Universidad de las Ciencias Informáticas
Facultad 6

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Titulo: Desarrollo de un sistema informático para editar las configuraciones básicas de ficheros de mapas

Autor: Oscar Matos Torres

Tutor: Ing. Yenier Jimenez Morales

Junio 2011, La Habana, Cuba.

Pensamiento

Estar preparado es importante, saber esperarlo es aún más, pero aprovechar el momento adecuado es la clave de la vida.

Arthur Schnitzler.

Declaración de autoría

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

Para que así conste firmamos la presente a los 23 días del mes de Junio del 2011.

_____	_____
Autor:	Tutor:
Oscar Matos Torres.	Ing. Yenier Jimenez Morales

Síntesis del tutor

Ing. Yenier Jiménez Morales

Profesor graduado en el año 2010 de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Profesor de la Facultad No.6. Actualmente imparte la asignatura optativa de Administración de GNU/Linux. Se desempeña como desarrollador de software en el proyecto Aplicativos SIG del centro productivo GEYSED perteneciente a la Facultad No.6.

Correo electrónico: yjmorales@uci.cu

Agradecimientos

Agradezco en este trabajo a las personas más importantes en mi vida que son mi padre y mi madre, este trabajo va dedicado a ustedes por el amor que me han dado y me han enseñado a brindar. A mis abuelos por su cariño, a mi hermana por inspirarme alegría y juventud. A mis tíos y primos por ser un gran apoyo y por quererme tanto. A mi novia Yanisleidy por ser mi apoyo en los momentos difíciles, por ser mi mano derecha y por su gran amor demostrado. A Orestes y a Carmen por tenerme como otro hijo y en general a toda su familia; a mi tutor Yenier por ser más que tutor, un gran amigo; a mis amigos, compañeros de grupo y a los que me han dado su amistad y cariño.

Dedicatoria

En este trabajo han sido muchas las personas que han estado apoyándome. En primer lugar le dedico este trabajo a mis padres por ser mis guías y lugar de consuelo en todo momento, a mi hermana, a mis abuelos, a mis tíos y primos, a mi novia y su familia, a mi tutor por apoyarme en todo momento, a mis amigos y compañeros de estudio y a todo el que una forma u otra ha colaborado en la realización de este trabajo.

Resumen

El georeferenciado de objetos a través de los años ha evolucionado desde simples mapas sobre papel hasta sistemas automatizados capaces de procesar considerables cantidades de datos espaciales en un corto tiempo. Producto del resultado de esta evolución surgen los Sistemas de Información Geográfica (SIG) como sistemas integrales capaces de gestionar cualquier tipo de información y ubicarla en mapas digitales. Existen SIG enfocados como aplicaciones web accesibles desde cualquier localización. Un recurso importante que este tipo de SIG explota es utilizar MapServer como servidor de mapas.

Como un componente fundamental el procesamiento de información dentro de MapServer se encuentra el fichero de mapa. El mismo define las relaciones entre objetos, puntos donde los datos espaciales serán localizados y define como serán dibujados los objetos en el mapa. El trabajar con MapServer requiere que en algún momento este tipo de fichero sea configurado para adaptarse a las necesidades propias de quien lo utilice. El presente trabajo de diploma tiene como objetivo desarrollar un sistema informático capaz de editar las configuraciones básicas de un fichero de mapa para MapServer.

Fueron analizados elementos teóricos relacionados con la tecnología mencionada, caracterizadas las herramientas y tecnologías empleadas en la construcción del sistema, definida la arquitectura del sistema construido y probado el resultado del proceso desarrollo.

Índice

Índice de Figuras

Índice de Tablas

Introducción

El hombre a través de la historia ha tenido necesidad de orientarse en el terreno. A medida que iba descubriendo nuevas tierras debía ubicarlas en mapas rústicos. Estos eran imprecisos y solo eran usados por muy pocas personas. Los primeros mapas que existieron fueron tallados en tablillas de arcilla y consistían en su mayor parte en mediciones de terrenos realizadas con el fin de cobrar impuestos. Posteriormente con el desarrollo del papel se fueron dibujando y mejorando su comprensión y exactitud.

Los primeros mapas dibujados sobre papel tenían forma circular y mostraban el mundo conocido hasta el momento agrupado en torno al mar Egeo y rodeado por el océano. Paralelamente al descubrimiento de nuevas tierras fueron ampliándose y para su mejor compresión se dibujaron en forma rectangular.

La utilización de esta forma de representación geográfica ha ido en escala creciente a través del tiempo. Con el desarrollo de la informática se logró representar mapas de forma digital y sobre los mismos representar todo tipo de información útil al hombre. Surgen pues los Sistemas de Información Geográfica (SIG).

Un SIG funciona como una base de datos con información geográfica[1] que se encuentra asociada por un identificador común a objetos gráficos de un mapa digital. De esta forma, señalando un objeto se conocen sus atributos e inversamente, preguntando por un registro de la base de datos se puede saber su localización en la cartografía (LOS SISTEMAS DE INFORMACIÓN, 1998). Constituyen una abundante fuente de información capaz de brindar todo tipo de información geográficamente referenciada.

Dentro de la estructura de los Sistemas de Información Geográfica existe un componente fundamental encargado de la representación final de los mapas: estos son los ficheros de mapas[2]. El fichero de mapa es un archivo de texto, con extensión .Map, en el que se incluye una serie de parámetros que definen las capas disponibles en el servicio, el estilo con que se representarán, su simbología y formato en que se generará la imagen (Ballari, 2008). Este archivo es el componente principal del

[1] Datos alfanuméricos
[2] Usualmente nombrado MapFile por sus siglas en inglés

servidor de mapas *MapServer*[3], constituye su archivo principal de configuración donde se incluye una serie de parámetros que definen las capas disponibles.

Este fichero consta de varias secciones. Cada sección se inicia con el nombre de la sección y termina con la palabra END. El contenido de las secciones consiste en la definición de determinados parámetros del tipo atributo - valor.

La edición de ficheros de mapas es brindada por varios software en el mercado a nivel mundial. No son de manera general sistemas completos de edición de este tipo de ficheros sino que se especializan en la edición de algunas partes específicas de los mismos. Los sistemas más utilizados en la edición de ficheros de mapas son el GIFfyCutter, desarrollado por la empresa FCODERSOFT, CyD Image Map desarrollado por la compañía CYDSOFT, CmapTool desarrollado por el Instituto de Cognición Humana y de la Computación, perteneciente a la Universidad de West Florida, Estados Unidos.

Libermap es un software cubano desarrollado en la Universidad de las Ciencias Informáticas (UCI[4]). Mejora el proceso de edición de ficheros de mapas utilizados en los SIG. Cuenta con una mayor cantidad de funcionalidades respecto a los analizados anteriormente, permitiendo una edición completa y profunda de un fichero de mapa. Fue desarrollado para que se desplegara en un entorno web por lo que no se encuentra disponible para usuarios sin conexión a una red informática. El Centro Productivo Geoinformática y Señales Digitales (GEySED), perteneciente a la Universidad de las Ciencias Informáticas, cuenta con un grupo de desarrolladores dedicados a la confección de Sistemas de Información Geográfica.

Con la diversificación del desarrollo de estos sistemas en el centro se tuvo la necesidad de que los archivos de mapas existentes pudieran ser modificados para su adaptación a nuevas necesidades. La edición de ficheros de mapas que se realiza en GEySED generalmente no se puede clasificar como profunda, sino que solo se cambian pocas configuraciones por lo que surge la necesidad de contar con una aplicación informática ligera encargada de la edición de configuraciones básicas de estos ficheros. Es necesario que este esa herramienta se ejecute sobre entornos donde no existan conexiones entre las estaciones de trabajo. Se pretende desarrollar una herramienta de escritorio que posibilite de forma rápida e intuitiva el proceso de

[3] Aplicación Common Gateway Interface(CGI) para construir aplicaciones que sirvan mapas a través de Internet
[4] Universidad cubana dedicada al desarrollo de la Informática.

edición de estos ficheros de mapas para dar solución a las crecientes demandas de los proyectos del Centro Productivo GEySED.

Ante la situación problemática anterior se define el siguiente **problema a resolver**:

¿Cómo agilizar la edición de las configuraciones básicas de un fichero de mapa utilizado por MapServer en los Sistemas de Información Geográfica del Centro Productivo GEySED bajo cualquier entorno de trabajo?

Se persigue como **objetivo general** desarrollar un sistema informático que incremente la agilidad del proceso de edición de las configuraciones básicas de los ficheros de mapas utilizados en MapServer.

En la presente investigación el **objeto de estudio** abarcado es el proceso de representación de información geográfica.

Se definió como **campo de acción** la edición de ficheros de mapas.

Con lo anteriormente definido se tiene como **idea a defender:** Si se desarrolla un sistema informático que permita la edición de las configuraciones básicas de ficheros de mapas se incrementará la agilidad del proceso.

Resultado a obtener: Se espera la construcción de un sistema informático que automatice la edición básica de ficheros de mapas acorde a las necesidades de los Sistemas de Información Geográfica desarrollados en el Centro Productivo GEySED.

Los **métodos científicos de la investigación** utilizados fueron:

Teóricos:

- **Analítico – Sintético:** Se realizará un análisis detallado de varios documentos, herramientas y bibliografías, permitiendo la comprensión de los elementos esenciales en el proceso de edición de ficheros de mapas.

- **Inductivo – Deductivo:** Se realizará un estudio detallado a las diferentes bibliografías que abarcan el tema relacionado con el proceso de edición de ficheros de mapas, lo cual permitirá llegar a una conclusión que contribuirá positivamente a la hora de determinar la herramienta de modelado, el lenguaje de programación, la metodología, el IDE, entre otros, para llevar a cabo el desarrollo del sistema.

- **Análisis Histórico – Lógico:** Este método posibilitará el estudio del arte de los procesos de edición de ficheros de mapas.

- **Modelación:** Este método permitirá mostrar mediante objetos, ilustraciones, tablas, diagramas u otros procedimientos, los procesos en el desarrollo de la aplicación de edición de ficheros de mapas.

Resultados esperados:

- ✓ Desarrollo de un sistema informático de edición de ficheros de mapas para Sistemas de Información Geográfica que utilicen MapServer.
- ✓ Un informe detallado con la documentación y la ingeniería utilizada en el desarrollo de la aplicación de edición de ficheros de mapas.

Para el desarrollo del trabajo de diploma fueron llevadas a cabo las siguientes tareas investigativas:

1. Caracterizar la estructura de los ficheros de mapas utilizados por los Sistemas de Información Geográfica que utilicen MapServer.
2. Caracterizar los sistemas informáticos de edición de ficheros de mapas desarrollados.
3. Fundamentar el uso de las tecnologías a utilizar para el desarrollo del sistema a desarrollar.
4. Identificar las funcionalidades que debe brindar el sistema.
5. Diseñar el sistema.
6. Implementar las funcionalidades del sistema.

El presente trabajo se encuentra estructurado en cuatro capítulos:

Capítulo 1. Fundamentación teórica: Son expuestos los principales elementos teóricos necesarios para la elaboración de la investigación. Son identificadas y debidamente justificadas las herramientas y tecnologías a utilizar para el desarrollo del sistema.

Capítulo 2. Características del sistema: Es descrito el dominio del problema tratado, se describe la solución propuesta basada en sus requisitos funcionales y no funcionales, son identificados los casos de uso.

Capítulo 3. Construcción del sistema: Describe la solución propuesta, basada en los requisitos identificados en el capítulo 2, centrándose en el diseño del sistema y refinándose la arquitectura del mismo.

Capítulo 4. Implementación y prueba: Incluye la implementación y pruebas del sistema informático. Es descrita la solución informática en términos de componentes de software. Conjuntamente son definidas las pruebas a las que será sometido.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

En la actualidad con el desarrollo de los Sistemas de Información Geográfica muchas disciplinas se han beneficiado. La forma en que han sido desarrollados ha evolucionado considerablemente. Un componente importante de este tipo de sistemas es el fichero de mapa.

En este capítulo se explicarán conceptos y definiciones relacionados con los ficheros de mapas, su proceso de edición y se caracterizarán las principales herramientas que editan ficheros de mapas.

1.2 Conceptos asociados al dominio del problema

Para lograr una mejor comprensión de algunos términos que se abordarán en la investigación se hace necesario fundamentar algunos elementos teóricos relacionados a los mismos.

SIG:

Variadas son las definiciones dadas de los Sistemas de Información Geográfica. Entre ellos se presentan los siguientes:

Un SIG se define como un conjunto de métodos, herramientas y datos que están diseñados para actuar, coordinar, capturar, almacenar, analizar, transformar y presentar todo tipo de información geográfica y sus atributos con el fin de satisfacer múltiples propósitos. Son una tecnología que permite gestionar y analizar la información espacial, y que surgió como resultado de la necesidad de disponer rápidamente de información para resolver problemas y contestar a preguntas de modo inmediato (LOS SISTEMAS DE INFORMACIÓN, 1998).

“Un Sistema de Información Geográfica 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, 2002).

Existen muchas definiciones de lo que es un SIG, una de las más aceptada por especialistas es la definida por el *National Center for Geographic Information and Analysis de EUA*., que plantea a los SIG como "Un sistema compuesto por elementos informáticos (hardware y software) y métodos diseñados para permitir la adquisición, gestión, manipulación, análisis, modelado, representación y salida de datos espacialmente referenciados, para resolver problemas complejos de planificación y gestión" (LINDE, 2000).

Con los SIG es posible capturar, almacenar, extraer, analizar y visualizar datos espaciales que enfatizan las relaciones espaciales entre los objetos que se están manejando. Estos pueden identificar una carretera como la frontera entre las zonas naturales y las zonas urbanas o como la intersección de dos calles, en vez de representarla sólo como una línea.

Fichero

Un fichero es una unidad lógica de almacenamiento que define el propio sistema operativo y cuyo significado está definido por su creador. Los ficheros están constituidos a nivel interno por un conjunto de registros lógicos.

Los ficheros o archivos se identifican por su nombre y su extensión. Según qué sistema operativo se utilice, podemos introducir un mayor o menor número de caracteres a este nombre. Por ejemplo MSDOS, solo permite 8 caracteres para el nombre, y en Windows hay ficheros con nombres de más de 8 caracteres (Rivera, 2008).

En informática, un archivo o fichero también es un conjunto de información que se almacena en forma virtual para ser leído y/o accedido por medio de una computadora. Las posibilidades de almacenamiento y clasificación son mucho más ricas en un sistema informático, ya que la información no ocupa un espacio físico y, por ende, es posible conservar millones de datos en un dispositivo muy pequeño. Inclusive, se puede guardar información de texto, audio o video en un mismo lugar sin inconveniente alguno.

A su vez, el sistema suele organizar taxonómicamente la información en forma automática, permitiendo que el usuario la encuentre con sólo ingresar palabras claves en un buscador específico, lo cual supone una operación rápida y útil cuando la

información almacenada es múltiple. A su vez, los sistemas informáticos suelen replicar los ficheros físicos y, así, organizar el contenido en carpetas y subcarpetas creadas y administradas por el usuario que son ubicadas en el disco interno y que pueden ser abiertas mediante accesos directos dispuestos en el escritorio virtual del ordenador (2009).

Edición de fichero

La edición de fichero es un proceso permanente en la computación. Al editar un fichero se pueden hacer diversas operaciones sobre el mismo, las más importantes son creación, apertura, cierre, borrado y extensión o modificación.

Creación de un fichero

El objetivo de esta operación es permitir a los usuarios la creación de nuevos ficheros. Se indican sus propiedades y características para que el sistema de ficheros pueda reconocerlo y procesarlo. En el proceso de creación del fichero debe registrarse la información necesaria para que el sistema pueda localizarlo y manipular sus registros lógicos. Para ello, el método de acceso debe obtener información sobre el formato y el tamaño de los registros lógicos y físicos, la identificación del fichero, la fecha de creación, su posible tamaño, su organización y aspectos de seguridad.

Apertura de un fichero

En esta operación el método de acceso localiza e identifica un fichero existente para que los usuarios o el propio sistema operativo puedan operar con él.

Cierre de un fichero

Mediante esta operación el método de acceso se encarga de "romper" la conexión entre el programa de usuario y el fichero, garantizando la integridad de los registros. Al ejecutar esta operación, el sistema se encarga de escribir en el dispositivo de almacenamiento aquella información que contienen los búfer asociados al fichero y se llevan a cabo las operaciones de limpieza necesarias (IES Alyanub, 2011). Una vez que es cerrado el fichero sus atributos no pueden ser accedidos por el método de acceso.

Borrado de un fichero

Esta operación elimina un fichero del directorio o tabla de contenidos correspondiente. El lenguaje de comandos del sistema operativo dispone de un comando para eliminar el identificador del fichero de la tabla de contenidos.

Extensión del fichero

Esta operación permite a los programas de usuario aumentar el tamaño de un fichero asignándole más espacio en el dispositivo de almacenamiento. Para realizar esta operación el método de acceso necesita conocer el identificador del fichero y el tamaño del espacio adicional que se debe asignar al fichero. En función de la organización del fichero, el método de acceso determinará si el espacio adicional que debe asignar debe ser contiguo al fichero o no. Mediante esta operación el atributo que indica el tamaño del fichero será modificado y se devolverá al programa de usuario con un código de estado. El único motivo para que esta operación no se lleve a cabo con éxito es que no haya suficiente espacio disponible en el lugar adecuado (Rivera, 2008).

Mapa

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

Iniciados con el propósito de conocer su mundo, y apoyados primeramente sobre teorías filosóficas, los mapas constituyen hoy una fuente importantísima de información, y una gran parte de la actividad humana está relacionada de una u otra forma con la cartografía (2009).

1.3 Objeto de Estudio

Para el desarrollo de la investigación se definió como objeto de estudio el proceso de representación de información geográfica. Para su mejor comprensión es preciso profundizar en el estudio de los ficheros de mapas.

1.3.1 *Descripción del fichero de mapa*

El fichero de mapa es una de las partes fundamentales del servidor de mapas MapServer[5]. MapServer es una plataforma de código abierto para la publicación de datos espaciales y aplicaciones de cartografía interactivas para la web. Permite la creación de Sistemas de Información Geográfica sobre entornos web, con el fin de visualizar, consultar y analizar información geográfica. La función de un fichero de mapa dentro de MapServer es contener la información necesaria para que el servidor de mapas procese los datos geográficos. Entre las funciones está indicar cómo se dibujará el mapa.

Este fichero presenta objetos con sus respectivas propiedades. Guarda todas las partes de un mapa, cada una por separada como un objeto, esta estructura permite un trabajo personalizado editando así su color, borde, ancho, alto y cambio de unidades.

Un fichero de mapa utilizado por MapServer quedaría estructurado como muestra la figura 1.

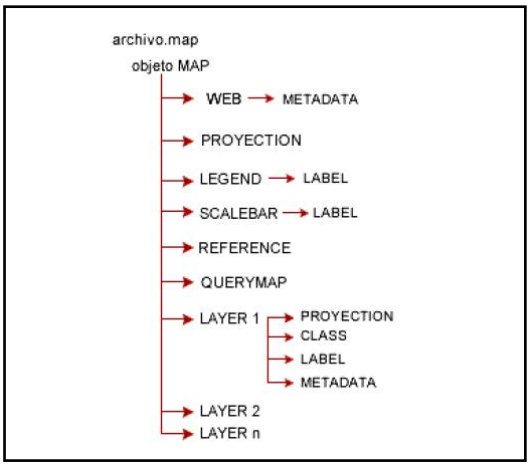

Fig. 1 Jerarquía de objetos en el fichero de mapas

La sección principal de un archivo de mapa es el objeto .Map, la cual anida a otras secciones.

Dentro de cada sección se definirá una serie de parámetros, algunos de los cuales son de obligatoria inclusión, mientras que otros son opcionales o tienen un valor asignado por defecto.

5

```
#
# Comienzo del  map file
#
MAP
  NAME larioja
  STATUS ON
  SIZE 400 350
  EXTENT 491642.4689 4641614.4994 609026.0001 4720900.9994
  UNITS METERS
  SHAPEPATH "data"
  IMAGECOLOR 255 255 255
  FONTSET "./font/fonts.txt"
  IMAGETYPE png
  ....
  ....
  ....
END
# fin del objeto map
```

Fig. 1 Ejemplo Parámetros de las secciones

Tabla 1. Parámetros de las secciones

Nombre del Parámetro	Descripción del parámetro
NAME	Nombre del archivo .Map
STATUS	on/off Establece si el mapa está activo o no. Puede existir interés solo en generar la escala gráfica y leyenda y no el mapa
SIZE	Ancho y alto en píxeles de la imagen de salida
EXTENT	[Xmin] [ymin] [xmax] [ymax] Extensión espacial del mapa a crear, en el sistema de referencia especificado en la sección PROJECTION
UNITS	[feet | inches | kilometers | meters | miles | dd]. Unidades de las coordenadas del mapa, usado para el cómputo de la escala gráfica y escala numérica. Debe estar definido en el sistema de referencia especificado en la sección PROJECTION.
SHAPEPATH	Nombre del directorio donde se almacenan los datos geográficos
IMAGECOLOR	[R] [G] [B]
FONTSET	Nombre completo del archivo y directorio que contiene el conjunto de fuentes
IMAGETYPE	[gif | png | jpeg | wbmp | gtiff | swf | userdefined]. Formato de salida

1.4 Sistemas informáticos que editan ficheros de mapas

La edición de ficheros de mapas es brindada por varios software en el mercado mundial. No son de manera general sistemas completos de edición de este tipo de formatos sino que se especializan en algunas de ellas

El software GIFfyCutter, desarrollado por la empresa FCODERSOFT, está montado bajo la plataforma Windows. Divide las imágenes de gran tamaño en pequeños fragmentos esto hace que los mapas que son muy grandes se puedan trabajar mejor.

La imagen se puede separar de acuerdo a las necesidades particulares de los usuarios, tanto vertical como horizontalmente, en tantos fragmentos como sea necesario, sin necesidad de realizar complicados cálculos previos ya que el programa lo hace todo de forma automática.

Tiene como limitante que es un software privativo teniendo que pagar para su uso y actualización sistemática.

CyD Image Map es un software en idioma inglés creado por la compañía CYDSOFT el cual funciona sobre Windows.

El programa ayuda a armar mapas de manera sencilla y genera todo el código fuente necesario a insertar en un SIG bajo un entorno web.

Crea fácilmente mapas de imágenes. Tiene como desventaja que solo genera el código fuente necesario para armar un mapa en una web. Es un software clasificado como una versión de prueba, la versión shareware permite un máximo de 30 ejecuciones por lo que no es recomendable su uso (Ficha técnica de CyD Image Map, 2008).

CmapTool es un software en idioma inglés clasificado como una herramienta multiplataforma capaz de funcionar bajo cualquier sistema operativo. Permite tanto el trabajo local individual, como en red, ya sea una red local, o en Internet, con lo que facilita el trabajo en grupo o colaborativo. Esta herramienta posibilita la navegación por los mapas realizados, lo que los convierte en interactivos. Se pueden enlazar e indexar prácticamente todo tipo de archivos (páginas web, imágenes, videos, sonidos, textos), con la posibilidad de añadir información contextual a cada uno de los conceptos o nodos del mapa (CMAPTOOL COMO HERRAMIENTA DE APRENDIZAJE VISUAL, 2007).

Tiene como desventaja que aunque sea un software libre no reconoce trabajos hechos en otros programas similares, el mismo guarda sus proyectos en formatos de imagen por lo que no permite otra forma hasta el momento, su documentación es poca y no pertenece a una compañía comercial.

AutoCAD MAP es la solución Autodesk 6 de Cartografía, basada en AutoCAD y en el módulo de gestión de información ADE 2.0, para la creación de mapas que permite digitalizar, mantener y trazar de manera precisa mapas más eficientemente que cualquier otro Sistema de Información Geográfica y creación de mapas o de CAD. AutoCAD MAP está dirigido a usuarios que deban realizar y mantener sus propios mapas. AutoCAD MAP ofrece funciones de digitalización e importación de datos desde otros sistemas (MapInfo, ArcInfo), funciones de limpieza, construcción, y topología 2D completa (Manual de Autodesk Map 5, 2007).

Por ser tan versátil necesita a menudo de PLUG-INS o aplicaciones específicas para cada especialidad. El mismo no permite trabajar con los archivos .Map y el valor del costo del paquete original y su licencia son elevados.

Libermap es un software nacional desarrollado en la Universidad de las Ciencias Informáticas (UCI). Mejora el proceso de edición de ficheros de mapas utilizados en los SIG. Cuenta con mayor cantidad de funcionalidades respecto a los analizados anteriormente, siendo capaz de editar un mapa completo. Fue desarrollado como un entorno web por lo que no se encuentra disponible para usuarios sin conexión a una red informática.

Herramientas y tecnologías a emplear

1.5 Aplicación de escritorio

Uno de los requisitos indispensables que debe contar el sistema a construir debe ser el ejecutarse en entornos de trabajo donde no exista una red. La velocidad de ejecución del sistema no se vería afectada por cargas en la red sino por el estado de la estación de trabajo donde se tenga instalada.

1.6 Lenguajes de Programación

CSharp

[6] Escritorio Automático

CSharp es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET. Es un lenguaje de programación que permite el desarrollo de aplicaciones para Internet, para móviles y aplicaciones de propósito general.

En el existen muchas clases implementadas en .NET, y están listas para permitirnos crear la gran mayoría de objetos que podemos llegar a necesitar en un programa de consola, tipo Windows o tipo web. Por ser un lenguaje con una plataforma privada y en la universidad se está migrando a software libre el mismo no se adoptará para la elaboración de esta solución informática (O@SIS, 2008).

C++

C++ es un lenguaje imperativo orientado a objetos, versión de C y se ha popularizado porque combina la programación orientada a objeto. Su ejecución sigue muy ligada al hardware subyacente manteniendo una considerable potencia para la programación a bajo nivel, aunque también permite un estilo de programación con un alto nivel de abstracción.

Tiene una velocidad de ejecución superior a muchos lenguajes. Algunas de sus extensiones tienen aplicación fuera del contexto de programación orientada a objetos pues muchos aspectos pueden ser usados independientemente de las clases.

Al estar ligado al hardware subyacente, las soluciones desarrolladas en C++ deben ser recompiladas para cada plataforma y presentan limitaciones para la escalabilidad hacia la Web. Por lo que no se adoptará para la elaboración de esta solución informática (Trupin, 2000).

Python

Python es un lenguaje de programación de alto nivel cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible.

Se trata de un lenguaje de programación multiparadigma ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico, es fuertemente tipado y es multiplataforma.

Ha incorporado gran cantidad de librerías. Dispone de varias funciones para el tratamiento de las cadenas de texto, números, archivos. Se pueden importar librerías en los programas para tratar temas específicos como la programación de ventanas. Por ser un lenguaje de alto nivel y al tratarse de un lenguaje interpretado, el

rendimiento es más bajo; por lo que no se adoptará para la elaboración de esta solución informática. (Company, 2009)

Java

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

La independencia de la plataforma, significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware. Este es el significado de ser capaz de escribir un programa una vez y que pueda ejecutarse en cualquier dispositivo (Otero, 2010).

El diseño de Java, su robustez, el respaldo de la industria y su fácil portabilidad han hecho de Java uno de los lenguajes con un mayor crecimiento y amplitud de uso en distintos ámbitos de la industria de la informática.

Hoy en día existen multitud de aplicaciones gráficas de usuario basadas en Java. El entorno de ejecución Java (JRE) se ha convertido en un componente habitual en las PC de usuarios de los sistemas operativos más usados en el mundo. Además, muchas aplicaciones Java lo incluyen dentro del propio paquete de la aplicación de modo que se ejecuten en cualquier PC (Otero, 2010).

Java constituye un lenguaje eficaz y muy versátil. Permite a los desarrolladores crear un sistema informático en una plataforma y ejecutarlo en otra distinta de la inicial. Se caracteriza por su potencia, y a la vez elimina las características menos usadas y más complejas de otros lenguajes como C y C++ (Carrero, 2011).

Se propone utilizar Java como lenguaje en el desarrollo de esta solución informática para que la misma sea multiplataforma. Java permite el diseño y construcción de aplicaciones de múltiples propósitos, cuenta con librerías bien documentadas para la gestión de interfaces de usuario y facilita la escalabilidad hacia la Web.

1.7 Entorno de Desarrollo Integrado (IDE[7])

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI) (Autors, 2011). Para el desarrollo del sistema se utilizará la plataforma NetBeans en su versión 6.9. Constituye una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes, soportando el desarrollo de todos los tipos de aplicación Java.

La plataforma ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma están la administración de módulos tales como configuración de usuario, de almacenamiento guardando y cargando cualquier tipo de datos, de ventanas y de las interfaces de usuario como son menús y barras de herramientas.

Ofrece funcionalidades para escribir, compilar, depurar y ejecutar programas. Es distribuido de manera libre, sin restricciones de uso. El desarrollador obtiene todas las herramientas necesarias para la creación de aplicaciones profesionales de escritorio, empresarial y para terminales móviles. Se ejecuta sobre varias plataformas (Windows, Mac OS X, GNU/Linux y Solaris). Facilita la gestión de las interfaces de usuario, la administración de las configuraciones del usuario y gestión de almacenamiento (Autors, 2011). Tiene como desventaja que al estar escrito en Java depende de tener instalado la maquina virtual de Java.

1.8 Metodología de desarrollo de software

Las metodologías de desarrollo de software definen un conjunto de procesos y actividades que guían a los desarrolladores de software durante el ciclo de vida de un proyecto, especificando los artefactos a construir y organizando los recursos humanos involucrados en roles dentro de un equipo de trabajo. Existen dos grandes grupos de metodologías: las ágiles y las robustas (2006).

1.8.1 Metodologías ágiles de desarrollo de software

Las metodologías ágiles o ligeras se distinguen por la simplicidad de sus reglas, prácticas, su orientación a equipos de desarrollo de pequeño tamaño, su flexibilidad ante los cambios y su ideología de la colaboración constante. Este tipo de

[7] IDE por sus siglas en inglés)

metodologías define pocos artefactos a generar y roles que asumir en el proceso de desarrollo de software. Identifica al cliente como un miembro activo más dentro del equipo de desarrollo. Están concebidas para proyectos donde los cambios en el desarrollo del sistema se produzcan con mucha frecuencia.

eXtremeProgramming, (XP) ó Programación Extrema

Promueve, para el éxito del desarrollo de un software, las relaciones interpersonales, así como el continuo aprendizaje de los miembros del equipo de trabajo y el fomento de la retroalimentación entre clientes y desarrolladores. Según esta metodología el cliente debe estar presente en todo momento junto al equipo de desarrollo en el proyecto pues tiene la responsabilidad de orientar el trabajo hacia lo que aporte mayor valor al negocio. El ciclo de vida en XP queda definido en seis fases: Exploración, Planificación de la Entrega, Iteraciones, Producción, Mantenimiento y Muerte del Proyecto (Kent Beck, 1999).

Durante el desarrollo del sistema para la edición de ficheros de mapas se esperan cambios en el equipo de desarrollo, por lo que se requiere que el proceso esté sólidamente documentado. No se esperan cambios frecuentes de los requisitos, por lo que esta metodología no será adoptada.

1.8.2 Metodologías robustas de desarrollo de software.

Las metodologías robustas de desarrollo de software o metodologías tradicionales surgen fruto de la acumulación de experiencias de determinadas empresas en la construcción de sistemas informáticos. A diferencia del nuevo enfoque ágil, ofrecen resistencia ante los cambios solicitados por un cliente. El equipo de trabajo conformado es mayor que el propuesto por las alternativas ágiles. Los artefactos generados son sustantivamente más numerosos que en las metodologías ágiles. Las metodologías robustas sugieren mantener un proceso lo más organizado y gestionado posible, evitando que la pérdida de personal dentro del grupo de desarrollo provoque que el proceso se vea afectado.

Proceso Unificado de Desarrollo de Software (RUP)

El Proceso Unificado Racional (*Rational Unified Process* en inglés, habitualmente resumido como RUP) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. El ciclo

de vida RUP es una implementación del desarrollo en espiral. Fue creado ensamblando los elementos en secuencias semi-ordenadas.

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.

RUP tiene como principales características la forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo),implementa las mejores prácticas en Ingeniería de Software, promueve un desarrollo iterativo, uso de arquitectura basada en componentes, mantiene un control de cambios y asegura la verificación de la calidad del software.

RUP se caracteriza por ser dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. El proceso de desarrollo de software se organiza a través de nueve flujos de trabajo organizados en cuatro fases de desarrollo. Los flujos de trabajo que define son: Modelado del Negocio, Levantamiento de Requisitos, Análisis y Diseño, Implementación, Prueba, Instalación, Gestión de Proyecto, Gestión de Configuración y Cambios y por último Ambiente. Define cuatro fases por las que todo proyecto informático debe ubicarse gradualmente: Inicio, Elaboración, Construcción y Transición (Jacobson, 2000).

Por lo que se propone utilizar esta metodología ya que permitirá mantener el proceso de desarrollo sólidamente documentado, por ser dirigida por casos de usos y centrado en la arquitectura. Además de contar con una verificación de la calidad del software.

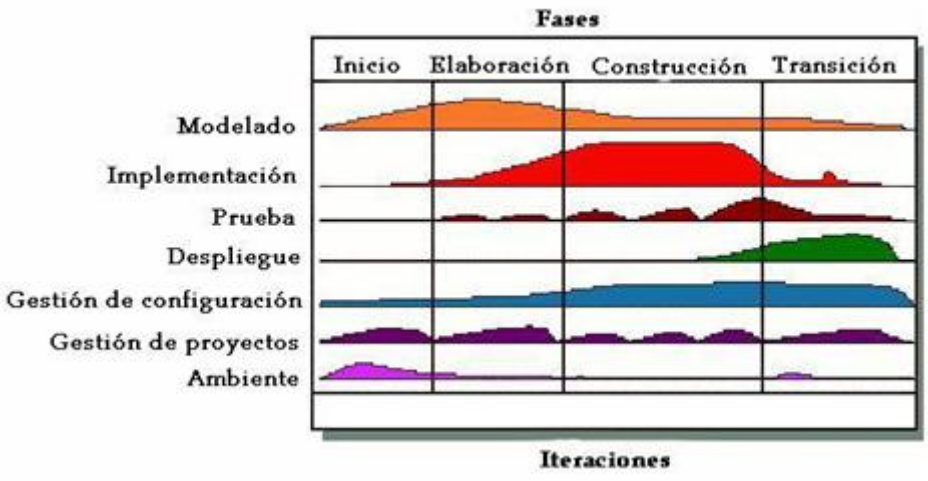

Fig. 2 Fases e Iteraciones dentro del ciclo de vida según RUP

1.9 Lenguaje de modelado y Herramienta CASE.

En la actualidad existen una serie de herramientas con este fin, las cuales han sido creadas por parte de sus desarrolladores con el objetivo de encontrar en ellas fiabilidad y eficiencia. Entre estas se encuentran: Visual Paradigm y Rational Rose Enterprise Edition, las que usan el Lenguaje de Modelado UML para la especificación, la visualización, la construcción y la documentación de los artefactos de los sistemas de software y también para otros tipos de sistemas (Informática, 1999).

1.9.1 Lenguaje Unificado de Modelado 2.0 (UML).

Lenguaje Unificado de Modelado (LUM o 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 (Group, 2011).

1.9.2 Visual Paradigm

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML (Visual Paradigm, 2002).

1.10 Conclusiones del capítulo

- La edición de las configuraciones de un fichero de mapa proporciona una personalización de los mapas mostrados a través de MapServer.
- La adopción de herramientas libres para el desarrollo de un sistema informático que edite un fichero de mapa permite no depender de sistemas que impongan licencias para su uso.

- Adoptar RUP como metodología de desarrollo de software ajustada al proceso de desarrollo proporciona que los artefactos generados queden debidamente documentado para el posterior análisis de otros miembros del centro GEySED.

Capítulo 2: Características del sistema:

2.1 Introducción

En el presente capítulo es descrito el dominio del problema tratado, se describe la solución propuesta basada en sus requisitos funcionales y no funcionales, son identificados los casos de uso y debidamente descritos textualmente.

2.2 Modelo de Dominio

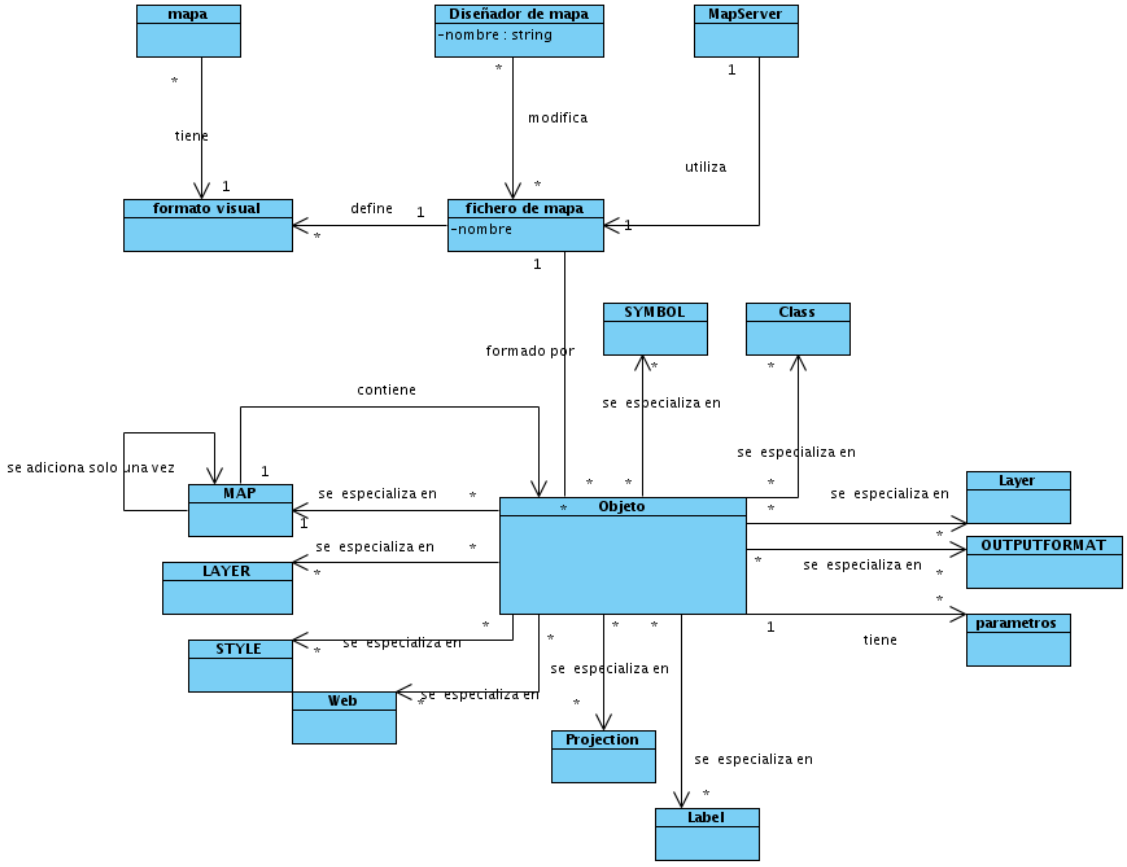

Fig. 3 Modelo de dominio

2.2.1 Breve descripción del Diagrama de Dominio

El diseñador de mapa representa a cada uno de los usuarios finales que interactúan con el sistema. Tiene la función de dirigir el proceso de gestión de edición de un fichero de mapa el cual utiliza el servidor de mapa MapServer para graficar los datos que este tiene descrito. Este fichero contiene varios objetos entre los que se

encuentran Map que es la propiedad de un mapa que contiene un valor además de la propiedad Capa (Layer[8]) la cual contiene un nombre como atributo.

2.2.2 Glosario de Términos del Dominio

Mapa: Un mapa es una representación gráfica y métrica de una porción de territorio generalmente sobre una superficie bidimensional pero que puede ser también esférica como ocurre en los globos terráqueos.

Capas del Mapa: Son los mapas que serán utilizados como capas para la formación del mapa topográfico a utilizar.

Diseñador de Mapas: Persona que trabaja en unos de los proyectos del centro GEySED que necesita editar cualquier fichero de mapa.

MapServer: MapServer permite la creación de Sistemas de Información Geográfica sobre entornos web, con el fin de visualizar, consultar y analizar información geográfica

Fichero de Mapa: El fichero de mapa es una de las partes fundamentales del servidor de mapas MapServer

Objetos: Conforman un fichero de mapa. Cada Objeto contiene palabras claves y otros objetos.

Map: Objeto principal del fichero de mapa el cual contiene todos los otros objetos que se encuentran bajo este.

Layer: Tipo de objeto. Marca el inicio de un objeto Layer dentro de un objeto Map.

2.2.3 Requerimientos Funcionales (RF)

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir, para que las peticiones del cliente queden satisfechas.

Tabla 2. Requerimientos Funcionales

A continuación se muestran los requerimientos funcionales del sistema:

RF. No.	Descripción
RF.1	Abrir un fichero de mapa desde un directorio local.
RF.2	Cerrar un fichero de mapa previamente abierto.
RF.3	Mostrar las configuraciones de un fichero de mapa.
RF.4	Guardar un fichero de mapa hacia un directorio local.
RF.5	Generar código ante una modificación realizada a un fichero de mapa

[8] Siglas en inglés

RF.6	Crear un fichero de mapa
RF.7	Validar acciones dentro de la gestión de objetos en un fichero de mapa
RF.8	Gestionar objetos de un fichero de mapa
RF.8.1	Adicionar objeto
RF.8.2	Eliminar objeto
RF.8.3	Modificar objeto
RF.9	Adherir una barra de escala
RF.10	Adherir una leyenda
RF.11	Cambiar el tipo de imagen de un mapa.
RF.12	Editar las dimensiones de salida de un mapa.
RF13	Editar la dimensión de la imagen de un mapa

Tabla 3. Requisito Funcional Abrir un fichero de mapa desde un directorio local

Nombre	RF.1. Abrir un fichero de mapa desde un directorio local.
Descripción	Permite que un fichero de mapa guardado en un directorio local en el ordenador sea cargado en el sistema.

Tabla 4. Requisito Funcional Cerrar un fichero de mapa.

Nombre	RF.2 Cerrar un fichero de mapa.
Descripción	Permite que un fichero de mapa previamente cargado en el sistema sea cerrado.

Tabla 5. Requisito Funcional Mostrar las configuraciones de un fichero de mapa

Nombre	RF.3 Mostrar las configuraciones de un fichero de mapa.
Descripción	Permite mostrar las configuraciones de un fichero de mapa previamente abierto.

Tabla 6. Requisito Funcional Mostrar mapa utilizando un fichero de mapa abierto

Nombre	RF.4 Mostrar mapa utilizando un fichero de mapa abierto.
Descripción	Permite que un fichero de mapa sea visto en su forma visual.

Tabla 7. Requisito Funcional guardar un fichero de mapa hacia un directorio local

Nombre	RF.5 Guardar un fichero de mapa hacia un directorio local.
Descripción	Permite que un fichero de mapa ya cargado sea salvado en un directorio local en el ordenador.

Tabla 8. Requisito Funcional Generar código en el fichero de mapa

Nombre	RF.6 Generar código en el fichero de mapa

Descripción	Permite generar el código en un fichero de mapa después de cualquier edición al mismo.

Tabla 9. Requisito Funcional Crear un fichero de mapa

Nombre	RF.7 Crear un fichero de mapa
Descripción	Permite crear un fichero de mapa

Tabla 10. Requisito Funcional Buscar errores sintácticos en el código

Nombre	RF.8 Buscar errores sintácticos en el código.
Descripción	Permite analizar el código de un fichero de mapa después de cualquier edición al mismo buscando errores sintácticos.

Tabla 11. Requisito Funcional Gestionar capas

Nombre	RF.9 Gestionar objetos
Asociados	RF.9.1, RF.9.2, RF.9.3
Descripción	Permite gestionar los objetos de un fichero de mapa abierto o creado.

Tabla 12. Requisito Funcional Adicionar capa

Nombre	RF.9.1 Adicionar objeto
Descripción	Permite adicionar un objeto a un fichero de mapa previamente cargado o creado.

Tabla 13. Requisito Funcional Eliminar capa

Nombre	RF.9.2 Eliminar capa
Descripción	Permite eliminar un objeto de un fichero de mapa previamente cargado o creado.

Tabla 14. Requisito Funcional Modificar capa

Nombre	RF.9.3 Modificar objeto
Descripción	Permite modificar un objeto de un fichero de mapa previamente cargado.

Tabla 15. Requisito Funcional Adherir una leyenda

Nombre	RF.11 Adherir una leyenda
Descripción	Permite adicionar una leyenda a un fichero de mapa previamente cargado.

Tabla 16. Requisito Funcional Cambiar el tipo de imagen de un mapa

Nombre	RF.12 Cambiar el tipo de imagen de un mapa.
Descripción	Permite cambiar el tipo de imagen de un fichero de mapa ya cargado.

Tabla 17.Requisito Funcional Editar las dimensiones de salida de un mapa

Nombre	RF.13 Editar las dimensiones de salida de un mapa.
Descripción	Permite editar las dimensiones de salida de un mapa previamente cargado.

Tabla 18. Requisito Funcional Editar la dimensión de la imagen de un mapa.

Nombre	RF.14 Editar la dimensión de la imagen de un mapa.
Descripción	Permite editar las dimensiones de la imagen de un mapa previamente cargado.

2.2.4 Requisitos no funcionales (RNF)

Los requerimientos no funcionales, son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa, define las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del mismo (Domínguez, 2003).

A continuación se muestran los requerimientos no funcionales del sistema:

RNF 1.Usabilidad

1.1 -La aplicación debe ejecutarse en cualquier sistema operativo en las estaciones de trabajo.

1.2 -La aplicación puede ser usada por cualquier persona con conocimientos básicos de la estructura de los ficheros de mapas utilizados por MapServer.

RNF 2.Fiabilidad

2.1- El sistema realizará una adecuada supervisión en la gestión de objetos.

RNF 3.Eficiencia

1.1 -Los algoritmos de procesamiento de información deben ser optimizados para que el tiempo de entregar las respuestas sea el menor posible.

RNF 4.Soporte

1.2 -A la aplicación se le brindará soporte y actualización en el tiempo que estime el equipo de desarrollo.

RNF 5.Portabilidad

5.1- El sistema será portable. Podrá ser ejecutado en cualquier sistema operativo en el cual se ejecute la maquina virtual de Java.

RNF 6.Software

1.1 -La aplicación será implementada en el sistema operativo GNU/Linux.

1.2 -La aplicación será implementada en el lenguaje Java.

1.3 -La aplicación será desarrollada en el IDE NetBeans 6.9.

1.4 -En las estaciones de trabajo donde se ejecute la aplicación debe estar instalada la máquina virtual de Java versión 1.5 o superior.

1.5 -Se utilizará para el desarrollo, diseño y modelado de los artefactos del ciclo de vida del software la herramienta de modelado Visual Paradigm 6.4.

RNF 7.Interfaces de Usuario

7.1-La aplicación debe contar con un diseño gráfico sencillo, fácil de entendimiento donde el usuario se sienta cómodo y con total conocimiento de la herramienta.

RNF 8.Interfaces de Software

8.1-La construcción de la aplicación funcionará bajo los conceptos de aplicación desktop.

RNF 9.Requisitos de Licencia

9.1- Para el desarrollo de la aplicación de acuerdo a los tipos de licencias de los componentes y herramientas que se proponen a utilizar se puede catalogar legalmente esta arquitectura de modelo libre, permitiendo la utilización, modificación y distribución de las mismas por terceros sin necesidad de obtener la autorización de sus respectivos titulares.

RNF 10.Requisitos Legales, de Derecho de Autor y otros.

10.1- La aplicación debe ajustarse y regirse por la ley, decretos leyes, decretos, resoluciones y manuales establecidos, que norman los procesos que serán automatizados.

10.2- Todas las herramientas de desarrollo deben ser de código abierto.

10.3- La aplicación se distribuye amparado bajo las normativas legales establecidas en el registro comercial emitido por las entidades jurídicas de la Universidad de las Ciencias Informáticas.

RNF 11.Hardware

11.1- 256 MB de memoria RAM instalada (requerimiento de la maquina virtual).

11.2- Procesador 512 MHz como mínimo.

2.3 Descripción del Sistema Propuesto.

2.3.1 Modelo de casos de uso del sistema

Actor del Sistema

Cada trabajador del negocio que tiene actividades a automatizar es un candidato a actor del sistema. Si algún actor del negocio va a interactuar con el sistema, entonces también será un actor del sistema.

Tabla 19. Descripción del actor del sistema

Actor del sistema.	Descripción
Diseñador de mapas	Persona que trabaja en unos de los proyectos del centro GEySED que necesita editar cualquier fichero de mapa.

Diagrama de Casos de Usos del Sistema.

Un caso de uso es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema.

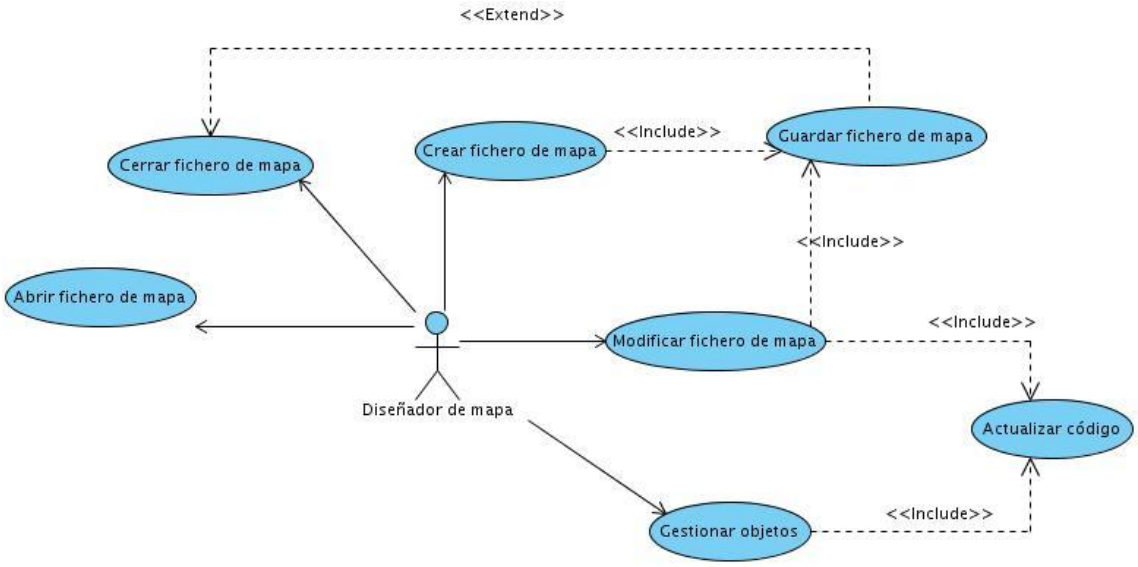

Fig. 4 Diagrama de casos de uso del sistema

Descripciones textuales de los Casos de Usos del Sistema.

Tabla 20. Descripción textual del Caso de Uso Abrir fichero de mapa

Caso de Uso:	Abrir fichero de mapa
Actor:	Diseñador de mapa
Propósito:	Permite que el diseñador abra un fichero de mapa desde una dirección local en la estación de trabajo.
Resumen:	El caso de uso inicia cuando el diseñador solicita la apertura de un fichero de mapa. El sistema le proporciona una interfaz para seleccionar el fichero y termina mostrando el código del mismo.
Precondiciones:	El fichero debe existir en el directorio seleccionado
Referencias	RF 1
Prioridad	Secundario
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Diseñador de mapa solicita abrir un fichero de mapa	1.1 Permite buscar el fichero en los directorios locales.
2. Selecciona el fichero a abrir	2.1 Verifica que sea un fichero de mapa lo que se solicita abrir. 2.2 Abre el fichero 2.3 Muestra el código del fichero de mapa 2.4 Muestra un árbol de estructuras que simbolizan los objetos del fichero de mapa. Culminando el caso de uso
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.2 Indica que el fichero que se solicita abrir no es un fichero de mapa.
Poscondiciones:	El fichero es abierto y sus datos mostrados

Tabla 21 . Descripción textual del caso de uso Crear fichero de mapa

Caso de Uso:	Crear fichero de mapa
Actor:	Diseñador de mapa
Propósito:	Permite que el diseñador cree un fichero de mapa
Resumen:	El caso de uso inicia cuando el diseñador solicita crear un nuevo fichero de mapa. El sistema brinda la posibilidad de crear los objetos que corresponden al

	fichero. Culminado el caso de uso.
Precondiciones:	
Referencias	RF 6
Prioridad	Critico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Diseñador de mapa solicita crear un fichero de mapa	1.1 Permite adicionar los objetos necesarios
2. Determina adicionar los objetos que considere necesarios	2.1 Se ejecuta el caso de uso Gestionar Objetos.
3.Guarda el fichero creado	3.1 Se ejecuta el caso de uso Guardar fichero de mapa. Culmina el caso de uso
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
Poscondiciones:	El fichero es creado y guardado

Tabla 22 Descripción textual del caso de uso Cerrar fichero de mapa

Caso de Uso:	Cerrar fichero de mapa
Actor:	Diseñador de mapa
Propósito:	Permite que el diseñador pueda cerrar un mapa cargado.
Resumen:	El caso de uso inicia cuando el usuario selecciona la opción de cerrar mapa y termina cuando el sistema cierra el mapa.
Precondiciones:	Debe existir un fichero de mapa abierto
Referencias	RF 2
Prioridad	Secundario
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando el actor selecciona la opción cerrar mapa.	1.1 El sistema verifica si se han hechos cambios en el mapa que no hayan sido guardados 1.2 El fichero es cerrado. Culmina el caso de uso
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**

	1.2 El usuario confirmará si desea guardar los cambios o si desea cerrar el fichero
2. Selecciona "guardar" o "no guardar" el fichero	1.3 en caso de seleccionar "guardar" se ejecuta el caso de uso Guardar Fichero de Mapa. Ir al caso de Uso Guardar fichero de mapa 1.4 es cerrado el fichero de mapa. Culminando el caso de uso
Poscondiciones:	El fichero es cerrado

Tabla 23 Descripción textual del caso de uso Modificar fichero de mapa

Caso de Uso:	Modificar fichero de mapa
Actor:	Diseñador de mapa
Propósito:	Permite que el diseñador pueda modificar un mapa abierto.
Resumen:	El caso de uso inicia cuando el Diseñador de mapas abre un fichero de mapa, y cambia alguna configuración de los objetos. Culminando del caso de uso
Precondiciones:	Debe existir un fichero de mapa abierto
Referencias	RF 1, RF 3
Prioridad	Critico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Diseñador de mapa selecciona el objeto a modificar 2. Solicita mostrar las configuraciones del objeto seleccionado	2.1 El sistema muestra las configuraciones del objeto seleccionado
3. Cambia los valores mostrados que estime necesario	3.1 Se ejecuta el caso de uso Actualizar Código. Terminando el caso de uso
Poscondiciones:	El fichero de mapa es modificado.

Tabla 24 Descripción textual del caso de uso Guardar fichero de mapa

Caso de Uso:	Guardar fichero de mapa
Actor:	Diseñador de mapa

Propósito:	Permite que el diseñador pueda guardar un fichero de mapa abierto.
Resumen:	El caso de uso inicia cuando el Diseñador de mapas solicita guardar un fichero de mapa. El sistema guardar en un directorio local el fichero de mapas, culminando del caso de uso
Precondiciones:	Debe existir un fichero de mapa abierto
Referencias	RF.4
Prioridad	Secundario
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Diseñador de mapa solicita guardar el fichero de mapas	1.2 El sistema permite que sea guardado el fichero en un diferente o en el directorio del fichero en cuestión.
2. Selecciona donde guardar el fichero	2.1 El sistema guarda el fichero , culminando el caso de uso
Poscondiciones:	El fichero de mapa es guardado

Tabla 25 Descripción textual del caso Actualizar código

Caso de Uso:	Actualizar código
Actor:	Diseñador de mapa
Propósito:	Permite que el diseñador pueda actualizar el código de un fichero de mapa que se haya modificado.
Resumen:	El caso de uso inicia cuando el Diseñador de mapas solicita actualizar el código de un fichero creado o modificado. El sistema verifica que existan cambios en los objetos y modifica el código ante estos cambios, culminando del caso de uso
Precondiciones:	Debe existir un fichero de mapa abierto o creado
Referencias	RF.5
Prioridad	Secundario
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Diseñador de mapa solicita actualizar el código de un fichero de mapas	1.1 El sistema verifica que existan objetos que tengan modificaciones o que estén creados. 1.2 El sistema actualiza el código correspondiente a cada uno de los objetos modificados o que estén creados ,

	culminando el caso de uso
Poscondiciones:	El código de los objetos es actualizado

Tabla 26 Descripción textual del caso **Gestionar objetos**

Caso de Uso:	Gestionar objetos
Actor:	Diseñador de mapa
Propósito:	Permite que el diseñador pueda gestionar los objetos de un fichero de mapas creado o abierto.
Resumen:	El caso de uso inicia cuando el Diseñador de mapas solicita realizar una de las opciones que trae consigo gestionar un objeto, crear, eliminar, buscar y modificar un objeto. El sistema permite que se realice la acción seleccionada, culminando del caso de uso
Precondiciones:	Debe existir un fichero de mapa abierto o creado
Referencias	RF.8, RF.8.1, RF.8.2 , RF.8.3
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Diseñador de mapa solicita gestionar un objeto	1.1 El sistema brinda las opciones que puede hacer sobre los objetos : • Adicionar objeto • Eliminar objeto • Mostrar objeto • Modificar objeto
2. Selecciona la opción que determine necesaria	2.1 En caso de seleccionar *Adicionar objeto* ir a la sección adicionar objeto 2.2 En caso de seleccionar *Eliminar objeto* ir a la sección adicionar objeto 2.3 En caso de seleccionar *Mostrar objeto* ir a la sección adicionar objeto 2.4 En caso de seleccionar *Modificar objeto* ir a la sección adicionar objeto
3. Solicita que sea actualizado el	3.1 Se ejecuta el caso de uso Actualizar código

código	
Sección Adicionar objeto	
Acción del Actor	**Respuesta del Sistema**
2.1 Seleccionar el objeto receptor del nuevo objeto 2.2 Seleccionar el nuevo objeto a adicionar	2.2.1 El sistema verifica que sea correcta la asignación 2.2.2 .muestra los campos necesarios que tiene que tener el nuevo objeto a adicionar
2.3 Llena los campos para el nuevo objeto 2.4 Solicita adicionar el objeto	2.3.1 Adiciona el objeto , culmina la sección
Flujo alterno	
	2.2.2 Muestra un mensaje de error indicando la invalidez de la acción, culmina el caso de uso
Sección Eliminar objeto	
	2.2.1 El sistema pide confirmación de la acción a realizar
2.3 Confirma la acción	2.3.1 En caso de ser aceptada se elimina el objeto
Sección Mostrar objeto	
	2.2.1 Muestra los datos del objeto identificado
Sección Modificar objeto	
	2.2.1 Muestra los datos del objeto identificado
2.3 Modifica los datos que estime necesarios	2.3.1 Se ejecuta el caso de uso Actualizar Código
2.4 Solicita guardar los cambios	2.4.1 Se ejecuta el caso de uso Guardar fichero de mapa
Poscondiciones:	El código de los objetos es actualizado

2.4 Conclusiones

- Se logró definir diagrama de dominio ya que los procesos de negocio no están bien definidos para el desarrollo de un modelo de negocio.
- Los requisitos funcionales definidos son los que el sistema debe poseer, estos permiten abarcar la edición básica de las configuraciones de un fichero de mapa.
- Los requisitos no funcionales definidos permiten que el sistema sea fácil de usar, con una interfaz de usuario intuitiva
- Los casos de uso identificados permiten que el proceso de desarrollo quede guiado y describan la forma en que serán desarrollado cada uno de los requisitos funcionales definidos.

Capítulo 3: Construcción del sistema

Introducción

Existe una serie de principios y buenas prácticas que a lo largo de los años de experiencia en la elaboración de arquitecturas de software han ayudado considerablemente a los ingenieros informáticos ante problemas comunes de organización y flujo de datos entre componentes de un sistema informático. Son comúnmente llamados patrones de arquitectura.

Un patrón es una solución a un problema en un contexto, codifica conocimiento específico acumulado por la experiencia en un dominio. Describe un problema que se instancia varias veces en un dominio determinado y propone la solución a ese problema. En el caso de los patrones arquitectónicos proponen un esquema organizativo estructural para los sistemas informáticos.

3.1 Patrón Modelo-Vista-Controlador

El patrón arquitectónico Modelo Vista Controlador (en lo adelante MVC) separa los datos de una aplicación, la lógica de la misma y las interfaces de usuario en tres estructuras distintas las cuales tienen definidas reglas de comunicación. El Modelo es referido a la representación de la información con la cual un sistema informático opera. La Vista se encarga de visualizar de una manera entendible los datos obtenidos desde el modelo.

El controlador responde ante los eventos de los usuarios de acuerdo con sus reglas de gestión de eventos definidas, pudiendo dirigir cambios en el Modelo. Adoptar MVC como patrón de arquitectura proporciona una buena organización del código generado, su reutilización y flexibilidad, ofrece un punto de entrada único para toda la aplicación y reduce el tiempo necesario para modificar y añadir mejoras a una aplicación. Es muy apropiado para aplicaciones Web.

3.2 Patrón arquitectónico de Tres Capas

Una arquitectura de software diseñada en capas consiste en la definición de niveles de abstracción, los cuales tienen una función específica permitiendo un diseño modular. Ello permite que se creen sistemas con un bajo acoplamiento entre sus módulos o componentes. Una variante de este patrón muy utilizada es la de Tres Capas.

Mediante la cual se fracciona al sistema informático en la capa de Presentación, la capa de Lógica del Negocio y la capa de Acceso a Datos.

“La capa de Presentación es la que se encarga de interactuar con el usuario mediante la interfaz de usuario. La capa Lógica del Negocio, llamada también como Lógica de la Aplicación, se encarga de realizar las tareas para las cuales está concebido el sistema. Es implementada utilizando un modelo orientado a objetos del dominio de la aplicación. Se encarga de controlar las operaciones de acuerdo con las reglas del negocio. La capa de Acceso a Datos es la que gestiona el almacenamiento de los datos, ya sea en una base de datos o en un fichero, así como la consulta a los mismos” (Flower, 2003).

Una restricción de este patrón consiste en que las capas inferiores no deben de conocer ni hacer llamadas a procedimientos implementados en capas superiores, sino que las funcionalidades que ellas ofrecen son accedidas desde niveles mayores (Larman, 1999).

Para el desarrollo de la solución se adoptará una arquitectura de tres capas debido a sus potencialidades para la reutilización y el aprovechamiento de los beneficios de una clara separación entre las funciones desplegadas en capas. Además, permite la estandarización y proporciona una distribución clara del trabajo entre los miembros de un equipo de desarrollo.

Buenas prácticas de aplicación recomendable en el diseño de software.

3.3 Patrones GRASP

Los patrones GRASP (General *Responsibility Assignment Software Patterns*, en español Patrones Generales de Software para la Asignación de Responsabilidades) describen los principios fundamentales de la asignación de responsabilidades a objetos (Larman, 1999).

En el desarrollo de esta solución se aplicaron los patrones GRASP: Experto, Creador, Controlador, No Hables con Extraños, Bajo Acoplamiento, Alta Cohesión.

El patrón Experto soluciona el problema de asignar una responsabilidad de forma general, al recomendar para esta función a la clase que maneje la información necesaria. El patrón Creador trata el problema de qué clase debe ser la encargada de instanciar a otra. El patrón Controlador ayuda a decidir quién se encarga de administrar un evento del sistema. El patrón Bajo Acoplamiento recomienda asignar

las responsabilidades de tal forma que se le dé soporte a una mayor reutilización y poca dependencia. El patrón Alta Cohesión aconseja mantener la clase lo más cohesionada posible para controlar la complejidad de la misma.

3.4 Patrones GoF

Los patrones Singleton y Fachada son patrones de diseño creado por “La Pandilla de los Cuatro” (Gang of Four, GoF). Mediante el patrón Singleton se garantiza que una clase solo pueda ser instanciada una sola vez.

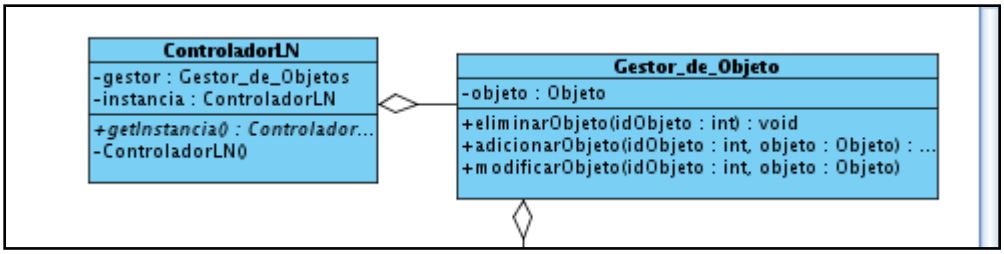

Fig. 5. Aplicación del Patrón Singleton

La clase ControladorLN tiene un objeto llamado instancia de la clase en cuestión, tiene el método getInstancia() el cual es público y estático que permite que sea accedido desde cualquier lugar del sistema sin necesidad de instanciar la clase a la cual pertenece, esto permite que sea devuelto el mismo objeto instancia. Y por último el constructor de la clase es privado para no permitir que no sea instanciada la clase desde fuera de su ámbito.

```
//--------- singleton ---------------------------
private static ControladorLN instancia = null;
public static ControladorLN getInstancia()
{
    if(instancia==null)
    {
        instancia = new ControladorLN();
        return instancia;
    }
    else return instancia;
}

public static ControladorLN anularInstancia()
{
    instancia = null;
    return instancia;
}
//-----------------------------------------------------

private ControladorLN()
{
   controladorAD = ControladorAD.getInstancia();
}
```

Fig. 6. Código donde se aplica el patrón Singleton

El patrón Fachada propone crear una clase que unifique las funciones de un conjunto de clases y que esta sea la encargada de colaborar con el sistema (Larman, 1999).

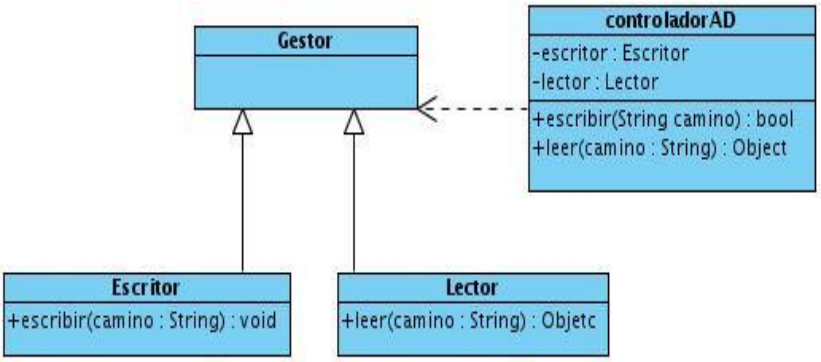

Fig. 7. Aplicación del Patrón Fachada

En la figura 8 se evidencia la utilización de este patrón en la clase ControladorAD. Esta clase sirve como una interfaz a la capa superior de Lógica de Negocio brindándole todas las funcionalidades.

3.5 Modelo de Diseño

Para una mejor comprensión de la solución y de acuerdo con la arquitectura de tres capas se han organizado las clases en tres paquetes principales: Presentación, Lógica del Negocio y Acceso a Datos. Cada uno de los paquetes se ha dividido a la vez en sub-paquetes de acuerdo con las funcionalidades de las clases contenidas. Cada paquete representa una capa de la arquitectura y está construido sobre su predecesor. Las clases pertenecientes a una capa están relacionadas con las clases de la capa inmediata inferior y desconocen a las clases de las capas superiores.

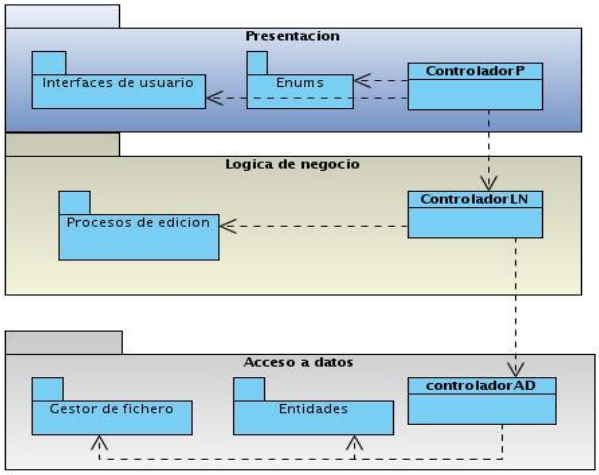

Fig. 8. Diagrama de paquetes

Acceso a Datos

La capa de Acceso a Datos es la encargada de la persistencia y recuperación de objetos, específicamente de la interacción de la aplicación con los ficheros de mapas guardados en un directorio local. El paquete Gestor de fichero cuenta con las clases de lector y escritor que son las clases encargadas de leer y escribir sobre el fichero .MAP. El paquete Entidades está con las clases que conservan los datos de las entidades en edición.

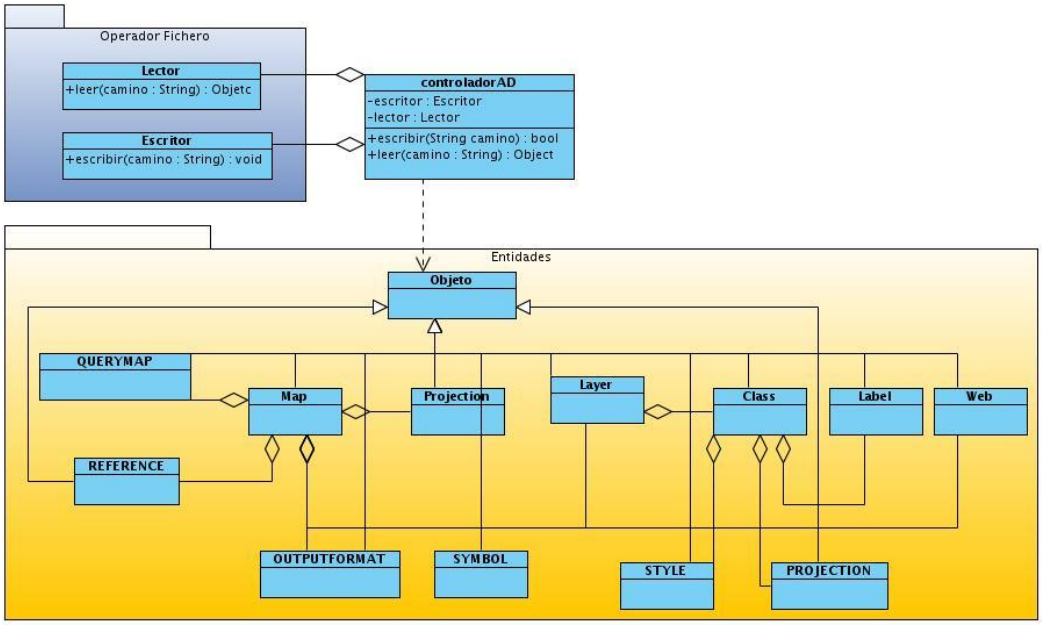

Fig. 9. Capa Acceso a Datos

Este paquete cuenta con las clases ControladorAD, Gestor, Lector Escritor y el paquete Entidades. La clase ControladorAD es la encargada de funcionar como un punto de comunicación entre la Capa Lógica de Negocio y el paquete Gestor de ficheros, esta clase se especializa en caso que se quiera gestionar un fichero de mapa copiado en el disco duro de la estación de trabajo. Otro paquete con que cuenta esta capa es el de Entidades, las clases de este paquete son las encargadas de almacenar la información básica de las entidades.

Tabla 27. Descripción de la clase ControladorAD

Nombre: ControladorAD	
Almacenar	
Nombre:	escribir(objeto:Objeto,bool modo)
Descripción	Garantiza la persistencia de un fichero de mapa.
Recuperar	
Nombre:	leer(): Objeto
Descripción	Garantiza la recuperación de un fichero de mapa

Lógica del Negocio

La capa Lógica del Negocio engloba a las clases encargadas de ejecutar las tareas y reglas que rigen el proceso (Soulie, 2009). En el diseño orientado a objetos la capa Lógica del Negocio se divide en otras menos densas. En esta solución el paquete de Lógica del Negocio está dividido en dos clases, la clase Gestor_de_Objeto y la clase ControladorLN.

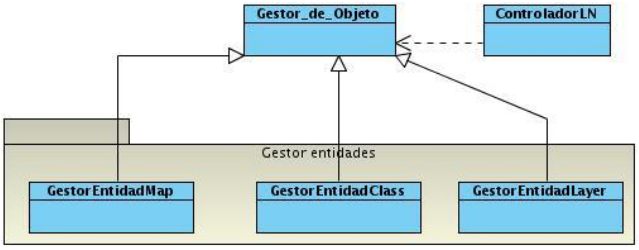

Fig. 10. Capa Lógica de Negocio

El paquete Gestor_de_Objeto es la base de la capa Lógica del Negocio. Esta clase contiene los procesos relacionados con la gestión de un fichero de mapa. La clase ControladorLN garantiza ser un punto de comunicación con la capa superior *Presentación* y con la capa *Acceso a Datos*.

Tabla 28. Descripción de la clase ControladorLN

Nombre: ControladorLN	
Almacenar	
Nombre:	escribir(objeto:Objeto,bool modo)
Descripción	Garantiza la persistencia de un fichero de mapa.
Recuperar	
Nombre:	leer(): Objeto
Descripción	Garantiza la recuperación de un fichero de mapa
Construir Objeto	
Nombre	Construir():Objeto
Descripción	Garantiza la creación de objetos que forman parte de un fichero de mapa.

Capa Presentación

La Capa Presentación es la encargada de lograr la comunicación directa entre el sistema y el usuario final a través de una interfaz gráfica. Una interfaz gráfica de

usuario consiste en un conjunto de componentes empleados por usuarios para comunicarse con los sistemas informáticos. Los usuarios dirigen el funcionamiento del sistema mediante la generación de eventos.

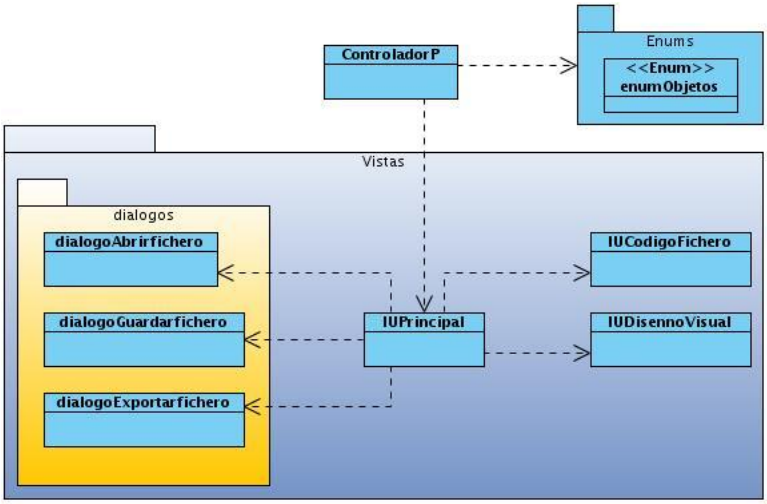

Fig. 11. Capa Presentación

La capa cuenta con la clase ControladorP encargada de gestionar todos los eventos visuales generados en la interfaz gráfica principal y delegar las responsabilidades al paquete correspondiente para que realice el procesamiento del evento a través de su controlador, también se comunica con la capa Lógica de Negocio.

Tabla 29. Descripción de la clase ControladorP

Nombre: ControladorP	
Atributos	
ventanaPrincipal: VentanaPrincipal	
controladorLN: ControladorLN	
Métodos	
Guardar	
Nombre:	guardar(datos:Objet,bool modo)
Descripción	Garantiza la persistencia de un fichero de mapa.
Abrir	
Nombre:	abrir(): Objeto
Descripción	Garantiza la recuperación de un fichero de mapa
Cargar fichero	
Nombre	Construir()
Descripción	Garantiza la representación de un fichero de mapas a través de componentes visuales.

La clase IUCodigo es un componente visual encargado de contener en texto plano el contenido de un fichero de mapa. La clase IUDisennoVisual se encarga de colocar a través de componentes visuales el código de un fichero de mapa.

3.6 Conclusiones del capítulo

- Adoptar una arquitectura de tres capas para el desarrollo de la solución permitió crear clases más cohesionadas, con lo cual se potencia el bajo acoplamiento y la reutilización de las mismas.

- El desacoplamiento de las entidades del negocio y de la capa de Acceso a Datos permite que esta pueda ser reutilizada en el desarrollo de otras herramientas.

- El diseño de la Capa Presentación permite la incorporación de nuevos componentes para representar los datos contenidos en un fichero de mapa. Por su parte, el diseño de la capa Lógica del Negocio permite su escalabilidad mediante la reutilización de las clases que representan las entidades del negocio.

- La abstracción de las entidades del negocio, de los procesos en los que están involucradas, potencia la reutilización y la escalabilidad del sistema en su conjunto.

Capítulo 4. Implementación y prueba

4.1 Introducción

En el presente capitulo se identifica el estándar de codificación a emplear en su desarrollo y la manera en que el sistema será desplegado. Por otra parte se realizan las pruebas al software como son las pruebas de caja negra aplicadas a las interfaces del sistema.

4.2 Estándares de Codificación

Un estilo o estándar de codificación es el conjunto de reglas o normas usadas para escribir código y que incluye una gran gama de aspectos dentro del proceso de codificación (UCI, 2010). El uso de un estilo de codificación contribuye a la comunicación dentro del equipo de desarrollo, al mantenimiento del software, y a la reducción de errores. La notación Camel consiste en escribir los identificadores con la primera letra de cada palabra en mayúsculas y el resto en minúscula: EndOfFile. Se denomina notación “Camel” porque los identificadores recuerdan las jorobas de un camello. Tiene dos variantes (UCI, 2010):

- UpperCamelCase, CamelCase o PascalCase: La primera letra también es mayúscula.
- lowerCamelCase, camelCase o dromedaryCase: La primera letra es minúscula.

Teniendo en cuenta que la solución se desarrollará en Java se decidió utilizar la notación CamelCase en identificadores de clases, y dromedaryCase para métodos y variables, siguiendo las recomendaciones de notación para este lenguaje de programación (UCI, 2010). Para la sangría se determinó utilizar el estilo BSD19, debido a las ventajas de visibilidad que el mismo ofrece

```
public class ControladorP
{
    private static ControladorP controladorP ;
    VentanaPrincipal ventanaPrincipal ;
    String tituloInicial  = "Editor de Ficheros de Mapas";
    EstadoFichero estadoFichero;
    ControladorLN controladorLN;


    public VentanaPrincipal getVentanaPrincipal() {
        return ventanaPrincipal;
    }

    public void setVentanaPrincipal(VentanaPrincipal ventanaPrincipa
        this.ventanaPrincipal = ventanaPrincipal;
    }
```

Fig. 12. Segmento de código con el estilo aplicado

4.3 Modelo de Despliegue

"El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo" (Jacobson, 2000). La presente solución es una aplicación para entornos de escritorio por lo que será ejecutada en una estación de trabajo, independientemente de que la misma disponga de conexión de red o no. A continuación se muestra el modelo de despliegue de la solución.

Fig. 13. Modelo de Despliegue

4.4 Modelo de Implementación

El modelo de implementación describe cómo los elementos del modelo de diseño, como las clases, se implementan en términos de componentes, como ficheros de código fuente, ejecutables. El modelo de implementación describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados y cómo dependen unos de otros (Jacobson, 2000).

Para la presente solución se implementaron tres subsistemas: AccesoDatos, LógicaNegocio y Presentación. El subsistema AccesoDatos contiene las clases encargadas de la persistencia y recuperación de las configuraciones de los ficheros de mapas, El subsistema LógicaNegocio está compuesto a su vez por el subsistema ProcesosGestion encargado de controlar la gestión de los objetos. El subsistema Presentación es el encargado de contener las funcionalidades de gestión de las

interfaces de usuario. A continuación se muestra una vista global del modelo de implementación de la solución propuesta.

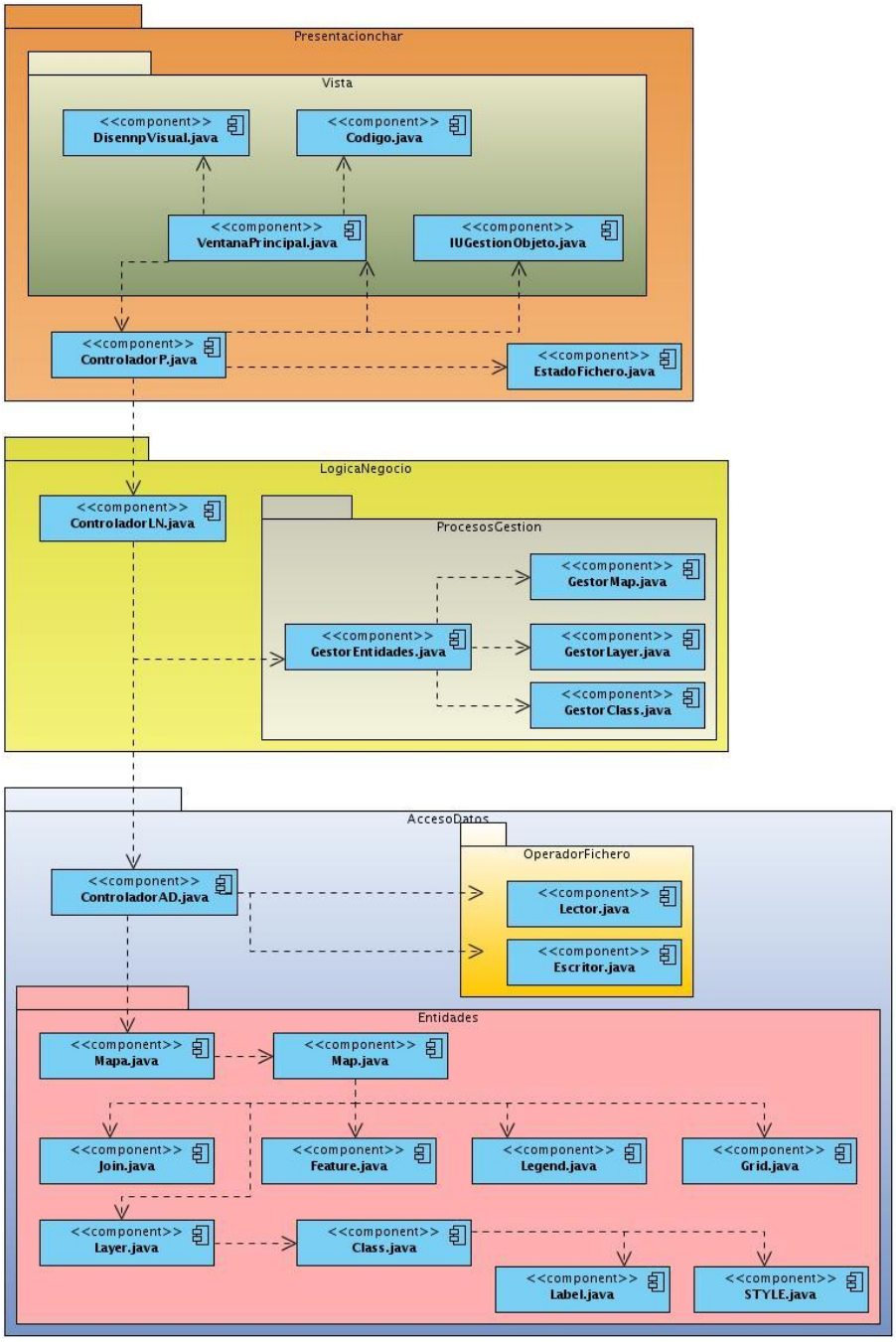

Fig. 14. Modelo de Implementación

4.5 Pruebas

En pruebas de software, conociendo una función específica para la que fue diseñado el producto, se pueden diseñar pruebas que demuestren que dicha función está bien

realizada. Dichas pruebas son llevadas a cabo sobre la interfaz del software, es decir, de la función, actuando sobre ella como una caja negra, proporcionando unas entradas y estudiando las salidas para ver si son o no las esperadas.

Las pruebas de caja negra son aquellas que se enfocan directamente en el exterior de la aplicación, sin importar el código, son pruebas funcionales en las que se trata de encontrar fallas en las que este no se atiene a su especificación.

Caso de Uso Abrir Fichero de Mapa

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Abrir fichero de mapa	EC 1.1 : Abrir el fichero correcto	Esta funcionalidad permite abrir un fichero de mapa desde un directorio local en la estación de trabajo	1. Hacer clic sobre la funcionalidad o en el menú abrir fichero. 2. Seleccionar la dirección del fichero a abrir. 3. Hacer clic en el botón aceptar. 4. Se abre el fichero seleccionado.
	EC 1.2 Abrir un fichero que no sea un fichero de mapa		1. Hacer clic sobre la funcionalidad o en el menú abrir fichero. 2. Seleccionar la dirección del fichero a abrir. 3. Hacer clic en el botón aceptar. 4. Se muestra un mensaje de error.

V1 : Fichero de mapa (V-válido, I-inválido)

Id del escenario	Escenarios	V1	Respuesta del sistema	Resultado de Prueba
EC 1.1	Abrir el fichero de mapa correctamente	V	El sistema Abre un nuevo fichero de mapa	Satisfactorio
EC 1.2	Abrir otro fichero que no sea un fichero de mapa	I	El sistema muestra un error al cargar el fichero.	Satisfactorio

Tabla 30. Resultado de Pruebas de caja negra Caso de uso Abrir Fichero de Mapa

Caso de Uso Gestionar Objeto

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
1. Adicionar Objeto	EC 1.1: Adicionar un objeto permisible a otro objeto	Esta funcionalidad permite Adicionar un objeto a un fichero de mapa.	1. Hacer clic sobre el objeto al cual se le va a adicionar el nuevo objeto. 2. Hacer clic en la funcionalidad Insertar

	existente.		objeto. 3. Seleccionar el objeto que se va a adicionar. 4. Llenar los campos que se soliciten. 5. Hacer clic en el botón adicionar. 6. Se adiciona el objeto.
	EC 1.2 Adicionar un objeto no valido.		1. Seleccionar el objeto al cual se le va a adicionar el nuevo objeto. 2. Hacer clic en la funcionalidad Insertar objeto. 3. Seleccionar el objeto que se va a adicionar. 4. Llenar los campos que se soliciten. 5. Hacer clic en el botón adicionar. 6. Se muestra un mensaje de error.
2. Eliminar objeto	EC 2.1 Eliminar un objeto seleccionado		1. Seleccionar el objeto que se desea eliminar 2. Seleccionar la opción eliminar objeto 3. Pedir confirmación para eliminar el objeto 4. En caso de ser aceptada eliminar el objeto 5. En caso de no ser aceptada mantener el objeto en el fichero de mapa.
	EC 2.2 Eliminar un objeto sin haber sido seleccionado		1. Seleccionar la opción eliminar objeto 2. Mantener el objeto en el fichero de mapa.
3. Modificar objeto	EC 3.1 Modificar un objeto seleccionado		1. Seleccionar el objeto que se desea modificar 2. Seleccionar el objeto que se desea eliminar 3. Cambiar las configuraciones de sus valores 4. Actualizar el código
	EC 3.2 Modificar un objeto sin haber sido seleccionado		1. Mantener el objeto en el fichero de mapa.

V1 : Tipo de Objeto (V-válido, I-inválido)

Id del escenario	Escenarios	V1	Respuesta del sistema	Resultado de Prueba
EC 1.1	Adicionar un objeto válido.	V	El sistema Adiciona un nuevo objeto.	Satisfactorio
EC 1.2	Adicionar un objeto no válido.	I	El sistema muestra un error al adicionar un nuevo objeto no válido.	Satisfactorio
EC 2.1	Eliminar un objeto seleccionado	V	El sistema elimina el objeto seleccionado	Satisfactorio
EC 2.2	EC 2.2 Eliminar un objeto sin haber sido seleccionado	I	El sistema mantiene el objeto en el fichero de mapa	Satisfactorio
EC 3.1	EC 3.1 Modificar un objeto seleccionado	V	El sistema muestra las configuraciones del fichero seleccionado	Satisfactorio
EC 3.2	EC 3.2 Modificar un objeto sin haber sido seleccionado	I	El sistema mantiene el objeto en el fichero de mapa	Satisfactorio

Tabla 31. Resultado de Pruebas de caja negra Caso de Uso Gestionar Objeto

Caso de Uso Cerrar Fichero de Mapa

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
Cerrar fichero de mapa	EC 1.1 : Cerrar un fichero de mapa sin ediciones.	Esta funcionalidad permite Cerrar un fichero de mapa anteriormente abierto.	1. Hacer clic sobre la funcionalidad o en el menú cerrar fichero de mapa. 2. El sistema mostrará un mensaje de información. 3. Dar clic en el botón aceptar. 4. El sistema cerrará el fichero abierto.
	EC 1.2 Cerrar un fichero de mapas con ediciones.		1. Hacer clic sobre la funcionalidad o en el menú cerrar fichero de mapa. 2. El sistema mostrará un mensaje de información. 3. Dar clic en el botón guardar los cambios. 4. El sistema

			guardará los cambios 5. El sistema cerrará el fichero abierto.

V1 : Fichero Editado (E- Editado, S- Sin editar)

Id del escenario	Escenarios	V1	Respuesta del sistema	Resultado de Prueba
EC 1.1	Cerrar un fichero de mapa sin ediciones.	S	El sistema muestra un mensaje de confirmación si desea cerrar el fichero	Satisfactorio
EC 1.2	Cerrar un fichero de mapas con ediciones.	E	El sistema muestra un mensaje de verificación mostrando si desea guardar los cambios en el fichero.	Satisfactorio

Tabla 32. Resultado de Pruebas de caja negra Caso de Uso Cerrar Fichero de Mapa

4.6 Conclusiones del capítulo

- La adopción de un estándar de codificación por el equipo de desarrollo proporciona que el código sea escrito de una misma forma.
- La distribución de la aplicación en un fichero .JAR permite que sea ejecutada en cualquier estación de trabajo que tenga instalada la maquina virtual de Java.
- Se realizaron pruebas de caja negra al sistema permitiendo encontrar errores en las interfaces del sistema.

Conclusiones

El sistema propuesto en el presente trabajo de diploma permite la edición de ficheros de mapas, el cual constituye una herramienta capaz de crear un fichero de mapa en su totalidad y una completa edición del mismo logrando llevar este proceso a los laboratorios donde se desarrollan las aplicaciones SIG.

El desarrollo de la presente solución informática para la edición de ficheros de mapas permitió dar cumplimiento a los objetivos trazados al inicio de esta investigación, con la cual se pudo arribar a las siguientes conclusiones:

- Se obtuvieron todos los artefactos generados durante el proceso de desarrollo propuesto por la metodología RUP.
- Se obtuvo un sistema capaz de crear y editar un fichero de mapa (MapFile) logrando una herramienta de escritorio que posibilitó el proceso de edición de los ficheros de mapas utilizados en los Sistemas de Información Geográfica.

La solución elaborada contribuye al proceso de edición de ficheros .MAP y soluciona la situación problemática de realizar una aplicación de escritorio capaz de trabajar sin necesidad de una red local la cual soluciona el proceso de edición de ficheros de mapa en el desarrollo de los SIG en el centro productivo GEySED.

Bibliografía

Autors, NetBeans. 2011. NetBeans. [En línea] 2011. http://netbeans.org/index_es.html.

Ballari, Daniela. 2008. El archivo Map. [En línea] 2008. http://mapas.topografia.upm.es/geoserviciosOGC/documentacion/WMS/Anexo-archivo-map.pdf.

Carrero, David. 2011. Programación en Castellano. [En línea] 2011. http://www.programacion.com/java.

CMAPTOOL COMO HERRAMIENTA DE APRENDIZAJE VISUAL. **ROJAS, ANDRÉS FELIPE MONSALVE y GIRALDO, EVELIO. 2007.** 2007.

Company, Phyton. 2009. www.python.org. [En línea] 2009. http://www.python.org/doc/.

2009. Definición ABC. [En línea] 2 de Enero de 2009. http://www.definicionabc.com/general/fichero.php.

2009. diccionario. [En línea] 2009. http://www.1diccionario.com/buscar .

Domínguez, José Carlos Sanchez. 2003. Verificación de los requisitos no funcionales en el software crítico. [En línea] Junio de 2003. www.softwcare.com/pdf/publicaciones/ForumCalidad(n142).pdf.

Ficha técnica de CyD Image Map. **CyD Image Map. 2008.** 2008, pág. WWW.CYDSOFT.COM.

Ficha técnica de GIFfyCutter. **GIFfyCutter. 2009.** 2009, pág. WWW.FCODERSOFT.COM.

Flower, Martin. 2003. *Patterns of Enterprise Application Architecture.* s.l. : Addison Wesley, 2003.

Group, Object Management. 2011. UML Resource Page. [En línea] febrero de 2011. http://www.uml.org/.

IES Alyanub. 2011. [En línea] marzo de 2011. http://www.iesalyanub.es/spip.php?article95.

Informática, INSTITUTO NACIONAL DE ESTADISTICA E. 1999. Herramientas Case. [En línea] Nov de 1999. www.inei.gob.pe/biblioineipub/bancopub/Inf/Lib5103/Libro.pdf.

Jacobson, Ivar, Boch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : s.n., 2000.

Kent Beck, Andres Cyntia. 1999. *Extreme Programming Explained.* s.l. : Addison-Wesley Professional, 1999.

Larman, Craig. 1999. *UML y Patrones.* s.l. : Patience Hall , 1999.

LINDE. 2000. ¿QUÉ ES UN S.I.G.? LA EXPERIENCIA DEL I.M.I. DEL AYUNTAMIENTO. [En línea] 2000. http://www.mappinginteractivo.com/plantillaante..

LOS SISTEMAS DE INFORMACIÓN. **RODRÍGUEZ, N. 1998.** 1998.

Manual de Autodesk Map 5. [En línea]

Manual de Autodesk Map 5. **ABCdatos. 2007.** 2007, pág. http://www.abcdatos.com/tutoriales/tutorial/z6978.html.

2006. Metodologias de desarrollo del software. [En línea] 2006. http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema04.pdf.

O@SIS. 2008. www.lawebdelprogramador.com. [En línea] 2008. http://oasis.cisc-ug.org/letzhune/cisc/tutoriales/tercero/lenguaje%20prog%20c%23.pdf.

Otero, Abrahan. 2010. JavaHispano. [En línea] 2010. http://www.javahispano.org/contenidos/es/curso_de_programacion_java__abraham_otero/?menuId=ARTICLES&onlypath=true.

Rivera, Javier Fernandez. 2008. Ficheros. [En línea] 2008. http://aurea.es/wp-content/uploads/ficheros-archivos.pdf.

Soulie, J. 2009. [En línea] 2009. http://www.cplusplus.com/doc/tutorial/..

Trupin, Joshua. 2000. Programacion C++. [En línea] 2000. www.msdn.microsoft.com/msdnmag/default.asp.

UCI. 2010. Entorno Virtual de Aprendizaje. Práctica Profesional. [En línea] 2010. http://eva.uci.cu/mod/resource/view.php?id=29419..

Visual Paradigm, Emagister. 2002. Emagister. [En línea] 2002. http://www.emagister.com/modelado-uml-visual-paradigm-cursos-1603500.htm.

YAGÜEZ, LANGHI &. 2002. Sistema de Información Geográfica (S.I.G.). [En línea] 2002. http://www.inta.gov.ar/barrow/info/documentos/SIG/que_es_sig.htm..

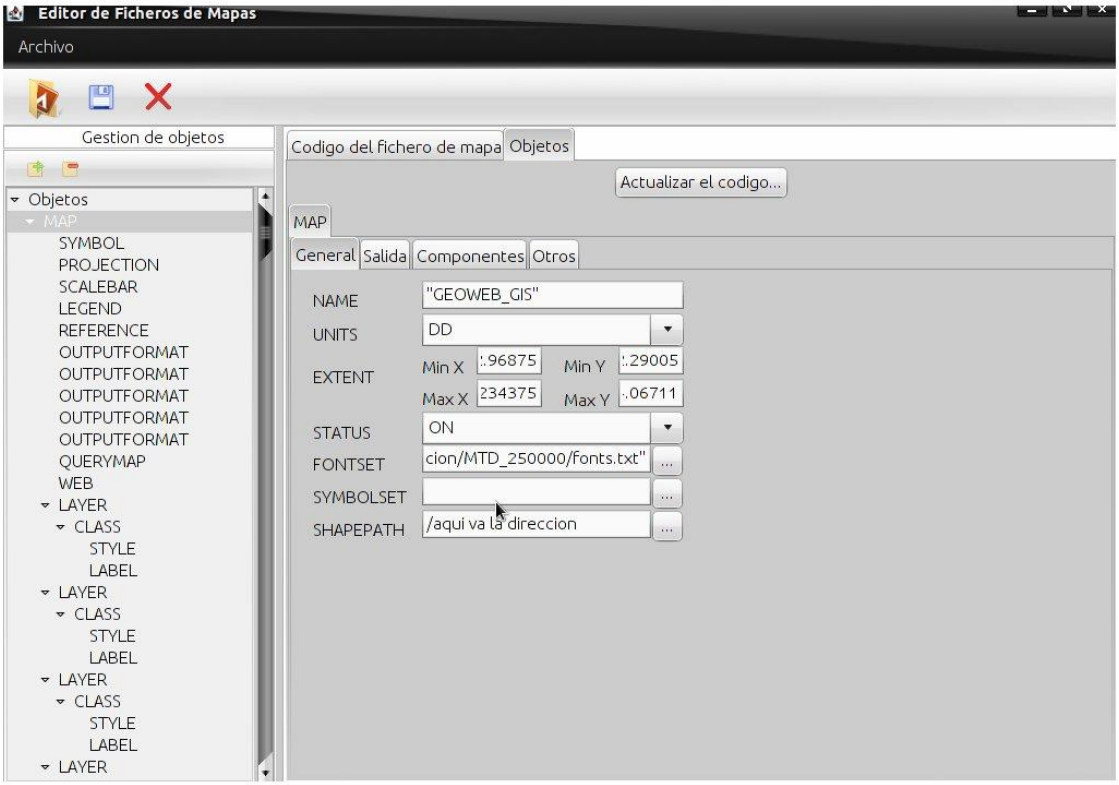

Fig. 15. Anexo 1. Prototipo utilizado para la obtención de requisitos