Tesis defendida por

Oscar Iván Hernández González

y aprobada por el siguiente Comité

Dr. Jorge Torres Rodríguez

Director del Comité

Dr. Hugo Homero Hidalgo Silva

Miembro del Comité

Dr. Raúl Rivera Rodríguez

Miembro del Comité

Dr. José Antonio García Macías

Coordinador
del Posgrado en Ciencias de la Computación

Dr. Jesús Favela Vara

Director de la
Dirección de Estudios de Posgrado

Noviembre de 2013

CENTRO DE INVESTIGACIÓN CIENTÍFICA Y DE EDUCACIÓN SUPERIOR DE ENSENADA, BAJA CALIFORNIA

Programa de Posgrado en Ciencias

en Ciencias de la Computación

Encapsulamiento de datos geoespaciales para interoperabilidad con sistemas de información geográfica basados en la Web

Tesis

que para cubrir parcialmente los requisitos necesarios para obtener el grado de Maestro en Ciencias

Presenta:

Oscar Iván Hernández González

Ensenada, Baja California, México

2013

Resumen de la tesis de Oscar Iván Hernández González, presentada como requisito parcial para la obtención del grado de Maestro en Ciencias en Ciencias de la Computación.

Encapsulamiento de datos geoespaciales para interoperabilidad con sistemas de información geográfica basados en la Web

Resumen aprobado por:

Dr. Jorge Torres Rodríguez

Director de Tesis

En los últimos años, el Web Mapping y los WebGIS (Sistemas de Información Geográfica basados en la Web) han evolucionado considerablemente; desde la simple presentación de mapas estáticos, hasta la visualización y análisis dinámico de datos. Esta evolución ha propiciado un incremento en la demanda de acceso y procesamiento de datos geográficamente referenciados (denominados en lo sucesivo como datos geoespaciales) y en consecuencia el desarrollo de conjuntos de datos geoespaciales heterogéneos (con distinto formato, método de acceso y manipulación); esta heterogeneidad, aunada al hecho de que los principales conjuntos de datos geoespaciales están diseñados para entornos de escritorio, dificulta la interoperabilidad y la compartición de los mismos en la Web.

El objetivo de este trabajo de investigación es el diseño, desarrollo e implementación de una plataforma Web interoperable que permita la visualización de datos geoespaciales multiorigen y heterogéneos de manera transparente, mediante el encapsulamiento de formatos de datos geoespaciales de escritorio (como el estándar Shapefile de ESRI) a los denominados "microformatos" geoespaciales como los estándares GML (Geography Markup Language), KML (Keyhole Markup Language) y WMS (Web Map Service) de la OGC (Open Geospatial Consortium). Las herramientas utilizadas fueron de código abierto (Software libre), por las ventajas de seguridad y transparencia que éstas brindan.

El resultado es una Plataforma Web capaz de desplegar información georreferenciada de manera transparente y eficaz, aún para conjuntos de datos con gran carga de información, gracias al encapsulamiento en microformatos tanto de las características geográficas, como de la información tabular relacionada a éstas. Gracias a la naturaleza Web de los microformatos utilizados, los tiempos de respuesta, el consumo de recursos de red y el tiempo de procesamiento de servidor fueron minimizados.

Palabras Clave: **Sistemas de Información Geográfica, WebGIS, Interoperabilidad, WMS, GML, KML, Bases de Datos Espaciales**

Abstract of the thesis presented by Oscar Iván Hernández González, as a partial requirement to obtain the Master of Science degree in Computer Science.

Geospatial data encapsulation for Web-based geographic information systems interoperability

Abstract approved by:

Dr. Jorge Torres Rodríguez

Thesis Director

Over the last years, Web Mapping and WebGIS (Web-based Geographic Information Systems) have evolved considerably; from the simple visualization of static maps, to the dynamic visualization and analysis of data. This evolution has led to an increased demand for access and processing of geographically-referenced data (hereafter referred to as geospatial data) and consequently, the developement of heterogeneous geospatial datasets (data with different format, access method and manipulation). This heterogeneity, along with the fact that the main geospatial datasets are designed for desktop environments, renders the interoperability and data sharing over the Web too difficult to be performed.

The objective of this research work is the design, development and implementation of a Web interoperable platform capable of performing transparent visualization of multisource heterogeneous geospatial data, through the encapsulation of desktop-based geospatial data (such as ESRI Shapefile format) into the so called geospatial "microformats" such as the OGC (Open Geospatial Consortium) standards GML (Geography Markup Language), KML (Keyhole Markup Language) and WMS (Web Map Service). The development tools and support technologies used were Open Source (free Software), due to the security and transparency advantages that these tools offer.

The result was a Web platform capable of displaying georeferenced information in an efficient and transparent manner, even for large datasets, thanks to the encapsulation of both the geographical features and its related tabular data into microformats. Due to the Web nature of the used microformats, response times, network resources usage, as well as server processing time were minimized.

Keywords: **Geographic Information Systems, WebGIS, Interoperability, WMS, GML, KML, Spatial Database**

Dedicatoria

A mis padres, por su invaluable e incondicional apoyo a lo largo de mi vida y mi trayectoria académica.

Agradecimientos

Al CICESE por brindarme la oportunidad de realizar este posgrado.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por el apoyo económico brindado durante mis estudios de Maestría.

Al Dr. Jorge Torres Rodríguez, así como a los miembros del Comité: Dr. Hugo Hidalgo Silva y Dr. Raúl Rivera Rodríguez.

A la Dirección de Telemática por el apoyo técnico y las facilidades otorgadas para el desarrollo de ésta tésis.

A mis hermanos y a todas las personas que de alguna u otra forma estuvieron conmigo durante este proceso.

Contenido

Página

Contenido

Lista de figuras

Figura Página

Lista de tablas

Capítulo 1

Introducción

Los Sistemas de Información Geográfica (GIS por sus siglas en inglés) son componentes informáticos (Software) que combinan el componente semántico de las características terrestres (principalmente descritas por atributos organizados en tablas) con su rigurosa representación geométrica y georreferenciada. Estos sistemas han alcanzado gran popularidad gracias a su impacto en muchos campos de la ciencia y la industria.

Los GIS han estado en constante evolución. Un ejemplo de tal evolución es el WebGIS, el cual es la implementación de los GIS y todas sus funcionalidades y capacidades sobre la WWW (World Wide Web); esto es: Un Sistema de Información Geográfica con un enfoque distribuído. La implementación de estos sistemas requiere la conjunción de retos en campos como la Geomática, Redes de computadoras y Telecomunicaciones, así como de Tecnologías de la Información.

El crecimiento del WebGIS ha sido muy rápido. Esto ha generado el desarrollo de muchos Sistemas de Información Geográfica con datos espaciales heterogéneos (con distinto formato, método de acceso y manipulación). La necesidad de interoperabilidad e intercambio de información espacial entre diferentes Sistemas WebGIS es vital en áreas como la atención y respuesta a emergencias y desastres, estudios hidrológicos y de planeación urbana, etc. Esta heteorgeneidad, aunada al hecho de que los principales conjuntos de datos geoespaciales están diseñados para entornos de escritorio, dificulta la interoperabilidad y la compartición de los mismos en la Web.

1.1 Antecedentes

El desarrollo de tecnologías GIS ha sido influenciado de manera importante por los avances en las Tecnologías de Información (TI). De hecho, los avances en el desarrollo de tecnologías GIS han reflejado estrechamente el avance en el desarrollo de tecnologías computacionales y de redes de telecomunicaciones, evolucionando desde los GIS de escritorio hasta los enfoques distribuídos (GIS basados en la Web). Estos avances incluyen la expansión de enlaces a Internet de alta velocidad y bajo costo, así como una nueva generación de computadoras habilitadas para dar soporte a estos enlaces de banda ancha.

Al igual que las tecnologías computacionales, las tecnologías en GIS basados en Web está en constante cambio. Como menciona Ulloa (2013) en su trabajo, el WebGIS desde su concepción, ha ido de la mano con el desarrollo de las tecnologías de presentacioń de datos para Web. La Figura 1 muestra la evolución de estos sistemas, desde la visualización de mapas estáticos, hasta su evolución a Web Mapping estático y Web Mapping interactivos, para finalmente llegar a los GIS dinámicos basados en la Web (Tsou y Peng, 2003).

La publicación de mapas estáticos distribuía mapas en una página Web en forma de imágenes estáticas de mapas en formatos gráficos tales como Portable Document Format (PDF), GIF o JPEG. Este enfoque se basó en las etapas tempranas de la tecnología Web, esto es, servidores HTTP basados en direcciones URL (acrónimo del inglés Uniform Resource Locator) para servir mapas previamente generados en la Web. Estos mapas eran parte del documento HTML y los usuarios no podían interactuar ni alterar los mapas de ninguna manera.

La segunda etapa de evolución la comprendió el Web Mapping estático. Ésta in-

volucraba el uso de formas HTML y CGI para enlazar los datos ingresados por el usuario al navegador Web por medio de GIS o programas de mapeo en los servidores. Los usuarios elaboraban peticiones desde el navegador Web usando formas HTML personalizadas, posteriormente las peticiones se enviaban al CGI mediante un servidor HTTP para invocar motores de mapeo o GIS; dichos motores creaban el mapa basados en las peticiones del usuario y generaban la imagen sobre la marcha. Esta imagen se enviaba vía HTTP al usuario en el navegador Web. Sin embargo, una de las desventajas de este enfoque es que el desempeño del protocolo HTTP con la tecnología CGI es muy lento, poco práctico y "stateless", esto es, interacciones en las que cada petición es manejada por el servidor como una transacción independiente, no relacionada con ninguna petición anterior.

La tercer etapa es la de Web Mapping interactivo, donde se agrega interactividad e inteligencia al lado del cliente Web, esto mediante el uso de scripts tales como HTML dinámico y/o aplicaciones del lado del cliente como plug-ins, controles ActiveX, y applets Java. Algunas consultas podían ser procesadas del lado del cliente sin necesidad de enviar peticiones al servidor.

La cuarta etapa es la de GIS Distribuído; en esta etapa, la interacción cliente/servidor es realizada por medio de Servicios Web, XML (eXtended Markup Language), API (Application Programming Interface), y tecnologías como Java Beans y .Net. Es importante hacer notar las ventajas que esta etapa (específicamente la utilización de servicios Web) trae consigo, estas ventajas se mencionan a continuación:

- Interoperabilidad e integración con sistemas existentes.
- Estandarización de protocolos (HTTP y XML).
- Fácil implementación.

- Accesibilidad y descentralización de servicios.

Sin duda, estas ventajas funcionales se ven reflejadas en el desarrollo de aplicaciones GIS Distribuídas, gracias a la utilización de XML como base para la definición de figuras geométricas y atributos, además del uso extensivo de HTTP como principal protocolo de transferencia.

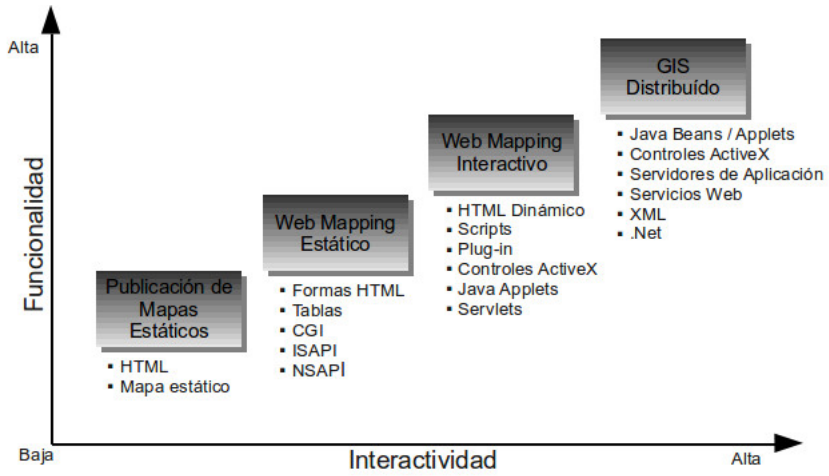

Figura 1. Evolución del GIS basado en Web

1.2 Planteamiento del problema

Las soluciones WebGIS tradicionales carecen de la capacidad para implementar el intercambio y comunicación estándar de datos en ambientes heterogéneos (Wang *et al.*, 2009). Para tratar de resolver estos problemas el OGC (Open Geospatial Consortium) ha desarrollado especificaciones para estandarizar las interfaces de mensajes. Estas especificaciones incluyen:

- Web Map Service (WMS) para mapas codificados como imágenes;

- Web Feature Servcie (WFS) para objetos geográficos o datos vectoriales;
- Geography Markup Language (GML): componentes XML para la codificación de objetos geográficos para su transporte.

De igual manera, los Servicios Web para GIS contribuyen al logro de estas metas. Park y Lee (2003) definen a los Servicios Web como sistemas de Software identificados por un URI (Uniform Resource Identifier), cuyas interfaces públicas y enlaces están definidos y descritos mediante XML. Esta tecnología es de las más populares en la industria computacional y se está convirtiendo en el principal patrón de trabajo y modelo de aplicación para la siguiente generación de aplicaciones de Internet. Su arquitectura cuenta con 3 roles: proveedor, solicitante y mediador o broker. El proveedor crea los servicios web y los hace disponibles a los clientes que quieran usarlos. El solicitante es una aplicación cliente consumidora de los servicios web. El broker, como servicio de registro, provee una forma para que el proveedor y el solicitante de servicios web interactúen (Figura 2).

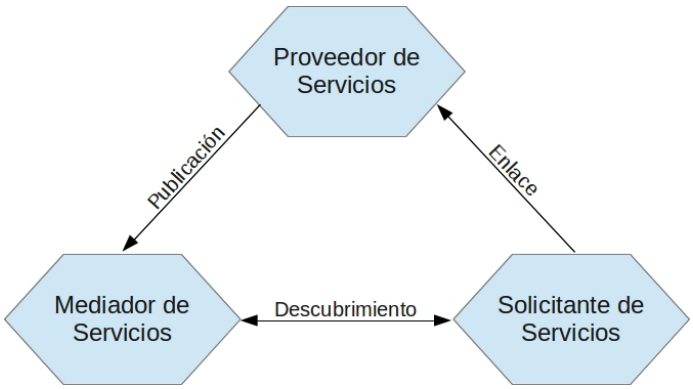

Figura 2. Arquitectura de los servicios Web.

Los protocolos y estándares asociados con los Servicios Web incluyen principalmente: SOAP (Simple Object Access Protocol), WSDL (Web Service Description Lan-

guage), UDDI (Universal Description Discovery and Integration), entre otros (Zhang y Liu, 2010).

Es así, que el presente trabajo está enfocado a transparentar el intercambio de información geoespacial en la Web, haciendo uso de las bondades que nos proporcionan los Servicios Web.

1.3 Objetivos

1.3.1 Objetivo general

Desarrollar una plataforma WebGIS interoperable basada en tecnologías de Código Abierto para compartir e intercambiar datos geoespaciales multiorigen, mediante su encapsulamiento en microformatos de datos geoespaciales.

1.3.2 Objetivos específicos

1. Estudiar estándares de formatos para datos geoespaciales existentes para WebGIS.
2. Definir problemas de compatibilidad Web de formatos geoespaciales de escritorio.
3. Diseñar y desarrollar módulos de encapsulamiento de formatos de datos geoespaciales a microformatos aptos para su despliegue en la Web.

1.4 Organización de la tesis

La organización del documento de tesis es la siguiente: en el Capítulo 2, se presenta una definición de los Sistemas de Información Geográfica basados en la Web (WebGIS) y sus principales características. El diseño y tecnologías de la plataforma, así como los requerimientos de Hardware y Software de la misma se detallan en los Capítulos 3 y 4

respectivamente. En el Capítulo 5 se describe la plataforma propuesta, así como las herramientas y los módulos que la conforman. El Capítulo 6 trata sobre la implementación práctica de la plataforma. Finalmente, en el Capítulo 7 presentamos las conclusiones y propuestas de trabajo futuro referentes a posibles mejoras de la plataforma.

Capítulo 2

Sistemas de información geográfica basados en la Web

2.1 Antecedentes de los sistemas de información geográfica

La naturaleza inquisitiva del hombre, siempre lo ha llevado a explorar, conocer, representar y administrar los lugares en los que habita, así como sus recursos naturales. Estos entornos, ya sea por influencia de la naturaleza o por acciones del mismo hombre, están en constante cambio; entonces, la importancia de contar con información actualizada en este respecto, lo ha llevado a incrementar, mejorar y, hasta cierto punto, automatizar las técnicas y herramientas utilizadas para obtener dicha información.

De esta manera, la Geografía, a través de sistemas de red, análisis de flujo de datos, conocimiento de factores económicos y la organización socio-política del territorio, se ha convertido en una disciplina independiente, que provee una visión sintética de nuestro planeta y la compleja relación entre los fenómenos físicos y los inducidos por el hombre.

Las actividades científicas de la Geografía en torno a la observación de la Tierra y sus características, se han expandido rápidamente, y cada vez más sectores de diversas disciplinas (economía, política, ecología) tienden a utilizar información territorial obtenida a partir de reconocimiento del terreno, Sistemas de Posicionamiento Global (GPS por sus siglas en inglés), fotogrametría tradicional y digital, sensado remoto (teledetección) multiespectral aéreo y satelital, tecnología LiDAR, etc. Los datos e información resultantes son representados mediante capas digitales administradas por Sistemas de Información Geográfica y Sistemas de Soporte a Decisiones. Estas canti-

dades tan grandes de datos deben ser forzosamente manejadas, organizadas y procesadas de manera adecuada para la correcta representación territorial. Estos elementos deben ser procesados de una manera interdisciplinaria e interoperable. En este sentido, la Geomática, satisface ampliamente los requerimientos.

Gomarasca (2009), define el término Geomática (derivado de las disciplinas geografía e informática) como un enfoque sistemático, integrado y multidisciplinario para la selección de instrumentos y técnicas apropiadas para la colecta, almacenamiento, integración, modelado, análisis, transformación, despliegue y distribución de datos georreferenciados provenientes de diferentes fuentes, con caraterísticas de exactitud y continuidad bien definidas y en formato digital.

Algunas de las disciplinas y tecnologías que constituyen a la Geomática son:

- *Ciencias computacionales:* para representar y procesar información a través del desarrollo de instrumentos tecnológicos (Hardware) y métodos, modelos y sistemas (Software).

- *Geodesia:* para determinar la forma y tamaño de la Tierra; define por un lado, la superficie de referencia en su forma completa, el geoide, así como en su forma simplificada, el elipsoide, y por el otro lado, el campo gravitacional externo como función del tiempo.

- *Fotogrametría:* para determinar la posición y la forma de los objetos mediante su medición por medio de imágenes fotográficas.

- *Teledetección:* para la adquisición remota de información relacionada con la naturaleza de los objetos, y para combinar métodos y técnicas para su procesamiento e interpretación subsecuente.

- *Sistema de Posicionamiento Global (GPS):* para proporcionar la posición tridimensional (3D) de objetos fijos o en movimiento, en espacio y tiempo, en toda la superficie terrestre, en tiempo real y bajo cualquier condición meteorológica.
- *Light Detection and Ranging(LiDAR):* para obtener información acerca de un objeto mediante la medición de la luz transmitida por un dispositivo láser y reflejada por este objeto.
- *Sistemas de Información Geográfica (GIS):* para hacer uso de la poderosa combinación de instrumentos capaces de recibir, grabar, invocar, transformar, representar y procesar datos geoespaciales.
- *WebGIS:* para distribuír datos geoespaciales almacenados remotamente en servidores dedicados para bases de datos, de acuerdo a arquitecturas de red complejas.

2.2 Sistemas de información geográfica

Un Sistema de Información Geográfica integra componentes computacionales físicos (Hardware), lógicos (Software) y datos, para la captura, administración, análisis y visualización de todo tipo de información geográficamente referenciada (ESRI, 2013).

2.2.1 Componentes de un GIS

Es importante especificar y definir cada uno de los componentes de los GIS, así como su organización. A continuación, se describe cada uno de estos componentes:

Hardware

Incluye todos los componentes de la computadora que aloja el sistema, así como todos los dispositivos periféricos. El desempeño de la computadora es, en gran parte, respons-

able de la eficiencia de las herramientas de software, así como de los periféricos utilizados para la alimentación de datos y para la representación de productos cartográficos, tales como digitalizador, instrumentos de escaneo, impresoras, etc.

Software

Se trata de Software diseñado específicamente para el procesado de datos geográficos; siendo capaz de manejar datos vectoriales, raster o híbridos. Los principales componentes de Software son los siguientes:

- Sistema Gestor de Bases de Datos (SGBD). El cual, a través del lenguaje SQL (Structured Query Language), administra y sirve los datos.
- Funciones básicas. Estas son gestionadas a través de interfaces de usuario y están dedicadas a: la entrada y validación de datos, almacenamiento de datos y gestión de bases de datos, análisis y procesamiento de datos y finalmente, salida de datos.

Datos

Los datos son parte esencial para un GIS; la calidad y precisión de los resultados, dependerá de la calidad y precisión de los datos alimentados al GIS. Es necesario que cada conjunto de datos que es alimentado al GIS indique su origen, confiabilidad, precisión y consistencia, a través de sus *metadatos*, que en esencia, son datos acerca de los datos (Figura 3). Para que los datos geoespaciales estén disponibles y sean útiles para la mayor cantidad de usuarios, éstos tienen que corresponder a un estándar o estándares predefinidos (i.e. los estándares de OGC y los Servicios Web).

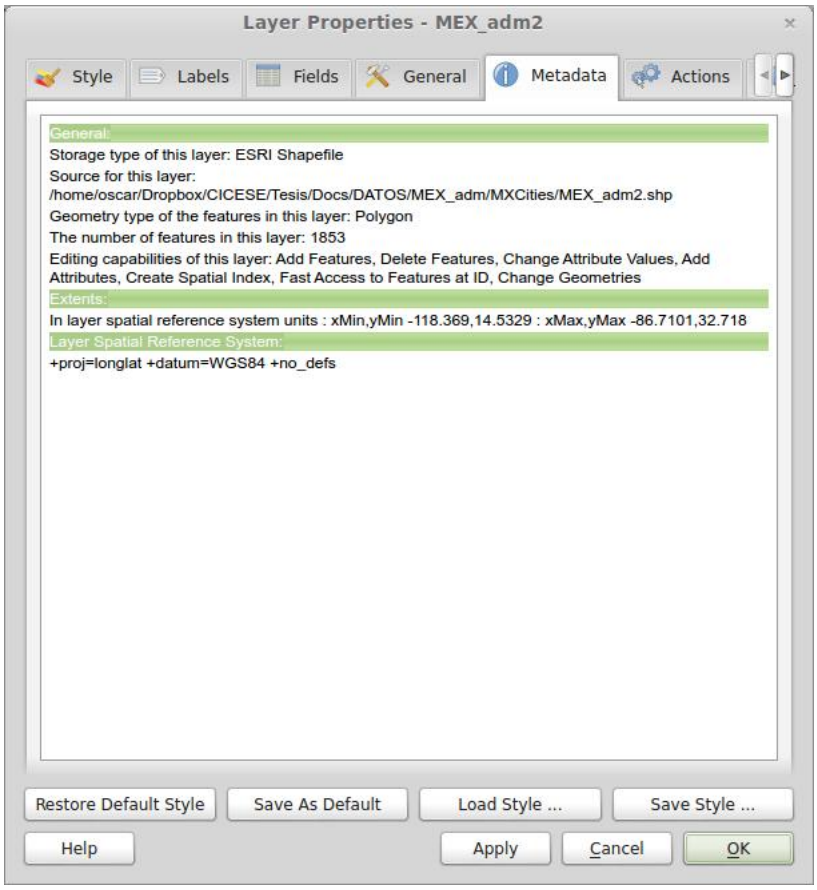

Figura 3. Visualización de metadatos en Quantum GIS.

2.2.2 Modelos de datos geoespaciales

Con base a las necesidades de los usuarios de un GIS, es necesario definir un modelo de datos geoespaciales eficiente, el cual pueda operar de manera correcta de acuerdo a las funcionalidades del sistema. El enfoque de modelado de datos geoespaciales se refiere a dos formatos distintos de datos: *vector* y *raster*. La Figura 4, muestra ambos formatos.

Datos vector

En este modelo, el territorio geométrico es descrito haciendo uso de los elementos (primitivas) propios de la cartografía digital:

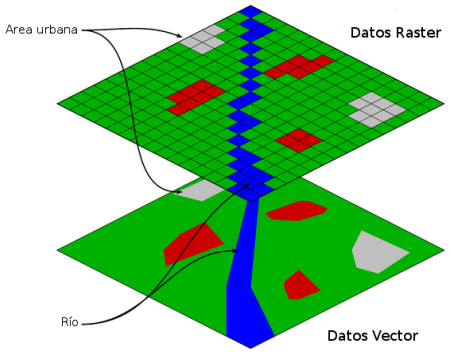

Figura 4. Formatos vector y raster de datos geoespaciales.

- *puntos:* que representan objetos descritos por un par de coordenadas (pozos, torres eléctricas, etc.);
- *líneas:* representadas por un conjunto de puntos interconectados (caminos, carreteras, etc.). A los conjuntos de líneas se les considera *polilíneas*;
- *polígonos:* Áreas encerradas por una multilínea, donde el primer y último vértice son coincidentes (campo de cultivo, lago, etc.). A los conjuntos de polígonos se les considera *multipolígonos*.

Este modelo presenta importantes limitaciones, especialmente en lo que concierne a digitalización de datos. Es necesario contar con especialistas para la supervisión constante, el tiempo de procesamiento es alto, lo cual muchas veces hace imposible contar con actualizaciones de datos en forma oportuna, como por ejemplo, aplicaciones de monitoreo de fenómenos cambiantes (e.g. huracanes). Por otro lado, el formato vector típicamente es más económico en términos de uso de espacio en disco y presenta ventajas en precisión y calidad de presentación (no presenta degradación en la visualización al hacer zoom). Algunos de los formatos vector más comunes son: Shapefile, TIGER, ESRI LYR.

Datos raster

En este modelo, el espacio representado es dividido en una cuadrícula usualmente homogénea conformado por columnas y filas. Cada elemento de la cuadrícula (denominado celda) define una porción discreta de suelo, su posición es definida por los valores de columna y fila dentro de la matriz. La celda contiene un valor numérico como único atributo de la porción de suelo en esa posición geográfica. Si dicha porción requiere más de un atributo, es necesario su almacenamiento en archivos raster diferentes. Este formato de datos requiere mucho espacio en disco para ser almacenado. Dado que su precisión depende de la extensión de terreno que cada celda ocupe, es necesario un equilibrio entre el grado de la resolución geométrica (i.e. el tamaño de la celda) y el tamaño del archivo. El formato Raster es particularmente apto para representar datos temáticos (datos que enfatizan un tema particular, por ejemplo el promedio de precipitación pluvial en un zona determinada). Algunos de los formatos raster más comunes son Digital Raster Graphic (DRG), ESRI Grid, GeoTIFF.

2.2.3 Bases de datos espaciales

Obe y Hsu (2011) definen una base de datos espacial, como aquella base de datos que define tipos especiales de datos para objetos geométricos, y que además permite almacenar datos geométricos (normalmente de naturaleza geográfica) en tablas de bases de datos regulares. Provee además funciones e indexado especiales para consulta y manipulación de datos usando SQL.

Una base de datos espacial proporciona tanto herramientas de almacenamiento como de análisis. Básicamente, la funcionalidad combinada de estas herramientas, puede dar respuesta a preguntas como “Dame una lista de restaurantes de comida

china en la zona centro de Ensenada", o inclusive a consultas más elaboradas como: "Dame una lista de las casas que se encuentran a menos de 1 km de La Lagunita en Ensenada, cuyo terreno sobrepase los 700 m^2".

2.3 Sistemas de información geográfica basados en la Web

Yang *et al.* (2005) definen al WebGIS como Sistemas de Información Geográfica que operan en una plataforma computacional basada en la Web. Son producto de la unión entre un GIS y una plataforma Web caracterizada por diversos protocolos técnicos, tales como el Protocolo de Transferencia de Hiper Texto (HTTP por sus siglas en inglés) soportado por servidores y navegadores Web. Sin embargo, su expansión no se ha limitado únicamente al HTTP tradicional; los WebGIS actuales incluyen todo tipo de clientes GIS que puedan interactuar con un servidor GIS a través de HTTP y Servicios Web, tales como WMS, WFS y CSW (Yang, 2011). El WebGIS provee una plataforma para la entrega de datos, así como de información y servicios geoespaciales a una gran variedad de usuarios a lo largo y ancho del planeta.

2.3.1 Arquitectura WebGIS

La arquitectura de WebGIS es muy similar a la arquitectura cliente/servidor. El procesado de datos geoespaciales está dividido en tareas del lado del servidor y del lado del cliente. Típicamente, un cliente es un navegador Web. El lado del servidor consiste en un servidor Web, software WebGIS y Bases de Datos Espaciales (Figura 5).

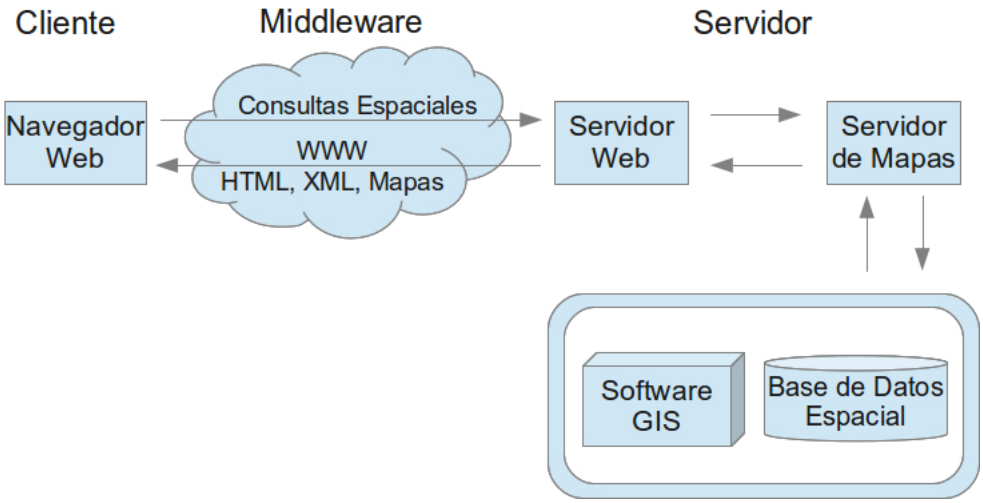

Figura 5. Arquitectura WebGIS.

2.3.2 Arquitectura del lado del cliente

En cuanto a la arquitectura del cliente, existen dos tipos: Arquitectura Delgada y Arquitectura Gruesa.

En la arquitectura delgada, los clientes tienen únicamente interfaces de usuario para comunicarse con el servidor y desplegar resultados. La Figura 6 muestra esta arquitectura; del lado del cliente se encuentra un navegador Web común y corriente con el cual interactúa el usuario, mientras que el lado del servidor consiste de un servidor Web y un servidor GIS, así como de un Servidor de Mapas Web (WMS).

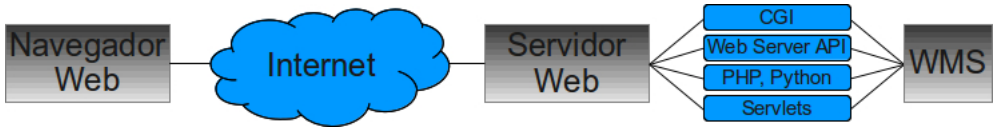

Figura 6. Arquitectura delgada de cliente.

En general, un navegador Web puede manejar documentos HTML e imágenes raster embebidas en formatos estándar. Para tratar otro tipo de formatos de datos,

como datos vectoriales, video o música, la funcionalidad del navegador debe ser extendida, ya que usando la misma arquitectura delgada, los formatos de datos vectoriales no podrían utilizarse. Para solucionar este problema la mayoría de los navegadores ofrecen un mecanismo que permite, que programas de terceros, trabajen en conjunto con el navegador a manera de Plug-in.

La funcionalidad de la interfaz de usuario ha evolucionado desde la simple obtención de documentos, hasta aplicaciones altamente interactivas basadas en Arquitectura Gruesa de Cliente. Este progreso incluye Formas HTML, CGI, JavaScript para incrementar las capacidades de la interfaz, Applets Java para proveer funcionalidad del lado del cliente, y APIs (como Google Maps API u Openlayers API) para dar soporte a los WMS. La Figura 7 muestra un ejemplo de Arquitectura Gruesa.

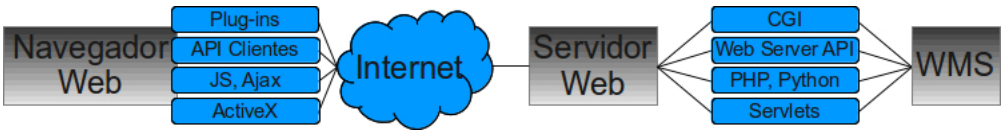

Figura 7. Arquitectura gruesa de cliente.

2.4 Microformatos de datos geoespaciales

Los microformatos de datos geoespaciales son formatos enfocados especialmente para la Web (basados en XML y otros lenguajes de marcado). Estos son comunmente utilizados para representar piezas individuales de datos geoespaciales, figuras dentro de un programa en ejecución, transferencia de figuras y formas de un programa a otro, y generalmente no son utilizados para el almacenamiento permanente de información. Estos microformatos son abiertamente soportados por compañías y organizaciones públicas y privadas como Google, Yahoo, Bing y Openlayers. Algunos ejemlos a considerar son

Well-known Text (WKT), Well-known Binary (WKB), GeoJSON, Geography Markup Language (GML) y Keyhole Markup Language (KML). A continuación se describen dos de los más importantes.

2.4.1 GML: Geography Markup Language

Es un esquema XML utilizado para expresar características geográficas. GML sirve tanto como lenguaje de modelado para sistemas geográficos, como formato abierto de intercambio para transacciones geográficas en Internet (OGC, 2013). Este formato es estándar y libre de plataforma, esto permite a los usuarios y desarrolladores describir conjuntos de datos geoespaciales genéricos que contienen puntos, líneas y polígonos, así como información tabular acerca de los mismos.

Li *et al.* (2008) menciona 4 principales categorías de primitivas geométricas en GML: puntos, curvas, superficies y sólidos. Dichas primitivas son utilizados para describir, por ejemplo, caminos, ríos y puentes y son codificadas como elementos dentro de documentos GML. Los valores de coordenadas de todas las geometrías son expresados, directa o indirectamente, como coordenadas vía las etiquetas “coordinates”, “coord”, “posList” o “pos”. La Figura 8 muestra un ejemplo de esquema de codificación de este lenguaje.

```
<?xml version="1.0" encoding="utf-8" ?>
<ogr:FeatureCollection
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://ogr.maptools.org/ vtmosaic_cut4326.xsd"
     xmlns:ogr="http://ogr.maptools.org/"
     xmlns:gml="http://www.opengis.net/gml">
  <gml:boundedBy>
    <gml:Box>
      <gml:coord><gml:X>-116.8551751459938</gml:X><gml:Y>31.60885186486473</gml:Y></gml:coord>
      <gml:coord><gml:X>-116.4097749033089</gml:X><gml:Y>31.99625765237976</gml:Y></gml:coord>
    </gml:Box>
  </gml:boundedBy>
```

Figura 8. Esquema de un archivo GML.

2.4.2 KML: Keyhole Markup Language

Ying-jun *et al.* (2009) define KML como un Lenguaje de Marcado descriptivo, basado en la sintaxis y formato de archivo de XML, utilizado para describir y almacenar información geográfica tal como puntos, líneas, superficies, información tridimensional y tabular, etc.

Google (2013a) describe a KML como un formato utilizado para mostrar datos geográficos en el navegador terrestre Google Earth. No obstante, también es posible desplegar el contenido de estos archivos por medio del API de Google Maps. Al igual que GML, este formato está basado en el estándar XML y cuenta con una estructura de etiquetas con elementos y atributos anidados. El esquema de codificación de este lenguaje se muestra en la Figura 9.

```
<?xml version="1.0" encoding="utf-8" ?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document><Folder><name>vtmosaic_cut4326</name>
<Schema name="vtmosaic_cut4326" id="vtmosaic_cut4326">
        <SimpleField name="Name" type="string"></SimpleField>
        <SimpleField name="Description" type="string"></SimpleField>
        <SimpleField name="Entity" type="string"></SimpleField>
        <SimpleField name="Handle" type="string"></SimpleField>
        <SimpleField name="Layer" type="string"></SimpleField>
</Schema>
  <Placemark>
        <Style><LineStyle><color>ff0000ff</color></LineStyle><PolyStyle><fill>0</fill></PolyStyle></Style>
        <ExtendedData><SchemaData schemaUrl="#vtmosaic_cut4326">
                <SimpleData name="Entity">Polyline</SimpleData>
                <SimpleData name="Handle">36</SimpleData>
                <SimpleData name="Layer">3165</SimpleData>
        </SchemaData></ExtendedData>
      <LineString><coordinates>-116.6941,31.8968 -116.6955,31.8955</coordinates></LineString>
  </Placemark>
```

Figura 9. Esquema de un archivo KML.

Sin embargo, debido al hecho de que el formato KML fue originalmente creado para ser visualizado dentro de Google Earth, la compatibilidad del despliegue y la visualización de este tipo de archivos en el API de Google Maps presenta ciertas restricciones en cuanto a la complejidad y el tamaño de los mismos. En la Tabla I se detallan dichas limitantes y restricciones.

Tabla I. Restricciones de tamaño y complejidad de KML. Obtenido de Google (2013b)

Tamaño máximo de archivo recuperado	3 MB
Tamaño máximo de archivo KML sin comprimir	10 MB
Número máximo de enlaces de red	10
Número máximo del total de recursos aplicables	1000

En el siguiente capítulo se presentará el diseño de la plataforma Web, las arquitecturas Web en las cuales se basó el diseño, y las tecnologías que se utilizaron para el desarrollo de la misma.

Capítulo 3

Diseño de la plataforma Web

El diseño de la plataforma está basado en una arquitectura de capas, cada una con tareas específicas, con un enfoque híbrido que engloba dos paradigmas: La Arquitectura de 3 capas y la Arquitectura MTV (Modelo-Templete-Vista) de Django.

3.1 Arquitectura de 3 capas

La arquitectura de 3 capas para aplicaciones Web está basada en una arquitectura cliente-servidor en la que cada una de las capas cumple con tareas específicas; las capas desconectadas no deben tener ningún tipo de comunicación y los cambios o modificaciones de rutinas afectarán únicamente a la capa que contenga la o las rutinas en cuestión. La Figura 10 muestra el esquema de esta arquitectura.

- La capa de *presentación* (conocida como Front-end) provee una interfaz de usuario y maneja las interacciones con el mismo. Esta capa no debe contener código de acceso o manipulación de datos ni lógica de negocios.

- La capa *lógica* (conocida como Middleware) contiene las premisas y reglas para el procesado de información. Esta capa no deberá contener interfaces de presentación ni código de acceso o manipulación de datos.

- La capa de *datos* (conocida como Back-end) es la capa de almacenamiento físico. Administra el acceso a la base de datos. No deberá contener presentación ni código de lógica de negocios.

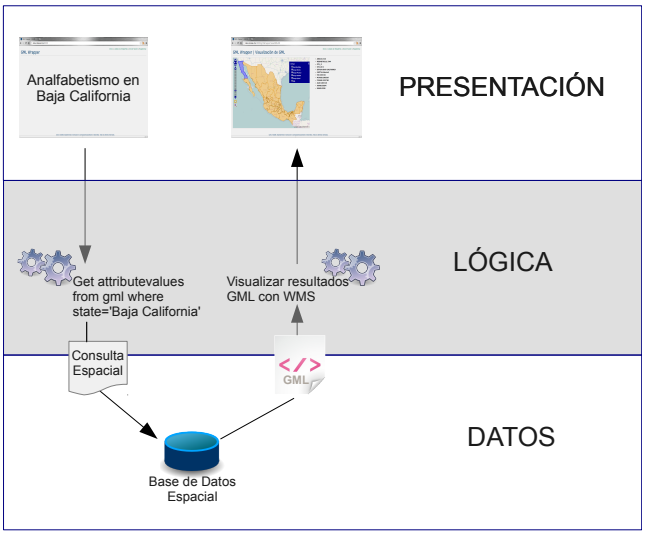

Figura 10. Arquitectura de 3 capas para aplicaciones Web.

3.2 Arquitectura MTV

Django (2013) diseñó la arquitectura MTV (Modelo-Templete-Vista) -vagamente basada en la arquitectura MVC (Modelo-Vista-Controlador)- para el desarrollo de su Framework (también denominado Django).

Bajo esta interpretación de MVC, la *Vista* describe los datos que serán presentados al usuario, pero no la forma en la cual serán presentados, es decir, especifica que datos verá el usuario, no como los verá; es así que la *Vista* es una función para un URL en particular, y dicha función describe que datos serán presentados.

La presentación de los datos corre a cargo del *Templete*. Es aquí donde se maneja la interacción entre el usuario y el sistema Web.

Finalmente, el *Modelo* representa la capa de datos de la aplicación. Contiene todo lo relacionado a los datos: como están estructurados, como acceder a ellos, su validación, etc.

3.3 Diseño y tecnologías de la plataforma

En esta sección desglosaremos las tecnologías utilizadas para el desarrollo de esta plataforma, presentándolas en un diagrama por capas. De igual forma presentaremos un diagrama con la estructura final de la plataforma.

3.3.1 Tecnologías de soporte

A continuación se desglosan las tecnologías utilizadas, partiendo de la capa del Front-end. La Figura 11 muestra el diagrama completo.

Figura 11. Diagrama de tecnologías de soporte.

- *Front-end.* En la capa de presentación, o Front-end, estarán todas las tecnologías Web que permitirán la generación y despliegue de la interfaz de usuario: HTML

5, Javascript, Ajax. De igual manera, los API de Google Maps y Openlayers para servicios WMS estarán alojados en esta capa.

- *Middleware.* En la capa de Middleware tendremos alojado Django, nuestro Framework para el lenguaje de desarrollo Python; este Framework cuenta con una aplicación para desarrollo geoespacial denominada Geodjango. Este Framework incluye diversas librerías para análisis y manejo de datos geoespaciales. Algunos ejemplos son: GEOS, OGR y GDAL.

- *Back-end.* En esta capa estarán ubicados los servicios de Bases de Datos que almacenarán los datos geoespaciales. Los SGBD serán PostGIS y SpatiaLite; de igual manera tendremos el servidor Web (Apache HTTP Server). Todos los servicios mencionados estarán corriendo en un entorno con el Sistema Operativo Linux Ubuntu Server 12.10 con arquitectura de 64 bits.

3.3.2 Estructura de la plataforma

La Figura 12 detalla la estructura y el flujo de los datos geoespaciales desde su inserción a la plataforma, almacenamiento en las bases de datos, encapsulamiento, y finalmente su visualización.

Como se puede apreciar, Geodjango, siendo la capa de Middleware, será el encargado de la interacción con el Back-end para la recepción, validación, procesamiento y almacenamiento de los datos geoespaciales de entrada (Shapefiles) en las bases de datos. De igual forma, la extracción de geometrías y atributos para su posterior encapsulamiento estará a cargo de Geodjango.

En el Front-end (capa de presentación) se procesarán los datos geoespaciales encapsulados (archivos GML y KML) y serán desplegados por medio de los API de los

Servicios WMS de Google Maps y Openlayers.

Es fácil apreciar en este esquema la separación de las 3 capas de la plataforma propuesta, siguiendo el principio *Don't repeat yourself* (DRY por sus siglas en inglés): *"Cada pieza de conocimiento debe tener una única, inequívoca y definitiva representación dentro de un sistema"*.

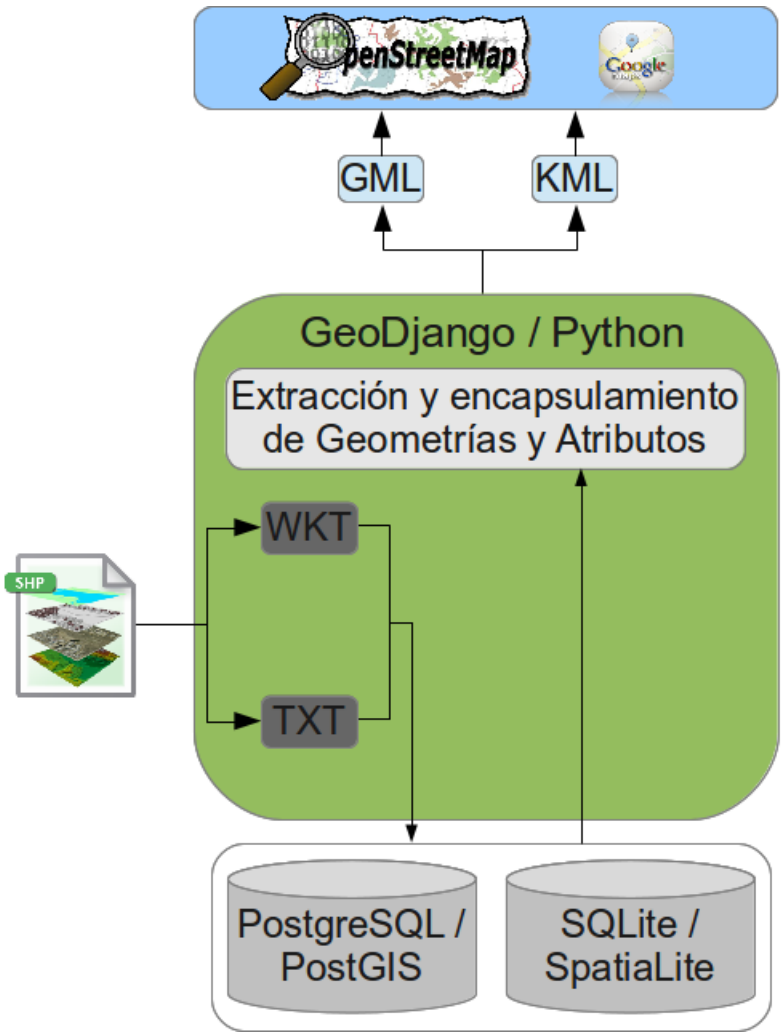

Figura 12. Estructura y flujo de datos geoespaciales de la plataforma.

3.3.3 Estructura de la base de datos

En la Figura 13 se muestra el diseño de tablas y relaciones de la base de datos implementada. Este diseño cuenta con cuatro tablas principales denominadas: Shapefile, Feature, Attribute y Attributevalue. Estas tablas fungirán como contenedores genéricos de shapefiles, ya que no se puede saber con antelación el tipo de figuras que contendrá un Shapefile, ni el número de atributos asociados a éstas. A continuación se detallan la función de cada tabla:

- *Shapefile.* Representará el archivo Shapefile importado por el usuario.
- *Attribute.* El archivo Shapefile tendrá asociados un número N de atributos; esta tabla contendrá un registro por cada atributo.
- *Feature.* Cada geometría del archivo Shapefile estará almacenada como un registro en esta tabla.
- *AttributeValue.* Cada geometría tendrá un número N de valores de atributos; esta tabla contendrá un registro por cada atributo.

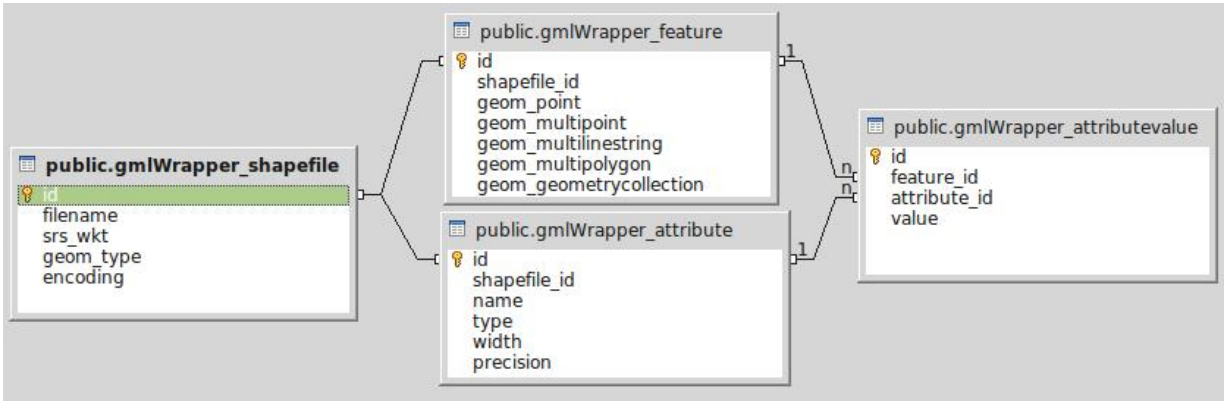

Figura 13. Esquema de tablas y relaciones de la base de datos.

En el siguiente capítulo, se discutirán los requerimientos funcionales necesarios para el desarrollo de la plataforma Web, incluyendo requerimientos de datos, requerimientos de Hardware y Software del servidor, y requerimientos de paquetería del cliente.

Capítulo 4

Requerimientos de la plataforma

Este capítulo trata los requerimientos tanto de Hardware y Software, como de formato y modelo de los datos geoespaciales proporcionados. El buen desempeño y correcto funcionamiento de la plataforma estará basado en una configuración específica de Hardware y Software, la cual permitirá el procesamiento de datos geoespaciales y visualización Web de los mismos de manera eficiente.

4.1 Requerimientos de datos geoespaciales

Shapefile es un formato vector para Software GIS desarrollado por ESRI. Es capaz de almacenar geometrías NO topológicas (no existen relaciones espaciales explícitas entre las geometrías) e información tabular de sus atributos en un conjunto de datos. Las geometrías son almacenadas como una figura que comprende un conjunto de coordenadas vector.

De acuerdo a Xia y Wei (2008), un Shapefile consta de 3 archivos distintos de datos: Un archivo principal (.shp), un archivo de índice (.shx) y una tabla de datos dBase (.dbf) donde se almacenará la información tabular del Shapefile; además, consta de un cuarto archivo de proyección (.prj) que especifica el sistema de coordenadas geográficas de los datos geoespaciales contenidos en el archivo principal.

Todas las figuras no nulas en un shapefile deberán ser del mismo tipo. Los tipos de figuras y sus respectivos valores se muestran en la Tabla II. Estos valores son codificados dentro de la estructura del encabezado de los archivos Shapefile y definen el tipo

de figura que contiene un Shapefile en específico.

Dado que los encabezados de un archivo Shapefile carecen de la sobrecarga de procesamiento de estructuras topológicas, presentan ventajas, como mayores velocidades de dibujo y edición. La Tabla III muestra la estructura del encabezado de los archivos Shapefiles (ESRI, 2013).

Tabla II. Tipo y valor de figuras en Shapefile

Valor	**Tipo Figura**
0	Figura nula
1	Punto
3	PoliLínea
5	Polígono
8	MultiPolígono
11	PuntoZ
13	PoliLíneaZ
15	PolígonoZ
18	MultiPuntoZ
21	PuntoM
23	PoliLíneaM
25	PolígonoM
28	MultiPuntoM
31	MultiPatch

Tabla III. Estructura del encabezado de un archivo Shapefile

Posición	**Campo**	**Valor**	**Tipo**	**Orden Byte**
Byte 0	Código de Archivo	9994	Entero	Big
Byte 4	No utilizado	0	Entero	Big
Byte 8	No utilizado	0	Entero	Big
Byte 12	No utilizado	0	Entero	Big
Byte 16	No utilizado	0	Entero	Big
Byte 20	No utilizado	0	Entero	Big
Byte 24	Longitud de Archivo	Longitud de Archivo	Entero	Big
Byte 28	Versión	1000	Entero	Little
Byte 32	Tipo de Figura	Tipo de Figura	Entero	Little
Byte 36	Caja Delimitante	Xmin	Doble	Little
Byte 44	Caja Delimitante	Ymin	Doble	Little
Byte 52	Caja Delimitante	Xmax	Doble	Little
Byte 60	Caja Delimitante	Ymax	Doble	Little
Byte 68*	Caja Delimitante	Zmin	Doble	Little
Byte 76*	Caja Delimitante	Zmax	Doble	Little
Byte 84*	Caja Delimitante	Mmin	Doble	Little
Byte 92*	Caja Delimitante	Mmax	Doble	Little

*No utilizados, con valor 0.0 si no es de tipo Measured o Z

4.2 Requerimientos de Hardware

Esta sección se enfocará exclusivamente en los requerimientos de Hardware del servidor que alojará los servicios necesarios para el funcionamiento de la plataforma, ya que será este el que aloje los servicios de procesamiento y almacenamiento de datos.

Como se menciona en la sección 3.3.1 del capítulo 3, la instalación y puesta en marcha de la plataforma, se efectuó sobre un servidor Linux Ubuntu Server 12.10. Este servidor fue instalado y configurado sobre un espacio virtual asignado dentro de un servidor VMware VCenter Server™ albergado en el Centro de Cómputo de la Dirección de Telemática del CICESE. La configuración de Hardware mínima (basada en los recursos asignados desde el servidor VMware) para el óptimo desempeño de la plataforma es la siguiente:

- *CPU.* Procesador AMD Opteron™ 6276, arquitectura x86 de 16 núcleos a 2300 MHz, con un Cache total de 32 MB. Gracias a sus 16 núcleos este procesador es capaz de manejar sin problemas crecientes cargas de trabajo, en ambientes virtualizados y aplicaciones de bases de datos.

- *RAM.* 3 GB. Dado que el servidor alojará distintos servicios (HTTP, Bases de Datos, Python), es necesario contar con suficiente Memoria RAM, además de la memoria de intercambio (swap) del Sistema Operativo.

- *Disco Duro.* 50 GB. La capacidad de almacenamiento es crucial, debido al volumen de información que potencialmente será almacenado en la base de datos.

- *Interfaz de Red.* Intel Corp. 82545EM Gigabit Ethernet Controller. El servidor cuenta con un enlace de cable de par trenzado sin blindaje (UTP por sus siglas en inglés) mediante una interfaz 1000Base-TX (con una velocidad nominal de

transmisión de datos de 1Gbit/s) que garantizará una tasa de transmisión de datos lo suficientemente rápida, capaz de manejar la gran cantidad de tráfico de datos geoespaciales.

4.3 Requerimientos de Software

Esta sección está dividida en dos partes: requerimientos del servidor y requerimientos del cliente.

Requerimientos del servidor

Para fines practicos, centralizamos todos los servicios en un solo servidor. A continución se enlista la paquetería necesaria:

- *Apache Web Server 2.2.22.* La naturaleza Web de la plataforma hace imperativa la implementación de un servidor que reciba las peticiones de los usuarios y regrese las respuestas correspondientes. Apache es un servidor HTTP de código abierto que permite una escalabilidad transparente (mediante la adición de módulos para compatibilidad con servicios de terceros), con un extenso número de usuarios y desarrolladores que dan pie a una gran cantidad de documentación en línea.

- *Python 2.7.3.* Es un lenguaje de programación interpretado, multiplataforma y de fácil integración. Python es de uso libre, incluso para productos comerciales, gracias a su licencia de código abierto.

- *Django Framework 1.5.1.* Framework de alto nivel para Python que permite un desarrollo rápido y un diseño limpio y pragmático. Este framework representa el middleware de la plataforma y el núcleo de la misma. Estará encargado de

la recepción y manipulación de los datos geoespaciales de entrada, la extracción de figuras geométricas de atributos y, finalmente su encapsulamiento en microformatos.

- *PostgreSQL 9.1.9.* Sistema Gestor de Bases de Datos de código abierto multiplataforma. El módulo espacial que se utiliza se denomina *PostGIS*. Este módulo habilitará espacialmente nuestra base de datos, permitiendo almacenar figuras geométricas y ejecutar consultas espaciales.

- *SQLite 3.7.13.* Librería que implementa un motor SQL de bases de datos autónomo, que interactúa directamente con el proceso que desea acceder a la base de datos (sin un proceso de servidor intermedio). El módulo espacial utilizado por SQLite es conocido como *SpatiaLite*.

Requerimientos del cliente

Dada la naturaleza Web de la plataforma propuesta, el cliente podrá ejecutar el Sistema Operativo de su preferencia (Linux, OSX, Windows o Solaris). Los requerimientos del cliente constan básicamente de un navegador Web de reciente generación, compatible con HTML 5 y Javascript para los API de los Servicios de Mapas. Algunos de estos navegadores son:

- *Google Chrome.* Navegador Web desarrollado por Google, que combina un diseño minimalista con un conjunto de herramientas para desarrolladores Web. Es de código parcialmente abierto y orientación multiplataforma.

- *Chromium.* Navegador Web de código abierto que propone un enfoque simple, rápido y seguro para la navegación en Internet. El código fuente de Google Chrome está basado en el código abierto de Chromium.

- *Mozilla Firefox.* Navegador Web multiplataforma de código abierto, desarrollado por la Corporación Mozilla. Al igual que Google Chrome, Firefox incorpora una serie de herramientas para desarrollo Web.

En el capítulo 5, se presentan la estructura de la plataforma Web y se describen de manera detallada los procesos de los módulos de carga y almacenamiento de archivos Shapefiles, así como los procesos de encapsulamiento y visualización de datos geoespaciales.

Capítulo 5

Descripción de la plataforma

En este capítulo se describirá el funcionamiento de la plataforma, el proceso de entrada de datos geoespaciales, su encapsulamiento y finalmente su visualización. De igual forma serán descritos los principales módulos que la conforman.

5.1 Control de acceso y autenticación

La información geoespacial que será alimentada a la plataforma de encapsulamiento podría contener información sensible o clasificada que, por razones de seguridad o privacidad, no deberán estar disponibles a todos los usuarios de la plataforma. Es por esto que se ha implementado un módulo de control de acceso y autenticación utilizando las interfaces de administración propias de Django (Figura 14). La plataforma manejará 3 tipos distintos de usuarios:

- *Administradores.* Estos usuarios tendrán acceso total a todos los módulos del sistema. Serán los únicos usuarios con privilegios para agregar y eliminar usuarios.
- *Propietarios.* Estos usuarios podrán cargar y eliminar Shapefiles. De igual manera, podrán visualizar los datos encapsulados.
- *Anónimos.* Estos usuarios únicamente podrán visualizar los datos encapsulados, no podrán cargar ni eliminar archivos Shapefile en la plataforma.

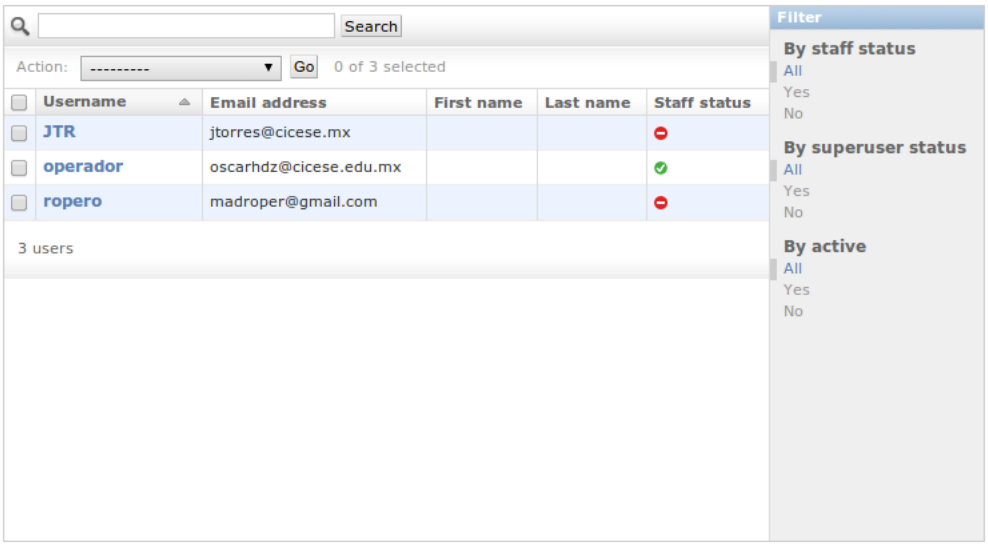

Figura 14. Interfaz de administración de cuentas del sistema.

5.2 Carga y validación de Shapefiles en la plataforma

La carga de los datos geoespaciales a la plataforma representa el primer paso para el encapsulamiento en microformatos. Como se mencionó en el Capítulo 4, los conjuntos de datos que habrán de alimentar al sistema deberán estar en formato Shapefile, el cual consta de 4 archivos distintos[1] (shp, shx, dbf y prj). Debido a esto, los datos deberán ser comprimidos en formato ZIP previo a su carga en el sistema.

La carga del Zip, la descompresión y validación de los contenidos del mismo (i.e. verificar la existencia de los 4 archivos correspondientes) así como la detección y corrección de errores en las geometrías del Shapefile (e.g. polígonos abiertos o con auto-intersecciones) se ilustran en el diagrama de flujo de la Figura 15.

[1]Los archivos Shapefile pueden contener además otros archivos con extensiones .sbn y.sbx. La presencia de estos archivos no afectará la carga del archivo

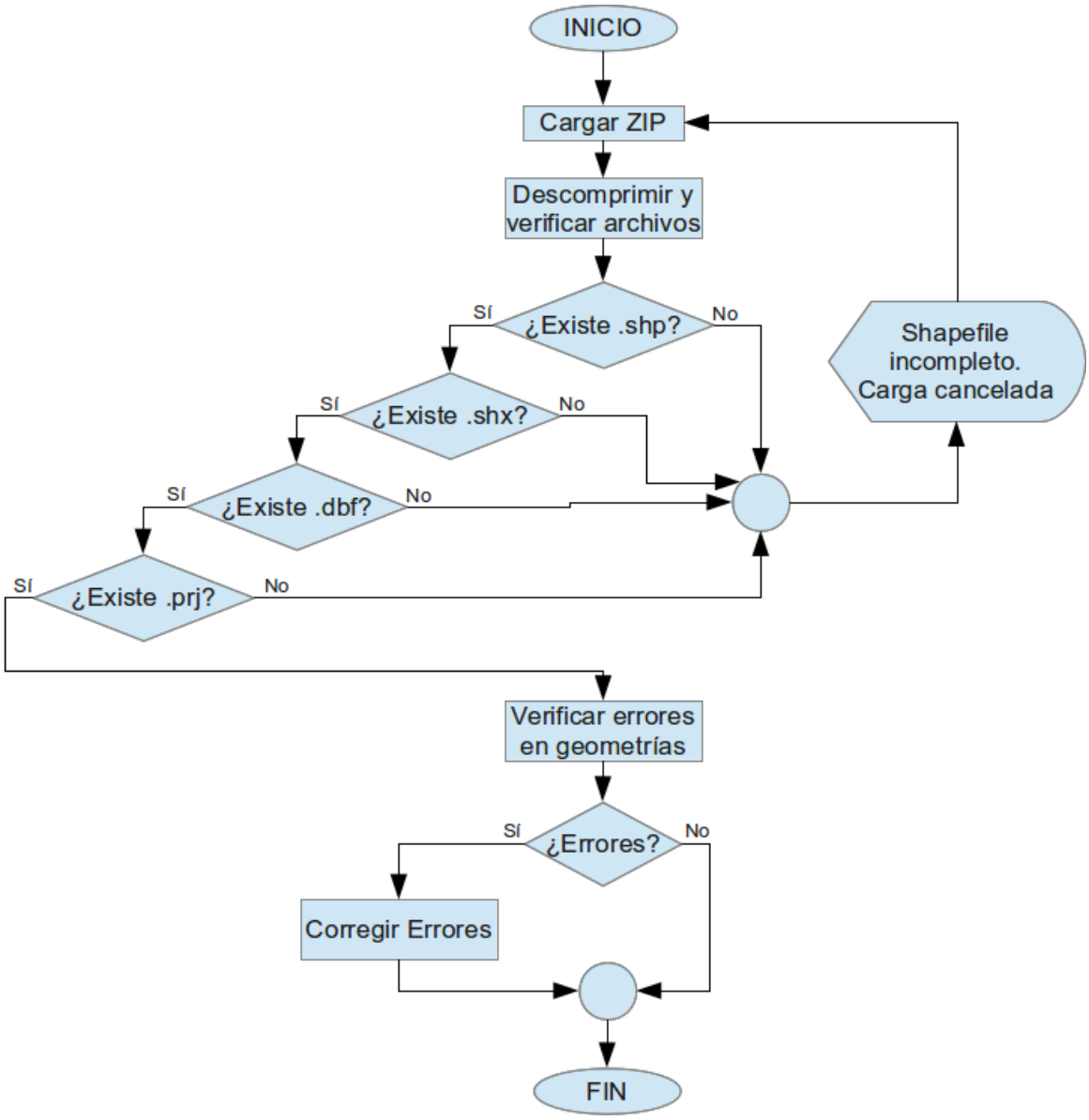

Figura 15. Diagrama de flujo de la carga y validación de archivos Shapefile.

5.3 Almacenamiento de Shapefiles en base de datos

Una vez que los procesos de carga, validación y corrección de errores en los datos geoespaciales han concluído satisfactoriamente, es necesario almacenar tanto la información geoespacial (figuras geométricas) como su información tabular dentro de la base de

datos. Para esto es necesario, en primera instancia, leer y extraer los contenidos del archivo Shapefile.

5.3.1 Extracción y almacenamiento de geometrías

En el caso de las figuras geométricas se efectúa un barrido del Shapefile, extrayendo cada figura geométrica y transformando su Sistema de Referencia Espacial (SRS por sus siglas en inglés) al sistema EPSG 4326 (valores de longitud y latitud sin proyección). Posteriormente se convierte la figura geométrica a formato WKT para su almacenamiento en la base de datos. Este proceso se detalla en el diagrama de flujo de la Figura 16.

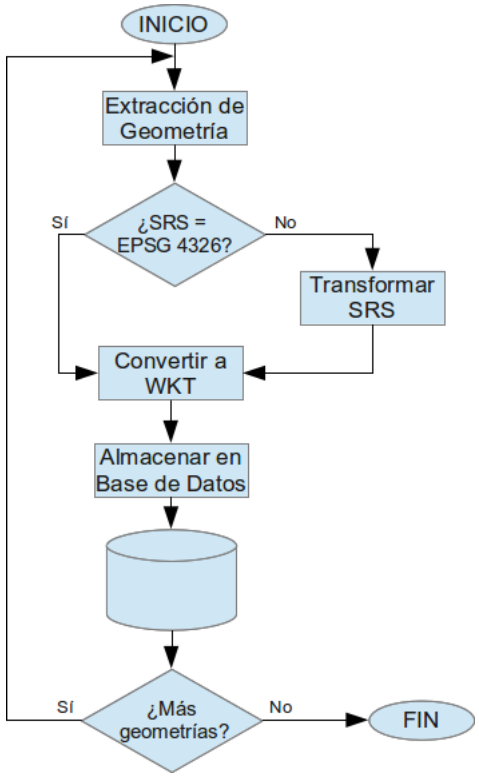

Figura 16. Diagrama de flujo de la extracción y almacenamiento de geometrías.

5.3.2 Extracción y almacenamiento de atributos

Una vez almacenadas las geometrías procedemos al almacenamiento de sus atributos. Para esto, al igual que con las geometrías, se efectúa un barrido de los atributos contenidos en el archivo Shapefile, durante el cuál, se extraerán sus valores para posteriormente almacenarlos dentro de la base de datos. El diagrama de la Figura 17 detalla este proceso.

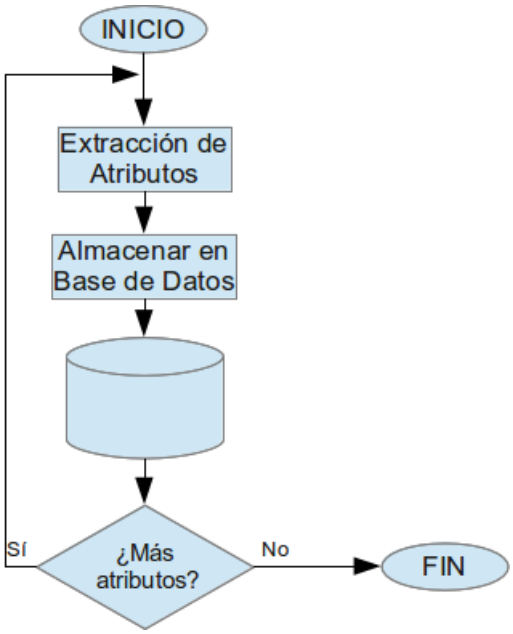

Figura 17. Diagrama de flujo de la extracción y almacenamiento de atributos.

5.4 Encapsulamiento en microformatos

El siguiente paso, una vez que las geometrías y sus atributos han sido almacenados en la base de datos, es el encapsulamiento de estos en los microformatos KML y GML. Una vez encapsulados estos datos geoespaciales podrán ser visualizados mediante WMS con la ayuda de los API de Google Maps y Openlayers.

Para realizar el encapsulamiento es necesario extraer las geometrías y los atributos desde la base de datos. Posteriormente, mediante un analizador sintáctico (comunmente denominado *parser*), se efectúa el encapsulamiento y se almacenan los archivos dentro del servidor. Este encapsulamiento se muestra en la Figura 18.

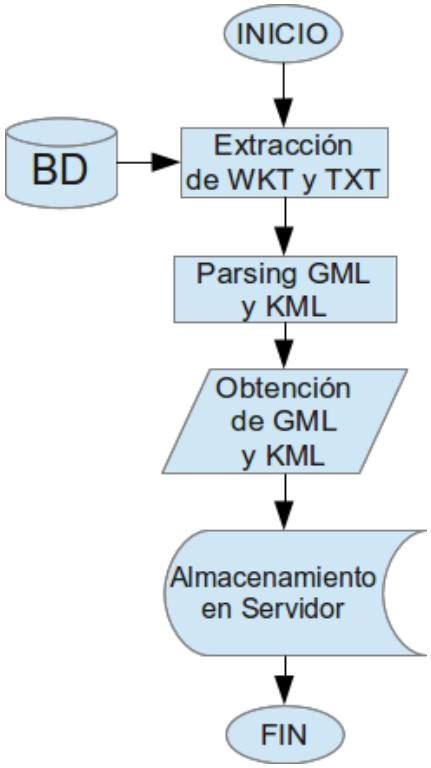

Figura 18. Diagrama de flujo del encapsulamiento en microformatos.

5.5 Visualización de datos geoespaciales encapsulados

Una vez realizado el encapsulamiento y almacenamiento de los archivos GML y KML, es posible efectuar la visualización de los datos geoespaciales y atributos contenidos en los mismos. Para esto se utilizarán los API proporcionados por Google Maps y Openlayers. Se optó por el uso de los API antes mencionados debido, por un lado, a

la popularidad de Google Maps y su soporte para archivos KML (aunque como ya se mencionó en el Capítulo 2, KML fue originalmente diseñado para desplegar información geoespacial en Google Earth). Por otro lado, Openlayers se está posicionando como uno de los visualizadores de datos geoespaciales más populares dentro de la comunidad de Software de Código Abierto. Esto, aunado al hecho de que Openlayers está basado en Javascript, brinda gran libertad al desarrollador en cuanto al potencial para visualizar los datos geoespaciales.

Además, la utilización de estos dos API obedece al hecho de que se desea evaluar el desempeño de ambos Servicios de Mapas en el despliegue de conjuntos de datos geoespaciales con distinto número de geometrías y atributos (desde decenas de polígonos, hasta millares de líneas).

5.5.1 Visualización con Google Maps API

El API de Google Maps fue implementado en esta plataforma para el despliegue de datos geoespaciales encapsulados en formato KML. Dentro de las relativas libertades que brinda este API, se encuentra la posibilidad de seleccionar la capa WMS de nuestra preferencia (Mapa, Terreno, Satélite e Híbrida), todas proporcionadas por Google. De igual forma el API de Google Maps proporciona, por defecto, herramientas de zoom y pan (panorámica) para la navegación dentro del mapa, así como herramientas para el popular Google Street View, el cual permite a los usuarios un recorrido virtual por las calles y carreteras del mundo (aunque esta funcionalidad no contemplará la visualización de los datos geoespaciales). La Figura 19 muestra estas herramientas dentro de Google Maps API.

Figura 19. Herramientas de navegación y visualización de Google Maps API.

5.5.2 Visualización con Openlayers API

La implementación del API de Openlayers en la plataforma se realizó para la visualización de datos geoespaciales en formato GML. Las libertades otorgadas por este API para la manipulación y visualización, tanto de datos geoespaciales como de capas WMS, son prácticamente absolutas.

Openlayers proporciona una extensa librería de controles que afectan la visualización y comportamiento del mapa. Dichos controles incluyen navegación dentro del mapa (zoom y pan), selector de capas, habilitación de control por teclado, etc.

Con la finalidad de facilitar y hacer más intuitiva la visualización de datos geoespaciales y la navegación dentro del mapa, se implementaron los siguientes controles:

- *Barra de zoom y panorámica.* Para una navegación más intuitiva dentro del mapa, mediante la manipulación de la barra de zoom y los botones de *panning* que permiten moverse en cualquier dirección cardinal.

- *Hand y Zoom Box.* Para navegación mediante arrastre de cursor y zoom personalizado mediante el dibujo de un rectángulo con el cursor sobre el área en la cual se desea hacer el acercamiento.

- *Navegación por teclado.* Para navegación por medio de teclado. Este control activa las teclas de dirección para el *panning* y las teclas + y - para zoom.

- *Selector de Capas.* Para visualización de capas WMS propias de Openlayers, así como de distintas organizaciones (incluídos las capas de Google Maps mencionadas anteriormente). Este control permite además, la activación o desactivación de la capa que despliega nuestros datos geoespaciales GML.

La Figura 20 muestra la implementación de las herramientas mencionadas dentro del API de Openlayers.

Figura 20. Herramientas de navegación y visualización de Openlayers API.

5.5.3 Extracción y visualización de atributos

En cuanto a la extracción y visualización de los atributos de las geometrías encapsuladas, Openlayers cuenta con una librería para carga de archivos de vector en varios formatos (para este caso GML) que permite la extracción de atributos de los mismos. Dado que toda la información geoespacial es obtenida directamente de los archivos encapsulados, no se efectúan peticiones a la base de datos, con lo cual, los tiempos de respuesta disminuyen considerablemente, siendo esta otra de las ventajas de la utilización de microformatos para despliegue de conjuntos de datos espaciales en la Web.

Una vez extraídos los atributos, la visualización en el navegador Web será efectuada mediante la transferencia de dichos atributos a objetos HTML5 vía Javascript. El diagrama de flujo de la Figura 21 muestra este proceso.

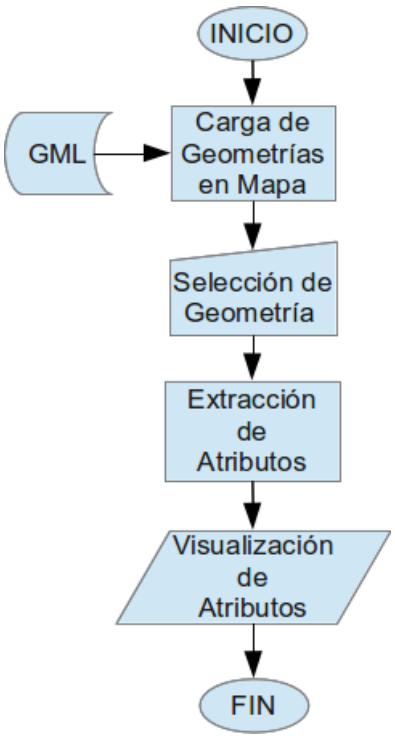

Figura 21. Diagrama de flujo de la visualización de atributos.

En el capítulo 6 se describirá la implementación práctica de la plataforma, utilizando conjuntos de datos reales obtenidos de diversas organizaciones, lo cual demuestra el potencial de esta plataforma Web.

Capítulo 6

Implementación práctica de la plataforma

En este capítulo se mostrará y describirá la implementación práctica de la plataforma Web, denominada GML Wrapper, mediante la utilización de diversos archivos Shapefile, obtenidos de diversas organizaciones nacionales e internacionales. Estos conjuntos de datos geoespaciales contendrán polilíneas (e.g. de vías terrestres, hidrología y curvas de nivel) y multipolígonos (e.g. áreas administrativas, uso del suelo, etc.), y tendrán diferentes codificaciones de caracteres y SRS.

Se detallará cada fase de la interacción del usuario con la plataforma, desde la carga del archivo hasta la visualización de los datos geoespaciales encapsulados. De igual manera se efectuarán comparaciones entre el desempeño de los Servicios de Mapas de Google y Openlayers en el manejo de formatos KML y GML respectivamente.

6.1 Acceso a la plataforma y autenticación

Para tener acceso a la plataforma Web se deberá acceder desde alguno de los navegadores Web mencionados en el Capítulo 4, al subdominio REIS del sitio cicese.mx desde el puerto 8000 (mediante el URL http://reis.cicese.mx:8000); es necesario especificar el puerto 8000 para acceder al servidor de desarrollo Django y no al directorio raiz del servidor Web Apache, ya que este lo utilizamos únicamente como servidor de producción y no de desarrollo, es decir, Apache únicamente establecerá las conexiones remotas al servidor.

La Figura 22 muestra la pantalla de inicio, donde el usuario podrá autenti-

carse mediante el inicio de sesión en caso de estar registrado como usuario, navegar en sesión anónima o bien, solicitar la obtención de nombre de usuario y contraseña al administrador.

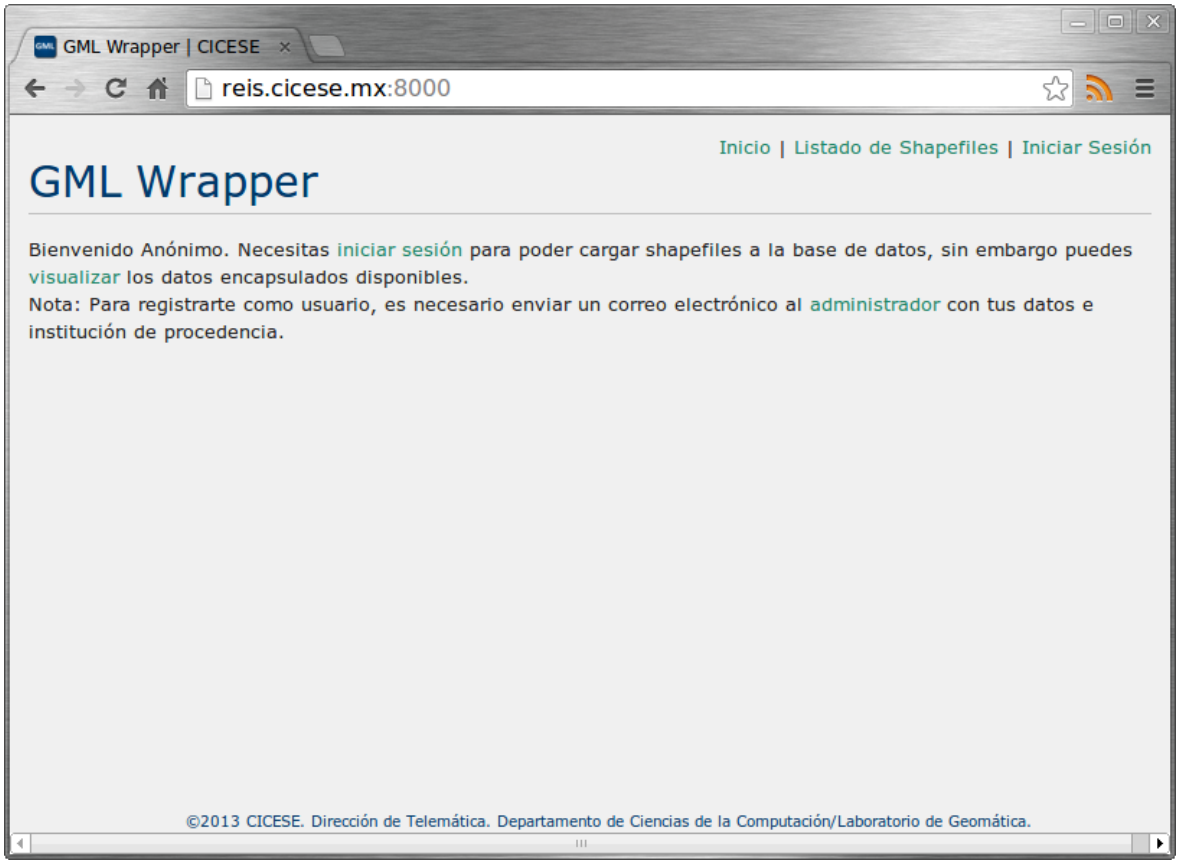

Figura 22. Pantalla de inicio de la plataforma GML Wrapper.

Si el usuario ya ha sido dado de alta y ha recibido una cuenta de autenticación para la plataforma, podrá acceder al módulo de inicio de sesión e introducir el nombre de usuario y contraseña otorgados por el administrador. La Figura 23 muestra la forma HTML para inicio de sesión. Para acceder a este módulo, se da click en en hipervínculo "Iniciar Sesión" del menú situado en la esquina superior derecha de la pantalla.

Figura 23. Forma para autenticación de la plataforma GML Wrapper.

6.2 Carga, almacenamiento y encapsulamiento de datos geoespaciales.

Una vez que el usuario ha iniciado sesión, éste podrá acceder al módulo de carga, visualización de datos y eliminación de Shapefiles dando click en el hipervínculo "Listado de Shapefiles" del menú. En este módulo, mostrado en la Figura 24, el usuario podrá cargar archivos Shapefile comprimidos al sistema. Una vez cargados, serán descomprimidos y validados como se detalló en el capítulo 5, para posteriormente ser almacenados en la base de datos para su encapsulamiento.

Figura 24. Módulo de carga y visualización GML Wrapper.

6.2.1 Carga y validación de archivos Shapefile

Para cargar un archivo Shapefile, se dará click en el botón "Importar Shapefile", el cual desplegará la forma HTML para la carga de archivos y selección de la codificación de caracteres correspondiente (Figura 25). Una vez cargado, el archivo será extraído, validado y, en caso de contener errores, estos serán corregidos.

Figura 25. Forma HTML para la carga de archivos Shapefile GML Wrapper.

La validación consistirá en verificar la integridad del archivo ZIP (en caso de tratar de cargar cualquier otro tipo de archivo, el sistema notificará que el archivo que se intenta subir no es válido) y comprobar que contenga los 4 archivos de Shapefile. De igual forma, si faltara alguno de estos archivos, la carga no se llevaría a cabo y se notificaría al usuario la falta del archivo.

6.2.2 Almacenamiento en base de datos

El tiempo de almacenamiento de geometrías y atributos en la base de datos variará en función del número de registros que se crearán dentro de la misma: Se almacenará un registro por cada atributo de cada geometría contenida dentro del archivo Shapefile. Esto es:

$$Regs = Geoms \times Atribs \tag{1}$$

donde *Regs* es el total de registros a ser almacenados dentro de la base de datos, *Geoms* es el total de figuras geométricas y *Atribs* el total de atributos.

Entonces, para un archivo Shapefile que cuente con 246 figuras geométricas y 11 atributos relacionados a ellas, se crearán 2706 registros en la base de datos.

Django cuenta con un monitor de eventos de servidor (conocido como log en el ámbito informático). Este monitor (mostrado en la Figura 26) registrará todas las peticiones hechas al servidor, así como sus respuestas, y los eventos y resultados relacionados con los procesos del mismo. Por ejemplo, mostrará el proceso de almacenamiento de registros en la base de datos, detalles de los errores encontrados (en caso de haberlos) y los detalles del encapsulamiento.

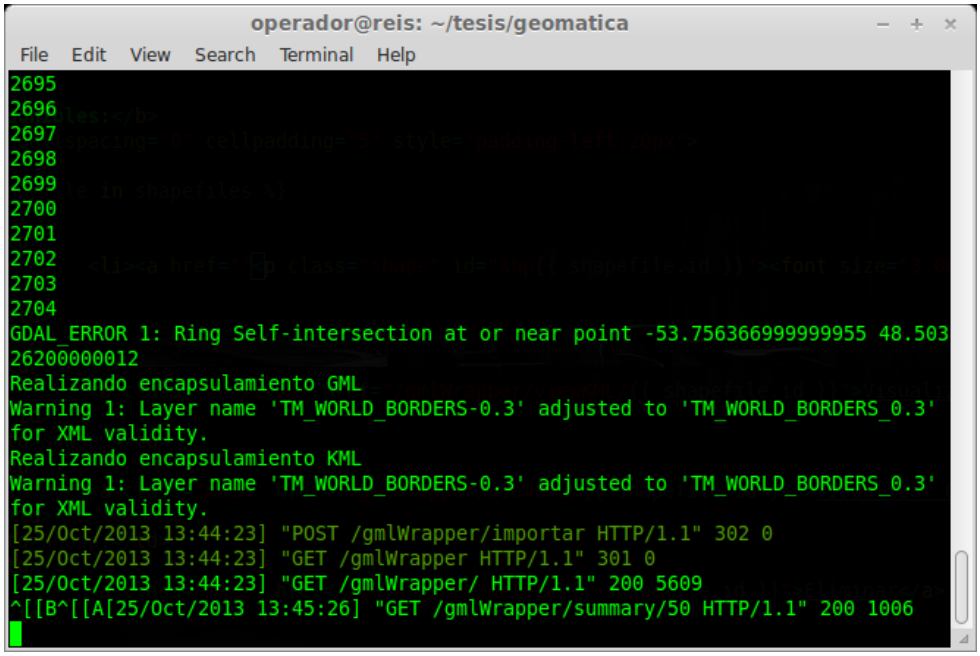

Figura 26. Monitor de eventos del servidor de GML Wrapper.

6.2.3 Encapsulamiento de datos geoespaciales

El encapsulamiento de información geoespacial se efectuará de manera automática e independiente del usuario, es decir, después de cargar los datos geoespaciales al sistema, el usuario no necesitará invocar ningun otro proceso en la plataforma para realizar el encapsulamiento.

El proceso de encapsulamiento será efectuado con la ayuda de librerías GDAL (Geospatial Data Abstraction Library) una vez que el contenido de los archivos Shapefile han sido almacenados en la base de datos.

Cuando la carga de información hacia la base de datos y el encapsulamiento finalicen, el usuario podrá visualizar los datos encapuslados accediendo al listado de Shapefiles desde el menú principal y posteriormente seleccionar que datos geoespaciales desea visualizar, ya sea GML o KML. Una vez hecho esto, se cargará el API correspondiente (Google Maps para visualización de archivos KML y Openlayers para visualización de archivos GML).

Visualización de archivos KML

Para la visualización de los datos encapsulados en formato KML, el Servicio de Mapas seleccionado fue Google Maps debido a que éste, junto con Keyhole, fueron los principales desarrolladores de este formato. No obstante, debido a que KML fue originalmente desarrollado para compatibilidad con Google Earth, la visualización de este tipo de archivos en Google Maps presenta algunas restricciones que alteran la presentación de la información geoespacial y la interacción cliente/servidor. (e.g. la carga de información geoespacial podría hacerse de forma incompleta y se necesitaría la uti-

lización de un parser externo[1] para entornos de red local).

La carga y visualización de datos geoespaciales en formato KML se lleva a cabo dando click en el hipervínculo "Visualizar KML". Dado el consumo masivo de recursos y el tiempo de procesamiento que toma el despliegue de los datos geoespaciales KML en el mapa, se optó por alertar al usuario acerca del consumo de recursos de desplegar archivos KML con gran cantidad de geometrías y atributos (Figura 27).

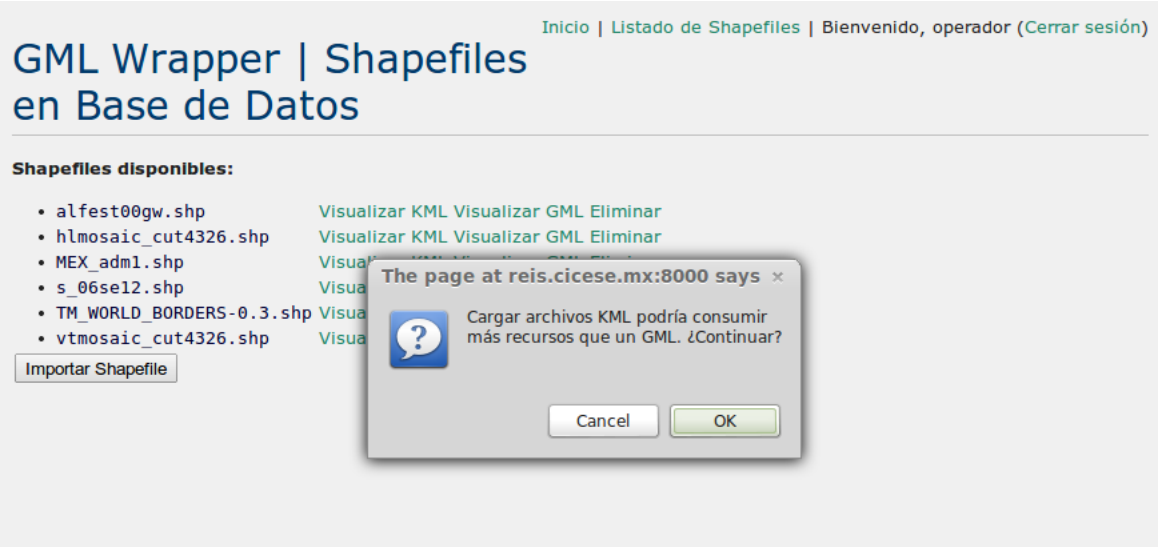

Figura 27. Alerta en visualización de archivos KML.

Al acceder al API de Google Maps las librerías de GeoXML3 comenzarán a realizar el parsing del archivo seleccionado para su visualización en el mapa. Dependiendo de la cantidad de geometrías, así como del número de atributos relacionados con las mismas, el tiempo de carga variará.

[1]Debido a que el archivo KML debe ser obtenido por un servidor de Google para su carga en Google Maps, éstos deben estar alojados en un servidor en Internet (públicamente accesible). Sin embargo, dado que la implementación y las pruebas se hicieron en un entorno local (en la red local de CICESE), fue necesario utilizar un *parser* que fungiera como el servidor de Google e interpretara el contenido del archivo KML.

En la Figura 28 se muestra un conjunto de datos geoespaciales desplegado en Google Maps, el cual consta de 246 polígonos, cada uno de los cuales representa un país (incluyendo sus respectivas islas y/o territorios no incorporados) y tiene ligados 11 atributos. Esto es, serán extraídos 2706 registros mismos que serán visualizados en la Web como geometrías (en Google Maps) y como texto HTML.

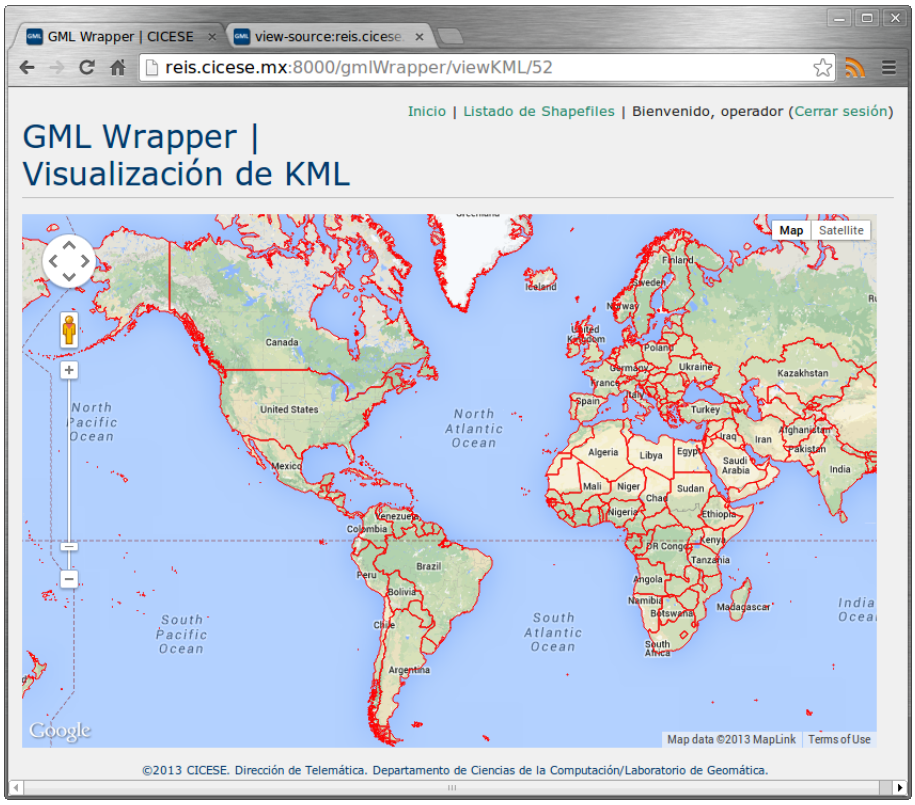

Figura 28. Visualización de archivos KML en Google Maps API.

El tamaño y complejidad (nivel de detalle y cantidad de nodos de los vectores) de los archivos KML que se desean desplegar serán factores cruciales en el tiempo total de carga y visualización de los mismos en el mapa. En la Sección 6.4 se hace una comparativa de tiempos de carga y visualización entre archivos KML y GML (con la misma información geoespacial) en los Servidores de Mapas antes mencionados.

Visualización de archivos GML

En el caso de la información geoespacial encapsulada en formato GML, se optó por la utilización de Openlayers como Servidor de Mapas base. Openlayers es capaz de desplegar datos vector en diversos formatos. A diferencia del API de Google Maps, Openlayers es capaz de extraer y realizar el parsing de información geoespacial en entornos de red de alcance local (Redes de Área Local) y amplio (Redes de Área Amplia), por lo cual no es necesario el uso de librerías externas para realizar el parsing. La Figura 29 muestra la visualización de un archivo GML en Openlayers.

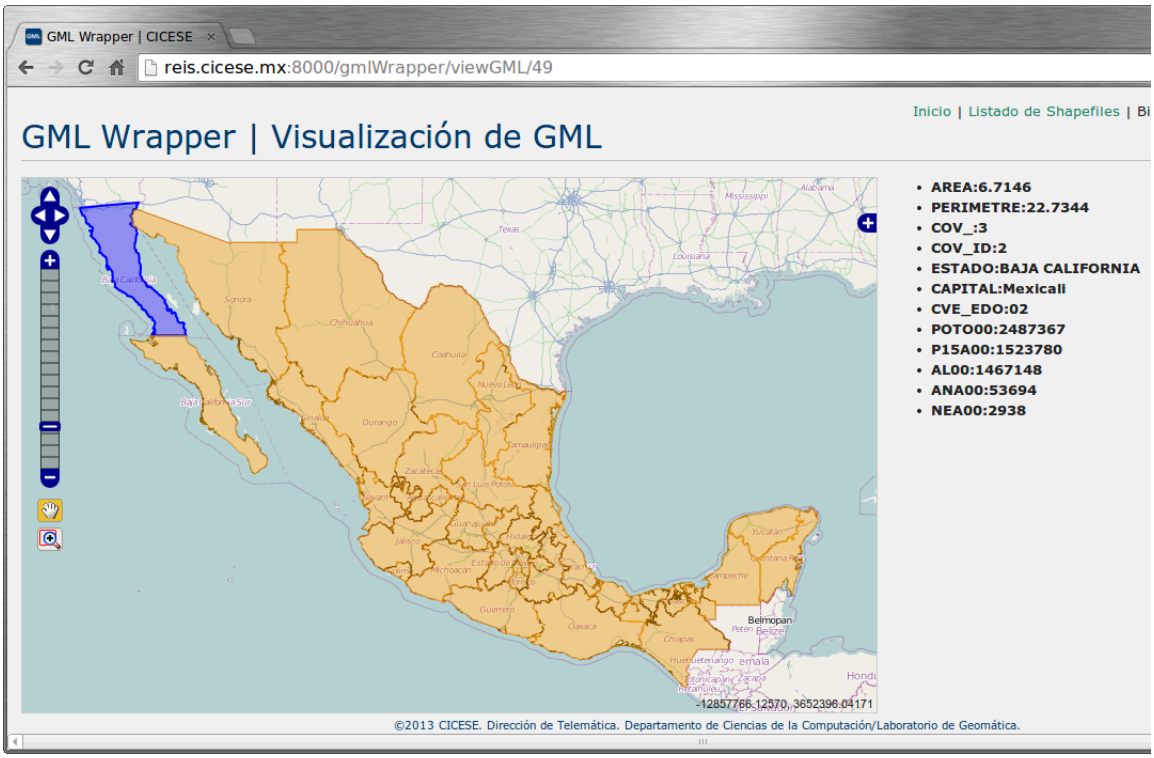

Figura 29. Visualización de archivos GML en Openlayers API.

Este archivo cuenta con 32 polígonos que representan a cada uno de los estados de la República Mexicana y contienen 12 atributos con información tabular referente

al índice de analfabetismo por entidad.

Otra de las bondades de Openlayers API es que proporciona la función binaria "ExtractAttributes" que, cuando es declarada con valor verdadero, extraerá los atributos de todas las geometrías contenidas dentro del archivo. La implementación de esta función dentro de la plataforma fue manejada a través de eventos del mouse:

- *Hover.* Este evento se activa cuando el cursor flota por encima del objeto (en este caso, la figura geométrica). Este evento dará lugar al despliegue de un objeto HTML el cual contendrá la información tabular relacionada con la geometría sobre la cual se encuentre el cursor. Cuando el cursor sea retirado del área comprendida por dicha geometría, los atributos desaparecerán.

- *Click.* Este evento es activado con el click primario del mouse. Al dar click sobre algúna geometría, esta será coloreada de color azul marino y sus atributos serán desplegados. La diferencia radica en que, aunque el cursor sea retirado de la geometría después de dar click, los atributos permanecerán visibles, como se muestra en la Figura 29).

6.3 Eliminación de datos geoespaciales de la base de datos

Para la eliminación de Datos Geoespaciales es necesario el nivel de usuario *Propietario* o *Administrador*. El hipervínculo "Eliminar" nos enviará directamente a la pantalla mostrada en la Figura 30. En esta pantalla tendremos dos opciones: Eliminar el contenido del archivo Shapefile de la base de datos o Cancelar la eliminación. Si se selecciona *Eliminar*, la información del archivo seleccionado será eliminada de manera definitiva de la base de datos. En cambio, si *Cancelar* es seleccionado, el usuario será redireccionado al listado de Shapefiles.

Figura 30. Eliminación de datos geoespaciales en GML Wrapper.

6.4 Comparativa de desempeño en la visualización de datos geoespaciales

Se optó por utilizar los API de Google Maps y Openlayers con la finalidad de realizar una evaluación del desempeño de ambos Servicios de Mapas en la carga y visualización de información geoespacial encapsulada.

Para la realización de esta evaluación se utilizaron 6 conjuntos de datos geoespaciales distintos, encapsulados en formatos KML y GML. La visualización de archivos KML se efectuó sobre el API de Google Maps, mientras que la visualización de archivos GML se realizó sobre el API de Openlayers. La Tabla IV detalla las características de dichos conjuntos.

Tabla IV. Conjuntos de datos geoespaciales utilizados para pruebas de visualización.

Conjunto de Datos	**Tipo de Geometrías**	**No. Geometrías**	**No. Atributos**	**Tamaño (MB)**
alfest00gw.kml	Polígonos	32	12	22.4
alfest00gw.gml	Polígonos	32	12	22.4
hlmosaic_cut.kml	Líneas	7458	29	15.0
hlmosaic_cut.gml	Líneas	7458	29	12.9
MEX_adm1.kml	Polígonos	32	12	15.8
MEX_adm1.gml	Polígonos	32	12	15.9
s_06se12.kml	Polígonos	56	6	68.9
s_06se12.gml	Polígonos	56	6	69.6
WORLD_BORDERS.kml	Polígonos	246	11	9.6
WORLD_BORDERS.gml	Polígonos	246	11	9.8
vtmosaic_cut.kml	Líneas	9913	29	19.1
vtmosaic_cut.gml	Líneas	9913	29	16.3

En general, el desempeño de Openlayers en el despliegue de datos geoespaciales fue mejor que el de Google Maps. Los tiempos de carga y visualización fueron contrastantes especialmente en la carga de polígonos con gran nivel de complejidad y detalle, siendo Openlayers más rápido que Google Maps. En el caso de los conjuntos de datos formados por líneas y/o polilíneas, los cuales contienen una mayor cantidad de geometrías, Google Maps presentó problemas con la carga y visualización de dichas geometrías. En esta sección se presenta la comparativa de desempeño de ambos Servicios de Mapas.

6.4.1 Tiempos de carga y visualización de datos geoespaciales

La evaluación de desempeño de la carga de conjuntos de datos geoespaciales sobre la Plataforma GML Wrapper se realizó con grupos de 2 conjuntos de datos idénticos, encapsulados en los formatos KML y GML (Tabla IV). La Figura 31 muestra la gráfica de los tiempos de carga de los conjuntos de datos geoespaciales compuestos por polígonos.

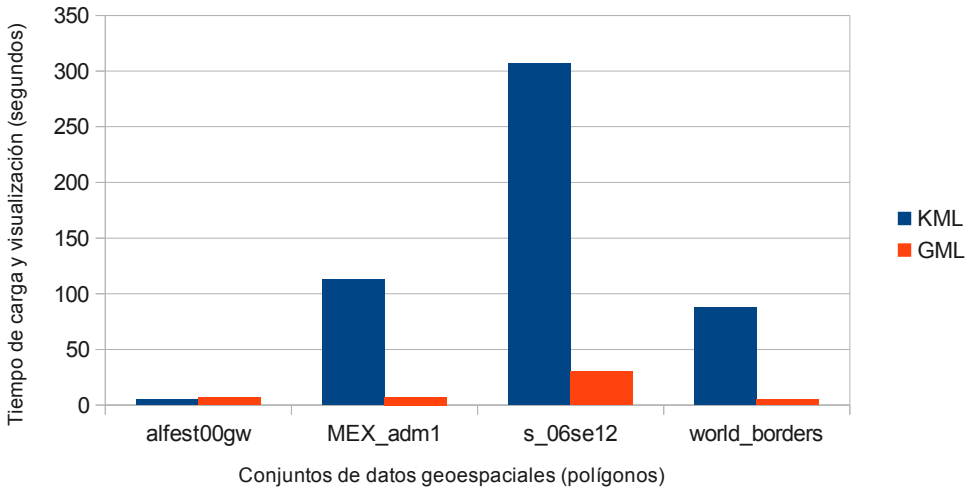

Figura 31. Tiempos de carga y visualización de polígonos.

Openlayers tuvo un desempeño mejor en la carga y visualización de polígonos. En el caso particular del conjunto de datos geoespaciales *s_06se12.kml*, el cual contiene la división política de los Estados Unidos de América con gran nivel de complejidad y detalle (68.9 MB contra 9.6 MB de WORLD_BORDERS que contiene todos los países del mundo), Google Maps colapsó después de una espera de 306.72 segundos. Por su parte, Openlayers cargó y desplegó dicha información geoespacial (encapsulada en formato GML) en tan solo 30.24 segundos.

Aunque Google Maps fue más rápido que Openlayers en la carga y visualización de líneas y polilíneas, éstas fueron cargadas de manera incompleta, debido a restricciones y limitaciones de despliegue de archivos KML (discutidas en el Capítulo 4). La Figura 32 muestra la gráfica de estos tiempos.

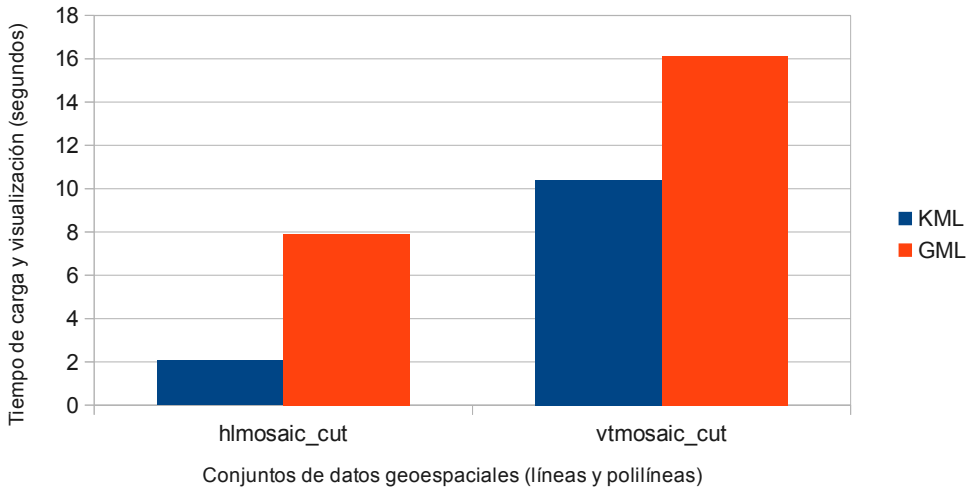

Figura 32. Tiempos de carga y visualización de líneas y polilíneas.

Las Figuras 33 y 34 muestran la comparativa entre la visualización del conjunto *hlmosaic_cut*, el cual contiene información hidrológica de la región de Ensenada B.C.

Figura 33. Carga y visualización de capa hidrológica KML en Google Maps.

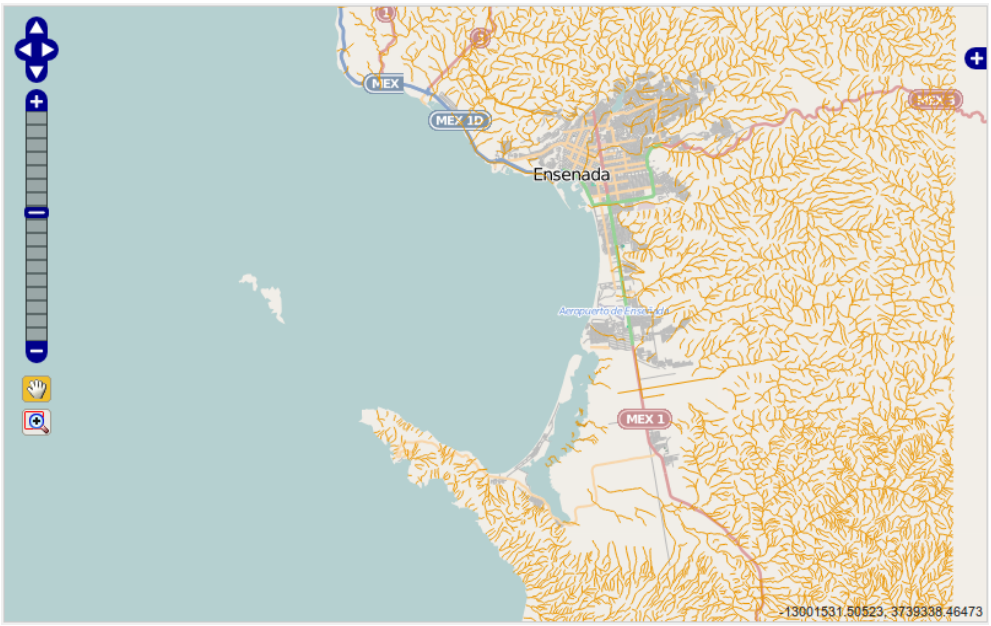

Figura 34. Carga y visualización de capa hidrológica GML en Openlayers.

Como puede apreciarse, el porcentaje de datos geoespaciales cargados en Google Maps fue del 5% aproximadamente, mientras que en el caso de Openlayers fue del 100%.

El segundo archivo cargado fue *vtmosaic_cut*, el cual contiene las vías de comunicación terrestres de la región de Ensenada B.C. (Figuras 35 y 36).

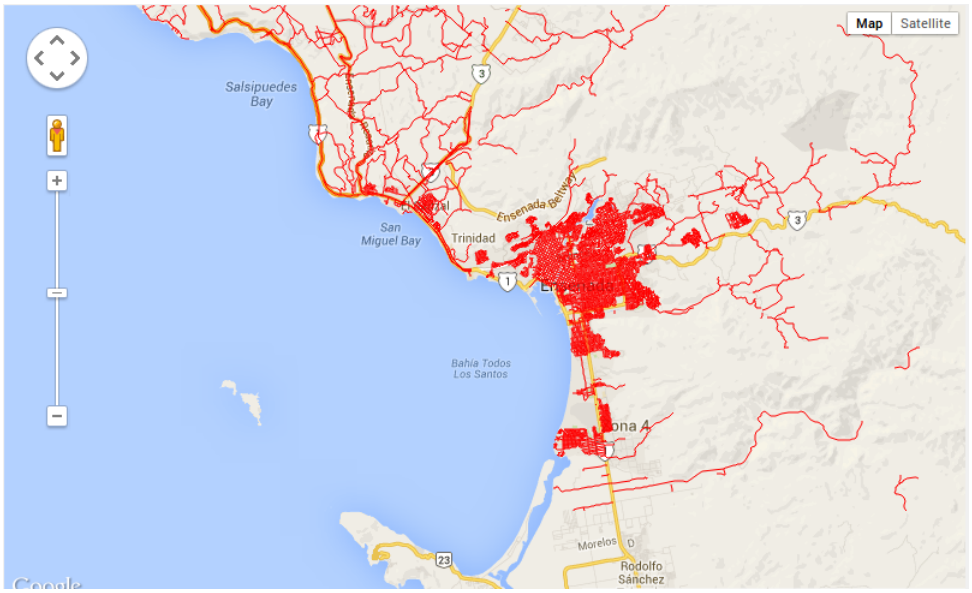

Figura 35. Carga y visualización de vías terrestres KML en Google Maps.

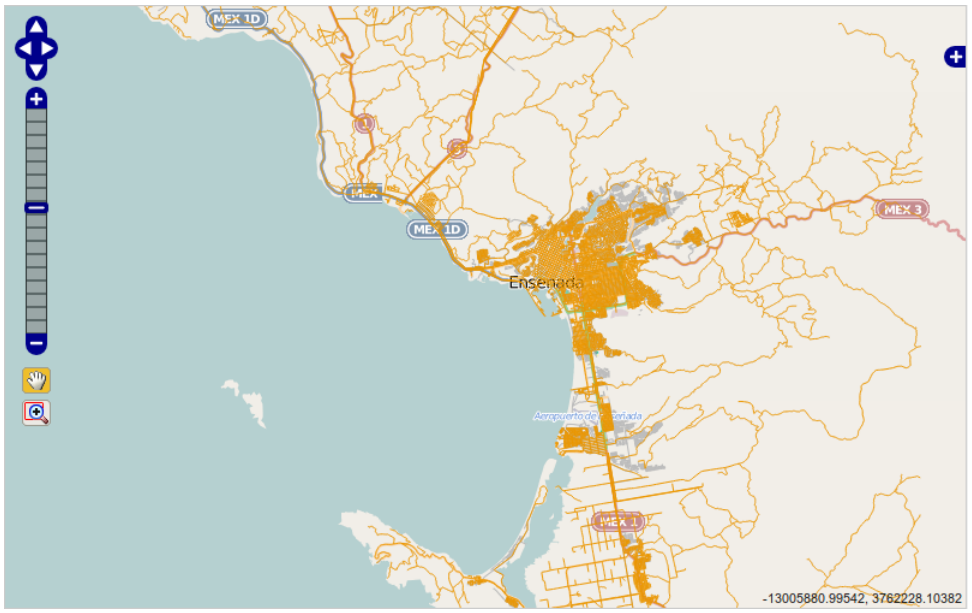

Figura 36. Carga y visualización de vías terrestres GML en Openlayers.

La carga de este último conjunto de datos en Google Maps, aunque incompleta, fue de alrededor del 90% del total de geometrías. En el caso de Openlayers, las geometrías fueron cargadas en su totalidad con un tiempo de 16.13 segundos.

Además de tener un mejor desempeño en la carga y visualización de información geoespacial, Openlayers ofrece una interacción más fluída y libre al usuario, gracias a la incorporación de diversas herramientas de navegación e interacción dentro del mapa (mencionadas en el Capítulo 5).

Con base en los resultados de este estudio comparativo de desempeño en la visualización de Datos Geoespaciales, podemos concluír que, aunque Google Maps es un excelente Servicio de Visualización de Mapas, su API presenta muchas limitaciones en el despliegue de datos vector KML en la Web, y para su correcta visualización sigue siendo necesaria la utilización de Google Earth o el Google Earth API[2] para navegadores Web. De tal suerte que Openlayers (de código abierto y con un número creciente de usuarios y desarrolladores) se posiciona como la mejor opción para la visualización de información geoespacial en la Web.

[2]El plugin de Google Earth fue descartado de este estudio debido a que actualmente este plugin no cuenta con compatibilidad para entornos Linux, contradiciendo la premisa de desarrollar un sistema multiplataforma.

Capítulo 7

Conclusiones y trabajo futuro

En este capítulo, se presentan las conclusiones de la implementación y las pruebas de desempeño realizadas. De igual forma, se presentan propuestas de trabajo a futuro orientadas al mejoramiento y crecimiento de la plataforma de encapsulamiento.

7.1 Conclusiones

En este trabajo se presentó el diseño, desarrollo e implementación de una Plataforma capaz de visualizar información geoespacial en la Web mediante el encapsulamiento de formatos vector orientados a Sistemas de Información Geográfica de escritorio (TIGER, Shapefile, LYR), en microformatos basados en XML aptos para su despliegue y visualización en la Web, tales como KML y GML. Estos formatos son estándares definidos por el Open Geospatial Consortium (OGC por sus siglas en inglés) y junto con el Servicio de Mapas WMS (también definido por la OGC) permitieron la visualización de datos geoespaciales de manera transparente y eficaz a través de la plataforma Web.

El desarrollo de la plataforma fue hecho con herramientas de código abierto, con la finalidad de maximizar la compatibilidad de implementación sobre cualquier sistema operativo y navegador Web. Del lado del cliente, no fue necesaria la instalación ni configuración de ningún paquete de Software. El único requisito fue la existencia de alguno de los navegadores Web mencionados en el Capítulo 4. Todas las librerías y paqueterías utilizadas para el desarrollo de esta plataforma son de distribución libre, por

lo que la implementación de la misma no está sujeta a restricciones ni licencias con costo.

La implementación y prueba de uso de la plataforma demostró ser una opción práctica y eficiente para la compartición y visualización de datos geoespaciales en la Web. Mediante el uso de esta plataforma, usuarios que se encuentren geográficamente separados y deseen compartir datos geoespaciales heterogénoes y en formatos basados en máquinas de escritorio, no se verán en la necesidad de instalar un Sistema de Información Geográfica de escritorio o ningún tipo paquetería, ya que sus conjuntos de datos geoespaciales serán encapsulados en microformatos estandarizados y adecuados para transporte y visualización a través de la Web, lo cual minimizará los recursos de red, almacenamiento y procesamiento.

7.2 Trabajo a futuro

Existen varias líneas de trabajo que pueden contribuir a la mejora de la plataforma Web. En esta sección mencionamos algunas de estas líneas a manera de sugerencia.

1. La adaptación de la plataforma de encapsulamiento con un Sistema WebGIS integral que, además de visualizar datos geoespaciales, sea capaz de realizar funciones de análisis espacial (sobre datos vector y raster) típicas de un GIS de escritorio, por ejemplo, análisis de cercanía, zonas de buffer, intersección de polígonos, etc. Integrar además módulos para visualización y análisis de nubes de puntos LIDAR con la finalidad de desarrollar una herramienta “ad hoc” orientada a proporcionar soluciones de una gran variedad de problemas específicos de diferentes tipos de usuarios.

2. La segmentación de la información geoespacial origen (proveniente de los Shape-

files) mediante consultas espaciales para el encapsulamiento selectivo de geometrías y atributos. De esta manera se disminuiría el tamaño de los datos geoespaciales (y por ende el consumo de recursos de almacenamiento y red), encapsulando únicamente las geometrías y atributos definidos en las consultas espaciales.

3. La descentralización y redundancia de los servicios alojados en el servidor mediante la implementación y alojamiento en la nube (conocido en el ámbito informático como *Cloud Computing*), haciendo uso de las bondades de servicios como *Software as a Service* (SaaS por sus siglas en inglés), *Platform as a Service* (PaaS) o *Infrastructure as a Service* (IaaS). Esto repercutirá en ahorro de recursos financieros al implementar todos los servicios en la nube, incluídos los aspectos de Hardware al no tener que invertir en equipos de cómputo para el alojamiento de los servicios.

Referencias bibliográficas

Django (2013). Django documentation. Accedido en Abril 22, 2013 de https://docs.djangoproject.com/en/dev/glossary/.

ESRI (2013). Esri shapefile technical description. Accedido en Abril 5, 2013 de http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.

Gomarasca, M. A. (2009). *Basics of Geomatics*. Springer. Milan.

Google (2013a). Keyhole markup language tutorial. Accedido en Septiembre 29, 2013 de https://developers.google.com/kml/documentation/kml_tut.

Google (2013b). Kml support in google maps. Accedido en Septiembre 29, 2013 de https://developers.google.com/kml/documentation/mapsSupport.

Li, Y., Imaizumi, T., Sakata, S., Sekiya, H., y Guan, J. (2008). Spatial data compression techniques for gml. En: *Frontier of Computer Science and Technology, 2008. FCST '08. Japan-China Joint Workshop*, pp. 79–84.

Obe, R. O. y Hsu, L. S. (2011). *PostGIS In Action*. Manning Publications Co. Boston, MA.

OGC (2013). Geography markup language. Accedido en Septiembre 29, 2013 de http://www.opengeospatial.org/standards/gml.

Park, N.-S. y Lee, G. H. (2003). Agent-based web services middleware. En: *Global Telecommunications Conference, 2003. GLOBECOM '03. IEEE*, Vol. 6, pp. 3186–3190.

Tsou, M.-H. y Peng, Z.-R. (2003). *Internet GIS*. John Wiley. San Diego, CA.

Ulloa, I. (2013). *Análisis y Visualización de nubes de puntos LiDAR en Internet*. Tesis de maestría, Centro de Investigacion Cientifica y de Educacion Superior de Ensenada B.C.

Wang, W., Li, C., Wu, Z., Luo, Y., Zeng, Q., Yang, X., Xie, N., y Zhao, X. (2009). A component-based management platform for multi-source spatial data. En: *Information Technology Journals, 2009*, Vol. 8, pp. 529–536, Agosto.

Xia, K. y Wei, C. (2008). Study on real-time navigation data model based on esri shapefile. En: *Embedded Software and Systems Symposia, 2008. ICESS Symposia '08. International Conference*, pp. 174–178.

Yang, C., Wong, D. W., Yang, R., Kafatos, M., y Li, Q. (2005). Performance-improving techniques in web-based gis. Vol. 19, pp. 319–342.

Yang, Huayi Wu, Q. H. Z. L. (2011). Webgis performance issues and solutions. En: *Advances in Web-based GIS, Mapping Services and Applications*, ISPRS Book Series, pp. 121–138. CRC Press.

Ying-jun, D., chong Yu, C., y jie, L. (2009). A study of gis development based on kml and google earth. En: *INC, IMS and IDC, 2009. NCM '09. Fifth International Joint Conference*, pp. 1581–1585.

Zhang, M. y Liu, W. (2010). Distributed interoperable gis data services based on web services. En: *Information Management and Engineering (ICIME), 2010 The 2nd IEEE International Conference on*, pp. 140 –143, Abril.