PROPUESTA DE UNA SOLUCIÓN DE SISTEMA DE INFORMACIÓN GEOGRÁFICA SOBRE KA-MAP

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

AUTOR: Franier González Avila

TUTOR: Ing. Armando Batista Piñeda

La Habana, Junio 2011

La felicidad humana generalmente no se logra con grandes golpes de suerte, que pueden ocurrir pocas veces, sino con pequeñas cosas que ocurren todos los días.

Benjamín Franklin (1706-1790). Estadista y científico.

DEDICATORIA

A quienes confiaron siempre en que podría lograr mis sueños: mis padres, Josefina y Francisco.

A mi hermano Frank por su confianza y siempre estar pendiente de mí.

A toda mi familia que me ha entregado su apoyo siempre.

A todas las personas que han creído en mí…

AGRADECIMIENTOS

A mi madre y mi padre por hacer todo lo posible para que yo saliera adelante en mi carrera.

A mis hermanos Frank y Dianelis por apoyarme en todo.

A mi novia Daymaris por estar siempre junto a mí, comprenderme y quererme tanto.

Al resto de mi familia gracias por apoyarme y darme aliento.

A Papito y Arlhey por estar estos cinco años en los momentos buenos y en los difíciles.

A Osmel, Diosbel, Rodolfo y todo el piquete que hicieron estos años divertidos.

A mi tutor por guiarme con este trabajo y por sus sugerencias.

A todos los que me ayudaron cuando lo necesité,

A la Revolución por permitirme estudiar en esta escuela.

A los que en algún momento me preguntaron por la tesis.

DECLARACIÓN DE AUTORÍA

Por este medio declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) para que hagan el uso que estimen pertinente del mismo.

Para que así conste firmo la presente a los ___ días del mes de junio del 2011.

Franier González Avila	Armando Batista Piñeda
Autor	Tutor

DATOS DE CONTACTO

TUTOR:

Ing. Armando Batista Piñeda

Graduado de Ingeniero Informático en el año 2006 en la Universidad de Holguín “Oscar Lucero Moya”

Categoría Docente: Instructor

Dirección de la institución: Carretera de San Antonio de los Baños Km 2 ½. Torrens. La Habana, Cuba

E-mail: armandobp@uci.cu

RESUMEN

Los Sistemas de Información Geográfica se han ido perfeccionando con el paso del tiempo y con los avances tecnológicos. Estos sistemas son de gran ayuda, ya que brindan al usuario la posibilidad de realizar una navegación intuitiva por la cartografía y además ayudan en la toma de decisiones en múltiples entidades. Tal es el caso de la Universidad de las Ciencias Informáticas, que además desarrolla varios SIG sobre plataformas libres, pues se ha demostrado que es una alternativa real al software propietario. Estas herramientas brindan numerosas funcionalidades, pero sin embargo resulta difícil su implementación; por tal motivo, disminuir el esfuerzo para dicho desarrollo se hace necesario. Es por ello que este trabajo de diploma abarca los aspectos fundamentales del cliente web para la publicación de datos cartográficos ka-Map, los conceptos más importantes, así como la descripción de la herramienta y una amplia explicación del objeto de estudio en la que está enmarcada la investigación y la necesidad de desarrollo de la misma. Se exponen algunas de las herramientas existentes en el mundo de los Sistemas de Información Geográfica, sus principales características y usos. Se fundamentan las elegidas para el desarrollo de la solución. Se realiza el análisis, diseño e implementación de la solución. Además probar que el producto cumple con las especificaciones definidas.

Palabras clave:

Ka-Map, datos geográficos, Sistema de Información Geográfica, software libre, visualización.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Existen referencias de que los primeros hombres que habitaron el planeta ya plasmaban en cuevas las rutas de emigración de animales que constituían su alimento (1), así como también límites de sus tierras para la actividad agrícola. Este ejemplo constituye tal vez el primer antecedente de los Sistema de Información Geográfica, más conocidos como SIG o GIS, por sus siglas en inglés.

Actualmente los SIG en el mundo son usados en diversas ramas como el trasporte urbano, correo postal, turismo de ciudad, ofertas de ventas, control epidemiológico, defensa civil, recursos minerales, recursos hidráulicos, planificación física, planificación y control de flotas de transporte, gestión de redes (datos, telefónicas, eléctricas), agricultura de precisión.

En la última década, el uso de los SIG se ha ido generalizando y ya es muy común su utilización por parte de un número considerable de empresas y entidades cubanas. Se puede mencionar el SIG para el manejo integral del ecosistema Sabana-Camagüey, primero de su tipo en el país. Se encuentra también el SIG-ESAC que no es más que el SIG para la gestión de la estadística de salud de Cuba. Tal es el caso de la Universidad de las Ciencias Informáticas (UCI), institución cuyo objetivo es convertirse en un centro de enseñanza superior de nuevo tipo que potencie la innovación y el uso de nuevas tecnologías.

La UCI contempla entre sus principales actividades la formación de profesionales, la investigación y la producción de software. Actualmente la universidad cuenta con varios centros de desarrollo que se especializan en diversas temáticas. Particularmente el centro de Geoinformática y Señales Digitales (GEYSED) cuenta con un departamento que se dedica al trabajo con los SIG. En los últimos años, este centro de desarrollo se ha interesado en la búsqueda de tecnologías y experimentación con estas con el fin de mejorar los resultados obtenidos.

Después de un estudio preliminar de los clientes web para SIG existentes en el mercado que han sido utilizados en otros proyectos en el mundo y Cuba se encuentra ka-Map con amplias condiciones para su utilización. A pesar de esto actualmente se carece de la experiencia necesaria para satisfacer posibles solicitudes de clientes múltiples en períodos cortos de tiempo. Además, la fuerza de trabajo con que se

cuenta no toda se encuentra a tiempo completo en la producción y está distribuida en varios y crecientes compromisos de las líneas de desarrollo existentes. Estas limitaciones implican un mayor esfuerzo de los recursos humanos y consumo de tiempo.

De la situación expuesta se identifica el siguiente **problema**: ¿Cómo contribuir a disminuir el esfuerzo y el tiempo necesario para brindar una solución a clientes interesados en publicar información cartográfica para el consumo en la Web sobre la tecnología libre ka-Map?

Para lograr resolver el problema planteado se propone desarrollar un prototipo funcional de Sistema de Información Geográfica que permita la visualización y consulta interactiva de información geográfica sobre ka-Map, lo cual figura como el **objetivo general** de esta investigación.

Para darle cumplimiento al objetivo presentado se hace necesario el análisis del proceso de desarrollo de Sistemas de Información Geográfica sobre tecnologías libres en entornos Web, lo cual constituye el **objeto de estudio** y el **campo de acción** se enmarca en la utilización de ka-Map en Sistemas de Información Geográfica.

Para dar cumplimiento al objetivo general se trazan las siguientes **tareas**:

- Establecer los fundamentos teóricos sobre el proceso de desarrollo de Sistemas de Información Geográfica sobre tecnologías libres en entornos Web.
- Seleccionar herramientas y metodologías adecuadas para el desarrollo del sistema.
- Identificar los requisitos del sistema a fin de lograr los objetivos planteados.
- Diseñar las actividades que tributen a la construcción de la solución.
- Implementar el diseño realizado.
- Probar que el producto cumpla las especificaciones definidas.
- Documentar el proceso realizado para su socialización.

El desarrollo exitoso de las tareas expuestas anteriormente permitirá el cumplimiento de la **idea a defender** de esta investigación: si se desarrolla un prototipo funcional de Sistema de Información Geográfica que permita la visualización y consulta interactiva de información geográfica sobre ka-Map

entonces debe disminuir el esfuerzo y el tiempo necesario para brindar una solución a clientes interesados en publicar información cartográfica para el consumo en la Web sobre esta tecnología.

Para una correcta comprensión y realización del trabajo se hace uso de los siguientes **métodos científicos**:

Análisis-histórico-lógico, para obtener información de antecedentes que existen respecto a otros SIG desarrollados tanto en Cuba como en el mundo que sirvan de punto de partida para el software que se desea construir.

Analítico-sintético, con el objetivo de buscar información referente a las herramientas a utilizar para la construcción de SIG, permitiendo de esta forma extraer los elementos más importantes y significativos de cada una para evaluar las ventajas y desventajas de las mismas.

Entrevista, con el objetivo principal de obtener datos reales de los hechos fundamentales que caracterizan la situación actual del proceso de desarrollo de SIG en el centro GEYSED, para identificar los requisitos funcionales y no funcionales con que contará la aplicación a construir.

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

El trabajo de diploma está conformado por cuatro capítulos.

En el *capítulo 1* se abordan los aspectos fundamentales para la comprensión del problema planteado, los conceptos más importantes, así como la descripción de la herramienta y una amplia explicación del objeto de estudio en la que está enmarcada esta investigación y la necesidad de desarrollo de la misma.

Algunas de las herramientas existentes en el mundo de los SIG se exponen en el *capítulo 2*, así como sus principales características y usos. Se fundamentan además las elegidas para el desarrollo de la solución.

La presentación de la solución propuesta se hace en el *capítulo 3*. En este se detallan los requisitos del sistema a fin de lograr los objetivos planteados. Se realizará el análisis y diseño para la construcción de la solución.

Por último, en el *capítulo 4* se exponen los elementos sobre la implementación del diseño realizado y las pruebas para que el producto cumpla con las especificaciones definidas.

1

1 FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se introducen conceptos importantes para un mejor entendimiento de este trabajo. Se describen las principales tendencias relacionadas con los SIG y algunas herramientas y tecnologías existentes, fundamentando la selección de las empleadas para desarrollar la solución.

1.2 Conceptos asociados al dominio del problema

En este epígrafe se dan a conocer los principales conceptos asociados que se relacionan con este trabajo, para lograr una mejor comprensión y visión general del tema a tratar.

Las primeras referencias de lo más semejante a un SIG datan de 1854 cuando el Dr. John Snow logra ubicar el foco de contagio de una epidemia de cólera tras utilizar métodos cartográficos únicos hasta el momento, que no solo representaban la realidad, sino que por primera vez analizaban conjuntos de fenómenos geográficos dependientes (2). Alrededor de la década de 1870 se creó un SIG por parte de una compañía de trenes de Irlanda que utilizó la superposición de acetatos, técnica que bajo el nombre de "foto litográfica" (donde los mapas eran separados en capas) alcanza un gran auge a comienzos del siglo pasado.

A partir de la década de los 50's surgen las primeras aplicaciones de software de cartografía automatizada (CAD[1] y CAM[2]) y las primeras bases de datos para operar con los atributos en el computador. En esa época solamente se obtenían mapas de buena factura pero nada más. Con la llegada de 1960 surgen los primeros sistemas que integran las bases de datos con las figuras cartográficas y esta nueva característica nunca dejaría de ser empleada desde entonces.

[1] Diseño Asistido por Ordenador.
[2]Fabricación Asistida por Ordenador.

Los SIG como se conocen hoy, son el resultado de la automatización de pesadas labores de producción cartográfica ligadas desde un inicio a los sistemas digitales y evolucionando propiamente desde la década de los 60's hasta la actualidad.

El primer SIG que logró cierta eficiencia fue el llamado CGIS[3] en 1962, estaba estructurado más que todo en polígonos y fue orientado a la gestión de los vastos recursos naturales de Canadá con información cartográfica relativa a tipos y usos del suelo, agricultura, espacios de recreo, vida silvestre y aves acuáticas. Su creador Roger Tomlinson está considerado como "el padre de los SIG".

Según el Diccionario de Organización y Representación del Conocimiento: **sistema** es el conjunto de elementos interrelacionados y regidos por normas propias, de modo tal que pueden ser vistos y analizados como una totalidad. El sistema se organiza para producir determinados efectos, o para cumplir una o varias funciones (3). Del planteamiento anterior se puede definir un sistema como un todo integrado de diversas estructuras interrelacionados entre sí con un objetivo común.

La **información** es cualquier publicación de un grupo de conocimientos obtenidos por la investigación, la comunicación, la instrucción .Un conjunto de datos que al relacionarse adquieren sentido o un valor de contexto o de cambio (4).

De esta forma **Sistema de información Geográfica** no es más que un conjunto de procedimientos sobre una base de datos no gráfica o descriptiva de objetos del mundo real que tienen una representación gráfica y que son susceptibles de algún tipo de medición respecto a su tamaño y dimensión relativa a la superficie de la tierra. A parte de la especificación no gráfica el SIG cuenta también con una base de datos gráfica, con información geo-referenciada o de tipo espacial y de alguna forma ligada a la base de datos descriptiva. La información es considerada geográfica si es mesurable y tiene localización (5).

La **cartografía** es la ciencia que se ocupa de la preparación y construcción de los mapas y cartas náuticas, reproduciendo en un plano la superficie terrestre (6). Estos **mapas** son la representación geográfica de la Tierra o parte de ella en una superficie plana (7) y constituyen el elemento fundamental de todo SIG.

[3] Sistema de Información Geográfica de Canadá ***(***Canadian Geographic Information System***).***

Para ser localizados los elementos en los mapas se utilizan las **coordenadas**. Que son el valor de los componentes de un vector (4). En otras palabras los valores que determinan la posición de un punto en un espacio dimensional y el sistema de ejes para el posicionamiento de un punto en el plano o en el espacio.

Los elementos cuando están posicionados en el mapa con sus coordenadas y sus datos pasan a ser **objetos geo-referenciados**: neologismo que refiere al posicionamiento con el que se define la ubicación de un objeto espacial (representado mediante punto, vector, área, volumen) en un sistema de coordenadas y datos determinados. Este proceso se utiliza frecuentemente en los Sistemas de Información Geográfica (8).

Un **modelo** se entiende como una representación de un objeto, sistema o idea, de forma diferente al de la entidad misma. El propósito de los modelos es ayudarnos a explicar, entender o mejorar un sistema. Un modelo de un objeto puede ser una réplica exacta de éste o una abstracción de las propiedades dominantes del objeto (9). En términos geográficos los SIG utilizan para visualizar su información principalmente dos modelos el **raster**, que consiste en una imagen formada por los colores o tonos de gris de una cuadrícula, en particular los pixeles del monitor (4) y el **vectorial** es una imagen digital formada por objetos geométricos independientes (segmentos, polígonos, arcos, etc.), cada uno de ellos definido por distintos atributos matemáticos de forma, de posición, de color, entre otros.

1.3 Estado del arte del cliente Web ka-Map

En particular ka-Map es una API[4] JavaScript multiplataforma para desarrollar aplicaciones Web cartográficas altamente interactivas. Su núcleo está desarrollado en PHP y es el complemento ideal para MapServer, ya que dota a los mapas de una interactividad y diseño muy por encima de los que MapServer ofrece por sí solo.

En el mundo se han desarrollado varias aplicaciones con esta API que no han dejado dudas en cuanto a la efectividad de ka-Map por ejemplo:

[4] Interfaz de Programación de Aplicaciones (del inglés Application Programming Interface).

- La página Oregon Coastal Atlas que puede ser utilizado por los visitantes para ver datos previamente almacenados en sus servidores de datos http://www.coastalatlas.net/maps/.
- El Proyecto Agrinova de la Universidad Politécnica de Valencia el cual desarrolló un visor con esta herramienta para el estudio y geo-referenciación de parcelas agrícolas. (10)
- El sitio Web http://www.openmapsihlwald.ch/ que consiste en las rutas para los visitantes de una zona de recreación (bosque) llamado Sihlwald cerca de Zúrich, Suiza.

En estos sitios se utiliza la interfaz predefinida de ka-explorer la cual por medio de botones y controles visuales permite la interacción con los mapas, pero esta se modifica de varias maneras como en el primer ejemplo donde es incluida dentro del CMS Joomla. En otros proyectos se le han añadido herramientas personalizadas como la búsqueda jerárquica mediante la cual se permite hacer una búsqueda con esta característica a partir de elementos administrativos geo-referenciados. La búsqueda directa que posibilita a través del atributo nombre obtener los datos asociados a él. Así como una función de impresión mejorada la cual puede ser general o específica, donde la general imprime lo que se muestra en el visor, mientras la específica lo hace con el mapa solicitado y sus datos.

Es común ver en este cliente Web para SIG la eliminación de botones y características que su uso no es muy útil con respecto a los requisitos que se desean lograr con el fin de simplificar su uso y aumentar sus ventajas con respectos a los restantes competidores del mercado.

Aún en el país el uso de ka-Map es muy limitado y no se presentan aplicaciones que lo usen.

1.4 Estudio comparativo entre ka-Map y herramientas similares

Los clientes web para SIG son aplicaciones que se utilizan para la visualización de la información geográfica y permiten su manejo a través de herramientas básicas de navegación y análisis. Han alcanzado gran importancia en los últimos tiempos gracias a la optimización de recursos en Internet y a las nuevas tecnologías desarrolladas para perfeccionar la experiencia de los usuarios en los navegadores web.

La mayoría de los proyectos están creados en base a dos paradigmas: UMN[5] MapServer y OpenLayers. Los clientes que se basan en UMN MapServer fueron creados años atrás valiéndose de las particularidades que este dispone: mapa, escala, mapa de referencia, herramientas de navegación básica, identificación de objetos espaciales; y su API llamada MapScript que ha sido implementada en diferentes lenguajes de programación como PHP, Python, Java, Perl y Ruby, y que sigue su progreso sumando funcionalidades como el etiquetado y la generación de gráficos de barra. Por otro lado se encuentra, la nueva generación de clientes que utilizan OpenLayers a causa de su óptimo desempeño en labores de renderización en la web y el apoyo de diferentes empresas. (11)

[5] MapServer fue originalmente desarrollado por la Universidad de Minnesota (UMN)

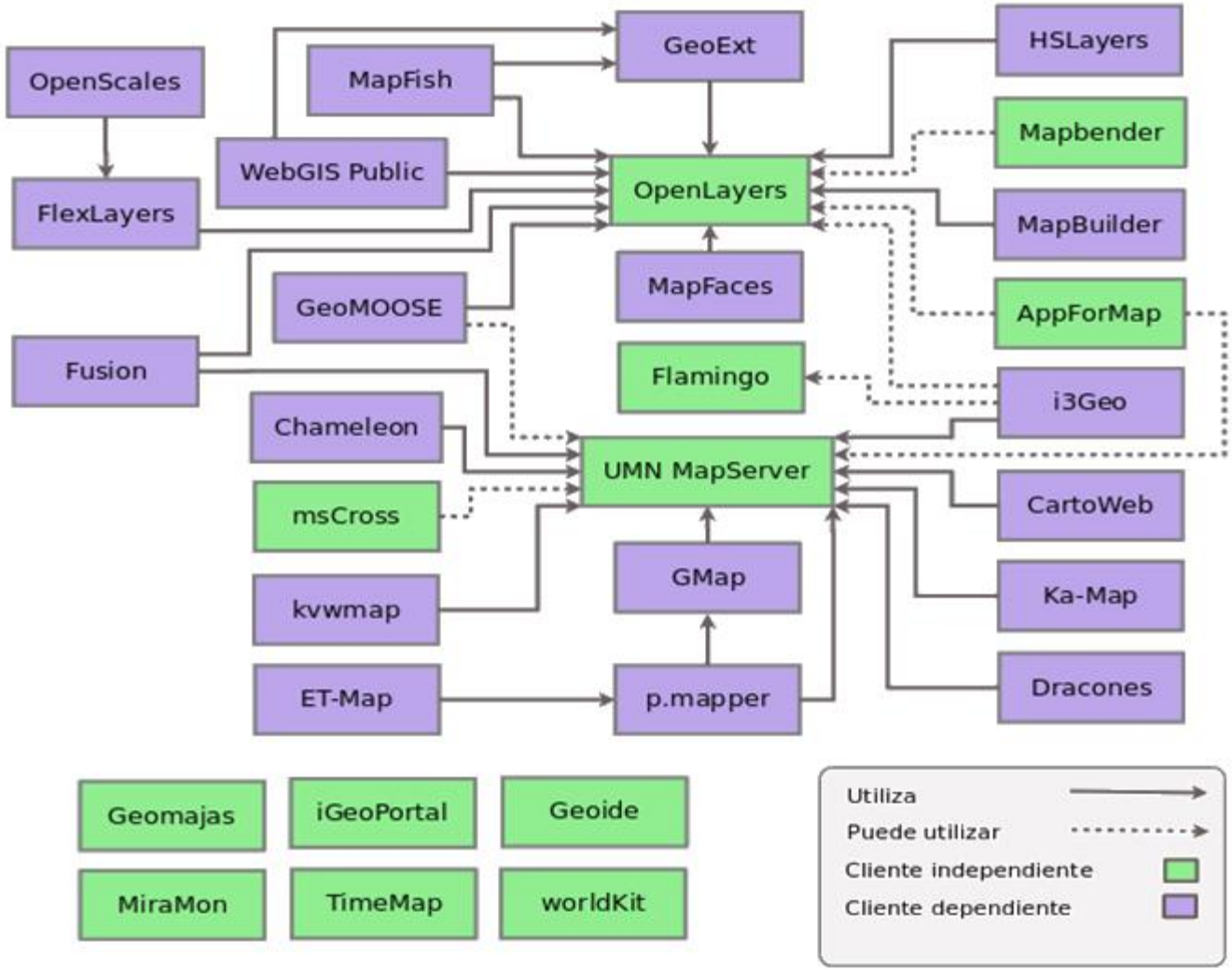


Figura 1 Dependencia entre clientes web para SIG

Existen otros proyectos que se crearon de manera independiente como Geomajas, iGeoPortal, TimeMap, entre otros. Que no utilizan para nada los paradigmas antes mencionados. Así como también están los que opcionalmente pueden usar UMN MapServer por medio de MapScript (AppForMap, GeoMOOSE y MsCross) y los que permiten de manera adicional renderizar sus mapas con OpenLayers (AppForMap, MapBender e i3Geo). (11)

A continuación se muestra una comparación respecto a la descripción general de las aplicaciones (Tabla 1) y sus características técnicas (Tabla 2). Para su mejor comprensión ventajas tendrá un color verde y desventajas en rojo. Está basada principalmente en proyectos de software libre y de código abierto manifestando en forma de ventajas y desventajas distintas características como: licencia, país de origen, lenguaje de programación, necesidad de programas servidores, entre otras.

Tabla 1 Descripción general de los clientes Web

Parámetro	CartoWeb	Chameleon	Ka-Map	MapBender	MapFish	OpenLayer
Licencia	**GNU GPL**	Chameleon License	MIT	**GNU GPL**	**GNU GPL 3**	BSD-style
País de origen	Francia, Suiza	Canadá	Canadá	Alemania	Francia, Suiza	USA
Empresa o entidad de origen	Camptocamp S.A.	DM Solutions Group	DM Solutions Group	CCGIS	Camptocamp S.A.	Metacarta
Documentación	Idiomas: Inglés, Francés Formatos: HTML; PDF; Wiki Niveles: Usuarios; Desarrolladores	Idiomas: Inglés Formatos: HTML; Wiki Niveles: Usuarios; Desarrolladores	Idiomas: Inglés, Francés, Japonés Formatos: Wiki Niveles: Usuarios; Desarrolladores	Idiomas: Alemán; Inglés; Formatos: Wiki, ODP,PDF, Video Niveles: Usuarios; Desarrolladores	Idiomas: Francés; Inglés Formatos: Blog; HTML; Wiki Niveles: Usuarios; Desarrolladores	Idiomas: Francés; Inglés; Portugués Formatos: Blogs; Wiki, HTML Niveles: Usuarios; Desarrolladores
Apoyo de OSGEO[6]	**No**	**No**	**No**	**Si (Graduado)**	**No**	**Si (Graduado)**
Observaciones	Posee una arquitectura orientada a objetos (estándar SOAP). Se obtiene todo su potencial cuando se asocia con PostgreSQL/PostGIS.	Posee una API Javascript bien documentada y consolidada. Crecimiento acelerado a través de widgets personalizados.	Requiere PHP/MapScript. Está diseñado para usar cacheo tanto como sea posible y para renderizar rápidamente mapas con tiles.	Maneja seguridad, usuarios y grupos a través de una base de datos en MySQL o PostgreSQL.	Es un framework. Está basado en Pylons. Del lado del cliente utiliza y extiende OpenLayers, GeoExt y ExtJS.	Desarrollado principalmente para visualizar GeoServicios. Soporta reproyección. Soporta SLD. Funcionalidades básicas de edición en línea.

Tabla 2 Características técnicas de los clientes Web

[6] La Open Source Geospatial Foundation ha sido creada para promover y construir el software geoespacial abierto de la más alta calidad. El objetivo de la fundación es promover el uso y desarrollo colaborativo de proyectos liderados por la comunidad.

Parámetro	CartoWeb	Chameleon	Ka-Map	MapBender	MapFish	OpenLayer
Lenguaje en el que está escrito	PHP	Javascript; PHP	Javascript; PHP	Javascript; PHP	Javascript Python	Javascript
Lenguaje de programación que admite su API	PHP	Javascript; PHP	Javascript; PHP	PHP	Java; Javascript; PHP; Python	Javascript
Geo-servicios que consume	WMS; WFS	WMS	WMS; WFS	WMS; WFS; WFS-T	WMS; WFS	WMS; WFS
Dependencia de servidor de mapas	MapServer	MapServer	MapServer	**No**	**No**	**No**
¿Requiere plugins privativos?	**No**	**No**	**No**	**No**	**No**	**No**
¿Incluye componente de metadatos?	**No**	**No**	**No**	**No**	**No**	**Sí**
Formatos de datos soportados	GeoJSON; PostGIS	PostGIS; Web Map Context	Shapefile	GeoJSON; GeoRSS; GML; Web Map Context	PostGIS	GeoRSS; GML; Google Maps; Ka-Map TMS; Microsoft Virtual Earth; Multimap; NASA WorldWind; KML; WKT; Yahoo

En esta comparación se valida que la API ka-Map está al nivel de todos los proyectos semejantes en el mundo y sobresale por su sencillez, lenguaje potente (PHP y Javascript) y rápido renderizado de mapas gracias al diseño que tiene que utiliza al máximo el cacheo. Para realizar las comparaciones se visitó cada una de las páginas oficiales de las herramientas aquí vistas.

1.5 Funcionamiento y especificidades técnicas de ka-Map

Ka-Map, esta herramienta de código abierto instalable en Linux, Unix, y Windows trabaja directamente con MapServer, aunque funciona con cualquier servidor web que soporte PHP. La plataforma MapServer se encarga del tratamiento, programación y generación de mapas en la parte del servidor. En la parte del

cliente ka-Map proporciona una API en Javascript para realizar las peticiones de datos desde el navegador, y en la parte del servidor una serie de scripts PHP que gestionan las respuestas de las extensiones de MapServer.

Su interfaz llamada ka-explorer se puede aprovechar sin mucho esfuerzo y entre sus funciones está la apariencia estilo Windows, herramientas para consulta, búsqueda e impresión. Su personalización es sencilla solo hay que modificar el fichero CSS. Tiene una serie de características importantes como la interactividad al ofrecer un paneo continuo sin recargar la página. Zoom a escalas pre-establecidas con barra de escala y leyenda. Control de capas opcional del lado del cliente que permite que estas se vuelvan visibles instantáneamente, aunque con muchas imágenes esto provoca una ralentización del navegador. (12)

Ka-Map usa los principios de AJAX, o Web 2.0, y requiere de navegadores que soporten la versión 1.5 de Javascript y manipulación DOM. Por esta razón los exploradores web muy antiguos simplemente no son soportados. Funciona principalmente en Firefox, Netscape 7, Safari, Opera 8 o superior e Internet Explorer 6 o superior.

1.6 Conclusiones

El presente capítulo recogió el resultado de todo un estudio referente a los principales conceptos asociados a la solución del problema planteado y sus relaciones. Se abordó todo lo vinculado al objeto de estudio con un análisis profundo del funcionamiento y las especificidades técnicas para el desarrollo de sistemas de información geográfica sobre la plataforma ka-Map así como la comparación entre este cliente Web para SIG y otras tecnologías similares. Luego del cual se decide usar ka-Map por ser de tecnología libre que proporciona todas las características necesarias para un correcto funcionamiento y utilidad como son un paneo continuo sin recargar la página, mapa de referencia, instalable en sistemas operativos Linux, Windows y Unix, navegación por teclado así como control de capas y búsqueda.

2

2 HERRAMIENTAS Y TECNOLOGÍAS

2.1 Introducción

En este capítulo se aborda mediante una descripción detallada las características de las herramientas y tecnologías que serán usadas para el desarrollo de la aplicación. Resaltando aquellas que se proponen para lograr la solución del problema planteado. Abordando desde servidor de mapas que se usará, pasando por las herramientas de desarrollo hasta los lenguajes que se utilizaran para la programación.

2.2 MapServer 5.0

Los servidores de mapas proveen un alto manejo de la información geográfica. Por una parte el cliente tiene acceso a los datos en su conformación original, de forma tal que es capaz de realizar consultas tan complicadas como las que haría un SIG. Un servidor de mapas actúa enviando desde un navegador, una sucesión de páginas HTML, con una cartografía relacionada entre sí en formato de imagen (puede ser, una imagen GIF o JPG). Las versiones iniciales de servidores de mapas sólo admitían efectuar funciones elementales de visualización y consultas alfanuméricas simples. En las versiones modernas es posible ejecutar funciones mucho más avanzadas.

MapServer es un entorno de desarrollo de código abierto que permite construir aplicaciones Web que utilicen mapas, es decir, datos espaciales (tanto raster como vectoriales). No se trata pues de un SIG. (13). Fue pensado para crear mapas geográficos configurables mediante un archivo llamado mapfile que permitan al usuario acceder a los contenidos de la aplicación. Se trata de un software que produce mapas en un entorno CGI (Common Gateway Interface), que es un mecanismo de comunicación que posibilita a un cliente (aplicación externa) solicitar datos de un programa ejecutado en un servidor Web.

Sus principales características son: (14)

- Ofrece salida cartográfica avanzada.
- Totalmente configurable gracias al archivo mapfile.
- Es totalmente personalizable mediante plantillas de salida.

Conceptos generales:

- Soporta el etiquetado de geometrías, y trata la colisión de etiquetas.
- Los siguientes elementos del mapa se muestran automáticamente: escala, mapa de referencia, leyenda,
- Incluye fuentes TrueType.
- Soporte para scripting y entornos de desarrollo más populares.
- Soporte para PHP, Python, Perl, Ruby, Java y C#.
- Soporte multiplataforma, Linux, Windows, Mac OS X, Solaris.
- Soporte para multitud de formatos raster y vectoriales.
- Soporte para formatos TIFF/GeoTIFF, EPPL7, y muchos otros gracias a la librería GDAL.
- Soporte para ESRI shapefiles, PostGIS, ESRI ArcSDE, Oracle Spatial, MySQL y muchos otros gracias a la librería OGR.

2.3 NetBeans IDE 6.1

Un entorno de desarrollo integrado es esencialmente un software que primeramente se instala en la máquina del programador y cuyo principal objetivo es el de facilitar el desarrollo de otro software. Normalmente consiste en un editor de código, un compilador y/o un intérprete, automatización de algunas tareas básicas y un depurador.

Netbeans es un entorno de desarrollo libre y de código abierto fundado por Sun Microsystems, pensada inicialmente para desarrolladores Java. Permite escribir, compilar, depurar y ejecutar programas. Está escrito en Java, por lo que es multiplataforma, y puede servir para cualquier otro lenguaje de programación. Soporta C/C++, Ruby y PHP. Existe además un número importante de módulos para extender el IDE Netbeans. Es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo.

Características: (15)

- Soporte JavaScript.
- Sintaxis resaltada.
- Completamiento de código y análisis de tipeo.
- Soluciones rápidas (Quick Fixes) y verificación de sintaxis.
- Refactorización.
- Soporte de estructuras spring.
- Agregado de la librería Spring Framework 2.5.
- Asistentes para configuración de archivos XML y controladores Spring Web MVC.
- Fácil creación de aplicaciones remezcladas (mashup).
- Operaciones de arrastrar y soltar dentro del entorno POJO, Servlet, JSP y servicios web RESTful para que NetBeans IDE genere todo el código para acceder a los servicios.
- Soporte de web APIs tales como Google, Facebook, Yahoo y YouTube.

2.4 Visual Paradigm

Visual Paradigm, es una herramienta UML (Lenguaje de Modelado Unificado) profesional que soporta el ciclo de vida completo del desarrollo de software. 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.

Características:

- Ofrece un entorno de creación de diagramas para UML versión 2.1.
- Generación de código.
- Generación de bases de datos.
- Transformación de diagramas de entidad.
- Relación en tablas de base de datos.
- Es disponible en múltiples plataformas.
- Diseño centrado en casos de usos y enfocado al negocio que genera un software de mayor calidad.
- Capacidad de ingeniería directa e inversa.

- Soporta aplicaciones Web.

2.5 PHP

Del inglés hypertext preprocessor (acrónimo recursivo). (16)

Un lenguaje de programación utilizado mayormente para desarrollar servicios web. PHP es un lenguaje de fácil aprendizaje, distribuido en forma gratuita, que permite interactuar con muchos sistemas de gestión de bases de datos. PHP es un lenguaje de scripting ampliamente utilizado para fines generales que es especialmente adecuado para el desarrollo web y puede ser embebido en páginas HTML.

Características:

- Es un lenguaje multiplataforma tanto para diversos Sistemas Operativos, como servidores HTTP y bases de datos.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- El código se actualiza continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.
- Capacidad de conexión con una gran cantidad de bases de datos como MySQL, PostgreSQL, Oracle, MS SQL Server, Sybase mSQL, Informix, entre otras.
- Permite aplicar técnicas de programación orientada a objetos.
- Orientado completamente al desarrollo de aplicaciones Web dinámicas con acceso a información almacenada en una base de datos.
- Permite la integración con varias bibliotecas externas, generar documentos en PDF y analizar código XML.

2.6 Javascript

JavaScript es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C. (17)

Características:

- Al ser un lenguaje interpretado, no requiere compilación. El navegador del usuario se encarga de interpretar las sentencias JavaScript contenidas en una página HTML y ejecutarlas adecuadamente.
- Es dinámico por lo que se puede cambiar totalmente el aspecto de la página a gusto del usuario.
- Es un lenguaje orientado a eventos. Cuando un usuario pincha sobre un enlace o mueve el puntero sobre una imagen se produce un evento. Mediante JavaScript se pueden desarrollar scripts que ejecuten acciones en respuesta a estos eventos.
- Permite la programación orientado a objetos. El modelo de objetos de JavaScript está reducido y simplificado, pero incluye los elementos necesarios para que los scripts puedan acceder a la información de una página y puedan actuar sobre la interfaz del navegador.
- Es soportado por la gran mayoría de los navegadores como Internet Explorer, Netscape, Opera, Mozilla Firefox, entre otros.
- Soporta cuatro tipos de datos, pero no es necesario declarar el tipo de las variables, argumentos de funciones ni valores de retorno de las funciones.

2.7 AJAX

AJAX, por sus siglas en inglés Asynchronous JavaScript And XML (JavaScript y XML asíncronos), es un conjunto de tecnologías aplicadas de forma concreta, que permite una técnica de desarrollo Web para crear aplicaciones interactivas. Éstas se ejecutan en el navegador del usuario, y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la velocidad de interacción en la misma.

Características:

- Las páginas son más responsivas: tiene la capacidad de obtener información para actualizar un mismo documento sin que el navegador modifique su estado; de esta forma la página no tiene que recargarse cada vez que se necesite un cambio. Esto aumenta la interactividad, la velocidad (cuantificable en al menos el tiempo necesario para refrescar una página) y la usabilidad.
- Maximización del uso de las vías de comunicación llevan a mayor eficiencia del sistema: la técnica permite un mayor uso de utilizar información por demanda sin que el usuario pague un costo por el

mismo (no se tiene que refrescar la página) por lo que en realidad al buscador sólo tiene que enviarse la información que el usuario necesita.

- Posee gran rapidez en las operaciones que realiza.
- Se ahorra tiempo de procesamiento en el servidor Web ya que una gran parte del procesamiento se realiza en el lado del cliente.
- Las aplicaciones son más interactivas, responden a las interacciones del usuario más rápidamente, al estilo aplicaciones de escritorio.
- Actualiza porciones de la página en vez de la página completa.

2.8 Metodologías

Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información. Una metodología está formada por fases, cada una de las cuales se puede dividir en sub-fases, que guiarán a los desarrolladores de sistemas a elegir las técnicas más apropiadas en cada momento del proyecto y también a planificarlo, gestionarlo, controlarlo y evaluarlo.

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

2.8.1 Metodologías pesadas

Las metodologías pesadas o tradicionales como también se les conoce están basadas en normas provenientes de estándares seguidos por el entorno de desarrollo, poseen cierta resistencia a los cambios a la vez que son impuestas externamente. Se caracterizan por un proceso mucho más controlado, con numerosas políticas y normas. Existe un contrato prefijado y el cliente interactúa con el equipo de desarrollo mediante reuniones. Los equipos tienen muchos integrantes y generalmente están distribuidos. La arquitectura de software es esencial en el desarrollo de la aplicación y se expresa mediante modelos. Ejemplos de este tipo de filosofía para desarrollar software está Rational Unified Procces (RUP), Microsoft Solution Framework (MSF), Iconix y Win-Win Spiral Model.

Rational Unified Procces (RUP)

El Proceso Unificado de Desarrollo (RUP) es una metodología para el desarrollo de software orientado a objetos. Es una disciplina de trabajo, con el fin de conseguir un software más eficiente, definido como un conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software.

El Proceso Unificado está basado en componentes, lo cual quiere decir que el sistema de software en construcción está formado por componentes de software interconectados a través de interfaces bien definidas. Es un proceso que define claramente quién, cuándo, cómo y qué debe hacerse, y como su enfoque está basado en modelos, utiliza un lenguaje bien definido para tal fin, el UML.

RUP posee tres características esenciales está dirigido por los Casos de Uso: que orientan el proyecto a la importancia para el usuario y lo que este quiere, está centrado en la arquitectura: que relaciona la toma de decisiones que indican cómo tiene que ser construido el sistema y en qué orden, y es iterativo e incremental: donde divide el proyecto en mini proyectos donde los casos de uso y la arquitectura cumplen sus objetivos de manera más depurada. (18)

Características específicas de RUP:

- Dirigido por casos de uso.
- Centrado en la arquitectura.
- Iterativo e incremental.
- La metodología RUP está dividida en cuatro fases de desarrollo del producto:
- Inicio: El objetivo en esta etapa es determinar la visión del proyecto.
- Elaboración: En esta etapa el objetivo es determinar la arquitectura óptima.
- Construcción: En esta etapa el objetivo es llevar a obtener la capacidad operacional inicial, o sea crear el producto.
- Transición: El objetivo es llegar a obtener el producto listo del proyecto.

RUP describe como obtener, organizar y documentar la funcionalidad y restricciones requeridas. Así como también a documentar y monitorear las alternativas y decisiones. Las nociones de Casos de Uso y de Escenarios utilizadas en RUP han demostrado ser una manera excelente de capturar los requisitos funcionales y asegurarse que direccionan el diseño, la implementación y la prueba del sistema, logrando así que el sistema satisfaga las necesidades del usuario.

Modelización del software visualmente:

RUP muestra cómo hacer la modelación de un software visualmente para capturar la estructura y comportamiento de arquitecturas y componentes. Las abstracciones visuales ayudan a comunicar diferentes aspectos del software; comprender los requisitos, ver como los elementos del sistema se relacionan entre sí, mantener la consistencia entre diseño e implementación y promover una comunicación precisa. El estándar UML, creado por Rational Software, es el cimiento para una modelización visual exitosa.

Verifica la calidad del software:

Es necesario evaluar la calidad de un sistema respecto de sus requisitos de funcionalidad, confiabilidad y rendimiento. La actividad fundamental es la prueba (testing), que permite encontrar las fallas antes de la puesta en producción. RUP asiste en el planeamiento, diseño, implementación, ejecución y evaluación de todos estos tipos de testing.

El aseguramiento de la calidad se construye dentro del proceso, en todas las actividades, involucrando a todos los participantes, utilizando medidas y criterios objetivos, permitiendo así detectar e identificar los defectos en forma temprana.

Controla los cambios al software:

La capacidad de administrar los cambios es esencial en ambientes en los cuales el cambio es inevitable. RUP describe como controlar, rastrear y monitorear los cambios para permitir un desarrollo iterativo exitoso. Es también una guía para establecer espacios de trabajo seguros para cada desarrollador, suministrando el aislamiento de los cambios hechos en otros espacios de trabajo y controlando los

cambios de todos los elementos de software (modelos, código, documentos, etc.). Describe como automatizar la integración y administrar la conformación de versiones del software a publicarse.

2.9 ¿Por qué RUP?

Este método es el que más se adecúa a las necesidades del proyecto porque a pesar de este no ser considerado de gran envergadura si requiere de mucha documentación. El desarrollo del prototipo puede ser altamente controlado y se define claramente quién, cuándo, cómo y qué debe hacerse. Los Casos de Uso utilizados en RUP han demostrado ser una manera excelente de capturar los requisitos funcionales y asegurarse que direccionan el diseño, la implementación y la prueba del sistema, logrando así que el sistema satisfaga las necesidades del usuario. Permite detectar e identificar los errores prematuramente ya que el aseguramiento de la calidad se construye dentro del proceso utilizando medidas y criterios objetivos.

Se centra en la arquitectura la cual muestra la visión común del sistema completo, que describe los elementos del modelo que son importantes para la construcción del software, creándose de esta manera los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo. RUP propone que se desarrolle el software mediante iteraciones, comenzando por los casos de uso relevantes desde el punto de vista para la arquitectura y además que cada fase se desarrolle en iteraciones, donde en cada iteración se involucran actividades de todos los flujos de trabajo, aunque se desarrollan fundamentalmente algunos flujos más que otros.

2.10 Conclusiones

En este capítulo se identificaron y describieron las principales herramientas, lenguajes y metodologías propuestas para desarrollar la aplicación. Esta decisión está basada en los requisitos del proyecto sobre todo y la facilidad que brindan al desarrollador. Se va a trabajar sobre la plataforma ka-Map, usando MapServer como servidor de mapas, la metodología de desarrollo de software RUP, Visual Paradigm como herramienta case, como lenguaje de programación PHP, Javascript y Ajax y como IDE de desarrollo NetBeans. Con esta selección se proporciona una solución eficiente al problema planteado.

3

3 PRESENTACIÓN DE LA SOLUCIÓN

3.1 Introducción

En este capítulo se hace una representación de la solución propuesta a través del modelo de dominio representando los principales conceptos asociados al entorno donde se emplazará el sistema a desarrollar. Se detallan los requisitos funcionales y no funcionales que deberá cumplir la solución propuesta, casos de usos que se generan a partir de los requisitos funcionales y una explicación de cada uno de ellos a través de las descripciones textuales, se presentan los diagramas de casos de uso del sistema y se describen los actores.

3.2 Modelo de dominio

Como la propuesta de solución puede ser personalizado por cualquier cliente o empresa interesada, resultó difícil encontrar procesos de negocio bien estructurados que permitieran realizar un modelo completo de dicho negocio; por lo que se tomó la decisión de realizar un modelo de dominio.

En (18) se plantea que el modelo de dominio es aquel que reúne los objetos más importantes del sistema a implementar. Estos objetos simbolizan los sucesos que ocurren en el entorno en el que trabaja el sistema. Este modelo permite ayudar a los usuarios, clientes y desarrolladores a utilizar un vocabulario común para poder entender el contexto en que se sitúa el sistema.

Las clases del dominio aparecen en tres formas típicas: (1) Objetos del negocio que representan cosas que se manipulan en el negocio. (2) Objetos del mundo real y conceptos de los que el sistema debe hacer un seguimiento. (3) Sucesos que ocurrirán o han ocurrido.

El Modelo de Dominio se aplica cuando:

- Los flujos de información no están claros (múltiples orígenes, sólo eventos, sucesos).
- Imposibilidad de determinar subsistemas (exceso de interconexiones).
- Solapamiento de responsabilidades.
- Múltiples responsabilidades.
- Difícil establecimiento de reglas de funcionamiento.

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.

3.3 Diagrama de clases del Modelo de Dominio

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. En la Figura 2 se muestra el diagrama de clases del Modelo del Dominio correspondiente a la aplicación.

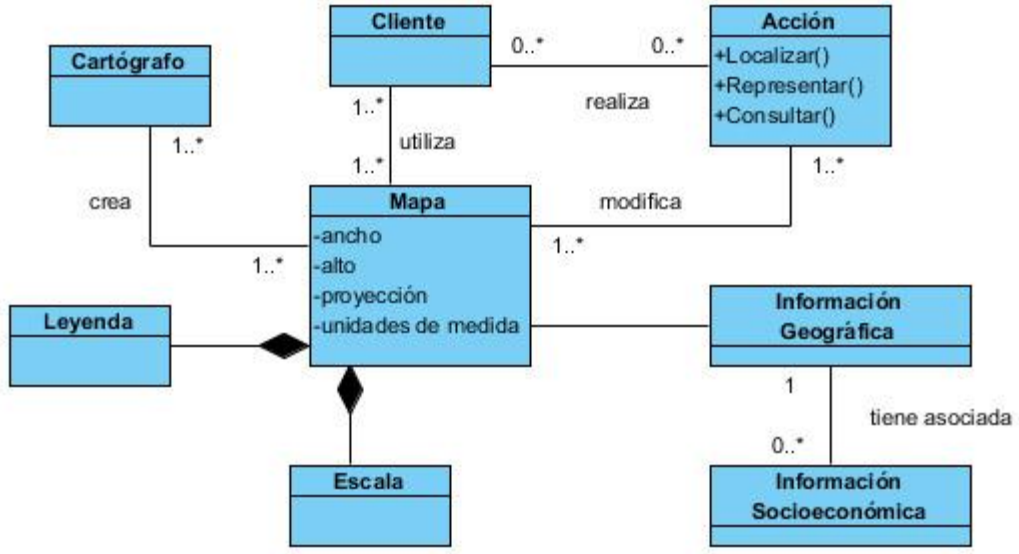


Figura 2 Diagrama de clases del Modelo de Dominio

3.4 Requisitos del sistema

El proceso de desarrollo de una aplicación está regido por varias actividades que han de ser llevadas a cabo por una guía metodológica, garantizando la calidad del producto y la satisfacción de los clientes y desarrolladores. La primera etapa dentro del proceso de la Ingeniería de Software, es la Ingeniería de Requisitos, que cumple un papel primordial en la realización de aplicaciones, ya que se enfoca en un área fundamental: la definición de lo que se desea producir. Los requisitos de software son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema de desarrollo y las restricciones en las que debe operar.

Los requisitos del sistema pueden dividirse en requisitos funcionales y requisitos no funcionales.

Los requisitos funcionales: Definen las funciones que el sistema será capaz de realizar. Describe las transformaciones que el sistema realiza sobre las entradas para producir salidas.

Los requisitos no funcionales: Tienen que ver con características que de una forma u otra pueden limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema), seguridad, portabilidad, estándares, etc.

3.4.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. Para el componente propuesto se definen los siguientes requisitos funcionales (RF):

El sistema debe cumplir:

RF-1 Realizar Zoom Más.

Consiste en aumentar la vista del mapa.

RF-2 Realizar Zoom Menos.

Consiste en disminuir la vista del mapa

RF-3 Desplazar la vista del mapa.

Consiste en que el usuario con el ratón pueda desplazar el mapa en cualquier dirección.

RF-4 Mostrar/Ocultar capa.

El sistema debe permitir que el usuario pueda seleccionar cualquier capa del mapa para ser visualizado en el mismo.

RF-5 Visualizar coordenadas.

El sistema debe permitir que el usuario pueda visualizar según la posición puntual en la que se encuentra en el mapa, las coordenadas (X, Y) correspondientes a ese punto.

RF-6 Exportar mapa.

El sistema debe permitir que el usuario pueda exportar un mapa o vista de éste a diferentes formatos (JPEG, PNG, GIF, PDF y TIFF).

RF-7 Localizar por coordenadas.

El sistema debe permitir que el usuario pueda localizar cualquier objeto del mapa de acuerdo con las coordenadas (X, Y) proporcionadas.

RF-8 Medir distancia.

El sistema debe permitir que el usuario pueda realizar trazos por el mapa y visualice la distancia total acumulada.

RF-9 Calcular área.

El sistema debe permitir que el usuario pueda realizar trazados formando una región determinada para poder visualizar el cálculo del área.

3.4.2 Requisitos no funcionales

Usabilidad: La aplicación podrá ser usada por personas con conocimientos elementales en el uso de computadoras.

Apariencia: Debe brindar una interfaz amigable, interactiva, intuitiva y de fácil comprensión para el usuario, facilitando en todo momento la interacción de este con el sistema.

Disponibilidad: El sistema debe estar disponible las veinticuatro horas del día.

Portabilidad: La aplicación debe ser multiplataforma, debe correr tanto en la plataforma Windows como en Linux.

Hardware: Para las PC cliente será necesario que tengan tarjeta de red, al menos 512 MB de RAM, procesador a 1 GHz como mínimo. El servidor requerirá tarjeta de red, al menos 2 GB de memoria RAM, al menos 10 GB de espacio en disco, procesador con una velocidad mínima de 3 GHz

Software: La aplicación se visualizará en cualquier navegador que soporte los estándares de la Consorcio de Estándares Web (W3C).

3.5 Descripción del sistema

Luego de haber realizado la descripción de los requisitos, quienes componen las funcionalidades esenciales del sistema, se pasa a la descripción del Modelo de Casos de Uso del Sistema. Dicho modelo está formado por actores, casos de uso y las relaciones que existen en ellos. Describiendo el sistema en cuanto a su utilización mediante diagramas utilizando el lenguaje de modelado.

3.5.1 Definición de los Actores del Sistema

Un actor es el papel que juega un usuario, en el cual este puede manejar información o solamente ser un contenedor pasivo de información. Representa un ser humano, un software o una máquina que interactué con el sistema. (18)

Tabla 3 Definición de los Actores

Actor	Descripción
Usuario	Cualquier persona que desee utilizar las funcionalidades de la aplicación.

3.5.2 Definición de los Casos de Uso del Sistema

Un caso de uso constituye una técnica utilizada para describir el comportamiento del sistema, donde se define la secuencia de acciones que obtienen resultados de valor para un actor del sistema. En el sistema se definieron los siguientes Casos de Uso:

Tabla 4 Definición de los Casos de Uso

Casos de Uso del Sistema	Prioridad
Modificar vista de mapa	Crítico
Realizar selección de capas	Secundario
Visualizar elementos del mapa	Secundario
Exportar mapa	Crítico
Localizar coordenadas	Crítico

Medir distancia	Crítico
Calcular área	Crítico

3.5.3 Diagrama de Casos de Uso del Sistema

Los diagramas de caso de uso del sistema muestran la relación de los actores y los casos de uso en un sistema. Se utilizan para ilustrar los requisitos de la aplicación ya que muestran cómo reacciona el sistema ante eventos que ocurran en el mismo.

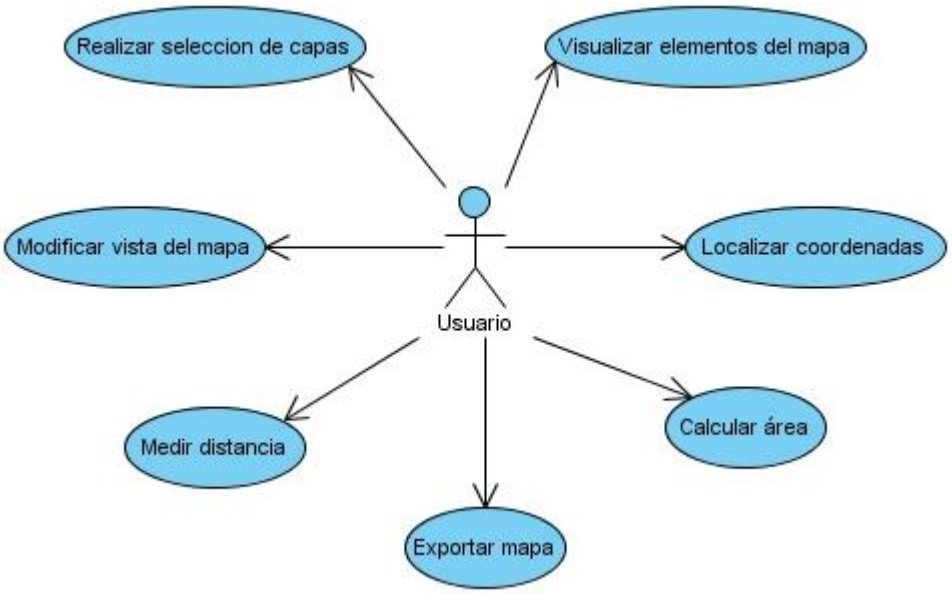


Figura 3 Diagrama de Casos de Uso

3.5.4 Descripción de los Casos de Uso del Sistema (CUS)

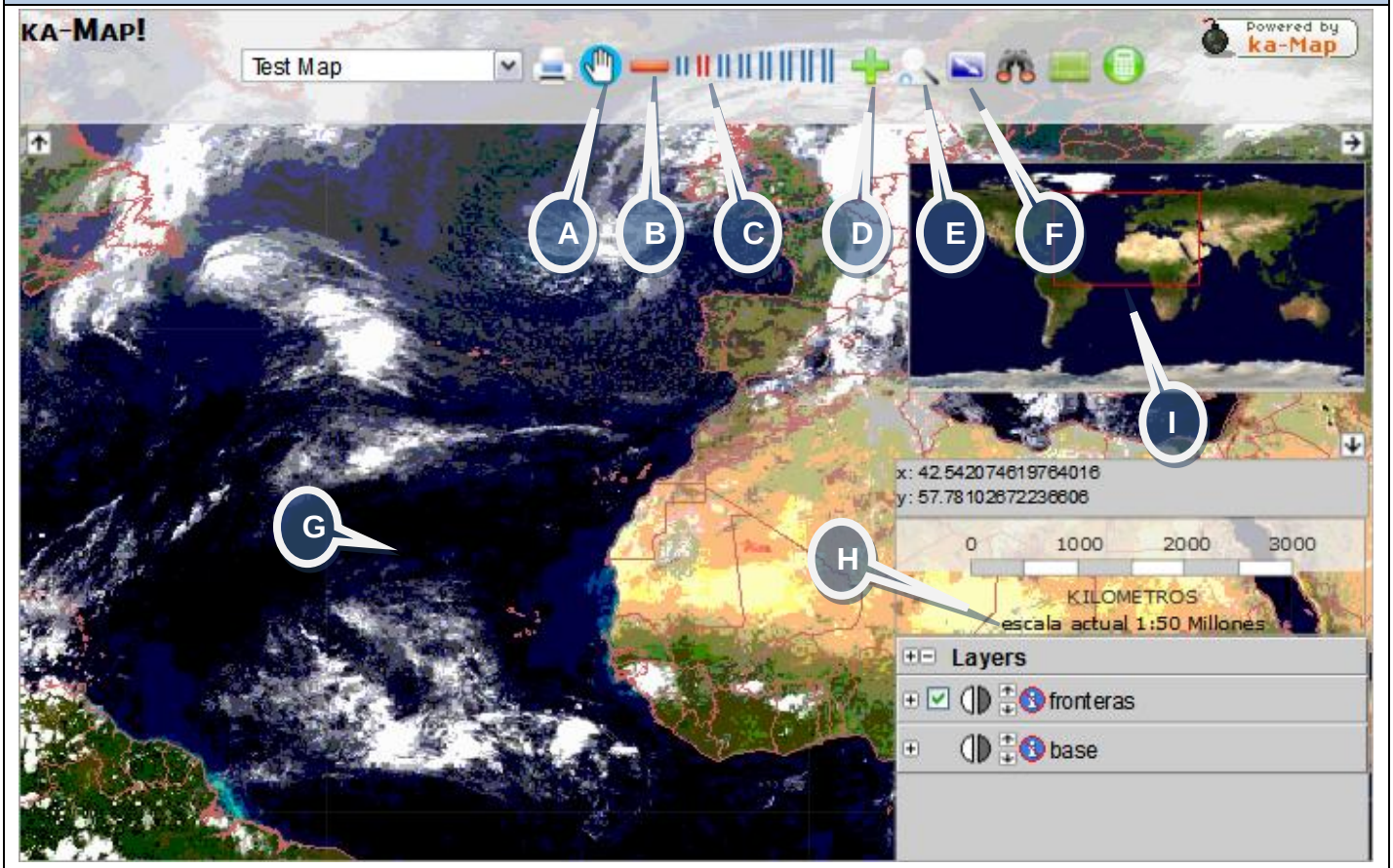
A continuación se describen textualmente cada uno de los casos de uso identificados. Estos presentan características individuales que hacen necesario dicha representación para un mejor entendimiento.

Tabla 5 Descripción textual del CUS Modificar vista de mapa

Caso de Uso Modificar vista de mapa			
Actores	Usuario		
Propósito	Este caso de uso se lleva a cabo con el objetivo de cambiar la forma de visualizarse el mapa.		
Descripción	El caso de uso se inicia cuando el usuario desea mover, ampliar o recentrar el mapa y termina cuando el sistema muestra el mapa como el usuario desea.		
Precondiciones			
Referencias	RF-1, RF-2, RF-3		
Flujo de eventos			
	Acción del actor		Respuesta del sistema
1	El usuario selecciona una de las siguientes opciones: Paneo (A) Zoom Menos (B) Zoom Gradual (C) Zoom Más (D) Ampliar Zona (E)	2	El sistema realiza la acción según la selección del usuario: Si selecciona la opción a) ver sección “Paneo”. Si selecciona la opción b) ver sección “Zoom Menos”. Si selecciona la opción c) ver sección “Zoom Gradual”. Si selecciona la opción e) ver sección “Ampliar

	Zoom Completo (F)		Zona”. Si selecciona la opción f) ver sección “Zoom Completo”.

Prototipo de interfaz



Sección Paneo

	Acción del actor		Respuesta del sistema

		3	El sistema cambia el puntero del ratón correspondiente a la acción de “mover”.
4	El usuario presiona el botón izquierdo del ratón sobre la vista del mapa (G).	5	El sistema captura la posición (coordenadas) donde se realizó el evento.
6	El usuario mantiene presionado el botón izquierdo del ratón y arrastra hacia la dirección deseada.	7	El sistema mueve el mapa junto con el movimiento del usuario, sin recargar la página.
8	El usuario libera el botón izquierdo del ratón.	9	El sistema fija el mapa en la posición donde se realizó el evento.
Sección Zoom Menos			
		3	El sistema aumenta la vista del mapa (G), actualiza (aumenta) la escala (H), actualiza el mapa de referencia (I), actualiza el indicador de “Zoom Gradual” (C).
Sección Zoom Gradual			
3	El usuario presiona el botón izquierdo del ratón sobre una de las barras del indicador de “Zoom Gradual” (C).	4	El sistema aumenta o disminuye la vista del mapa (G) según la escala que representa la barra seleccionada.
Sección Zoom Más			
		3	El sistema disminuye el área de la vista del mapa (G), actualiza (reduce) la escala (H), actualiza el

			mapa de referencia (I), actualiza el indicador de "Zoom Gradual" (C).
Sección Ampliar Zona			
		3	El sistema cambia el puntero del ratón correspondiente a la acción de "Ampliar Zona".
4	El usuario ejecuta un clic sobre un punto en la vista del mapa (G).	5	El sistema disminuye el área de la vista del mapa (G) con el punto seleccionado por el usuario como centro, actualiza (reduce) la escala (H), actualiza el mapa de referencia (I), actualiza el indicador de "Zoom Gradual" (C).
Flujo alterno			
4	El usuario presiona el puntero del ratón sobre la vista del mapa (G), y arrastra sobre la zona que desee ampliar.	5	El sistema dibuja una figura de forma rectangular mientras el usuario traza la zona.
6	El usuario suelta el botón del ratón.	7	El sistema disminuye el área de la vista del mapa (G) con la zona seleccionada por el usuario como centro, actualiza (reduce) la escala (H), actualiza el mapa de referencia (I), actualiza el indicador de "Zoom Gradual" (C).
Sección Zoom Completo			

		3	El sistema regresa el mapa a su escala original si se han producido cambios en la vista (G). Actualiza la escala (H), el mapa de referencia (I), y el indicador de “Zoom Gradual” (C) a sus valores por defecto.

Tabla 6 Descripción textual de CUS Realizar selección de capas

Caso de Uso Realizar selección de capas			
Actores	Usuario		
Propósito	Este caso de uso se ejecuta con el propósito de cambiar las capas seleccionables y visibles de un mapa.		
Descripción	El caso de uso se inicializa cuando el usuario desea seleccionar o deseleccionar alguna capa y termina cuando el sistema muestra las capas seleccionadas por el usuario.		
Precondiciones	Tiene que existir al menos una capa que no sea capa base en el archivo de configuración mapfile.		
Referencias	RF-4		
Flujo de eventos			
	Acción del actor		Respuesta del sistema
1	El usuario selecciona la capa que desea hacer visible. Dando clic en la casilla (A)	2	El sistema marca la casilla, y muestra al instante

	habilitado para el efecto.		la capa (B) elegida por el usuario.
Flujo alterno			
1	El usuario selecciona la capa que desea hacer invisible. Dando clic en la casilla (A) habilitado para el efecto.	2	El sistema desmarca la casilla, y oculta al instante la capa elegida por el usuario.
Prototipo de interfaz			

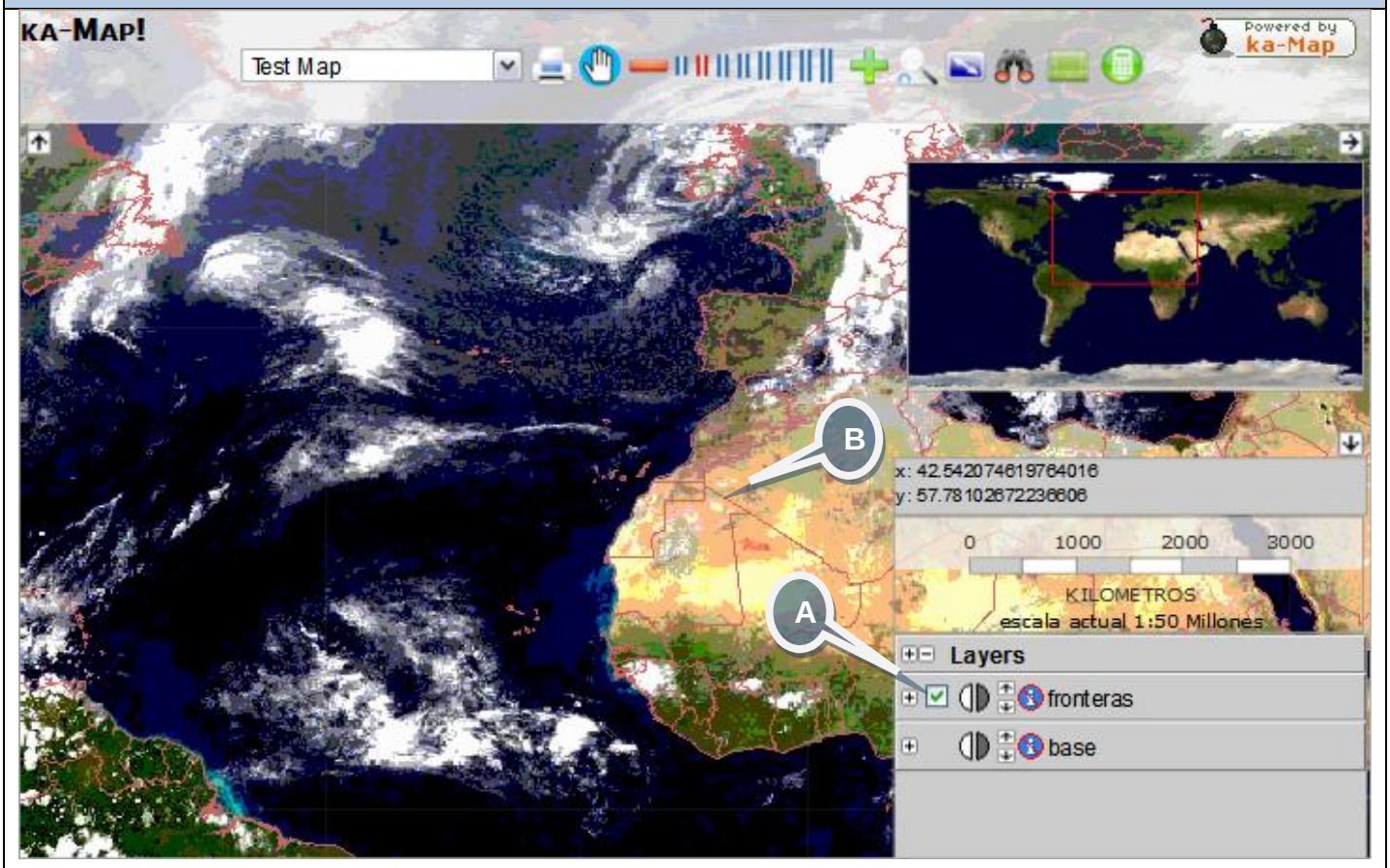


Tabla 7 Descripción textual de CUS Visualizar elementos del mapa

Caso de Uso Visualizar elementos del mapa			
Actores	Usuario		
Propósito	Este caso de uso se lleva a cabo con el objetivo de visualizar los elementos del mapa, como coordenadas, mapa de referencia y escala gráfica.		
Descripción	El sistema debe tener un mapa de referencia, así como siempre visible la escala gráfica al actor hacer Zoom y mostrará las coordenadas de donde se encuentre ubicado el cursor.		
Precondiciones			
Referencias	RF-5		
Flujo de eventos			
	Acción del actor		Respuesta del sistema
1	El usuario quiere visualizar una de las siguientes opciones: Coordenadas (A) Escala (B) Mapa de referencia (C)	2	El sistema muestra: Si desea ver opción a) ver sección “Coordenadas”. Si desea ver opción b) ver sección “Escala”. Si desea ver opción c) ver sección “Mapa de referencia”.
Sección Coordenadas			

3	El usuario mueve el puntero sobre la vista del mapa (D).	4	El sistema muestra las coordenadas (X, Y) que corresponde al punto donde está ubicado el puntero (A).
Sección Escala			
3	El usuario realiza cualquier operación de Zoom Menos (E) sobre la vista mapa (D).	4	El sistema muestra los cambios efectuados por el usuario actualizando la escala (B).
Flujo alterno			
3	El usuario realiza cualquier operación de Zoom Gradual (G) sobre la vista mapa (D).	4	El sistema muestra los cambios efectuados por el usuario actualizando la escala (B).
5	El usuario realiza cualquier operación de Zoom Más (H) sobre la vista mapa (D).	6	El sistema muestra los cambios efectuados por el usuario actualizando la escala (B).
7	El usuario realiza cualquier operación de Ampliar Zona (E) sobre la vista mapa (D).	8	El sistema muestra los cambios efectuados por el usuario actualizando la escala (B).
9	El usuario realiza cualquier operación de Zoom Completo (E) sobre la vista mapa (D).	10	El sistema muestra los cambios efectuados por el usuario actualizando la escala (B).
Sección Mapa de referencia			
3	El usuario realiza cualquier operación de Paneo (F) sobre la vista mapa (D).	4	El sistema muestra los cambios efectuados por el usuario actualizando el mapa de referencia (C).

Flujo alterno			
3	El usuario realiza cualquier operación de Zoom Menos (E) sobre la vista mapa (D).	4	El sistema muestra los cambios efectuados por el usuario actualizando el mapa de referencia (C).
5	El usuario realiza operación de Zoom Gradual (G) sobre la vista mapa (D).	6	El sistema muestra los cambios efectuados por el usuario actualizando el mapa de referencia (C).
7	El usuario realiza cualquier operación de Zoom Más (H) sobre la vista mapa (D).	8	El sistema muestra los cambios efectuados por el usuario actualizando el mapa de referencia (C).
9	El usuario realiza cualquier operación de Ampliar Zona (E) sobre la vista mapa (D).	10	El sistema muestra los cambios efectuados por el usuario actualizando el mapa de referencia (C).
11	El usuario realiza cualquier operación de Zoom Completo (E) sobre la vista mapa (D).	12	El sistema muestra los cambios efectuados por el usuario actualizando el mapa de referencia (C).
Prototipo de interfaz			

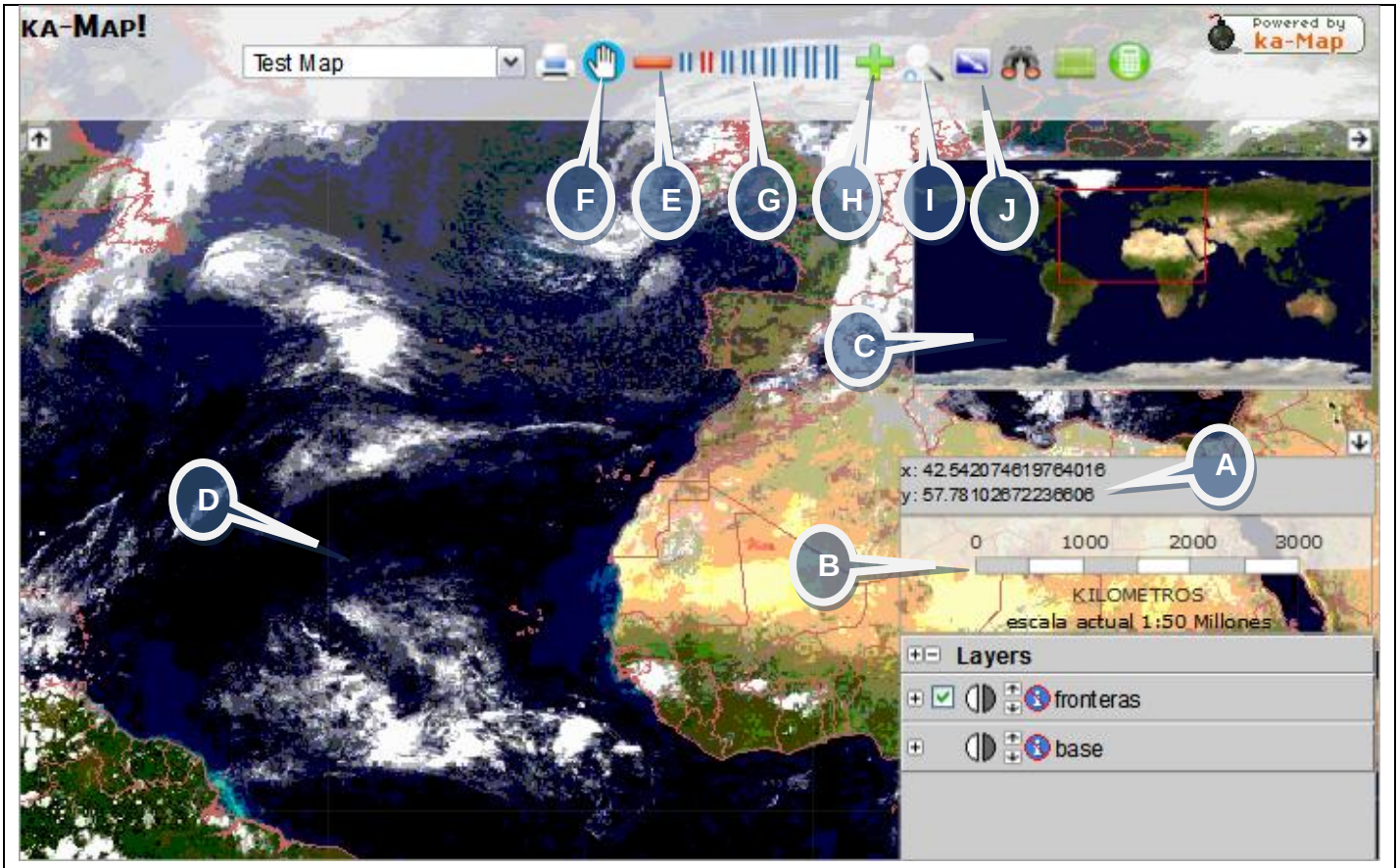


Tabla 8 Descripción textual de CUS Exportar mapa

Caso de Uso Exportar mapa	
Actores	Usuario
Propósito	El caso de uso se lleva a cabo con el objetivo de exportar el mapa.
Descripción	El caso de uso se inicia cuando el actor desea exportar el mapa visualizado

<table>
<tr><td colspan="2"></td><td colspan="2">en la pantalla y termina cuando el sistema envía el fichero en el formato deseado.</td></tr>
<tr><td colspan="2">Precondiciones</td><td colspan="2"></td></tr>
<tr><td colspan="2">Referencias</td><td colspan="2">RF-6</td></tr>
<tr><td colspan="4">Flujo de eventos</td></tr>
<tr><td></td><td>Acción del actor</td><td></td><td>Respuesta del sistema</td></tr>
<tr><td>1</td><td>El usuario selecciona la opción exportar mapa(A).</td><td>2</td><td>El sistema muestra una ventana con los formatos disponibles (B).</td></tr>
<tr><td>3</td><td>El usuario escoge en que formato desea ver el mapa y presiona el botón izquierdo del ratón</td><td>4</td><td>El sistema crea el archivo y lo envía a la aplicación por defecto para el formato deseado que tenga el usuario.</td></tr>
<tr><td colspan="4">Flujo alterno</td></tr>
<tr><td>3</td><td>El usuario escoge en que formato desea ver el mapa y presiona el botón derecho del ratón.</td><td>4</td><td>El sistema permite que el usuario pueda descargar el mapa.</td></tr>
<tr><td colspan="4">Prototipo de interfaz</td></tr>
</table>

Tabla 9 Descripción textual de CUS Localizar por coordenadas

Caso de Uso Localizar por coordenadas	
Actores	Usuario
Propósito	Con este caso de uso se pretende que el usuario pueda localizar cualquier objeto del mapa de acuerdo con las coordenadas (X, Y) proporcionadas.
Descripción	El caso de uso inicia cuando el usuario introduce un par de coordenadas y termina cuando el sistema muestra el punto en el cual están ubicadas en el mapa.

Precondiciones			
Referencias	RF-7		
Flujo de eventos			
	Acción del actor		Respuesta del sistema
1	El usuario selecciona Buscar punto en el mapa.	2	El sistema muestra dos ventanas en las cuales se introducen el valor X (A) y el valor Y (B) de las coordenadas del punto.
3	El usuario introduce los valores	4	El sistema verifica que estén dentro de la extensión del mapa.
		5	Si las coordenadas son válidas se muestra el punto en el mapa con una imagen representativa (C).
Flujo alterno			
		5	Si las coordenadas no se encuentran dentro de la extensión del mapa el sistema muestra un mensaje de alerta al usuario donde le dice las dimensiones del mapa (D).
Prototipo de interfaz			

Tabla 10 Descripción textual de CUS Medir distancia

Caso de Uso Medir distancia	
Actores	Usuario
Propósito	Con este caso de uso se quiere que el usuario pueda realizar trazos por el mapa y pueda visualizar la distancia total acumulada.
Descripción	El caso de uso inicia cuando el usuario desea medir la distancia entre dos puntos y entonces recorre el mapa con el cursor terminando con la presentación de la distancia recorrida por el sistema.
Precondiciones	

Referencias	RF-8		
Flujo de eventos			
	Acción del actor		Respuesta del sistema
1	El usuario selecciona Medir distancia (C).	3	El sistema a partir del segundo clic muestra la distancia entre los puntos en total (B) en la barra de herramientas.
2	El usuario con el ratón traza una ruta que inicia con el primer clic y puede hacer varios más por el mapa (A).		
Prototipo de interfaz			

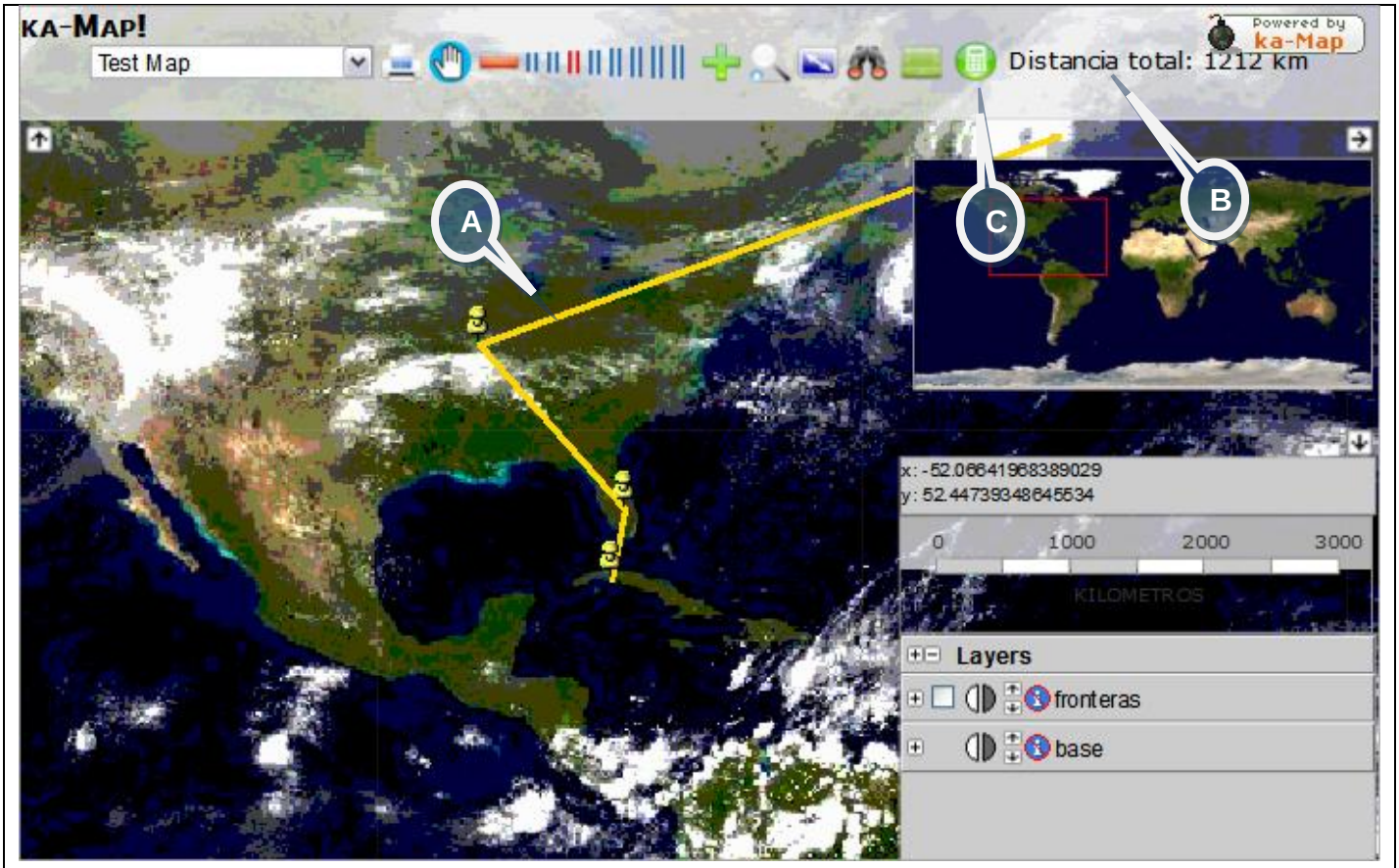
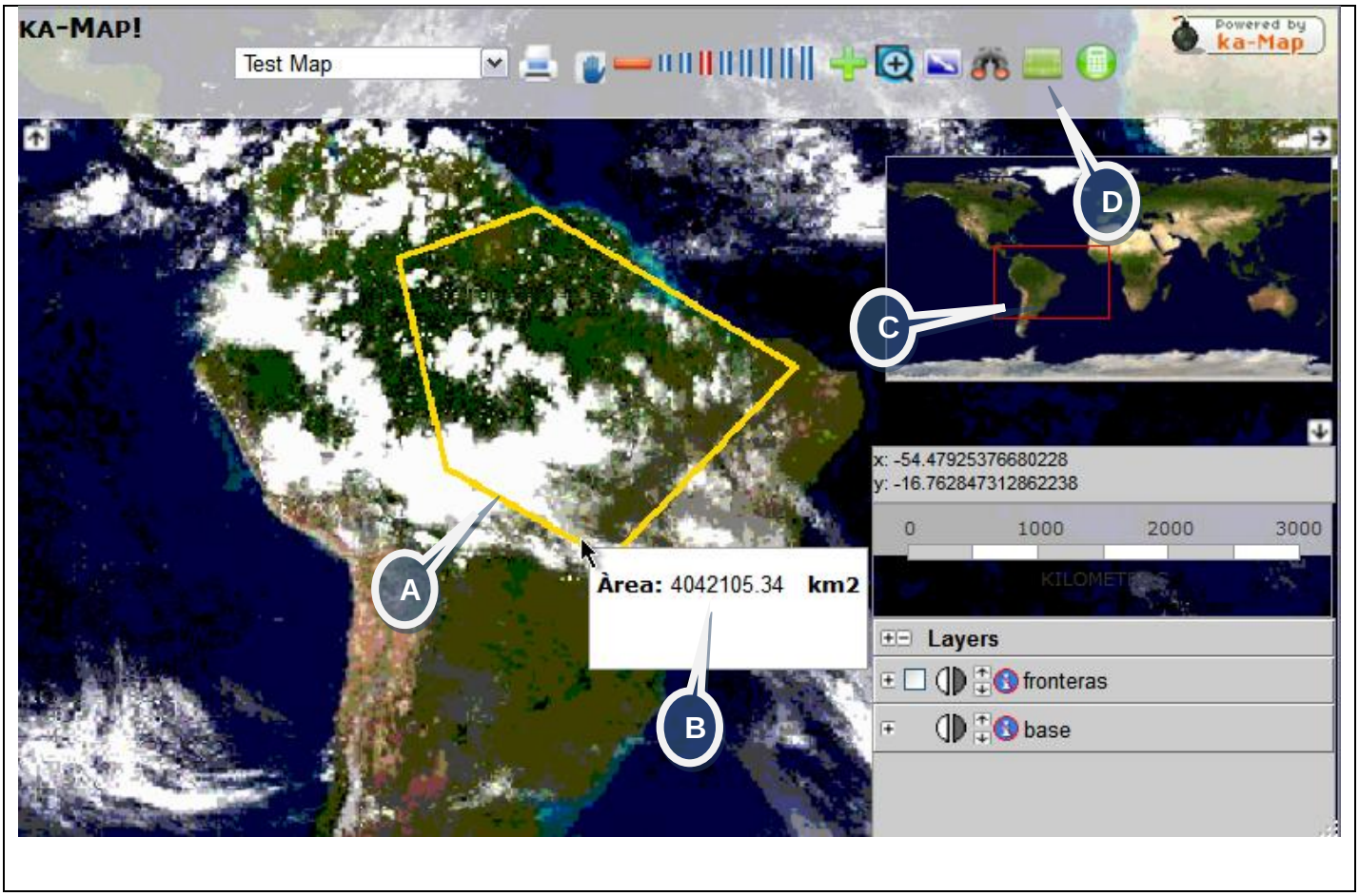


Tabla 11 Descripción textual de CUS Calcular área

Caso de Uso Calcular área	
Actores	Usuario
Propósito	Este caso de uso se lleva a cabo con el objetivo de calcular el área de una región seleccionada por el usuario.
Descripción	El caso de uso inicia cuando el actor realiza un trazado de un área la cual el sistema

<table>
<tr><td></td><td colspan="3">calcula y muestra un valor.</td></tr>
<tr><td>Precondiciones</td><td colspan="3"></td></tr>
<tr><td>Referencias</td><td colspan="3">RF-9</td></tr>
<tr><td colspan="4">Flujo de eventos</td></tr>
<tr><td></td><td>Acción del actor</td><td></td><td>Respuesta del sistema</td></tr>
<tr><td>1</td><td>El usuario selecciona Calcular área (D).</td><td></td><td></td></tr>
<tr><td>2</td><td>El usuario traza un polígono con el cursor donde el punto inicial siempre es igual al punto final (A).</td><td>3</td><td>El sistema muestra el área del polígono seleccionado en cuadro de texto (B).</td></tr>
<tr><td></td><td></td><td>4</td><td>Para deseleccionar el área el usuario presiona clic derecho en el ratón.</td></tr>
<tr><td colspan="4">Flujo alterno</td></tr>
<tr><td></td><td></td><td>3</td><td>Si al polígono no se le puede calcular el área el sistema muestra un mensaje de error en el mismo cuadro de texto.</td></tr>
<tr><td colspan="4">Prototipo de Interfaz</td></tr>
</table>



3.6 Conclusiones

En este capítulo se definieron importantes aspectos correspondientes al desarrollo de la solución propuesta a partir de un análisis de las funcionalidades que debe poseer la aplicación. Se identificaron los actores así como las acciones que realizan con los Casos de Uso del Sistema con los que interactúan. Se aclararon las características del sistema que se desea construir y se crearon las bases para las restantes fases del proceso de diseño e implementación del mismo.

4

4 CONSTRUCCIÓN DE LA SOLUCIÓN

4.1 Introducción

En este capítulo se lleva a cabo la construcción de la solución propuesta en los capítulos anteriores. Se describe la arquitectura del sistema, los patrones de diseño e implementación además de las pruebas realizadas al producto para verificar que todo lo antes planteado se cumpliera. Además se realizan los diagramas de clases del diseño.

4.2 Patrones de arquitectura

Un patrón de arquitectura es un intento de formalizar la forma en que se comunica y reutiliza la experiencia de diseño. Los patrones capturan la experiencia probada de diseño de software. Así como también describen problemas recurrentes que surgen en determinados contextos. Describen esquemas de soluciones probados antes. Son muy buena herramienta para los no expertos. Con ellos se da un paso más hacia la ingeniería de software debido a que permiten que gente común haga cosas que antes requerían conocimientos avanzados (21).

Sus características más conocidas son:

- Atacan problemas recurrentes que ocurren en situaciones específicas y dan una solución.
- Documentan experiencias de diseño existentes y bien probadas.
- Identifican y especifican abstracciones de alto nivel.
- Proveen un vocabulario común y comprensible.
- Son una forma de documentar la arquitectura del software.
- Facilitan la construcción de software con propiedades definidas.
- Ayudan a construir arquitecturas heterogéneas y complejas.

4.2.1 Arquitectura en capas

La aplicación se divide en tres capas lógicas distintas, cada una de ellas con un grupo de interfaces bien definidas. La primera capa se denomina *capa de presentación* y normalmente consiste en una interfaz gráfica de usuario. La *capa intermedia*, o capa de negocio, que contiene la lógica, y la tercera capa, la *capa de acceso a datos*, maneja los datos necesarios para la aplicación.

Figura 4 Representación de la arquitectura en tres capas

En ka-Map existe un conjunto de clases que manejan la interfaz (kaMap, kaLegend, kaKeymap, jsGraphics, entre otras), estas clases se comunican con otras (Init) en el nivel de lógica que ejecuta las operaciones correspondientes a una herramienta determinada y estas a su vez se comunican con la capa de acceso a datos que suministra la información geográfica.

Esta separación entre la lógica de aplicación de la interfaz de usuario añade una enorme flexibilidad al diseño de la aplicación. Pueden construirse y desplegarse múltiples interfaces de usuario sin cambiar en absoluto la lógica de aplicación siempre que esta presente una interfaz claramente definida a la capa de presentación. El prototipo que se desarrolla utiliza archivos shape como fuente de datos, sin embargo, gracias a esta arquitectura es posible cambiar la fuente de datos a PostgreSQl por ejemplo, con pocas implicaciones en el resto de las capas.

4.3 Patrones de diseño

Un patrón de diseño es una solución efectiva a problemas que surgen con el desarrollo de aplicaciones informáticas relacionados con el diseño de interfaces. Un patrón de diseño debe tener como

características principales que debe ser reusable para que así pueda ser adaptable a distintas situaciones y la otra que su efectividad se haya demostrado.

Los patrones de diseño, ayudan en el aprendizaje al programador inexperto, facilitan diseños mucho más flexibles, modulares y reutilizables. Contribuyen a describir las interfaces, identificando sus elementos principales y las relaciones existentes entre ellas.

4.3.1 Patrones de Asignación de Responsabilidades

Los Patrones de Asignación de Responsabilidad (GRASP[7]) describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades (19).

Los patrones GRASP que se utilizaron en la aplicación fueron (20):

Patrón Experto: Un modelo de clase puede definir docenas y hasta cientos de clases de software, y una aplicación tal vez requiera el cumplimiento de cientos o miles de responsabilidades. Si estas se asignan en forma adecuada, los sistemas tienden a ser más fáciles de entender, mantener y ampliar, y se presenta la oportunidad de reutilizar los componentes en futuras aplicaciones. Se debe asignar la responsabilidad al experto en información que en este caso sería la clase que cuenta con la información necesaria para cumplir la responsabilidad. Además se conserva el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. Esto provee un bajo nivel de acoplamiento. En ka-Map la clase kaLegend se especializa en el manejo de la información correspondiente a la leyenda del mapa, por ejemplo, los datos y acciones sobre el color, borde y otros elementos simbólicos de las capas deben ser solicitados a esta clase solamente.

Patr*ón Creador:* La creación de objetos es una de las actividades más frecuentes en un sistema orientado a objetos. En consecuencia, conviene contar con un principio general para asignar las responsabilidades concernientes a ella. Este patrón plantea la necesidad de asignarle a una clase la responsabilidad de crear una instancia de otra clase siempre y cuando agregue los objetos de la clase, los contenga, registre las instancias de estos objetos y los utilice específicamente. El propósito fundamental de este patrón es

[7] Acrónimo de General Responsability Assignment Software Patterns (Patrones de Software para la asignación General de Responsabilidad).

encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento. En ka-Map la clase kaDistance tiene la responsabilidad de crear instancias de otras clases como jsGraphics y kaMap, luego utiliza estos objetos para realizar las diferentes acciones de dibujado y otras sobre los objetos que componen el mapa.

Patrón Alta Cohesión: Asigna una responsabilidad de modo que la cohesión siga siendo alta, la cohesión no es más que la medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Una baja cohesión hace muchas cosas no afines o realiza trabajo excesivo. En ka-Map la clase kaTool debe manejar varios eventos para la activación/desactivación de las herramientas, esta clase bien pudiera encargarse de realizar este trabajo pero también debe gestionar la lógica propia de las herramientas. Para evitar el trabajo excesivo de esta clase se define la clase _eventManager que comparte estas responsabilidades afines.

4.4 Diseño

En el modelo de diseño, se conforma el sistema y se encuentra la arquitectura para que soporte los requisitos funcionales y los no funcionales. El objetivo fundamental del modelo de diseño es adquirir una comprensión con profundidad de los aspectos relacionados con los requisitos no funcionales y restricciones relacionados con los lenguajes de programación. Con el diseño se crea una entrada para la actividad de implementación futura ya que se capturan los conceptos de las clases, interfaces y sistemas que serán necesarios para la solución.

4.4.1 Diagrama de Clases del Diseño

Un diagrama de clases del diseño Web permite describir gráficamente un conjunto de clases, así como las relaciones entre ellas, logrando de esta forma una muestra del sistema importante para la implementación. Muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema. Principalmente, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas.

Los diagramas de clases también son la base para los diagramas de componentes. Los diagramas de clases son importantes no sólo para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas ejecutables, aplicando ingeniería directa e inversa. En este documento se incluyen los principales diagramas modelados. Con el objetivo de lograr un mejor entendimiento del modelo del diseño, se decide representar en un paquete las clases del núcleo de ka-Map, las cuales son utilizadas en el diseño de todos los casos de uso. En la Figura 5 se muestra el diagrama correspondiente.

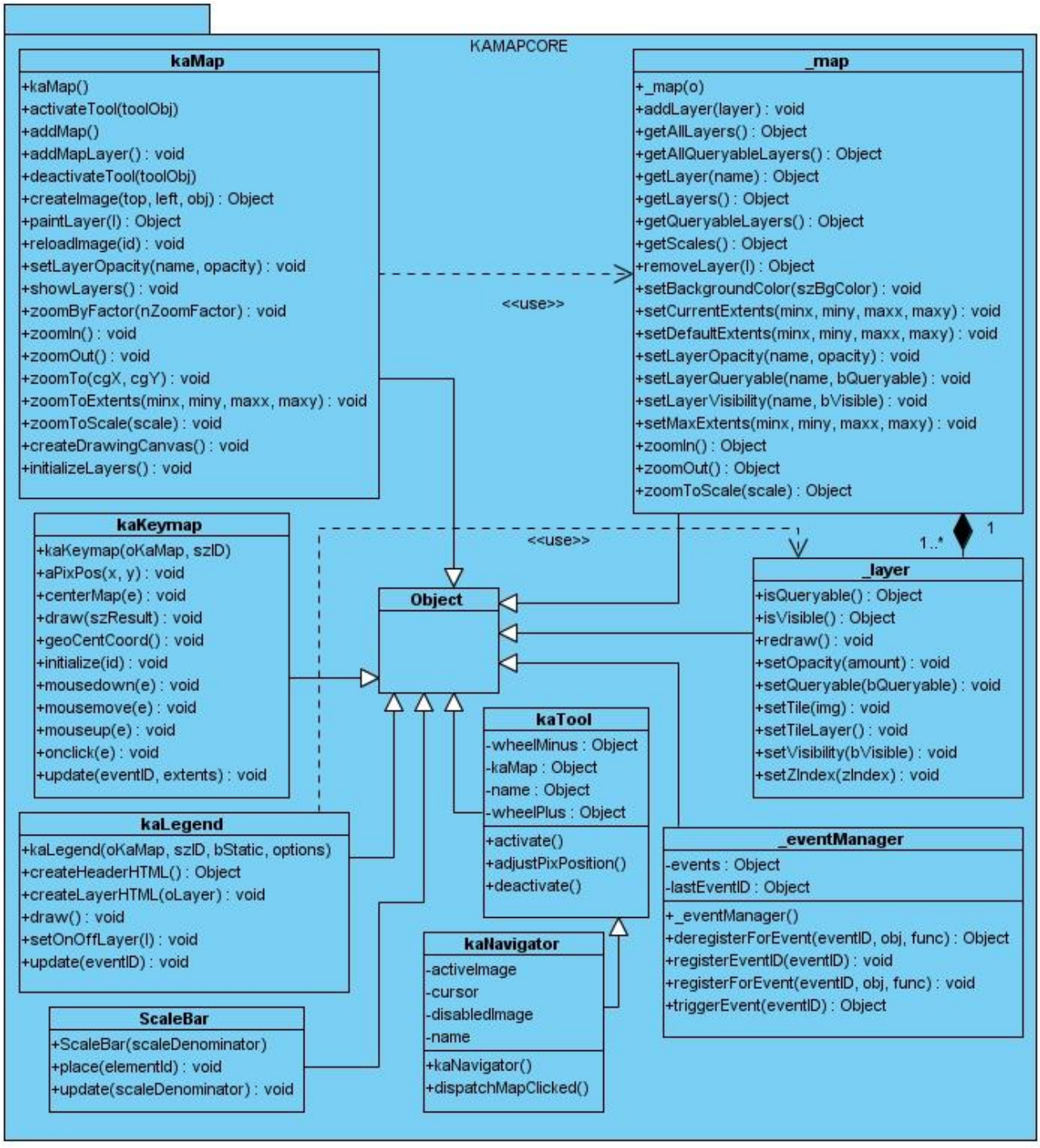


Figura 5 Clases del núcleo de ka-Map

En la Figura 6 se muestra el diagrama de clases del diseño del Caso de Uso Medir distancia y Calcular área (Figura 7).

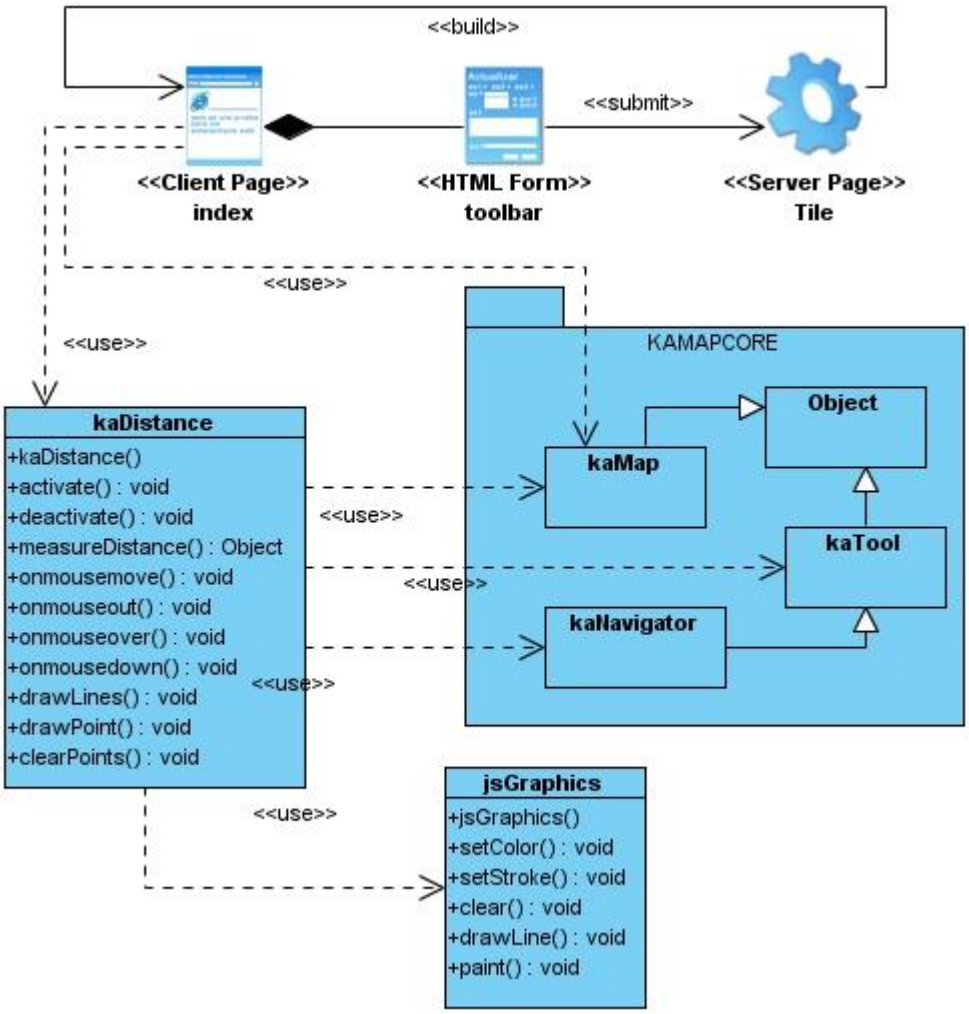


Figura 6 Diagrama de clases del diseño CUS Medir distancia

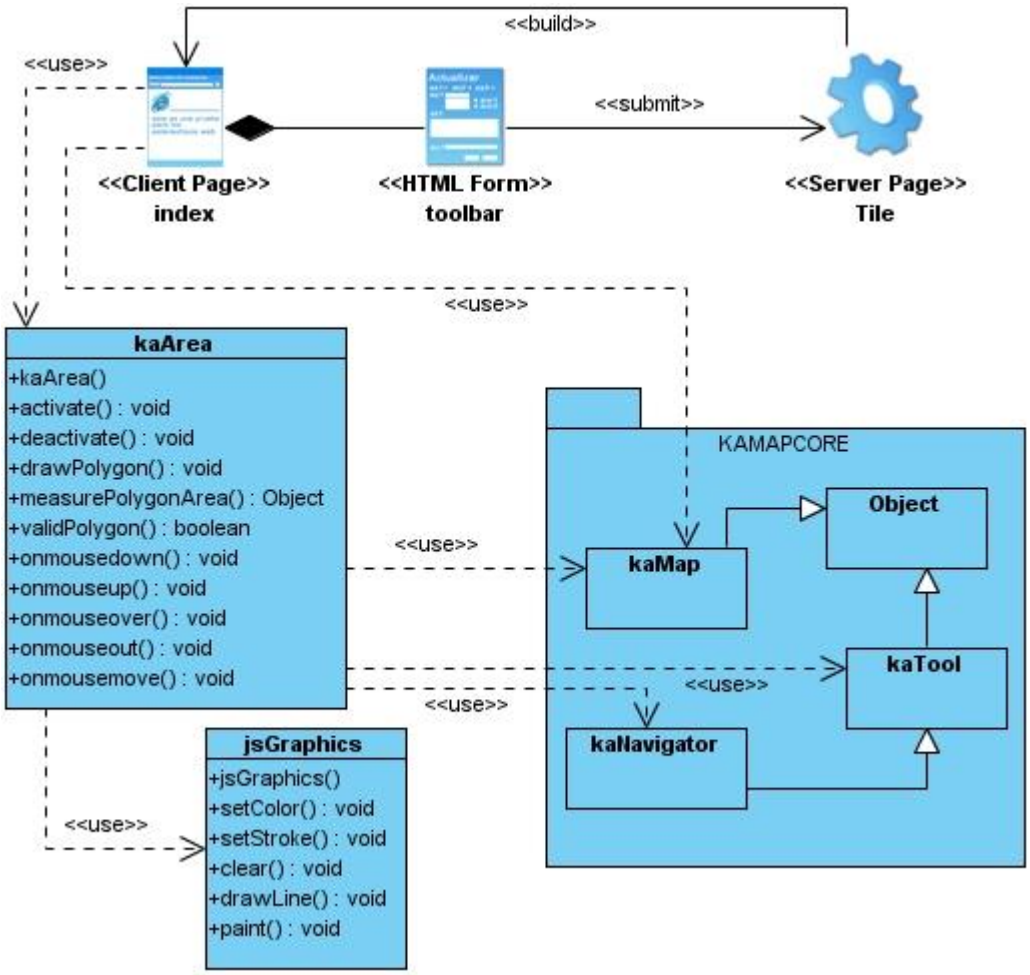


Figura 7 Diagrama de clases del diseño CUS Calcular área

A continuación se presenta el diagrama de clases del diseño del Caso de Uso del Sistema Visualizar elementos del mapa.

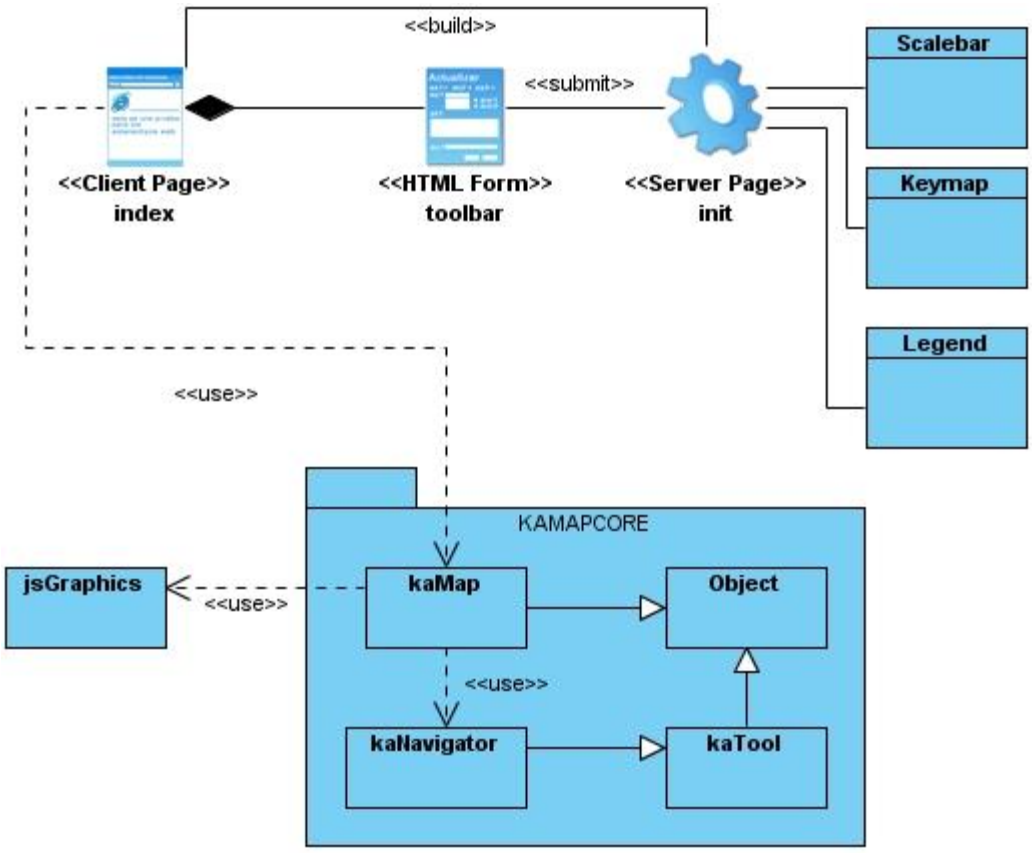


Figura 8 Diagrama de clases del diseño CUS Visualizar elementos del mapa

4.4.2 Diagrama de Interacción

Estos diagramas muestran una interacción, que consiste de un conjunto de objetos y sus relaciones, incluyendo los mensajes que puedan ser realizados entre ellos. Son importantes para modelar los aspectos dinámicos de un sistema y para construir sistemas ejecutables a través de ingeniería hacia adelante e ingeniería inversa. Comúnmente contienen objetos, enlaces, mensajes. Pueden servir para visualizar, especificar, construir y documentar los aspectos dinámicos de una sociedad particular de objetos, o pueden ser usados para modelar un flujo particular de control de un caso de uso.

Los diagramas de interacción están conformados por los diagramas de secuencia y los diagramas de colaboración.

4.4.2.1 Diagrama de Secuencia

Los diagramas de secuencia destacan la ordenación temporal de los mensajes. Un diagrama de secuencia se forma colocando en primer lugar los objetos que participan en la interacción en la parte superior del diagrama, en forma horizontal. Normalmente, se coloca a la izquierda el objeto que inicia la interacción, y los objetos subordinados a la derecha. A continuación, se colocan los mensajes que estos objetos envían y reciben a lo largo del eje Y, en orden de sucesión en el tiempo, desde arriba hasta abajo. Esto ofrece al lector una señal visual clara del flujo de control a lo largo del tiempo.

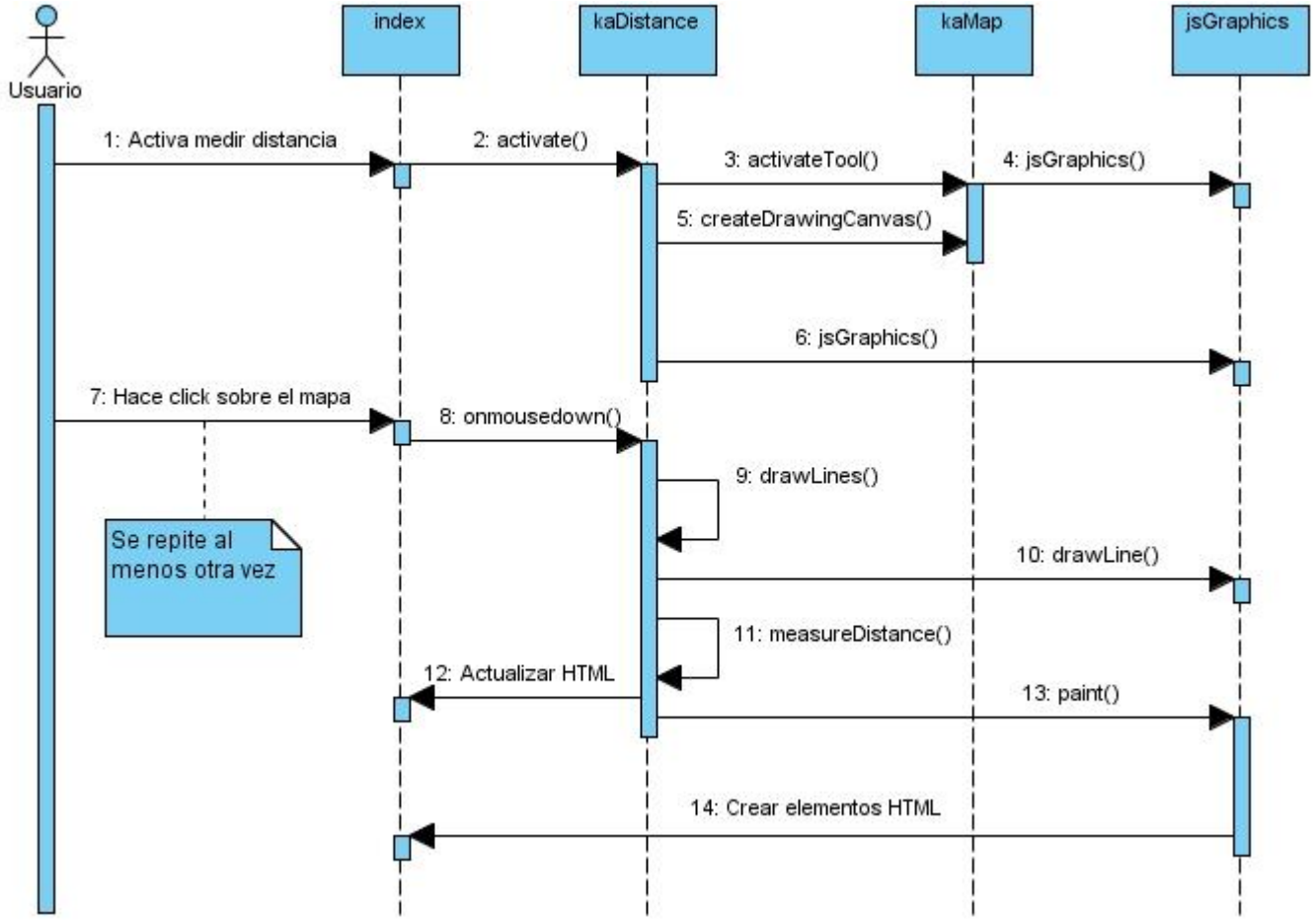


Figura 9 Diagrama de secuencia CUS Medir distancia

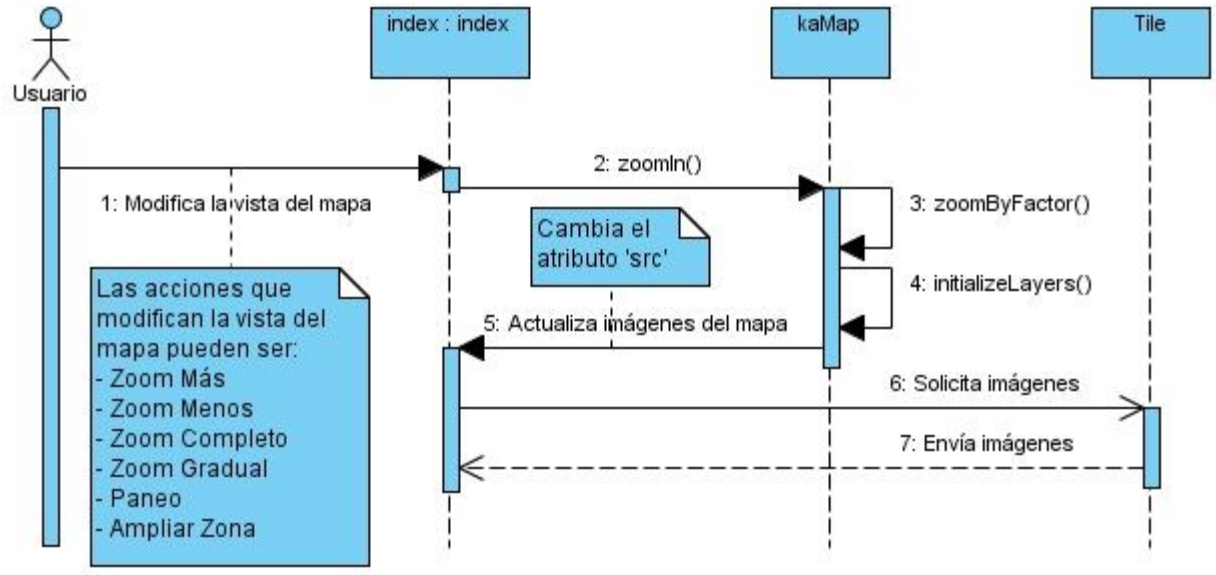


Figura 10 Diagrama de secuencia CUS Modificar vista del mapa

4.4.3 Diagrama de Componentes

Muestra como el sistema está dividido en componentes y las dependencias entre ellos, proveen una vista arquitectónica de alto nivel del sistema. Con este diagrama se visualiza más fácil el camino a tomar para la implementación y permite tomar decisiones respecto a las tareas de este tipo.

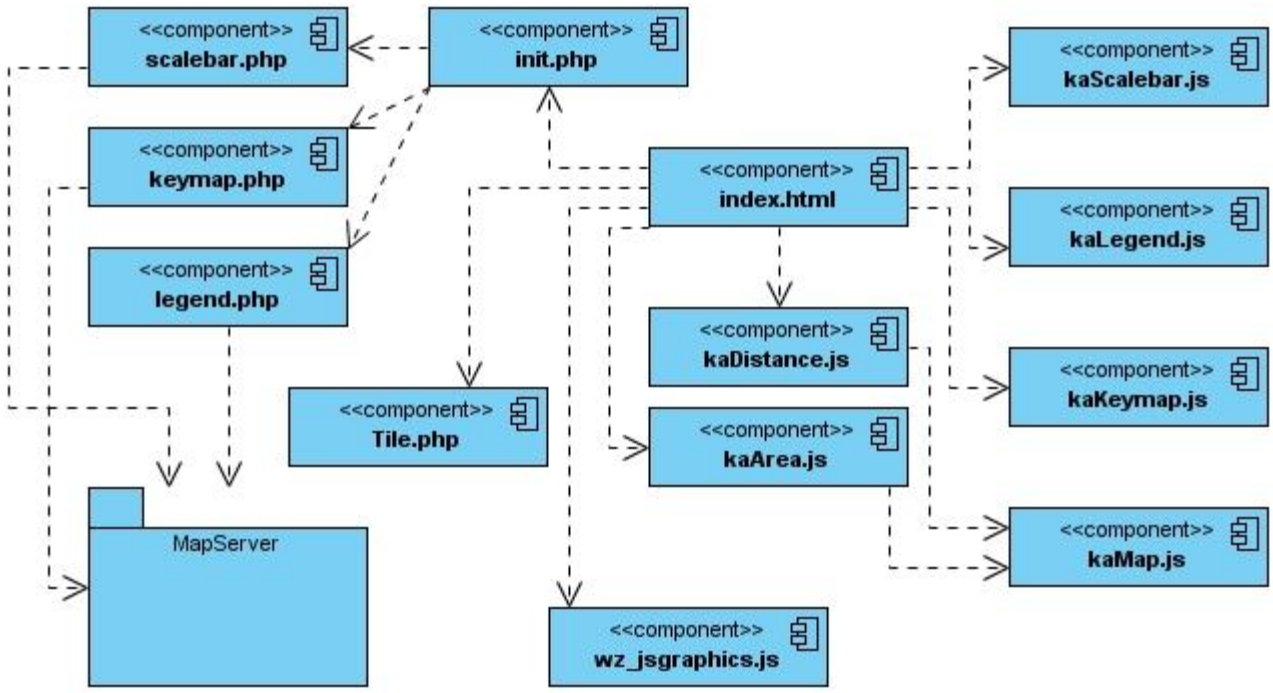


Figura 11 Diagrama de componentes

4.5 Diagrama de Despliegue

El Diagrama de Despliegue se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes.

Figura 12 Diagrama de despliegue

4.6 Validación de la solución propuesta

La validación de la solución se realizó con los casos de prueba de caja negra que se refiere a las pruebas que se llevan a cabo sobre la interfaz del software, por lo que los casos de prueba pretenden demostrar

que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta, así como que la integridad de la información externa se mantiene. Esta prueba examina algunos aspectos del modelo fundamentalmente del sistema sin tener mucho en cuenta la estructura interna del software. Las celdas de la tabla contienen V, I, o N/A. V indica válido, I indica inválido, y N/A que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante.

4.6.1 Diseño de prueba del Caso de Uso Medir distancia

Descripción general: Se inicia cuando el usuario desea medir la distancia total entre varios puntos.

Tabla 12 Diseño de caso prueba del CUS Medir distancia

Nombre de la sección	Escenario	Descripción	Flujo donde empieza
SC 1: Medir distancia	EC 1.1: El usuario selecciona con un clic la opción “Medir distancia” de la barra de herramientas, después realiza un trazo sobre el mapa dando clic en los puntos que darán la distancia total acumulada entre ellos.	El sistema traza una línea entre los lugares seleccionados por el usuario. Los puntos se identificaran con una imagen. El cálculo de la distancia se expresa en la unidad de medida de la escala del mapa en la barra de herramientas.	Barra de herramientas. Medir distancia.

Tabla 13 Matriz de Caso de Prueba Medir distancia

ID del escenario	Escenario	Variable	Respuesta del sistema

EC 1.1	Medir distancia	N/A	El sistema muestra en la barra de herramientas la distancia total que hay entre todos los puntos seleccionados por el usuario.

4.6.2 Diseño de prueba del Caso de Uso Calcular área

Descripción general: Se ejecuta cuando el usuario desea calcular el área de una zona seleccionada.

Tabla 14 Diseño de caso prueba del CUS Calcular área

Nombre del Escenario	Escenario	Descripción	Flujo donde empieza
SC 2: Calcular área.	EC 2.1: El usuario selecciona con un clic la opción “Calcular área” de la barra de herramientas, realiza tantos clics como desee para crear el polígono en la zona del mapa en la que se calculará el área.	El sistema traza un polígono entre los puntos seleccionados por el usuario. El cálculo del área se expresa en la unidad de medida del mapa, en un cuadro de texto junto al puntero.	Barra de herramientas. Calcular área.
	EC 2.2 El usuario selecciona con un clic la opción “Calcular área” de la barra de herramientas, realiza clics y crea un polígono de formato	El sistema traza un polígono entre los puntos seleccionados por el usuario. Muestra en un mensaje en el puntero que	Barra de herramientas. Calcular área.

	incorrecto.	el polígono es incorrecto	

Tabla 15 Matriz del Caso de Prueba Calcular área

ID del escenario	Escenario	Variable	Respuesta del sistema
EC 2.1	Cálculo de área exitoso	N/A	El sistema muestra el cálculo del área seleccionada por el usuario.
EC 2.2	Incorrecta la creación del polígono	N/A	Se le informa al usuario que el polígono es incorrecto.

4.6.3 Diseño de prueba del Caso de Uso Localizar coordenadas

Descripción general: Se ejecuta cuando el usuario desea localizar coordenadas (X, Y) en el mapa.

Tabla 16 Diseño de caso prueba del CUS Localizar coordenadas

Nombre del Escenario	Escenario	Descripción	Flujo donde empieza

SC 3: Buscar coordenadas.	EC 3.1: El usuario selecciona con un clic la opción “Buscar” de la barra de herramientas, introduce los valores de ‘X’ e ‘Y’	El sistema muestra una imagen en el lugar exacto donde están las coordenadas.	Barra de herramientas. Buscar.
	EC 3.2: El usuario selecciona con un clic la opción “Buscar” de la barra de herramientas, introduce los valores de ‘X’ e ‘Y’ fuera de la extensión del mapa	El sistema lanza una excepción indicando que los datos entrados son incorrectos y muestra los valores válidos.	Barra de herramientas. Buscar.
	EC 3.2: El usuario selecciona con un clic la opción “Buscar” de la barra de herramientas, no introduce ningún valor en el campo ‘X’ o ‘Y’.	El sistema muestra una imagen en las coordenadas donde el campo vacío se toma como valor ‘0’.	Barra de herramientas. Buscar.

Tabla 17 Matriz del Caso de Prueba Localizar coordenadas

ID del escenario	Escenario	Variable 1: Coordenada X	Variable 2: Coordenada Y	Respuesta del sistema
EC 3.1	El usuario introduce correctamente las coordenadas.	V	V	El sistema muestra el punto exacto con una imagen.

EC 3.2	El usuario introduce algún valor incorrecto.	Una variable o ambas toma valor I	Se le informa al usuario con una alerta que las coordenadas son incorrectas y le muestra cual es el rango aceptable.
EC 3.3	El usuario no introduce ningún valor.	Una variable o ambas toma valor N/A	El sistema muestra el punto exacto con una imagen. Donde el valor nulo se toma como '0'.

Ya diseñados los casos de prueba se verifica que se cumplieron los requisitos funcionales establecidos. Con el método de Caja Negra, se pretendió probar cada uno de los elementos que componen la interfaz, permitiendo así validar la solución propuesta. Los diseños de caso de prueba correspondientes a los demás Casos de Uso pueden ser consultados en el Anexos.

4.7 Conclusiones

En el presente capítulo se mostró el diseño detallado de la aplicación completa, así como la arquitectura de la misma basada en patrones de arquitectura y diseño necesarios para la construcción del sistema. Mediante la realización del Modelo de Despliegue, se describe como está distribuida física y lógicamente la arquitectura del sistema y sus conexiones. La realización del Modelo de Implementación, permitió detallar los componentes creados para el desarrollo de la aplicación y la relación entre ellos. Además se presenta las pruebas realizadas al mismo las que demostraron que la aplicación cumple con las funcionalidades previstas.

CONCLUSIONES GENERALES

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

- El uso de la metodología de desarrollo de software RUP, el lenguaje de modelado UML y la herramienta CASE Visual Paradigm, permitieron la ejecución del proceso con éxito y obtener los resultados esperados de esta investigación por lo que se propone su uso en sistemas similares al realizado.
- Las herramientas y tecnologías utilizadas, al ser software libre, están acordes a las políticas del país respecto a la soberanía tecnológica. Además de esto, el diseño del sistema permite la implementación de nuevas características funcionales para adaptarlo a las necesidades del cliente.
- Las funcionalidades de medir distancia, calcular área y localización por coordenadas constituyen un valor agregado disponible para futuros desarrollos sobre el producto.
- La documentación generada del trabajo realizado y el resultado obtenido contribuyen a disminuir el esfuerzo y el tiempo necesario para utilizar ka-Map en el desarrollo de un Sistema de Información Geográfica.

En general, se consideran cumplidos los objetivos trazados al desarrollar el prototipo funcional de un Sistema de Información Geográfica sobre ka-Map, pues el mismo cumple todos los requisitos diseñados y las funcionalidades agregadas muestran los resultados esperados.

RECOMENDACIONES

Como resultado del desarrollo y las conclusiones de esta investigación, se expresan las siguientes recomendaciones:

- Incorporarle más funcionalidades a la aplicación.
- Realizar mejoras en cuanto al diseño del sistema, teniendo en cuenta la opinión de los usuarios una vez desplegado este primer prototipo.

GLOSARIO DE TÉRMINOS

Cartografía temática: Forma parte de lo que se denomina generalmente la representación cartográfica. Permite la elaboración de imágenes gráficas particulares que traducen las relaciones espaciales de uno o varios fenómenos, de uno o varios temas. La cartografía temática es un útil de análisis, de ayuda a la decisión y de comunicación ampliamente utilizado para representar una o varias variables.

Paneo: Mover la vista del mapa sin acercarla o alejarla.

Raster: Imagen formada por los colores o tonos de gris de una cuadrícula, en particular los pixeles del monitor. (4)

Objeto: En cartografía se refiere a la representación digital de una entidad o rasgo. (4)

Datos: Del latín datum, lo que se da. Antecedente necesario para llegar al conocimiento exacto de algo o para deducir las consecuencias legítimas de un hecho. Documento, testimonio, fundamento. Información dispuesta de manera adecuada para su tratamiento por un ordenador. (7)

UML (Unified Modeling Language): Lenguaje Unificado de Modelado, usado para modelar sistemas de software.

Visual Paradigm: Herramienta de Ingeniería de Software Asistida por Computadora (CASE, por sus siglas en inglés, Computer-Aided Software Engineering) que permite realizar ingeniería tanto directa como inversa. Utiliza UML como lenguaje de modelado y RUP como base de desarrollo.

Mapfile: Es el principal archivo en la configuración de MapServer que define los datos a ser usados en la aplicación, muestra y consulta los parámetros. Incluye una serie de parámetros que definen las capas disponibles para el mapa interactivo, el estilo con que se mostrarán esas capas, su simbología, el formato en que se generarán las imágenes, el sistema de referencia, tamaño de las imágenes, entre otros.

También contiene información acerca de cómo se debe dibujar el mapa, la leyenda y el resultado de realizar una consulta, controlando la forma en que las salidas de mapas y las leyendas de MapServer se deben presentar en la página HTML. Habitualmente lleva la extensión .map.

Shapefile: Es un formato vectorial de almacenamiento digital donde se guarda la localización de los elementos geográficos y los atributos asociados a ellos.

Zoom: Efecto de acercamiento o alejamiento de una imagen, donde se puede variar a voluntad, la distancia focal y, en consecuencia, el ángulo de visión obtenido por este sistema o por cualquier otro procedimiento.

Requisitos: Condición o cualidad que debe cumplir algo o alguien.

Software: Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

Open Source: El término Open Source nace cuando se pretende obtener aplicaciones comerciales para el software libre. Puesto a que no pueden obligar a pagar por el código, ofrecen asesoramiento y mantenimiento de sistemas. A la hora de llamar la atención de los inversores, el nombre de software libre (free software) no resultaba atractivo, por lo que se buscó un término alternativo que permitiera ofrecer a los inversores conceptos que no tuvieran nada que ver con “free”. La realidad no enseña que el concepto “free” significa en inglés tanto “libre” como “gratis”. Con el fin de eliminar toda referencia a “gratis”, se creó el concepto de Código Abierto (Open Source). Así otorgaron cobertura legal a aquellos programas que optaran por una licencia Open Source frente a una de software libre.

BIBLIOGRAFÍA

1. **Ministerio de Cultura francés.** Lascuax. [En línea] 2007. [Citado el: 3 de 2 de 2011.] http://www.lascaux.culture.fr/#/en/00.xm.

2. **York University.** York. [En línea] 2009. [Citado el: 23 de 5 de 2011.] http://www.york.ac.uk/depts/maths/histstat/snow_map.htm.

3. **Universidad República Oriental del Uruguay.** KO dictionary. [En línea] [Citado el: 15 de 11 de 2010.] http://www.eubca.edu.uy/diccionario/.

4. Diccionario de Sistemas de Información Geográfica. [En línea] [Citado el: 1 de 12 de 2010.] http://siga.cna.gob.mx/SIGA/Diccionarios/glosario.htm.

5. **Carmona, Álvaro y Monsalve, Jhon Jairo.** Monografías. [En línea] 1999. [Citado el: 1 de 12 de 2010.] http://www.monografias.com/trabajos/gis/gis.shtml.

6. **Poyato, Cristobal.** Cpoyato. [En línea] [Citado el: 2 de 12 de 2010.] www.cpoyato.com/Glosario/glosarioC.htm.

7. **DRAE.** Real Academia Española. *Diccionario de la Real Academia Español- Vigésima Segunda Edición.* [En línea] 2010. [Citado el: 1 de 12 de 2010.] http://buscon.rae.es/draeI/SrvltGUIBusUsual?TIPO_HTML=2&TIPO_BUS=3&LEMA=dato.

8. **WIKIPEDIA.** Wikipedia. [En línea] 2008. [Citado el: 2 de 12 de 2010.] http://es.wikipedia.org/wiki/Georeferencia.

9. **Universidad Nacional de Colombia .** Cursos. [En línea] [Citado el: 2 de Octubre de 2010.] http://www.monografias.com/trabajos/gis/gis.shtml.

10. **Universidad Politécnica de Valencia.** Mappinginteractivo. [En línea] 10 de 2008. [Citado el: 23 de 10 de 2010.] http://www.mappinginteractivo.com/plantilla-egeo.asp?id_articulo=1531.

11. **GeoTux.** GeoTux. [En línea] [Citado el: 6 de 10 de 2010.] http://geotux.tuxfamily.org/index.php?option=com_myblog&task=view&id=180&Itemid=59.

12. **ka-Map.** *Ka-Map Wiki.* [En línea] 2006. [Citado el: 23 de 9 de 2010.] http://ka-map.ominiverdi.org/wiki/index.php/Links_to_some_ka-Map_applications.

13. **Kropla, Bill.** *Beginning MapServer. Open Source GIS Development.* s.l. : Ed. Apress, 2005.

14. **Ominiverdi.** MapServer. [En línea] 2009. http://MapServer.gis.umn.edu/.

15. **NetBeans.** NetBeans. [En línea] 2009. [Citado el: 20 de 2 de 2011.] http://netbeans.org/community/releases/61/index_es.html.

16. **Universidad de La Habana.** fBIOinformatica. [En línea] 2010. [Citado el: 4 de 2 de 2011.] http://fbio.uh.cu/sites/bioinfo/glosario.html.

17. **CODEBOX.** CodeBox. [En línea] [Citado el: 28 de 1 de 2011.] http://www.codebox.es/glosario.

18. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Addison Wesley, 2000.

19. **Asakawa, Gabriel, Hurtado Bustamante., Diana Paola y Suárez Valencia, Juan Pablo.** Universidad del Valle. [En línea] 2005. [Citado el: 15 de 3 de 2011.] http://eisc.univalle.edu.co/materias/Material_Desarrollo_Software/exposiciones2005B/polimorfismoFabricacionPura_G01/presentacion.ppt.

20. **Larman, Craig.** *UML y Patrones Introducción al análisis y diseño orientado a objetos.* México : PRENTICE HALL, 1999. 970-17-0261-1.

21. **Universidad de Chile.** U-Cursos. [En línea] [Citado el: 11 de 6 de 2011.] https://www.u-cursos.cl/ingenieria/2008/1/CC31B/1/.../174674.

22. **Beck, Kent.** *Extreme Programming Explained.* Madrid : Adisson Wesley, 2000.

23. **Bosque, Sandr J.** *Sistemas de información Geográfica.* s.l. : Edit. Rialp, 1997.

24. **Berry, J.K.** *Beyond Mapping: Concepts, Algorithms and Issues in GIS.* Fort Collins : GIS World Books, 1993.

25. **Bolstad, P.** *GIS Fundamentals: A first text on Geographic Information Systems, Second Edition.* White Bear Lake : Eider Pres, 2005.

26. **Burrough, P.A. y McDonnell, R.A.** *Principles of geographical information system.* Oxford : Oxford University Press, 1998.

27. **Buzai, G.D.** *Análisis Socioespacial con Sistemas de Información Geográfica.* Buenos Aires : Lugar Editorial, 2006.

28. **Tomlinson, R.F.** *Thinking About GIS: Geographic Information System Planning for Managers.* s.l. : ESRI Press, 2005.

29. **Maguire, D.J., Goodchild, M.F. y Rhind, D.W.** *Geographic Information Systems: principles, and applications.* s.l. : Longman Scientific and Technical, 1997.

30. **Chang, K.** *ntroduction to Geographic Information System, 4th Edition.* McGraw Hill : s.n., 2007.

31. **Stefanov, Stoyan.** *Object Oriented JavaScript.* s.l. : Packt Publishing, 2008. 978-1-847194-14-5.

32. **Gutmans, Andi, Bakken, Stig Sæther y Rethans, Derick.** *PHP 5 Power Programming.* s.l. : Pearson Education, 2005. 0-131-47149-X.

ANEXOS

Anexo 1

Tabla 18 Diseño Caso de Prueba Exportar mapa

Nombre del Escenario	Escenario	Descripción	Flujo donde empieza
SC 4: Exportar mapa.	EC 4.1: El usuario selecciona con un clic la opción "Exportar mapa" de la barra de herramientas, selecciona el formato deseado.	El sistema lanza una ventana donde se encuentran los vínculos a los formatos posibles.	Barra de herramientas. Exportar mapa.

Tabla 19 Matriz de Caso de Prueba Exportar mapa

ID del escenario	Escenario	Variable	Respuesta del sistema
EC 4.1	Exportar mapa	V	El sistema muestra la región del mapa en el formato elegido.

Tabla 20 Diseño del Caso de Prueba Seleccionar capa

Nombre del	Escenario	Descripción	Flujo donde

Escenario			empieza
SC 5: Seleccionar capa	EC 5.1: El usuario puede habilitar y deshabilitar las capas que posee el mapa con un clic en la misma.	El sistema instantáneamente permite tanto la visualización como la desactivación de las capas seleccionadas por el usuario.	Leyenda Capas (Layer)

Tabla 21 Matriz Caso de Prueba Seleccionar capa.

ID del escenario	Escenario	Variable	Respuesta del sistema
EC 5.1	Seleccionar capa	V	El sistema muestra u oculta las capas habilitadas para uso del actor.

Tabla 22 Diseño del Caso de Prueba Visualizar elementos del mapa.

Nombre del Escenario	Escenario	Descripción	Flujo donde empieza
SC 6: Visualizar elementos.	EC 6.1: El usuario puede ver todos los elementos de la leyenda como capas, coordenadas donde se encuentra el cursor y escala	El sistema permite la visualización de la leyenda con todos sus elementos.	Ka-Map leyenda

	actual.		
	EC 6.2: El usuario puede ver un mapa de referencia de la vista actual y navegar en ella.	El sistema muestra el mapa de referencia.	Ka-Map Mapa de referencia

Tabla 23 Matriz Caso de Prueba Visualizar elementos del mapa.

ID del escenario	Escenario	Variable	Respuesta del sistema
EC 6.1	El usuario puede ver la leyenda con todos sus atributos.	V	El sistema muestra la leyenda.
EC: 6.2	El usuario puede ver el mapa de referencia y moverse con él.	V	El sistema muestra el mapa de referencia.