), corrección, compleción, comprensión, capacidad de verificación, consistencia externa e interna, capacidad de logro, concisión, trazabilidad, capacidad de modificación, exactitud y capacidad de reutilización. (PRESSMAN, 2005)

Para determinar la especificidad (ausencia de ambigüedad) de los requisitos se propone una métrica que se basa en la consistencia de la interpretación de los revisores, para esto es necesario conocer primeramente:

Q: Especificidad

Nui: Número de requisitos para los que los revisores tuvieron interpretaciones idénticas.

Nr: Total de requisitos en el sistema.

Nf: Requisitos Funcionales

Nnf: Requisitos no Funcionales.

Teniendo que:

Nr= Nf + Nnf

Nr= 20 +20

Nr= 40

En la presente investigación se realizaron 2 revisiones, las cuales se tratan a continuación.

Revisión 1 (Q1): En la primera revisión se pudo detectar que de los 20 requisitos funcionales los revisores tuvieron interpretaciones idénticas en 18, por lo que se detectaron 2 requisitos que presentaban problemas, estos fueron “Crear Vista” y “Deseleccionar”, en los dos casos los revisores no sabían claramente su significado. Entre los 20 requisitos no funcionales no se detectaron ambigüedades por lo que de un total de 40 requisitos, los revisores tuvieron interpretaciones idénticas en 38.

Q1= Nui/Nr

Q1= 38/40

Q1= 0.95

Revisión 2 (Q2): En la segunda y última revisión se corrigieron las dificultades encontradas en la primera revisión, sustituyendo el nombre “Crear Proyecto” por “Crear Vista” y “Deseleccionar” por “Deseleccionar elementos seleccionados”, al realizarse la presente revisión los revisores coincidieron en la interpretación de los 40 requisitos.

Q2= Nui/Nr

Q2= 40/40

Q2= 1

4.2.2 Aplicación de Métricas al Grado de Validación de requisitos

Para el grado de validación de requisitos se utiliza la métrica siguiente:

V = Nc / (Nc + Nnv)

Donde:

V: Grado de Validación.

Nc: Número de requisitos que se han validado correctamente.

Nnv: Requisitos que no han sido validados.

Posterior a la aplicación de la métrica a la especificación de requisitos se concluye que se validaron todos los requisitos tanto los funcionales como los no funcionales, por lo que se obtiene el siguiente grado de validación:

V = 40 / 40 + 0

V = 1

4.2.3 Aplicación de Métricas a los Casos de Uso del Sistema

Para mejorar la calidad de los Casos de Uso del Sistema, se les aplicó a los mismos un modelo de métricas, el cual consta de 4 atributos referentes a la calidad (completitud, comprensibilidad, concisión, no trivialidad). A continuación se explican de forma breve estos atributos:

Completitud: Grado en que se ha logrado detallar los casos de uso relevantes.

Comprensibilidad: Un caso de uso es comprensible si todos los lectores pueden entenderlo fácilmente.

Concisión: Un caso de uso es conciso cuando el mismo no incluye información innecesaria.

No trivialidad: Un caso de uso es no trivial cuando su secuencia de pasos conduce al actor a conseguir el objetivo que persigue la realización del caso de uso.

El siguiente modelo presenta para cada uno de estos atributos una serie de preguntas definidas que deben ser respondidas para cada CU, para la correcta aplicación de esta métrica y asegurar la calidad de los CU se requiere que cuando se le dé respuestas a estas preguntas si alguna no tiene una respuesta positiva para un CU dado indica que se deben hacer ajustes y una nueva revisión.

Tabla 4.1 Aplicación de Métricas a los CUS

Métricas	Preguntas
Completitud	1. ¿Hay respuestas a todas las peticiones que el actor del caso de uso hace al sistema y viceversa? 2. ¿Se contemplan todos los posibles escenarios para poder alcanzar el objetivo del caso de uso? 3. ¿Se especifican todas las secuencias alternativas a la secuencia normal? 4. ¿Se contemplan todas las posibles excepciones a la secuencia normal?
Comprensibilidad	1. ¿Es posible leer el caso de uso sin volver atrás en repetidas ocasiones? 2. ¿Es difícil seguir la secuencia normal del caso de uso por la presencia de las relaciones inclusión o extensión? 3. ¿Es difícil seguir la secuencia de pasos por la existencia de demasiados pasos alternativos? 4. ¿Se han desglosado demasiado los pasos de algún actor o del sistema provocando que el caso de uso avance a un ritmo muy lento? 5. ¿Aparecen pasos condicionales para expresar que el sistema

	comprueba una situación que permite al caso de uso continuar su realización?
Concisión	1. ¿Podría el caso de uso ser expresado con menos palabras? 2. ¿Existen elementos que se pueden obviar o aparecen anotaciones innecesarias y que dificultan la lectura del caso de uso? 3. ¿Aparecen demasiadas interacciones entre el actor principal del caso de uso y otros elementos del entorno?
No trivialidad	1. ¿Expresa el nombre del caso de uso un objetivo de un usuario que el sistema debe implementar? 2. ¿Conduce el caso de uso al actor a conseguir alguno de sus objetivos sin representar un conjunto de interacciones triviales?

Revisión 1:

Completitud

1. ¿Hay respuestas a todas las peticiones que el actor del caso de uso hace al sistema y viceversa?

Luego de haber revisado cada caso de uso se pudo comprobar que existen respuestas a todas las peticiones que el actor del caso de uso hace al sistema y viceversa.

2. ¿Se contemplan todos los posibles escenarios para poder alcanzar el objetivo del caso de uso?

En los casos de uso: “Crear Proyecto “, “Salir”, “Gestionar Capas” específicamente en la sección “Agregar Capas”, “Medir distancias” y en “Localizar punto” no se exponen todas las posibles formas de iniciar el CU por ejemplo en el CU “Crear Proyecto” existen dos formas de iniciarlo, seleccionando la opción nuevo proyecto en el menú archivo o en la barra de herramientas el icono correspondiente a esta acción para crear una nueva vista, en este caso solo se hacía referencia al primer mencionado.

3. ¿Se especifican todas las secuencias alternativas a la secuencia normal?

Tras revisar todos los casos de uso se pudo detectar que en el caso de uso “Abrir Proyecto” no se tiene en cuenta la posibilidad de que el usuario presionara cancelar en el explorador o que seleccionara un archivo de un tipo incorrecto.

4. ¿Se contemplan todas las posibles excepciones a la secuencia normal?

En el caso de uso “Navegar direccionalmente” no se contempla el caso de que el usuario al introducir el factor de desplazamiento se equivoque introduciendo algún número demasiado pequeño, negativo o que contenga letras incluidas. En el caso de uso “Gestionar Capas” específicamente a la hora de agregar un archivo vectorial o raster no se tiene en cuenta que el usuario seleccione un archivo no válido.

Comprensibilidad

1. ¿Es posible leer el caso de uso sin volver atrás en repetidas ocasiones?

En el caso del caso de uso gestionar capas se detectó que para entenderlo, por su complejidad, era necesario leerlo con detenimiento.

2. ¿Es difícil seguir la secuencia normal del caso de uso por la presencia de las relaciones inclusión o extensión?

Tras la revisión de los CU se detectó que existe la relación extensión en el caso de uso “Visualizar Mapa” pero que a pesar de esto no se hace significativamente difícil seguir la secuencia normal del CU.

3. ¿Es difícil seguir la secuencia de pasos por la existencia de demasiados pasos alternativos?

Tras la revisión de todos los CU no se detectaron problemas de este tipo.

4. ¿Se han desglosado demasiado los pasos de algún actor o del sistema provocando que el caso de uso avance a un ritmo muy lento?

Después de revisar los CU se concluyó que estos están correctos en este sentido.

5. ¿Aparecen pasos condicionales para expresar que el sistema comprueba una situación que permite al caso de uso continuar su realización?

Si se encuentran situaciones de este tipo, por ejemplo en el caso de uso “Autenticar Usuario” cuando el sistema verifica que el usuario y contraseña sean correctos para dar acceso a la aplicación.

Concisión

1. ¿Podría el caso de uso ser expresado con menos palabras?

Con la revisión de los CU se pudo comprobar que en este sentido los CU están correctos.

2. ¿Existen elementos que se pueden obviar o aparecen anotaciones innecesarias y que dificultan la lectura del caso de uso?

El caso de uso “Redimensionar Mapa” específicamente en las secciones “Zoom previo” y “Zoom siguiente” se detectó que se aludían especificaciones no necesarias, referentes a la forma en que el sistema realiza estas funcionalidades, como por ejemplo se explica que el sistema guarda en un arreglo los Zoom realizados sobre la vista y utiliza el mismo para saber cuál es el siguiente y anterior Zoom.

3. ¿Aparecen demasiadas interacciones entre el actor principal del caso de uso y otros elementos del entorno?

Por la revisión realizada a los CU se pudo observar que no existen demasiadas interacciones entre el actor principal del caso de uso y otros elementos del entorno.

No trivialidad

1. ¿Expresa el nombre del caso de uso un objetivo de un usuario que el sistema debe implementar?

En todos los casos, los nombres del CU expresan un objetivo de un usuario que el sistema debe implementar.

2. ¿Conduce el caso de uso al actor a conseguir alguno de sus objetivos sin representar un conjunto de interacciones triviales?

Tras el análisis realizado a cada CU se concluyó que los CU cumplen con esta condición.

Revisión 2:

En esta revisión se corrigieron los errores encontrados en la primera revisión, al hacérseles un análisis a los cambios realizados no se encontraron más problemas y se logró la satisfacción de los revisores con las soluciones mostradas aunque se precisaron algunos detalles que fueron resueltos al instante por lo que no fue necesario una tercera revisión. Tras la aplicación de la métrica expuesta anteriormente a los casos de uso se pudo observar que estos cumplen positivamente, lo que representa un resultado satisfactorio del trabajo realizado.

4.3 Conclusiones Parciales

La necesidad de la validación de los artefactos generados en el capítulo anterior mediante el uso de métricas, ha conducido a la elaboración del capítulo presentado con anterioridad, el cual comprende un conjunto de definiciones fundamentales de las métricas que contribuyen a un mejor entendimiento de las mismas para su utilización. Se realizaron dos revisiones en cuanto a la especificación de los requisitos donde en la primera se detectaron dos no-conformidades por el equipo de revisores las cuales fueron solucionadas permitiendo que para la segunda revisión los revisores coincidieran en todas sus interpretaciones, lo que a su vez permitió contar con un grado de validación de requisitos de 1 lo que quiere decir que todos los requisitos fueron validados. También se les aplicó métricas a los CUS en la cual se realizaron dos revisiones, en la primera revisión se encontraron un conjunto amplio de problemas que para la segunda revisión fueron arreglados, lográndose la satisfacción de los revisores esta vez. En general se puede decir que luego de la aplicación de las métricas se encontraron algunas deficiencias que fueron posteriormente solucionadas por el proceso de revisión, logrando así, validar los artefactos obtenidos durante el desarrollo del presente trabajo.

Conclusiones generales

Con la terminación de este trabajo se puede arribar a las siguientes conclusiones:

- La Metodología de Desarrollo de Software, el Lenguaje de Modelado y la Herramienta CASE utilizados, permitieron la realización del proceso con éxito y obtener los resultados esperados de esta investigación: la documentación técnica prevista, correspondiente a la modelación del análisis del proceso de desarrollo de un SIG de tipo escritorio sobre la herramienta gvSIG.

- Mediante el uso de técnicas usadas para la obtención de requisitos como la entrevista y la lluvia de ideas, además del estudio realizado sobre SIG existentes, se pudieron identificar un total de 21 requerimientos funcionales y 20 requerimientos no funcionales, los cuales fueron traducidos en las funcionalidades que el sistema debe ofrecer.

- Las métricas aplicadas, para la validación de los requisitos y los casos de uso del sistema, arrojaron algunas deficiencias que posteriormente fueron corregidas para obtener finalmente un resultado satisfactorio que avala en buena medida la calidad de los artefactos generados.

- En este trabajo se ha logrado resolver el problema formulado al modelar las especificidades desde el punto de vista funcional de un producto de software de tipo SIG sobre la herramienta gvSIG, utilizando las técnicas y herramientas de la Ingeniería de Software; llevando las necesidades de los clientes a un lenguaje más cercano a los desarrolladores que facilitará la consecuente etapa de construcción de dicho producto de notable valor para el buen desempeño del proyecto GENESIG en el área de los SIG de tipo escritorio.

Recomendaciones

Basándose en el trabajo realizado y en los resultados obtenidos en el mismo, el autor de esta investigación recomienda:

Diseñar e implementar una aplicación de escritorio SIG sobre gvSIG, tomando como guía y soporte el análisis realizado en este trabajo de diploma.

Tomar como aplicación base a dicha implementación, para que en futuras peticiones de clientes de aplicaciones SIG sobre gvSIG contar con un punto de partida mejor y así poder dar respuestas a estas solicitudes en menor tiempo y con mejor calidad.

Realizar un correcto refinamiento de requisitos con los futuros clientes con el fin de obtener las funcionalidades a agregar a la aplicación base para lograr personalizar la misma y lograr mayor satisfacción en los clientes.

Referencias Bibliográficas

CONSELLERIA DE INFRAESTRUCTURAS Y TRANSPORTE. Curso de gvS IG 1.1.2. 2008.

BERNI, J. A. J.; UREÑA, M. J. A., et al. Alternativas de software libre a los sistemas de información geográfica comerciales. 2005. [Citado: 24 de noviembre de 2009]. Disponible en: <http://www.cartesia.org/geodoc/ingegraf2005/gis10.pdf>

DÍAZ-ANTÓN, M. G.; M. A. PÉREZ, et al. Propuesta de una metodología de desarrollo de software educativo bajo un enfoque de calidad sistémica, Simón Bolívar, 2004. [Citado: 12 de diciembre de 2009]. Disponible en: <http://www.academia-interactiva.com/ise.pdf>

GRUPO GVSIG. Sitio Oficial de gvSIG. 2010. [Citado: 18 de enero de 2010]. Disponible en: <http://www.gvsig.org/web/projects/gvsig-desktop/funcionalidades>

JACOBSON, I.; G. BOOCH, et al. El Proceso Unificado de Desarrollo de Software. Addison - Wesley (Edición en español por la Pearson Educación S.A. traducido de The Unified Software Development Process, 1999). Madrid, 2000.

LETELIER, P. and E. A. SÁNCHEZ Metodologías Ágiles en el Desarrollo de Software, 2003. [Citado: 15 de diciembre de 2009]. Disponible en: <http://issi.dsic.upv.es/tallerma>

MARGALEF, J. G. y POLO, C. P. Estudio comparativo de herramientas SIG Libres aplicadas a contextos de cooperación al desarrollo. 2007. [Citado: 5 de diciembre de 2009]. Disponible en: <http://www.sigte.udg.es/jornadassiglibre/uploads/file/Comunicaciones_2/8.pdf>

MARTÍNEZ, E. A. Metodología. Hágase la luz. 2005. Disponible en: <http://www.crisol.cc/eamf-tec/metodologias/Metodologias.pps>

NCGIA. National Center for Geographic Information and Analysis. 1990. Disponible en: <http://www.ncgia.ucsb.edu>

PIATTINI, M. Análisis y diseño detallado de aplicaciones informáticas de gestión. 1996.

PRESSMAN, R. S. Ingeniería del Software. Un enfoque practico. Quinta edición Mc Graw-Hill/Interamericana de España, S.A. 2002.

RODRÍGUEZ, N.; SANABRIA., R. D. M., et al. Los sistemas de información geográfica. [Citado: 21 de noviembre de 2009]. Disponible en: <http://www.humboldt.org.co/humboldt/mostrarpagina.php?codpage=70001#top>

VALENCIA, D. D. S. I. Y. C. U. D. Proceso de desarrollo de software.

YAGÜEZ, J. C. D. y LANGHI, R. 2002. Sistema de Información Geográfica (S.I.G). 2002. [Citado: 28 de diciembre de 2009]. Disponible en: <http://www.inta.gov.ar/barrow/info/documentos/SIG/que_es_sig.htm>

Bibliografía

ACEBAL, C. F. y LOVELLE, J. M. C. eXtreme Programming (XP): un nuevo método de desarrollo de software. 2002. Disponible en: <http://www.ati.es/novatica/2002/156/156-8.pdf>

CANADA, N. R. Sistemas de Información Geográfica. Disponible en: <http://www.nrcan.gc.ca/ess>

CANÓS, J. H.; LETELIER, P., et al. Métodologías Ágiles en el Desarrollo de Software. 2008. Disponible en: <http://www.willydev.net/descargas/prev/TodoAgil.pdf>

CARBONELL, J. I. S. ¿Qué es un SIG? Disponible en: <http://www.nosolosig.com/%BFque_es_un_sig?.html>

CASTAÑEDA, K. Sistema de Gestión de Datos Geológicos. Módulo: Inventario Nacional de Recursos de Aguas Minerales. Rol de Analista. Tesis de pregrado, Universidad de las Ciencias Informáticas, 2009.

DÍAZ-ANTÓN, M. G.; M. A. PÉREZ, et al. Propuesta de una metodología de desarrollo de software educativo bajo un enfoque de calidad sistémica, Simón Bolívar, 2004. Disponible en: <http://www.academia-interactiva.com/ise.pdf>

ECHEVERRIA, X. V. Qué es un SIG. 2007. Disponible en: <http://www.navactiva.com/web/es/descargas/pdf/atic/sig1.pdf>

FERNÁNDEZ, Y. Diseño de la plataforma soberana para el desarrollo de Sistemas de Información Geográfica del Polo Geoinformática. Tesis de pregrado, Universidad de las Ciencias Informáticas, 2009.

GRUPO GVSIG. Sitio Oficial de gvSIG. [En línea] Disponible en: < http://www.gvsig.org/web/>

HERNÁNDEZ, M. E. Diseño del perfil de competencia para el rol de Analista. Tesis de pregrado, Universidad de las Ciencias Informáticas, 2009.

JACOBSON, I.; G. BOOCH, et al. El Proceso Unificado de Desarrollo de Software. Addison - Wesley (Edición en español por la Pearson Educación S.A. traducido de The Unified Software Development Process, 1999). Madrid, 2000.

LEO, N. D. ¿Que es un SIG (GIS)? [Consultado el: 16 de diciembre de 2009]. Disponible en: <http://www.fcagr.unr.edu.ar/mdt/GTS/Zonaedu/GIS7htm.htm>

LETELIER, P. Metodologías Ágiles en el Desarrollo de Software. Disponible en: <http://issi.dsic.upv.es/tallerma>

MARGALEF, J. G. y POLO, C. P. Estudio comparativo de herramientas SIG Libres aplicadas a contextos de cooperación al desarrollo. 2007. Disponible en: <http://www.sigte.udg.es/jornadassiglibre/uploads/file/Comunicaciones_2/8.pdf>

MOLPECERES, A. Procesos de desarrollo: RUP, XP y FDD. 2002. Disponible en: <http://www.willydev.net/InsiteCreation/v1.0/descargas/Articulos/General/cualxpfddrup.PDF>

MONSALVE, J. J. y CARMONA, A. D. J. Sistemas de Información Geográfica. Disponible en: <http://www.monografias.com/trabajos/gis/gis.shtml>

MONTES, D., RODRÍGUEZ, A. R. Manual de Analista. Tesis de pregrado, Universidad de las Ciencias Informáticas, 2009.

MUÑOZ, A., AGUILAR, N. y RAMIREZ, E. Kosmo-Plataforma SIG Libre Corporativa. 2007. [en línea] [Consultado el: 7 de diciembre de 2009]. Disponible en: <http://www.sigte.udg.es/jornadassiglibre>

PAZ, J. D. Desarrollo en comunidad con eXtreme Programming. 2005. Disponible en: <http://2005.guadec-es.org/download/articulos/Articulo%2005%20-%20Jose%20Dapena%20-%20Metodos%20y%20herramientas%20de%20desarrollo%20en%20comunidad.pdf>

RATIONAL SOFTWARE. RUP. 2007. Disponible en: <http://es.wikipedia.org/wiki/Rational_Unified_Process>

RODRÍGUEZ, H. C. Guía metodológica para el desarrollo de Sistemas de Información Geográfica en la Universidad de las Ciencias Informáticas. Ciudad de la Habana: s.n., 2008.

TAVARES, D. Análisis y diseño del sistema de manejo integral de perforación de pozos. Tesis de pregrado, Universidad de las Ciencias Informáticas, 2009.

Glosario

CAD: Diseño Asistido por Computadora, consiste en el uso de un amplio rango de herramientas computacionales que asisten a ingenieros, arquitectos y a otros profesionales del diseño en sus respectivas actividades.

Framework: Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

GPL: General Public License. Es una licencia que protege la creación y distribución de software libre. Está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

Hardware: Elementos físicos de un sistema informático como son la pantalla, el teclado, el ratón y otros.

Línea: Es una sucesión continua de puntos interminables e infinitos.

OGC: Open Geospatial Consortium. Entidad dedicada a la creación de estándares entre las diferentes empresas del sector que posibiliten la interoperación de sus sistemas de geoprocesamiento y facilitar el intercambio de la información geográfica en beneficio de los usuarios.

Plugins: Se trata de un programa que proporciona alguna funcionalidad específica a otra aplicación. Un complemento que se relaciona con otra aplicación para aportarle una función nueva, esta aplicación adicional es ejecutada por la aplicación principal e interactúan por medio de la API.

Polígono: Un polígono es una figura geométrica conformada por segmentos consecutivos no alineados, llamados lados.

Punto: Describe una posición en el espacio, determinada respecto de un sistema de coordenadas preestablecido.

Requisito (requerimiento): Circunstancia o condición necesaria para algo.

Software: Los procedimientos y reglas lógicas escritas en la forma de programas y aplicaciones, que definen el modo de operación de la computadora. Tienen carácter virtual y están almacenadas en los diferentes tipos de memoria de lectura/escritura.

WCS: Web Coverage Service. Es un servicio estándar definido por el OGC, es una especificación tecnológica que permite obtener capas raster en formatos SIG originales desde un servidor especifico compatible con WCS. Generalmente estas capas SIG son archivos muy voluminosos y el usuario sólo desea una parte de los mismos.

WFS: Web Feature Service. Es un servicio estándar definido por el OGC que ofrece una interfaz de comunicación que permite interactuar con los mapas como por ejemplo editar la imagen o analizar la imagen siguiendo criterios geográficos.

WMS: Web Map Service. Es un servicio definido por el OGC que produce mapas de datos referenciados espacialmente, de forma dinámica a partir de información geográfica. Este estándar internacional define un "mapa" como una representación de la información geográfica en forma de un archivo de imagen digital conveniente para la exhibición en una pantalla de ordenador.